D4.2
Intelligent D-Band wireless systems and networks initial designs

Editor:	Edwin Yaqub, Rachana Desai (RapidMiner)	
Deliverable nature:	Document, report (R)	
Dissemination level:	Public (PU)	
Date: planned | actual	31.10.2021	19.11.2021
Version | No. of pages	1.0	158
Keywords:	Artificial intelligence, machine learning, network management, radio placement, beamforming, channel estimation, complex event forecasting.	

Abstract

This deliverable gives the results of the ARIADNE project's Task 4.2: Machine Learning based network intelligence. It presents the work conducted on various aspects of network management to deliver system level, qualitative solutions that leverage diverse machine learning techniques. The different chapters present system level, simulation and algorithmic models based on multi-agent reinforcement learning, deep reinforcement learning, learning automata for complex event forecasting, system level model for proactive handovers and resource allocation, model-driven deep learning-based channel estimation and feedbacks as well as strategies for deployment of machine learning based solutions. In short, the D4.2 provides results on promising AI and ML based methods along with their limitations and potentials that have been investigated in the ARIADNE project.

Disclaimer

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 871464. This publication reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

Impressum

Full project title: Artificial Intelligence Aided D-band Network for 5G Long Term Evolution

Short project title: ARIADNE

Number and title of work-package: WP4, D-Band wireless network optimization leveraging ML principles

Number and title of task: T4.2, Machine Learning based network intelligence

Document title: D4.2, Intelligent D-band wireless systems and networks initial designs

Editor: Edwin Yaqub, Rachana Desai (RapidMiner)

Work-package leader: NCSRD

Copyright notice

List of authors

Company	Author	Contribution
CNRS	Marco Di Renzo	Contributions to Chapter 5.
CNRS-CentraleSupelec	Xuewen Qian	Contributions to Chapter 5.
EURESCOM	Halid Hrasnica	Final review of the document
NCSRD	Nikos Katzouris, Kyriakos Manganaris, Dimitris Selimis, Fotis Lazarakis	Contributions to Chapter 3 and 4. Editing of the document. Review of the document.
NOKIA	Tachporn Sanguanpuak, Heikki Halmetoja, Moamen Ibrahim	Contributions to Chapter 1 and 2.
RAPIDMINER	Edwin Yaqub, Rachana Desai, Ralf Klinkenberg	Contributions to Chapter 7. Editors of the document (Edwin Yaqub, Rachana Desai). Review of the document.
UOULU	Joonas Kokkoniemi	Review of the document.
UPRC	Alexandros-Apostolos A. Boulogeorgos, Droulias Sotiris, Angeliki Alexiou	Contributions to Chapter 6. Review of the document. Final review of the document (Angeliki Alexiou).

Intelligent D-band wireless systems and networks initial designs

List of Figures

List of Tables

Executive Summary

The present deliverable focuses on the Machine Learning perspectives in D-band wireless networks as AI/ML approaches hold the promise to deliver autonomous decision-making mechanisms in an efficient manner. This is arguably one of the most important requirements of Beyond5G (B5G) networks, considering the expected demand levels for low-latency, high bandwidth, continuous and reliable connectivity. Achieving performant and adoptable AI/ML outcomes also depends to a large degree on the operator's skill, domain and data understanding at the problem solving level as well as at the level of operationalization of solution. Hence, both of these aspects have been addressed in this deliverable, where chapters 1-5 focus on problem solving and chapters 6-7 highlight the broader applicability and deployment of AI/ML models. The remainder of the document is organized as follows.

Chapter 1 focuses on the radio placement optimization using multi-agent reinforcement learning (MARL) concept. It considers that the radios in a network act as independent agents and are working together in a shared environment so as to achieve a common goal, where the maximization of the total area under radio coverage is the main aim. An extension of the contribution is included in **Chapter 2**, which addresses beamforming optimization for multiple moving user equipments (UEs) in downlink transmission. It mainly focuses on implementing deep reinforcement learning (DRL) for codebook beam and beam tilt selection at the base station (BS) to serve the UEs. It also deploys the carrier-aggregation concept into both BS-UEs and BS-RIS-UEs scenarios.

Subsequently, **Chapter 3** presents a methodology for learning automata (finite state machines) from labelled symbolic sequences, such as discretized time-series data. This work addresses a novel technique for learning automata patterns from multivariate time-series that represents the evolution of the domain in time. This approach was evaluated on publicly available ViWi (Vision-aided Wirelesscommunication)/mobility data and later, the efficacy of this approach was demonstrated and compared to state-of-the-art automata learning techniques, achieving superior performance in terms of predictive accuracy (F1-scores) and complexity/interpretability of the learned automata. Additionally, **Chapter 4** describes the system level modeling that refers to a simulated geographic area where mobile users are modeled as points coming from a uniform poisson point process (PPP). Access points (AP) are generated from another independent and uniform PPP, while obstacles in the area are modeled as rectangles. The goal is the mapping of the received power of a user from every access point as function of the time. In order to do so, the position of each user has to be identified relatively to the serving AP and the obstacles. Sample results of the received power within the simulated area, are presented at the last part of the chapter.

Moreover, **Chapter 5** introduces a model-driven deep learning (MDDL)-based channel estimation and feedback scheme for wideband high-frequency massive hybrid multiple-input multiple-output (MIMO) systems, where the angle-delay domain channels' sparsity is exploited for reducing the overhead. To further reduce the uplink channel estimation and downlink channel estimation feedback overhead, only the received pilots on part of the subcarriers are fed back to the BS, which can exploit the MMV-LAMP (multiple-measurement-vectors learned approximate message passing) network to reconstruct the spatial-frequency channel matrix. Numerical results show that the proposed MDDL-based channel estimation and feedback scheme outperforms

the state-of-the-art approaches.

Chapter 6 and **Chapter 7** constitutes the last part of the deliverable. **Chapter 6** presents applications of THz wireless systems in the beyond fifth generation era and discusses their enabling technologies and fundamental challenges that can be formulated as AI problems. These problems are related to physical, medium/multiple access control, radio resource management, network and transport layer. For each of them, the AI approaches are reported, which have been recognized as possible solutions in the technical literature, emphasizing their principles and limitations with an objective to provide an insightful discussion concerning research gaps and possible future directions. Finally, we conclude with **Chapter 7**, where strategies for the deployment of AI/ML solutions are presented, highlighting important non functional aspects such as continuous retraining and collaboration among different ML pipelines that together provide a high level service. Different emerging and enabling paradigms are presented, which play a synergetic role in the operationalization and successful adoption of AI/ML solutions within the centralized, distributed and federated learning schemes.

Introduction

Focusing on the ARIADNE use cases and system requirements defined in deliverable D1.1 and based on the conducted research and developed hybrid AI/ML framework that optimizes resource allocation problems, the present deliverable extends the work to be composed of a stack of Artificial Intelligence, Machine Learning and model-driven methods that collectively work to produce both more accurate and more tractable models. These models could target issues, such as maximization/optimization of energy efficiency in wireless networks, optimal assignment between receivers and transmitters, optimal deployment density of base stations or network optimization through reconfigurable metasurfaces. The presented work thus delivers high quality solutions to the problems of interest in ARIADNE.

In this direction, this document is organized under two parts: i) specific problem solving (chapters 1-5) and ii) broader applicability of AI/ML methods (chapters 6-7). In the first part, Chapter 1 formulates one of the central design time problems namely the radio placement optimization, which is addressed using the multi-agent reinforcement learning (MARL) method. Chapter 2 presents beamforming optimizations for real-time interaction of mobile UEs with their environment using deep reinforcement learning (DRL). To identify complex patterns that form in an evolving network, Chapter 3 focuses on event pattern learning for Complex Event Forecasting (CEF). It introduces a declarative automata learning technique, based on the problem-solving methodology of Answer Set Programming (ASP) which can be extended to network scenarios. In Chapter 4, the system level modeling and simulation of D-band networks is described. The work aims to model environments realistically such that they contain static obstacles, static access points and mobile users. The delivered models can assist in proactive handling of connections by helping avoid line of sight blockages as the receivers move over time in a given environment. The work can be reused to represent further scenarios as well. Next, Chapter 5, focuses on model-driven deep learning (MDDL) based channel estimation and feedback scheme for wideband high-frequency massive hybrid multiple-input multiple-output (MIMO) systems, where the angle-delay domain channels' sparsity is exploited for reducing the overhead. In the second part of this deliverable, Chapter 6 focuses on applications of THz wireless systems in the beyond fifth generation era and discusses their enabling technologies and fundamental challenges that can be formulated as AI problems. Finally, Chapter 7 presents some important non-functional aspects that are necessary to be considered for successful adoption and operationalization of AI/ML solutions by highlighting strategies for deployment of AI/ML units as collaborative and reconfigurable pipelines that may be deployed among central or distributed nodes of the network and can be scaled up or down to deliver required quality of service.

Deliverable D4.2 and its findings constitute the subsequent activities of Work Package 4, within the ARIADNE project. The results of deliverable D4.1 are exploited and extended in deliverable D4.2, towards realizing intelligent D-band wireless systems and the initial designs of these networks.

Chapter 1

Radio Placement Optimization Using Multi-Agent Reinforcement Learning

1.1 Introduction

As wireless telecommunication technology advances towards ever higher frequencies, signals are more severely affected by blockages and various attenuating effects in the environment. A Line of Sight (LoS) connection is practically required for a good user experience on a handset utilizing spectrum from 100 GHz upwards. This places great importance on physical network design and positioning of the network equipment. Constraints on network cell size are placed by the requirement for LoS connection as well as attenuating effects such as rain that have to be considered when dealing with millimeter wavelength (mmWave) frequencies. This in turn leads to an increase in the number of required Access Points (AP) to serve the same area and number of users as on lower frequencies.

Building on our previous work in ARIADNE WP4, we present a formulation of radio placement optimization as a Multi-Agent Reinforcement Learning (MARL) problem. Specifically, we consider the radios in a network configuration as independent agents, working together in a shared environment to achieve a common goal, i.e. to maximize the total area under radio coverage.

1.2 Methods

This section introduces the main methods used. Section 1.2.1 presents process models used for modeling single-agent reinforcement learning (RL). Section 1.2.2 describes multi-agent reinforcement learning and presents a process model suitable for those scenarios. Section 1.2.3 covers the concept of self-attention and how that is used in our neural network policy.

For general information about RL, we refer the reader to the ARIADNE deliverable D4.1.

1.2.1 Process Models for RL

In single-agent RL the environments are typically assumed to be stationary. That is, the behavior and dynamics of the environment remain unchanged over time. These kinds of environments can be modeled as Markov Decision Processes (MDP). A standard MDP is a tuple of form

$$(S, A, P, R, \gamma), \tag{1.1}$$

where $S = \{s_1, s_2, \ldots, s_N\}$ is a set of possible environment states (state space), $A = \{a_1, a_2, \ldots, a_N\}$ is a set of possible actions (action space), P is a set of conditional state transition probabilities $P(s'|s, a) : S \times A \times S \rightarrow [0, 1]$ for state transition $s \rightarrow s'$, $R(s, a) : S \times A \rightarrow R$ is the reward function and $\gamma \in [0, 1]$ is the discount factor. At each discrete time step t , the environment is in some state s. The agent takes an action a, which results in new state s' with the probability $P(s'|s, a)$ and a reward $r = R(s, a)$. At the next time step $t + 1$, s' is assigned to s and the process repeats. The goal of an agent is to choose actions that maximize its expected future discounted reward at each time step t:

$$maxE[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)] \tag{1.2}$$

The simple form of MDP is only applicable if state s is fully observable by the learning agent. As interesting real-life problems are usually not fully observable, a broader model for the scenario is often needed. In cases where the full observability requirement does not hold an extended form of MDP, Partially Observable Markov Decision Process (POMDP) can be used. It is defined as

$$(S, A, P, R, \gamma, \Omega, O), \tag{1.3}$$

where S, A, P, R and γ are equal to their counterparts in MDP, $\omega = \{o_1, o_2, \ldots, o_N\}$ is a set of observations and O is a set of conditional observation probabilities $O(s', a) : S \times A \rightarrow \Omega$. In POMDP only the observation o is available to an agent, not the full state s. The agent thus has to internally maintain a model for what it believes to be the real state, based on the observations it has seen and the actions it has taken. For an agent in an POMDP setting, the goal then becomes selecting the actions that maximize expected future discounted reward given a belief state formed by the agent. Besides the fact that agent only sees observation o, operation remains identical to MDP case.

1.2.2 Multi-Agent Reinforcement Learning

Multi-agent reinforcement learning is a subfield of reinforcement learning that focuses on systems with multiple independent learning agents acting in a shared environment. By interacting with each other and the environment the agents try to achieve some goal, defined in terms of reward signals. The goals of individual agents may be aligned, opposed, or a mixture of the two.

Each agent makes a decision on every time step based on its policy. The policy receives the state of the environment as observed by the agent. It then produces an action that's applied to the environment along with actions from all the other agents. An

individual agent's policy therefore needs to account for the behavior of other agents as well as the environment dynamics in general. The behavior of other agents changing as they learn brings with it complexities not typically present in single-agent RL problems.

We can expand the POMDP definition for multi-agent case as

$$(S, A, P, R, \gamma, \Omega, N, O), \tag{1.4}$$

where S is the state space and γ is the discount factor as defined in single-agent MDP. Each agent $i \in N := \{1, \ldots, N\}$ selects an action $a_i \in i$ at each time step, A^i being the action space for agent i. $A := A^1 \times \ldots \times A^N$ is the joint action space for all agents and the joint action vector is $\boldsymbol{a} \in A$ State transition probability function is $P(s'|s, a) : S \times A \times S \to [0, 1]$, now in terms of the joint action vector. Reward function is defined as $R(s, \boldsymbol{a}, a_i, i) : S \times A \times N \to \mathbb{R}$ for a reward with both components shared across the agents and per-agent components. Each agent receives an individual partial observation $o_N \in \Omega$ according to the observation function $O(s, i) : S \times N \to \Omega$.

With action space, reward function and observation function conditioned on i, this model supports any mixture of cooperative and opposing goals and diverse agents with different observational and operational capabilities. Reward function can emphasize the value of joint action vector to orient behavior toward group-level goals and vice versa.

1.2.3 Attention

Attention in artificial neural networks mimics the functioning of attention in animals. It is a behavioral and cognitive process of focusing selectively on a discrete aspect of information while ignoring other information. For animals and humans alike, attention is necessary because there is vastly more information available in any given environment than can be effectively processed. Specifically concentrating on information that is relevant for the task or situation at hand allows more effective utilization of cognitive faculties. [1]

Computational attention has been studied at least from the late 1970s [1], but practical applications have only started appearing in recent years. Initially shown to outperform previously used deep learning models in natural language processing [2] and image processing [3, 4], it has since been shown to also work well in RL applications.

Generally speaking, the operations performed by an attention module could be summarized as weighted averaging. It takes in two arbitrary input sequences $X, Y \in \mathbb{R}^{n \times d_{in}}$ of n elements with d_{in} dimensions in each and uses one to "attend" to the other, producing an output sequence $Z \in \mathbb{R}^{n \times d_{in}}$.

Our focus is on a special case of attention called self-attention, where $X = Y$. Self-attention allows mapping arbitrary input to an output of identical dimensions, with the mapping being determined by the learned weights. The specific method under consideration is the Scaled Dot-Product Attention described in [5], which has been widely adopted to different use cases.

Using the definition of X above, we can define the scaled dot-product attention function as

$$\widetilde{V} = \mathrm{Attn}(Q, K, V) = \mathrm{softmax}\left(\frac{(XW_q)(XW_k)}{\sqrt{d_{in}}}\right) XW_v \tag{1.5}$$

where $\widetilde{V} \in \mathbb{R}^{n \times d_{in}}$ is the weighted value matrix or attention matrix and $W_q, W_k \in \mathbb{R}^{n \times d_k}$ and $W_v \in \mathbb{R}^{n \times d_v}$ are the learned weights used to generate $query$, key and $value$ matrices, respectively. d_k is the dimension for query and key spaces and d_v is the dimension of value space. The use of terms query, key and value is conceptually similar to their usage in the context of databases. That is, the operation can be thought of as retrieving information from a set of key-value pairs using a query. Using the corresponding weights, the input sequence is linearly projected to query, key and value spaces. The attention matrix is summed to the original input, resulting in a modified version where each token in the sequence is weighted with every other token considered:

$$Z = \widetilde{V} + X \tag{1.6}$$

Multi-head attention is another concept presented in [5]. In multi-head self-attention, the attention function is evaluated h times for the same input, h representing the number of parallel attention heads. Each attention head has different learned weights for W_q, W_k and W_v. Resulting h attention matrices are concatenated along the n-dimensional axis and linearly projected to produce $\widetilde{V}$. This was found to improve the results over single-head attention. The reasoning is that multiple different subspaces for representing the query, key and value allow the model to jointly attend to different representations, giving it access to more information.

The primary use case of attention in RL is to attend over some preprocessed features before they're further processed into actions. The work in [6] introduced a setup where attention is applied to a set of feature vectors, each representing features from a single agents observation. This allows agents to influence each other's decision, making and results in improved behavior.

1.3 Placement Optimization as Multi-Agent Reinforcement Learning Problem

For a generic formulation of the radio placement optimization system we consider a 2D environment with a finite grid of square cells. There are N agents representing the base stations and Reconfigurable Intelligent Surfaces (RISs). Each agent occupies a discrete position, i.e. one cell on the map. Two agents cannot occupy the same position. On every timestep t each agent receives an observation o and selects an action $a = \pi(a, o)$. The joint action vector $\boldsymbol{a}$ is evaluated by the environment, which produces the next state s' and joint reward $\boldsymbol{r} = \{r_1, r_2, \ldots, r_N\}$ where r_i is the reward received by agent number i.

The environment also has radio receivers and blockages that prevent signals from passing. We do not model individual user terminals; instead we consider the receivers to represent clusters of users or areas of high expected user density. Each receiver occupies a single cell. Blockers are rectangular areas that block a direct line of sight between radios, spanning any number of cells. For a base station or an RIS to be able to provide coverage, it needs to be next to a blocker (i.e. mounted to a wall or on the edge of a rooftop). Agents can move on top of the blockers, but will not provide coverage while they remain there.

All agents have a set of generic movement and rotation actions available. Movement actions move the agent to one of eight neighboring cells. Rotation actions change the

orientation of the agent by a fixed amount. Both have a corresponding “do nothing” action to all other actions. Agent can select “do nothing” for both movement and rotation to indicate that it considers its current configuration to be the best possible one. When all agents pick the “do nothing” action at the same timestep, this is interpreted as a “stop episode” action and the episode will be stopped. Effectively this acts as a voting mechanism for agents to jointly decide if the configuration is good enough. To prevent the agents from quitting prematurely and to foster exploration, the stop action can only be triggered on every 25th timestep.

For each episode in training, the agents, receivers and blockers are randomly placed on an empty map. Agents take actions on each timestep as described earlier and the episode progresses until either the last step of the episode has been reached or the joint “stop episode” action is selected.

Main reward signal comes from the change in relative total coverage over the last timestep. For instance, if coverage increases from 20% to 30% the corresponding reward is 0.1. All agents receive the same coverage reward. To encourage agents to seek locations where they can provide coverage, an additional reward is given for each agent individually when it moves from an invalid position to a valid one. A penalty of equal magnitude is given when an agent moves out of a valid position to an invalid one.

1.4 Placement Optimizer Implementation

1.4.1 MARL in RLlib

MARL is behind many of the most publicized achievements of machine learning in recent times including AlphaGo Zero, AlphaStar, OpenAI Five and more. However, MARL has always been a significant engineering challenge, unlike single-agent reinforcement learning. Supporting standard APIs for MARL is not abundantly available in many environments as easy and extensible as in RLlib. PettingZoo API was first released by RLlib to demonstrate a general and easy way of using MARL. Prior to PettingZoo, the numerous uses of MARL APIs almost exclusively inherited their design from the two most prominent mathematical models of games in the MARL literature which were Partially Observable Stochastic Games (POSGs) and Extensive Form Games (EFGs). Modeling multi-agent environments has numerous benefits, it allows for clearer attribution of rewards to different origins, allowing for various learning improvements. In addition, it is easy to handle by scaling up number of agents without additional complexity involved with the implementation.

RLlib supports handling of different algorithms in a simple setting. Multiagent in RLlib consists of relevant principles such as policies which are represented as objects and are considered black boxes. All gradient-based algorithms in RLlib declare a policy graph that includes a policy model and a policy loss. The policy graph runs in a distributed framework, collects experiences, and then improves the policy with gradient updates computed from batches of experiences. RLlib just manages the creation and execution of multiple policy graphs per environment, adds together the policies using some policy optimization algorithms.

One main component in reinforcement learning is the environment where the training happens. A multi-agent environment has multiple acting entities per step. In a

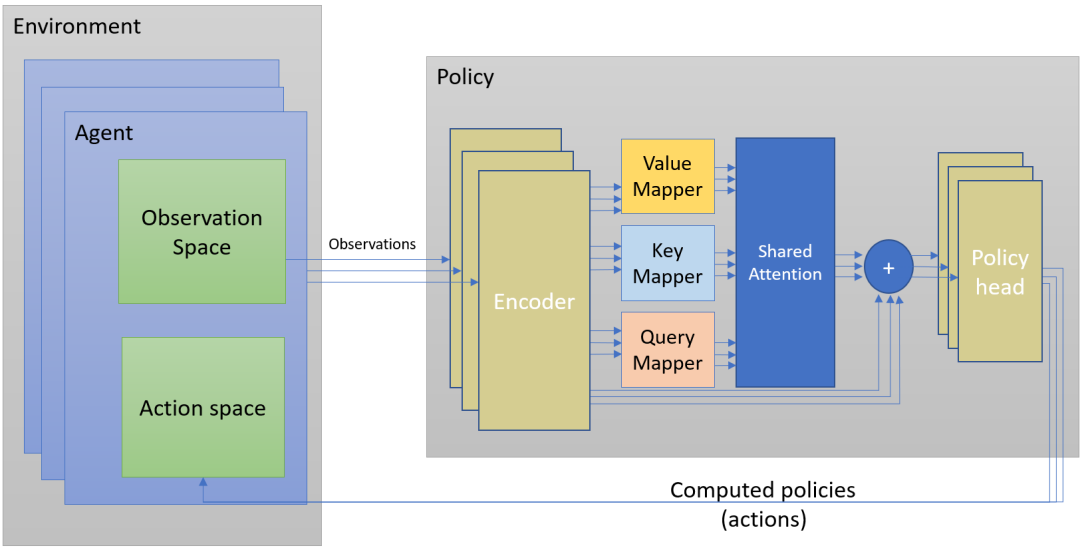


Figure 1.1: Multi-agent implementation overview for placement optimizer. LSTM state variables are also inputs to the model but omitted from picture.

multiagent environment in RLlib, an environment can have a varying number of agents over time, however, the number of distinct policies should be fixed.

1.4.2 Implementation

Figure 1.1 presents the high level structure of the system and the primary neural network blocks of the RL model.

The environment creates multiple agents and the required spaces for them as action space and observation space, those are then handled through the model under training. Observations are fed from the environment to the model as a top-down view image of the map with multiple information layers and a vector of scalar values, containing information such as whether the radio can provide coverage from its current position or not. Action space is as shown in Figure 1.2. Radios can do two types of action categories per step. Those actions are movement and rotation specific, radios can both move in a 2D-space in all directions and rotate clock and anti-clockwise. Actions are calculated by the policy, then the environment performs the action operation within the simulator class.

Simulator is the entity that translates that information to a real-world environment of training, it retains information of the map information including buildings, blockages, radios and UEs. It exposes its information externally to the environment and model level through observation space. Environment also computes rewards and penalties using simulator functions to determine the coverage values and determine whether placement of radios was optimal or not.

1.4.3 Policy

Our policy is represented by a neural network. Agents share a single policy model which has inputs and outputs for each agent, following the general setup presented in [6].

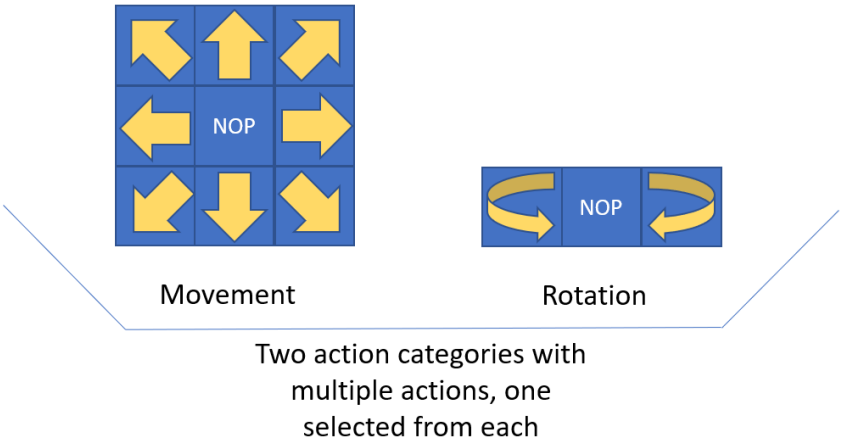


Figure 1.2: Actions allowed for agents

Agent observations are processed by encoder networks. The resulting agent-specific feature vector is fed to each of key, query and value mapper networks. Resulting vectors are combined into key, query and value matrices which go into the attention function. This produces a matrix, which is split back into vectors that are then added to the original features and used as inputs into the policy and value functions.

Feature encoders consist of CNN, dense and LSTM layers, with a CNN branch processing the map input and dense branch processing the scalar input. Results are concatenated and run through an LSTM layer to produce final features with time-domain information. Attention parameter mappers and different output heads all have a single dense layer.

1.4.4 Conclusion

This chapter presented our revised approach to automating the placement of base stations in the context of network design. It introduced the usage of multi-agent reinforcement and attention mechanism as enhancements over our base model previously presented in ARIADNE deliverable D4.1. A derivation of Markov Decision Process suited for multi-agent scenarios was also presented. Finally, it gave an overview into our implementation and the technologies used. Results were not included and we plan to provide them later as part of the next deliverables in this WP.

Chapter 2

Beamforming Optimization for Mobility Users based on Deep Reinforcement Learning

2.1 Introduction

Future wireless networks 5G and beyond 5G (B5G) are expected to meet the massive demand for data rates especially for the extended reality (XR), augmented reality (AR) and mixed reality (MR). In order to meet various requirements, the future new radio not only considers the sub-6 GHz but also takes millimeter-wave (mmWave) band and terahertz (THz) band into implementation. Downlink beamforming technique has attracted attention from both industry and academic. In most of the actual wireless communication scenarios, it is impossible to keep users (UEs) from moving. Therefore, the base stations (BSs) need to have effective mechanism to allocate proper beamforming vector to the moving UEs. This leads to our motivation to address this research direction in order to make the UEs' real time interaction with the environment easy by using deep reinforcement learning (DRL).

In general, the system capacity of wireless communications is represented by the weighted sum rate when the UEs have different priority and require different capacity. The summation of the achievable rate of all UEs can be also the overall throughput representation of the network. We consider two main scenarios for downlink transmission in this work (i) the base station serves moving UEs using mmWave and (ii) when there is blockage between the BS and UEs, we assume that the BS transmit signal to the reconfigurable intelligent surface (RIS) and then the RIS reflects/refracts signal by using analog phase shifters towards multiple UEs.

2.1.1 Contributions

- Our work is the first work which proposes a dynamic beamforming optimization with beam selection from the predefined codebook beam set and beam tilt selection using DRL. Moreover, for the BS-RIS-UEs scenario, we also use DRL to obtain the codebook beam selection, beam tilt at the BS and the angles at the RIS phase shifters simultaneously.

- Our work is a pioneer work to combine a dual-band connectivity and it can be called as carrier aggregation concept with the codebook beam and beam-tilt selection at the BS. For the BS-RIS-UEs scenario, a dual-band connectivity is firstly introduced here.
- We use Importance Weighted Actor-Learner Architecture (IMPALA) based based V-trace actor-critic algorithm for distributed DRL to implement our work. Therefore, this is the first work to use the IMPALA based distributed DRL in beamforming optimization multi-UE multi-input-single-output (MISO) concept. Our work is also the first one to implement IMPALA for BS-RIS-UEs scenario.[1]

2.2 Beamforming Optimization for Moving UEs

We consider downlink scenario where a multi-antenna base station (BS) is transmitting signal to multiple moving UEs. The UEs are randomly scattered in the service area and the UEs move in random directions. The BS serves all the UEs with maximum power all the time. We propose two main scenarios which are (i) The BS serves multiple UEs and (ii) When there is blockage between the BS to UEs, we deploy RIS to redirect signal from the BS to UEs. In both scenarios, the BS serve UEs and the BS connect to RIS using mmWave channel. The channel model is explained in the following subsection.

2.2.1 Channel Model

We consider 28 GHz mmWave channel in our proposed model for comparison with the 3.5 GHz low frequency band with Rayleigh fading channel. Assuming that $\mathbf{H}_{bk}$ is mmWave channel, where the line-of-sight (LoS) is the dominant path as the LoS is highly required as to maintain a stable mmWave link. Let L_bk be the paths for channel $\mathbf{H}_{bk}$ between BS-b and UE-k, we can write $\mathbf{H}_{bk}$ as [7],

$$\mathbf{H}_{bk} = \sqrt{\frac{N_t}{L_{bk}}}\left(g^1_{bk} a_{ULA}(\theta^1_{bk}) + \sum_{n=2}^{L_{bk}} g^n_{bk} a_{ULA}(\theta^n_{bk})\right) \tag{2.1}$$

where θ^1_{bk} and θ^n_{bk} denote the AoD for LoS and NLoS path. Note that the AoD for each non-LoS (NLoS) path n is assumed to be uniformly distributed $\theta_{bk} \in [0, 2\pi]$. The transmit array steering of ULA is $a_{ULA}(\theta) \in \mathcal{C}^{N_t \times 1}$. We can write $g^1_{bk} = v_{bk} d^{-\eta}_{bk}$, where v_{bk} is random complex gain with zero mean and unit variance, d_{bk} is a distance between the BS-b and UE-k, the pathloss exponent for LoS is η and for NLoS.

2.2.2 Scenario 1: A Multi-Antenna Base Station Serving Moving UEs

Considering the system where each BS employs a uniform linear array (ULA) of N_t antennas. Let $\mathcal{K}$ denote the set of UEs, each UE-$k \in \mathcal{K}$ has single antenna.

[1] We use the IMPALA based distributed DRL similar to our placement optimization work. More details could be found in ARIADNE deliverable D4.1.

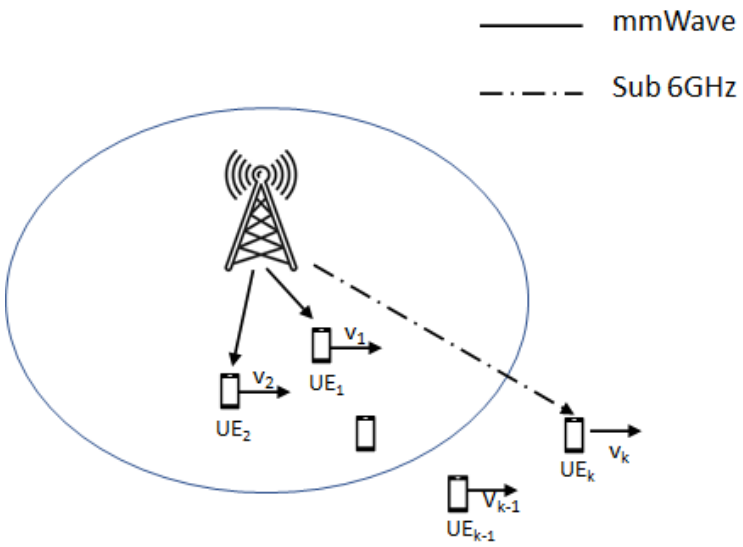


Figure 2.1: BS serves moving UEs

In Fig. 2.1, we consider a downlink transmission where a BS transmits signal to UEs using 28 GHz mmWave band. Each UE moves to random directions with the same or different velocity. We also deploy a dual-connectivity concept, i.e., when the UEs move to bad quality signal places, the BS switches the band from mmWave to 3.5 GHz to serve those UEs.

The received signal at each UE-k from the BS-b can be expressed as,

$$y_{bk} = \mathbf{H}_{bk}\mathbf{w}_{bk}x_{bk} + \sum_{k'=1,k'\neq k}^{K} \mathbf{H}_{bk'}\mathbf{w}_{bk'}x_{bk'} + n_{bk} \tag{2.2}$$

where $\mathbf{H}_{bk} \in \mathcal{C}^{1\times N_t}$ is the channel vector from the BS-b to UE-k, $\mathbf{w}_{bk} \in \mathcal{C}^{N_t\times 1}$ and x_{bk} are the beamforming vector and the transmit data symbol from the BS-b to UE-k, respectively, and $n_{bk} \sim \mathcal{CN}(0, \sigma_{bk}^2)$ is the additive Gaussian noise with variance σ_{bk}^2. The second term in (2.2) represents intracell interference.

We can write the SINR of the UE-k served as,

$$\gamma_{bk} = \frac{|\mathbf{H}_{bk}\mathbf{w}_{bk}|^2}{\sum_{k'=1,k'\neq k}^{K} |\mathbf{H}_{bk'}\mathbf{w}_{bk'}|^2 + \sigma_{bk}^2}. \tag{2.3}$$

Then, we use the SINR in (2.3) to compute the weighted sum rate of all K UEs as,

$$C(\mathbf{w}) = \sum_{k=1}^{K} \omega_k R_k, \tag{2.4}$$

where R_k is the data rate of the k-th UE, given by $R_k = \log_2(1+\gamma_k)$. ω_k represents the weight of the signal transmission to each UE. This can be applied when the UEs have different priority, i.e., some UEs need higher bandwidth and some need lower bandwidth. Note that when weight $\omega_k = 1$ the weighted sum rate equation in (2.4) becomes sum rate.

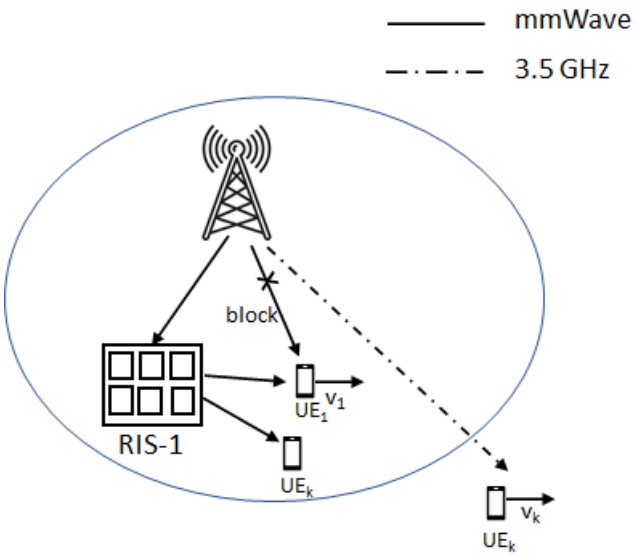


Figure 2.2: RIS serve mobility UEs

2.2.3 Scenario 2: A Single Base Station with RIS Serving Moving UEs

In Fig. 2.2, we consider a downlink transmission where there is blockage between the BS and the UEs. The BS connect to RIS using mmWave band and the RIS redirect signal by using analog phase shifters to reflect or refract signal towards the UEs. When the UEs move away from RIS coverage area to bad quality signal places, the BS switches the band from mmWave to 3.5 GHz to serve those UEs.

The received signal at the UE-k can be written as,

$$y_k = \mathbf{H}_{m,k}^H \mathbf{\Phi} \mathbf{H}_{b,m} \mathbf{w}_k \mathbf{x}_k + \sum_{k',k' \neq k}^{K} \mathbf{H}_{m,k}^H \mathbf{\Phi} \mathbf{H}_{b,m} \mathbf{w}_{k'} \mathbf{x}_{k'} + n_k \tag{2.5}$$

where $\mathbf{H}_{m,k}$ is the channel matrix between the RIS-m and UE-k, $\mathbf{H}_{b,m}$ is the channel matrix between the BS-b and RIS-m. $\mathbf{\Phi}$ is the analog phase shift arrays with the size N_{RIS}. The second term of (2.5) is co-channel interference. The SINR at the k-th UE is

$$\gamma_k = \frac{|\mathbf{H}_{m,k}^H \mathbf{\Phi} \mathbf{H}_{b,m} \mathbf{w}_k|^2}{\sum_{k',k' \neq k}^{K} |\mathbf{H}_{m,k'}^H \mathbf{\Phi} \mathbf{H}_{b,m} \mathbf{w}_{k'}|^2 + \sigma^2} \tag{2.6}$$

The weighted sum rate can be written as,

$$C(\mathbf{w}, \mathbf{\Phi}) = \sum_{k=1}^{K} \omega_k R_k. \tag{2.7}$$

In the first scenario in Section 2.2.2, our objective is to find the optimal $\mathbf{w}_k$ by solving weighted sum rate maximization problem as max $C(\mathbf{w})$ in (2.4) with respect to $\mathbf{w}_k$. In the second scenario in Section 2.2.3, we obtain the optimal $\mathbf{w}_k$ and $\mathbf{\Phi}$ by solving weighted sum rate maximization problem as max $C(\mathbf{w}, \mathbf{\Phi})$ in (2.7) with respect to $\mathbf{w}_k$ and $\mathbf{\Phi}$. We solve the weighted sum rate maximization problems of both scenarios

using deep reinforcement learning technique under the given particular CSI. Each CSI is used to construct the start and run the algorithm to obtain two matrices $\mathbf{w}$ and $\mathbf{\Phi}$ simultaneously.

Note that: in our current training model, we consider only a single BS, a single RIS and multi-UE scenario. However, our model can be extended for multi-BS multi-UE scenario, the main change of the received signal at each UE-k will need to include intercell interference from other BSs.

2.3 Deep Reinforcement Learning Method and Implementation

The MISO beam-UE pairing selection can be modeled using deep reinforcement learning. In reinforcement learning, all problems can be framed as markov decision process (MDP).

2.3.1 Bellman Equations

In this part, we discuss about Bellman equations since the Bellman equations are absolute necessary for trying to solve reinforcement learning problem. In RL, the environments are assumed to be stationary and can be framed as MDP. A fundamental property of all MDPs is that the future states depend on the current state. This is because the current state is supposed to have all the information about the past and the present. Hence, the future depends only on the current state.

The MDP involves four sets of components: state (s), actions (a), transition probabilities $P(s'|s,a)$, and reward $r(s,a)$. Solutions of MDP are policies. A policy is a strategy and a rule specifying what action to execute in every possible state, denoted as $\pi(s)$. In order to solve the MDPs, the policies need to be searched to maximize the rewards obtained by the agents [8]. The sum of rewards, from the state s, is the utility of the policy $U_\pi(s)$.

$$U_\pi(s) = \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t), \tag{2.8}$$

where $\gamma (0 < \gamma < 1)$ is a discounted factor. The expected utility following the policy π from the state s is the state value function $V_\pi(s)$ of the policy, which is not random:

$$V_\pi(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)] \tag{2.9}$$

State-action value function $Q_\pi(s,a)$, also called Q-value of the policy is the expected utility of taking action a from state s, then following policy π:

$$Q_\pi(s,a) = \sum_{s'} P(s'|s,a)[r(s,a,s') + \gamma V_\pi(s')] \tag{2.10}$$

When it is not in the end state, the value is equal to the Q-value of the policy. This yields the Bellman equation:

$$V_\pi(s) = \sum_{s'} P(s'|s,a)[r(s,a,s') + \gamma V_\pi(s')] \tag{2.11}$$

Bellman equation is a recursive equation, as shown. Therefore, to find the optimal policy, the vale iteration of policy iteration can be utilized. The value iteration is to get directly at the maximum expected utility. $V_{opt}(s)$ is assigned as the optimal value attained by any policy, and $Q_{opt}(s)$ is the optimal $Q-value$ of any policy. At the Bellman optimality equation, the optimal policy can be written as,

$$V^{t}_{\text{opt}}(s) = \text{max}_{a\in\text{Actions}} \sum_{s'} P(s'|s,a)[r(s,a,s') + \gamma V^{(t-1)}_{\text{opt}}(s')] \tag{2.12}$$

Policy iteration randomly initializes the policy π and then solves the Bellman equation to get $V_{\pi}(s)$. Then update the policy according to the greedy policy until it converges.

2.3.2 Policy Based Algorithms

By solving the Bellman equation in (2.11) to get the optimal policy in (2.12), it is called on-policy. Policy-based methods directly search for the optimal policy by maximizing the agents'expected long-term reward $V_{\pi}(s)$ in (2.9). The policy is parameterized by a function approximator $\pi(a,s)$. The policy gradient methods are used to performing gradient ascent on the objective $V_{\pi}(s)$ in (2.9).

For each gradient update, the agent needs to interact with the environment and collect trajectories. When computing the gradients for policy updates, the value function can be used together with the sampled rewards to improve the quality of the updates. The combination of policy and value functions into one RL agent is called an actor-critic architecture, where the "critic" estimates the value function and the "actor" updates the policy distribution in the direction suggested by the critic (such as with policy gradients) [9]. Both critic and actor functions are parameterized with neural networks.

Due to the lag between when actions are generated by the actors and when the learner estimates the gradient, we need to decouple actor-learner architecture. Accordingly, the V-trace which is a novel off-policy actor-critic algorithm has been introduced [10]. The goal of an off-policy RL algorithm is to use trajectories generated by some policy μ, called the behavior policy, to learn the value function $V_{\pi}(s)$ in (2.9) of another policy π. Thus in the on-policy case, V-trace reduces to the on-policy n-steps Bellman update and this property allows one to use the same algorithm for off- and on-policy data.

The chosen approach for beamforming optimization implementation is a variant of the actor-critic method based V-trace actor-critic algorithm known as IMPALA [10] [2].

2.4 Implementation of DRL for Beamforming Optimization

In deep RL, the neural network are used to approximate the agent's optimal strategy or policy. How we model this problem based on deep RL is the key for the problem formulation. Basically, we need to define the state, action and the reward function. In our problem, let the state be the UE performance metric (SINR) and UE channel

[2]We implement the beamforming optimization based Impala similar to our placement optimization work, more details about V-trace and Impala can be found in [10] and placement optimization work

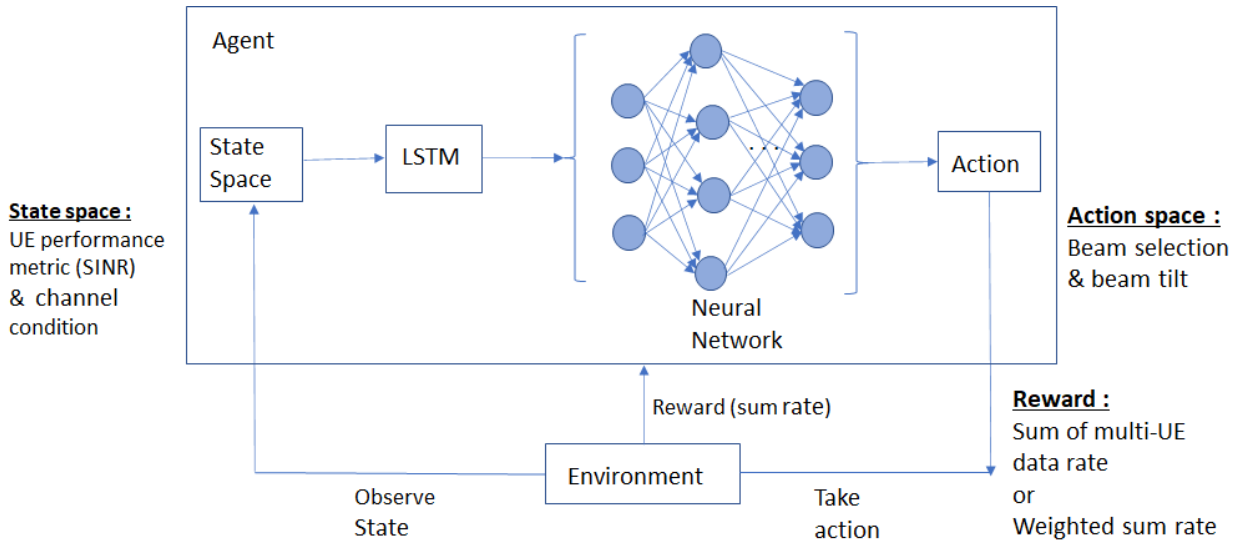


Figure 2.3: Illustration of the proposed DRL for beamforming optimization with the association of state space, environment, and action

condition and the beams/UEs have been selected so far. Given the system's current state, the neural network learns to predict either a distribution over actions or the action expected reward. The observation space or called as state space are fed from the environment to the long-short-term-memory (LSTM) layers in order to help the neural network to have memory. The BS can perform codebook selection and tilt selection per step. The tilt selection can be from -45 degree to 45 degree. Actions are selected by the policy, which is represented by neural network. Then the environment perform the action operation.

We show the proposed DRL implementation with association of state space, environment and action space in Fig. 2.3.

For the scenario 1 in Section 2.2.2, we define the actions space as beam selection and beam tilt at the BS in Fig 2.4. The rewards function can be written as weighted sum rate in (2.4) or the sum of multi-UE data rate when $\omega_k 1 = 1$ in (2.4).

For the scenario 2 in Section 2.2.3, we define the actions at the BS to be beam selection and beam tilt towards RIS. The selection of angle of the phase shifters are the actions at the RIS in Fig. 2.5

At each time transmission time interval (TTI), the BS selects the best beam to allocate to each UE from the predefined beams. In our case, we generate codebook beams as the predefined beam set at the BS. However, our implementation can be applied to any predefined beam set, i.e., discrete Fourier transform (DFT) based beamforming and etc.

2.4.1 Structure of DRL used for the implementation of the beamforming optimization algorithm

In the scenario 1 in Section 2.2.2, we use channel coefficients matrix between BS-UEs to feed into neural network. The neural network learn to predict the action expected

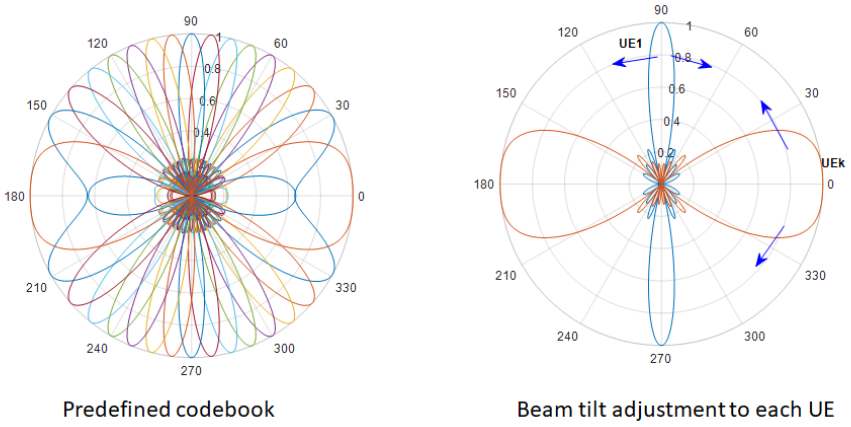


Figure 2.4: Codebook selection and beam tilt adjustment

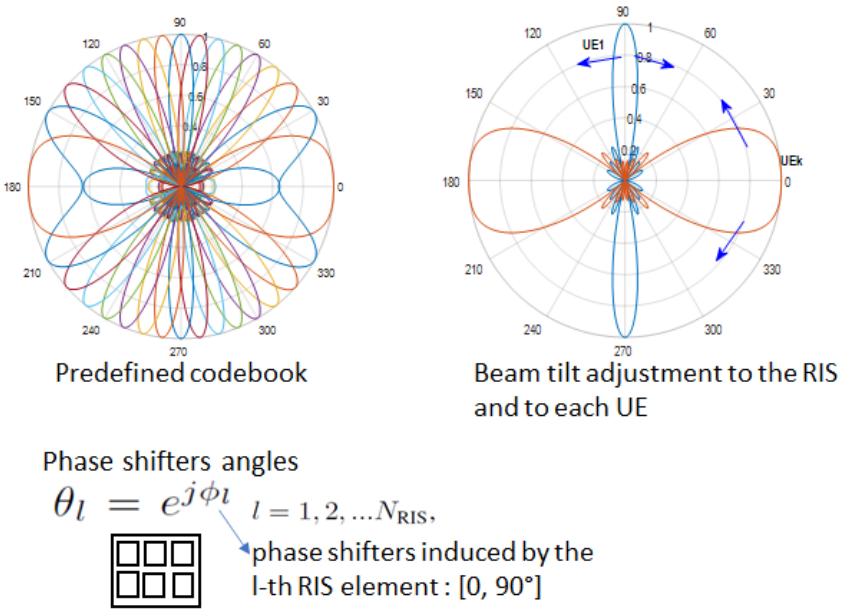


Figure 2.5: Codebook selection, beam tilt adjustment and RIS phase shifts selection

reward. Once we obtain the optimal codebook beam and beam tilt, we compute the SINR and obtain the weighted sum rate accordingly. Similarly as in the scenario 2 in Section 2.2.3, we use the channel coefficients matrix between BS-RIS and RIS-UEs to feed into neural network.

Since the BS deploys dual-band connectivity therefore when the UEs move away from BS and from RIS coverage area, the BS can switch to use 3.5 GHz instead of using mmWave channel to serve UEs with bad signal quality. Implementation Status : We are using the channel coefficients as input into neural network in both scenarios. The implementation is based on the Fig.2.3.

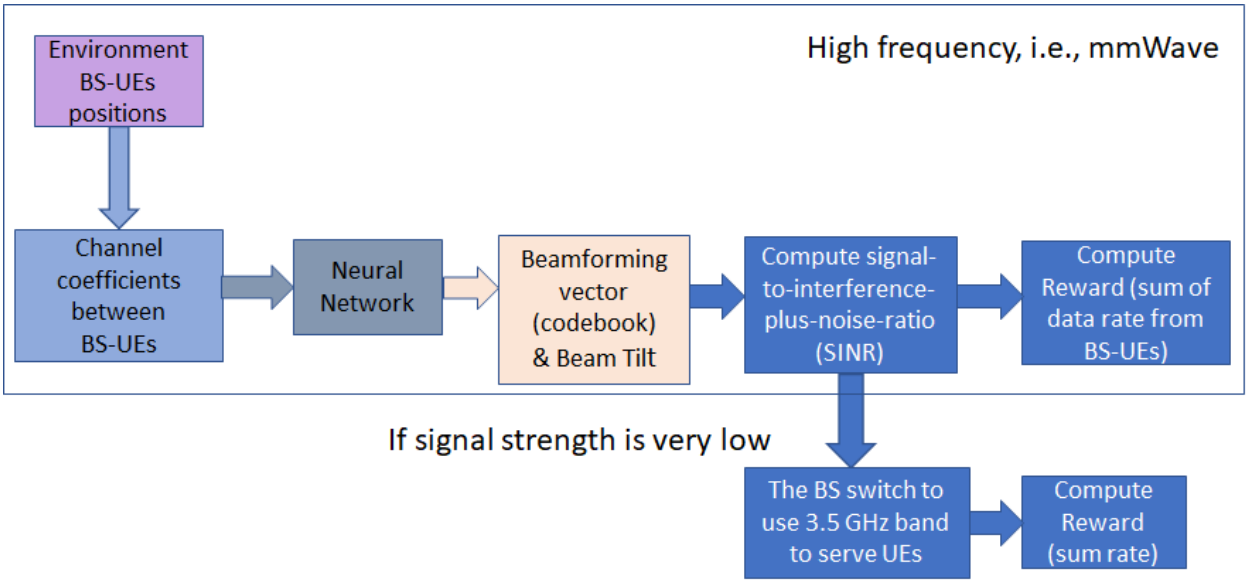


Figure 2.6: WorkFlow of the model using channel coefficients between BS-UEs as input for neural network

2.5 Conclusion

In our work, we consider two scenarios which are (i) A single BS serves multiple moving UEs and (ii) A single BS serves multiple moving UEs through RIS, where there is blockage between the BS and UEs. In the first scenario, our work is the first work that implements DRL to select the best beams from the predefined beam set and and select the beam-tilt angles to allocate to multiple moving UEs. In the second scenario, our work is the first work that implements the DRL to select the best beams and beam-tilt angles at the BS and select the phase shifters angles at the RIS simultaneously.

Furthermore, our work is a pioneer work to propose carrier aggregation concept in these two scenarios. In the first scenario, we implement carrier aggregation concept such that the BS can serve moving UEs using mmWave, when the UEs move to places with bad signal quality. The BS can switch the band from mmWave to 3.5 GHz to serve those UEs. In the second scenario, we propose that the BS serves moving UEs through RIS using mmWave band, when the UEs move away from the RIS and move to places with bad signal quality. The BS can switch the band from mmWave to 3.5 GHz to serve those UEs. Remark: we are working on training the model, we will include the results in next deliverable under the same work package.

Chapter 3

Machine Learning Complex Event Forecasting Patterns

3.1 Introduction

In this chapter we present symbolic learning techniques for the automatic extraction of automata-based complex event patterns from data. An automaton learnt by the proposed techniques represents a sequential pattern of *simple events* over time, i.e. time-stamped pieces of information that represent occurrences within the modeled domain. This pattern corresponds to a *complex event* of interest, i.e. a temporal combination of simple events, that needs to be monitored/traced in a given domain. Using dedicated *complex event forecasting (CEF)* techniques, the learnt automaton may be used for forecasting the complex event occurrence ahead of time, from early signs observed in the input data stream. Forecast complex events may subsequently be used for timely drawing insights from the data and implementing proactive measures.

For instance, in ARIADNE CEF techniques are used as a means towards timely detecting an imminent blockage event before it actually occurs. This information may subsequently be used for re-allocating the user that is about to be blocked to a different access point, thus helping to minimize disruption of service incidents.

The presented work is a first step towards the end-goal of event pattern learning for CEF, where we introduce a declarative automata learning technique, based on the problem-solving methodology of *Answer Set Programming (ASP)* [11], which allows to delegate the automata learning process to highly efficient and optimized off-the-shelf solvers. We compare our approach to a number of state-of-the-art automata learning techniques on telecommunications data with the goal being that of learning for blockage prediction and we demonstrate empirically the superiority of our technique in terms of predictive accuracy, simplicity and comprehensibility of the learnt automata.

The rest of this chapter is structured as follows: In Section 3.2 we present the necessary background and discuss related work on CEF and machine learning for CEF. In Section 3.3 we present the proposed automata learning approach, while in Section 3.4 we present a preliminary experimental evaluation. Finally, in Section 3.5 we discuss current and future work and conclude.

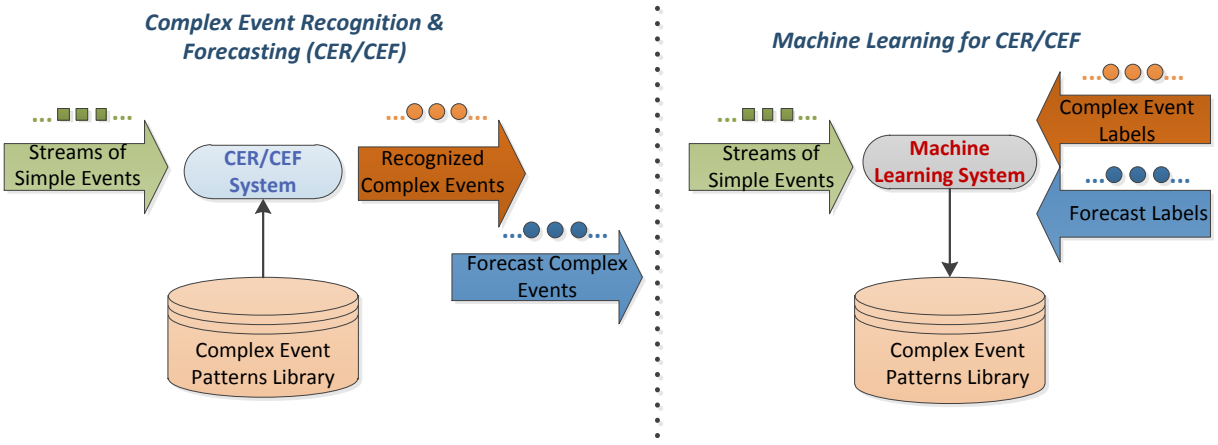


Figure 3.1: An illustration of the complex event recognition and forecasting tasks (left) and high-level view of the related machine learning task that we address.

3.2 Background and Related Work

In this section we present the necessary background material on complex event detection and forecasting, machine learning and automata learning techniques. We also include an extensive presentation of related work on these issues to highlight the differences of the techniques developed in ARIADNE with existing methods.

3.2.1 Complex Event Forecasting

Figure 3.1 presents the main tasks that motivate the presented work. The left-hand side of the figure presents an overview of the complex event recognition and forecasting (CER/F) tasks [12–14], which are essentially concerned with continuously matching patterns over streams of incoming event-based data. The input to a CER/F system consists of (heterogenous and possibly geographically distributed) streams of so-called simple events (SEs), which refer to any kind of time-stamped information that represents an occurrence within the monitored domain. On the output side, the system recognizes/detects, or even forecasts occurrences of so-called complex events (CEs), which refer to interesting/critical situations that need to be traced, or forecast ahead of time. Such situations are defined via a library of CE patterns, which specify spatio-temporal dependencies and constraints between the input SEs, potentially in addition to other contextual domain knowledge. On many occasions, such patterns are assumed to be known beforehand, e.g. provided by domain experts. The recognition task then amounts to efficiently matching such patterns against the streaming input, and the forecasting task to estimating the likelihood of the completion of such patterns in the future (a full match), based on early observations (partial pattern matches), while taking into account estimated statistical properties of the input streams.

There are several learning-related challenges involved in the CEF task. First, if the CE patterns are known beforehand, statistical learning techniques (typically unsupervised) are required, in order to learn how to generate useful forecasts with these patterns, i.e. identify the sub-patterns whose matchings over the incoming data may be used as reliable forecasters of a full pattern match. Existing CEF techniques [14, 15] rely on such estimates of sub-patterns' statistical support in the incoming data, in order

to perform forecasting with hand-crafted patterns.

A significantly more challenging ML-related task is learning the actual patterns that are to be forecast, which are often unknown. The right-hand side of Fig. 3.1 illustrates the case. Streams of SEs, along with labelled data corresponding to CE occurrences, or to the validity of forecasts thereof, are passed as input to an ML algorithm, whose task is to construct a library of accurate and interpretable CE patterns from the training data. This is essentially a structure learning task, variants of which have been explored in the literature, to some extent, although none of the existing approaches is appropriate for the CEF task. Existing approaches learn patterns in the form of shapelet combinations extracted from time series data, rule-like structures in various event specification languages [16–19], or logical theories in temporal formalisms [20–23]. All these approaches share a key limitation that make them inappropriate for the CEF task. First, although the learnt rule-like models may be used for detecting event occurrences by matching the premise of a rule in the input data, using such models for forecasting CEs ahead of time is less straightforward, since rules point to concurrency, i.e conditions, expressed in the body of a rule, that occur simultaneously. Rules that express some sort of sequential conditions need to be converted into some form of automata in practice, to be operational for CEF, otherwise, such a rule fires only after the sequential condition is matched in the data (which is useless for forecasting). Second, with the exception of the shapelet extraction technique, which, however, is limited to time series data and cannot reason with existing domain knowledge, all other approaches make the assumption that a known "dictionary" of SEs is available, and moreover, that the input to the learner consists of such SE occurrences, rather than the actual raw sensory data. Therefore, these techniques assume that some third-party ML tools are in place, which process the typically unstructured input and extract the (known) SEs of value.

3.2.2 Related Work

In striking contrast to CER (its a posteriori recognition/detection counterpart), the CEF task has not received significant attention in the literature, despite the fact that "forecasting" – in general – has attracted considerable attention in various related research areas, such as time-series forecasting [24], sequence prediction [25–28], temporal mining [29–32] and process mining [33]. However, the CEF task differs significantly from all the aforementioned tasks and the need for robust CEF techniques has been acknowledged in the literature, as evidenced by several conceptual proposals [34–36]. We next briefly discuss the limitations of the aforementioned families of approaches with respect to the CEF task.

Time-series forecasting is an area with some similarities to CEF and a significant history of contributions [24]. However, it is not possible to directly apply techniques from time-series forecasting to CEF for the following reasons: (i) Time-series forecasting typically focuses on sequences of (mostly) real-valued variables and the goal is to forecast relatively simple patterns. On the contrary, in CEF we are also interested in categorical values, related through complex patterns and involving multiple variables; (ii) Crucially, techniques for time-series forecasting are unable to incorporate domain knowledge into the prediction task. Such knowledge, however, is omnipresent in most event-based applications, and its exclusion from AI-based solutions is not a real option; (iii) time-series forecasting methods do not provide a language with which we

can define complex patterns, but simply try to forecast the next value(s) from the input stream/series. In CER, the equivalent task would be to forecast the next input event(s) (SEs). This task in itself is not very useful for CER though, since the majority of CE instances should be ignored and do not contribute to the detection of CEs. CEs are more like "anomalie" and their number is typically orders of magnitude lower than the number of SEs. One could possibly try to leverage techniques from SE forecasting to perform CE forecasting. At every timepoint, we could try to estimate the most probable sequence of future SEs, then perform recognition on this future stream of SEs and check whether any future CEs are detected. It has been experimentally observed [14] that such an approach yields sub-optimal results. It almost always fails to detect any future CEs. This behavior is due to the fact that CEs are rare. As a result, projecting the input stream into the future creates a "path" with high probability but fails to include the rare "paths" that lead to a CE detection.

Another related field is that of prediction of discrete sequences over finite alphabets and is closely related to the field of compression, as any compression algorithm can be used for prediction and vice versa [25, 27]. The main limitation of these methods is that they also do not provide a language for patterns, cannot utilize domain knowledge and focus exclusively on next symbol prediction, i.e., they try to forecast the next symbol(s) in a stream/string of discrete symbols. As already discussed, this is a serious limitation for CER. An additional limitation is that they work on single-variable discrete sequences of symbols, whereas CER systems consume streams of events, i.e., streams of tuples with multiple variables, both numerical and categorical.

Forecasting methods have also appeared in the field of temporal pattern mining [29–32]. From the perspective of CER, the disadvantage of these methods is that they usually target simple patterns, defined either as strict sequences or as sets of input events. Moreover, the input stream is composed of symbols from a finite alphabet, as is the case with the compression methods mentioned above.

Lately, a significant body of work has focused on event sequence prediction and point-of-interest recommendations through the use of neural networks (see, for example, [37, 38]). These methods are powerful in predicting the next input event(s) in a sequence of events, but they suffer from limitations already mentioned above, i.e. they do not provide a language for defining complex patterns, their focus is thus on SE forecasting and they cannot utilize domain knowledge. Besides, evidence from related fields, such as time series forecasting, indicates that statistical methods have often been proven to be more accurate and less computationally demanding than deep ML approaches [39].

An additional, crucial limitation of deep learning methods is the opaque, black-box nature of the learnt models and their brittleness, as demonstrated by extreme sensitivity to hyper-parameters and the fact that deep networks are easily "fooled" by irrelevant data characteristics. These limitations make deep models inappropriate for mission-critical tasks, which are typically involved in CER/F applications. Resorting to explainable AI (XAI) techniques as a workaround is often not sufficient, since explanations generated by XAI techniques, which, themselves typically rely on ML methods, are often inaccurate, non-representative of the model's inner workings, or plain wrong [40, 41], which generates issues of trust in the explanations and makes XAI a less reliable tool for trustworthy explainability, pointing to interpretable ML techniques (i.e. relying on ML methods that learn interpretable models) as a viable alternative. Additionally, explanations delivered by XAI techniques are typically convoluted and over-

focused to low-level details and models' internals, and omissive of high-level domain-specific aspects [42]. This makes them potentially useful to trained practitioners (e.g. data scientists), but less so to non-experts in ML and data science, such as domain analysts. Additionally, deep learning techniques are data-hungry, requiring vast amounts of training data. In contrast, CE of interest are typically rare, or infrequently observed in the data, which makes deep learning techniques inappropriate for learning such patterns, and instead calls for techniques (such as those found in the symbolic learning realm), capable of generalizing adequately from extremely scarce, by highly informative training data.

Compared to the previous categories for forecasting, the field of process mining is more closely related to CER [43]. An important difference between CER and process mining is that processes are usually given directly as transition systems, whereas CER patterns are defined in a declarative manner. The transition systems defining processes are usually composed of long sequences of events. On the other hand, CER patterns are shorter, may involve Kleene-star, iteration operators (usually not present in processes) and may even be instantaneous. A CEF system cannot always rely on the memory implicitly encoded in a transition system and has to be able to learn the sequences of events that lead to a (possibly instantaneous) CE. Another important difference is that process prediction focuses on traces, which are complete, full matches, whereas CER focuses on continuously evolving streams which may contain many irrelevant events. A learning method has to take into account the presence of these irrelevant events.

3.2.3 The Wayeb Complex Event Forecasting Engine

Wayeb is a CEF engine that relies on symbolic automata to forecast future occurrences of complex events. It is based on a highly expressive formal language for defining such patterns in the form of symbolic regular expressions, which are converted to automata at run-time. This language is based on widely accepted formal computational models for CER and stream processing [13, 44, 45] and thanks to that language Wayeb possesses a clear (both declarative and operational), compositional semantics.

The main idea behind symbolic automata is that each transition, instead of being labeled with a symbol from an alphabet, is equipped with a unary formula from an effective Boolean algebra [45]. A symbolic automaton can then read strings of elements and, upon reading an element while in a given state, can apply the predicates of this state's outgoing transitions to that element. The transitions whose predicates evaluate to `true` are said to be "enabled" and the automaton moves to their target states.

Given a symbolic automaton representing a complex event pattern, Wayeb uses this automaton to learn a probabilistic model, typically a Markov chain, that encodes dependencies among the events in an input stream, in terms of statistical correlations between event occurrences. The probabilistic model is learned from a portion (e.g. initial segment) of the input stream, which serves as a training set, and it is then used to derive forecasts about the expected occurrence of the CE encoded by the automaton. By means of this probabilistic model, Wayeb is able to output forecasts on the occurrence of a target complex event in the future (full match of the pattern represented by the automaton), based on "early signs" observed in the streaming input (partial matches of the pattern, with a high statistical correlation to a full match, i.e. once such

partial matches are observed, a full match follows with a high probability). In addition to the forecast itself, Wayeb also outputs the probability of the future occurrence and a time interval indicating how "far" into the future the complex event occurrence is expected.

Learning the probabilistic model of SE dependencies from the input data correspond to a "parameter learning" task for the automaton at hand. Significantly more challenging is the "structure learning" task of learning the automaton itself. This is crucial, since in most event-based applications the interesting CE patterns are not known, or they are frequently subject to change as the input streaming data evolve over time. Wayeb operates on hand-crafted patterns, provided by domain experts, which on most occasions are not specified, or, they are only crude approximations of critical situations that are to be forecast.

The issue that we address in what follows is the structure learning task of automatically extracting such symbolic automata-based patterns from data, in order to use them with Wayeb for complex event forecasting.

3.2.4 Automata Learning Techniques

Automata learning, also known as *automata induction* is an active research field with a long history [46, 47]. The learning setting in most existing approaches is similar: A sample of labeled (positive/negative) sequences is presented to the learner, which generates an automaton that accepts the positive and rejects the negative ones. Learning the minimal, most-compressive automaton (i.e. one with a minimal number of states and transitions) is a hard computational task and a large family of approaches trade optimality for efficiency, aiming to learn a larger automaton that correctly accounts for the input sequences. The most well-known and frequently used technique in this family of algorithms is called *state-merging* and it is adopted by state-of-the-art and widely used in practice automata induction algorithms such as RPNI (Regular Positive/Negative Inference) and EDSM (Evidence-Driven State-Merging) – see [47] for a review of these algorithms and their many variations over the years.

State-merging algorithms attempt to learn an automaton that is consistent with the training set (i.e. correctly accepts/rejects positive/negative sequences therein) and is hopefully compressive enough to generalize well. Such algorithms typically start from the *Prefix Tree Acceptor* (PTA) of the training set, a tree whose branches are all prefixes of the positive training sequences (i.e. eventually all positive sequences). State-merging algorithms work by "folding" a PTA into a smaller graph by merging states that represent the same mappings from suffixes to labels, using the negative sequences as a constraint set. An automaton that results from this process is guaranteed to accept all positive sequences and reject all negative ones. From an efficiency perspective, state-merging algorithms are effective, since the PTA may be constructed in time polynomial in the size of the training set and the actual state-merging process is usually realized via scalable, greedy heuristic techniques. On the other hand, in practice such techniques often result in large, over-fitted automata that generalize poorly and are hard to inspect and interpret. An additional reason for this is that almost all existing automata learning techniques, including the techniques of the state-merging family, aim for automata that are consistent with the training set, i.e. they accept all positive sequences and reject all negative ones. This implies that even small amounts of noise in the training data result

in a disproportional increase in the learnt automata size, which in turn affects the ability to generalize and results in over-fitted models. In contrast to such heuristic techniques, our proposed method is guaranteed to learn optimal automata, by jointly optimizing a training objective that takes into account the training error and the size of the learnt model. It thus learns much more compressive automata that have better performance on unseen data, as compared to state-merging techniques. Moreover, while the latter can only induce deterministic automata, our approach can also learn non-deterministic ones. This is important, since non-deterministic automata are simpler and thus easier to learn, while they may be easily converted into equivalent deterministic automata by standard algorithms.

Another family of automata induction techniques that are closely related to the approach that we present here are satisfiability (SAT)-based techniques. Such algorithms translate the automata induction problem into a set of boolean formulae, whose models (assignments of truth values to the variables of the formulae) may be translated into automata structures that are consistent with the training set. Our approach is similar in spirit: We encode the automata induction problem as a logic program whose models correspond to solutions of the problem (i.e. learnt automata). However, there are a number of important differences between SAT-based approaches and ours: First, our approach is significantly simpler and more intuitive to implement. Second, it naturally supports reasoning, which can be directly incorporated into the automata induction process. This is important, since it allows to take into account existing domain knowledge, draw inferences and discover relations that are not explicitly present in the input sequences. In turn, the latter allows for learning *symbolic automata*, whose transitions are labeled with logical predicates rather that mere symbols, which may significantly simplify the learning process and compress the learnt model. Finally, similar to state-merging algorithms SAT-based techniques for automata induction aim to correctly account for the entirety of the training data, which may result to over-fitting.

Recently, automata structures are being used in deep reinforcement learning applications to represent policies [48, 49], which are typically learnt from sequences, using symbolic automata learning techniques. These policies are used by the deep learning system as are (instead of having the system learn the policy on its own). This line of work has demonstrated significant reduction in convergence time, as compared to blind exploration. Another recently proposed application is using automata learning as a means towards interpretable early time series classification [50], as opposed to mainstream deep learning-based techniques (based on e.g. LSTMs), which generate black-box models. The learning techniques in all the aforementioned applications have several limitations, including the inability to learn symbolic automata, their limitation to the uni-variate case (i.e. training examples in these techniques consist of a single sequence) and their inability to incorporate reasoning with domain knolwedge in the automata learning process.

3.3 Learning Automata-Based Forecasting Patterns in Answer Set Programming

Answer Set Programming (ASP). In what follows a *rule* r is an expression of the form $\alpha \leftarrow \delta_1, \ldots, \delta_n$, where α is an *atom*, called the *head* of r, (den. $head(r)$) $\delta_i's$ are

literals (possibly negated atoms), which collectively form the *body* of r (den. $body(r)$) and commas in the bodies of rules denote conjunction. We denote the set of positive (non-negated) and negative (negated) literals in $body(r)$ by $body^+(r)$ and $body^-(r)$ respectively.

A rule is *ground* if it contains no variables and a grounding of a rule r is called an instance of r. A (Herbrand) *interpretation* is a collection of true ground facts. An interpretation I satisfies an atom α if and only if $\alpha \in I$. I satisfies a ground rule if and only if satisfying each literal in the body implies that the head atom is also satisfied and it satisfies a non-ground rule r if it satisfies all ground instances of r. A *constraint* in an expression of the form $\leftarrow \delta_1, \ldots, \delta_n$, which is syntactic sugar for $\texttt{false} \leftarrow \delta_1, \ldots, \delta_n$. An interpretation I is a model of a logic program Π (collection of rules) if it satisfies every rule in Π and it is a minimal model if no strict subset of I has this property. An interpretation I is an *answer set* of Π if and only if it is a minimal model of the reduct of Π, i.e. the negation-free, ground program that results by removing from the ground version of Π all rules with a negated body literal not satisfied by I and removing all negated literals from the bodies of the remaining rules.

A *choice rule* is an expression of the form $\{\alpha\} \leftarrow \delta_1, \ldots, \delta_n$ with the intuitive meaning that whenever the body $\delta_1, \ldots, \delta_n$ is satisfied by an answer set I of a program that includes the choice rule, instances of the head α are arbitrarily included in I (satisfied) as well. A weak constraint is an expression of the form $: \sim \delta_1, \ldots, \delta_n.[w]$, where δ_i's are literals and w is an integer. The intuitive meaning of a weak constraint c is that the satisfaction of the conjunction $\delta_1, \ldots, \delta_n$ by an answer set I of a program that includes c incurs a cost of w for I. Inclusion of weak constraints in a program triggers an optimization process that yields answer sets of minimum cost.

Answer Set Programming (ASP) [11] is methodology for solving combinatorial optimization problems using the constructs defined above (i.e. facts, rules, logic programs, (weak) constraints and choice rules). This methodology is fully declarative and it summarized as follows: Given a combinatorial optimization problem P, we first encode it into a logic program Π in a way that answer sets (models) of this Π correspond to solutions to the original problem P. We then use dedicated, off-the-self tools (answer set solvers) to compute the answer sets of Π from within which solutions to P may be extracted. The ASP problem-solving methodology is based on a generate-and-test approach. In particular, choice rules are responsible for the "generate" part, by nondeterministically proposing alternative models that could represent viable solutions to the original problem at hand. Constraints and weak constraints are responsible for the "test" part, by discarding solutions that violate these constraints. Weak constraints, in particular, allow for optimization: In contrast to hard constraints, which need to be satisfied at all times, weak constraints may be violated at a price, namely the weight w associated to the constraint. A cumulative penalty score is associated to an answer set that violates a weak constraint. This penalty is proportional to the violated constraint's weight (multiplied by the number of times that the constraint is violated in the answer set). The inclusion of weak constraints in a program triggers an optimization process, which yields an answer set of minimum cumulative penalty (i.e. an answer set that adheres to the weak constraints in the original program as much as possible).

Our automata learning technique that we present in this section is based on ASP. In particular, we model the automata learning process as an abductive reasoning task with regard to a meta-interpreter, i.e. a simple logic program that captures the functionality of an automaton. Using the training sequences (the input simulations) as obser-

vations that need to be explained, we utilize ASP's generate-and-test problem-solving strategy to learn compressive automata that discriminate between blockage and non-blockage incidents by accepting/rejecting the latter.

The input (and hence, training data) that we assume in what follows consists of multivariate time-series data, representing the evolution through time of different streams of interest, e.g. consider two streams related to a user's (x, y)-coordinates over time, and a third one related to the strength of the received signal in each point in time. Labeling such multivariate sequences as ones that eventually "lead/do not lead" to a blockage incident (positive/negative examples respectively), the goal is to learn a symbolic automaton the discriminates between the two as much as possible.

An automaton (finite state machine) is a graph-based structure M described as a tuple $M = \langle Q, \Sigma, q_0, \delta, F \rangle$, where Q is a finite set of states, $q_0 \in Q$ is the *start state*, Σ is a finite set of symbols, $\delta : Q \times \Sigma \to Q$ is a state *transition function* and $F \subseteq Q$ is a set of *final states*. Given an input sequence $x_1, \ldots, x_n$, where $x_i \in \Sigma$, an automaton M transitions through the sequence of states $s_0, \ldots, s_n$, where $s_i \in Q$, $s_0 = q_0$ and $s_i = \delta(s_{i-1}, x_i)$. An automaton M is *deterministic* if for each state $s \in Q$ and each symbol $x \in \Sigma$ there is at most one transition $\delta(s, x)$ and it is called *complete* if there is exactly one transition. On the other hand, M is *non-deterministic* if there may be multiple transitions from some state with the same symbol. A deterministic automaton M accepts an input sequence if and only if the last state s_n in the state sequence , is a final state (i.e. $s_n \in F$) and it rejects the sequence otherwise. A non-deterministic automaton accepts the sequence if some transition sequence results in a final state and it rejects the sequence if all transition sequences result in a non-final state.

Given a set of training sequences the goal of automata learning task that we address is to learn an automaton whose size does not exceed a given threshold and which makes as few mistakes as possible on the training set. Definition 1 provides a more detailed account of our learning setting.

Definition 1 (**Learning Setting**). *Let S^+ and S^- be two sets of sequences consisting of symbols from a fixed alphabet Σ. S^+ and S^- jointly represent a training set of positive and negative examples respectively. Let q_0 be a fixed state (which denotes a start state) and N a positive integer that represents a "state budget". A learner is requested to learn a transition function $\delta : Q \times \Sigma \to Q$, where $Q \ni q_0$ is a set of states, such that the corresponding automaton $M = \langle Q, \Sigma, q_0, \delta, F \rangle$ has the following properties:*

1. *$|Q| \leq N$.*
2. *M minimizes the training error, defined as the sum of false positive (FP) and false negative (FN) mistakes that M makes on the training set. An FP mistake corresponds to a negative sequence $s \in S^-$, which is accepted by M, while an FN mistake corresponds to a positive sequence $s \in S^+$, which is rejected by M.*

Formally, denoting by $\mathcal{M}_N(\Sigma)$ the class of automata over alphabet Σ with at most N states, the training objective pursued by our automata learning framework is:

$$\underset{M \in \mathcal{M}_N(\Sigma)}{\operatorname{argmin}} \left(\sum_{s \in S^+} \mathsf{rejects}(M, s) + \sum_{s \in S^-} \mathsf{accepts}(M, s) \right).$$

3.3.1 Representing Input and Output

To implement the learning setting of Definition 1 in ASP we use a logic-based representation of the input (the sequences in the training set) and the output (the learnt automata).

Regarding the input, we assume that each training sequence has a unique id and a label (1 for positive and 0 for negative sequences in S^+, S^- respectively). We encode a training sequence as a set of ground facts of the form $\texttt{seq}(seqId, x, t)$, where $seqId$ is the id of the sequence and x is the symbol at position t in the sequence. We also use a predicate $\texttt{label}/2$ to represent the label of the sequence.

Example 1. Logical Representation of the Input Sequences Assume that `abbc` is a positive sequence with id id_1 and `aaa` is a negative sequence with id id_2, over alphabet $\Sigma = \{a, b, c\}$. According to the above, the logic-based representation of these sequences will be respectively:

$$\texttt{seq}(id_1, a, 1).\ \texttt{seq}(id_1, b, 2).\ \texttt{seq}(id_1, b, 3).\ \texttt{seq}(id_1, c, 4).\ \texttt{label}(id_1, 1).$$
$$\texttt{seq}(id_2, a, 1).\ \texttt{seq}(id_2, a, 2).\ \texttt{seq}(id_2, a, 3).\ \texttt{label}(id_2, 0).$$

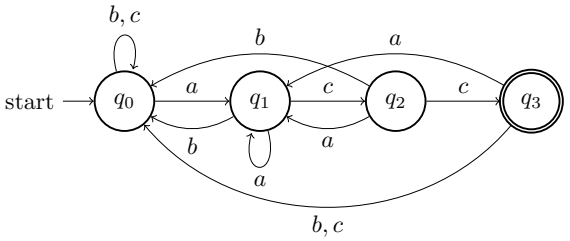
In the multivariate case a training example may consist of multiple sequences, each representing a "signal" obtained by the evolution of a relevant domain attribute in time. We can represent such multivariate examples using essentially the same representation as before, by simply extending the alphabet to account for the different features that correspond to the sequences in a training example. For instance, assume three sequences as per the example given earlier, corresponding to a user's (x, y)-coordinates and the signal strength over time.

Example 2. Logical Representation of Multivariate Input Assume that a training example with id id_3 is labeled as positive (i.e. it represents a blockage incident), it "goes on" for three time steps and consists of the following length-three sequences for *x_coordinate, y_coordinate* and *signal strength* respectively:
$x\_coord:\ geb$
$y\_coord:\ cgj$
$strength:\ abb$

Note that the symbols in these example sequences correspond to value intervals for the corresponding measurements, which may be obtained by discretizing the sequences. The logical representation of this simulation is:

$$\texttt{seq}(id_3, x\_coord(g), 1).\ \texttt{seq}(id_3, x\_coord(e), 2).\ \texttt{seq}(id_3, x\_coord(b), 3).$$
$$\texttt{seq}(id_3, y\_coord(c), 1).\ \texttt{seq}(id_3, y\_coord(g), 2).\ \texttt{seq}(id_2, y\_coord(j), 3).$$
$$\texttt{seq}(id_3, strength(a), 1).\ \texttt{seq}(id_3, strength(b), 2).\ \texttt{seq}(id_3, strength(b), 3).$$
$$\texttt{label}(id_3, 1).$$

Regarding the output of Definition 1, we represent an automaton as a set of ground facts of the form $\texttt{transition}(q_1, x, q_2)$, where q_1, q_2 are states and x is a symbol from



$\texttt{transition}(q_0, b, q_0).\ \texttt{transition}(q_0, c, q_0).\ \texttt{transition}(q_0, a, q_1).$
$\texttt{transition}(q_1, a, q_1).\ \texttt{transition}(q_1, b, q_0).\ \texttt{transition}(q_1, c, q_2).$
$\texttt{transition}(q_2, b, q_0).\ \texttt{transition}(q_2, a, q_1).\ \texttt{transition}(q_2, c, q_3).$
$\texttt{transition}(q_3, b, q_0).\ \texttt{transition}(q_3, c, q_0).\ \texttt{transition}(q_3, a, q_1).$
$\texttt{start}(q_0).\ \texttt{final}(q_3).$

Figure 3.2: A four-state deterministic automaton over the alphabet $\Sigma = \{a, b, c\}$ and its logical representation. A double circle denotes a final state and commas in transition labels denote disjunction. Therefore, the automaton in the figure moves e.g. from state q_3 to state q_0 upon encountering either a "b", or a "c".

$$\texttt{inState}(\texttt{Id}, q_0, 0) \leftarrow \texttt{seq}(\texttt{Id}, \_, \_).$$

$$\texttt{inState}(\texttt{Id}, \texttt{S}_2, \texttt{T}+1) \leftarrow \texttt{inState}(\texttt{Id}, \texttt{S}_1, \texttt{T}), \texttt{transition}(\texttt{S}_1, \texttt{X}, \texttt{S}_2), \texttt{seq}(\texttt{Id}, \texttt{X}, \texttt{T}).$$

$$\texttt{accepted}(\texttt{Id}) \leftarrow \texttt{inState}(\texttt{Id}, \texttt{S}, \texttt{T}), \texttt{end}(\texttt{Id}, \texttt{T}), \texttt{final}(\texttt{S}).$$

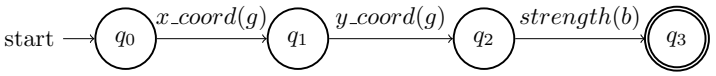
Table 3.1: An automata interpreter.

the alphabet. The interpretation of this representation is that the automaton moves from state q_1 to state q_2 upon encountering the symbol x. We also use the predicates $\texttt{start}/1$ and $\texttt{final}/1$ to represent start and final states.

Example 3. Logical Representation of Automata Figure 3.2 illustrates an example automaton and its logical representation using the $\texttt{transition}/3$, $\texttt{start}/1$ and $\texttt{final}/1$ predicates.

3.3.2 Abductive Automata Learning

The main idea towards realizing the learning setting of Definition 1 in ASP is to use *abduction* with respect to an automata interpreter. The latter is a logic program that describes the functionality of an automaton. Abduction is a form of explanatory reasoning that seeks to generate explanations for a set of observations, subject to a number of constraints, which restrict the space of possible explanations. In logic programming (and hence in ASP) such explanations are derived from a given logic program that represents some background knowledge and are given in terms of a set of predefined domain predicates (referenced in the background knowledge), called *abducible* predicates. In our case, the observations that are to be explained are the training sequences, the background knowledge used to derive explanations from is the automata interpreter and the *abducible* predicate is the $\texttt{transition}/3$ predicate that is used to



$\texttt{transition}(\texttt{q}_0, \texttt{x\_coord}(\texttt{g}), \texttt{q}_1).\ \texttt{transition}(\texttt{q}_1, \texttt{y\_coord}(\texttt{g}), \texttt{q}_2).$
$\texttt{transition}(\texttt{q}_2, \texttt{strength}(\texttt{b}), \texttt{q}_3).\ \texttt{start}(\texttt{q}_0).\ \texttt{final}(\texttt{q}_3).$

Figure 3.3: An example automaton, and its logical representation, accepting the multivariate simulation of Example 2.

encode an automaton in a logical form. Let us precede a more detailed account of the approach by a formal definition of abduction.

Definition 2. *(**Abductive Task**) An abductive task is defined as a tuple $\langle B, G, A \rangle$, where B is a logic program that represents some domain knowledge, G a set of observations, often called "goals" in the form of a set of logical facts and A a set of predicate signatures called abducible predicates. A predicate signature is an expression of the form p/n, where p is a predicate symbol and n is an integer that denotes p's arity. Given an abductive task, an abductive solution to the task is a set of ground logical facts Δ, each with a signature from A, such that $B \cup \Delta \models G$.*

Table 3.1 presents an automata interpreter that will play the role of the background knowledge B from Definition 2. Recall that variables start with an upper-case letter. The interpreter in Table 3.1 defines what an automaton does when it processes a sequence. The $\texttt{inState}(Id, S, T)$ predicate states that the automaton is in state S at step T of processing the sequence Id. The first rule in Table 3.1 simply states that for all sequences the automaton is at state q_0 at step 0 (i.e. q_0 is always assumed to be the start state).

The second rule states that while processing the sequence Id, the automaton moves to state S_2 at step $T + 1$ if it was in state S_1 in the previous step (T) and there is a transition from S_1 to S_2, which uses a symbol X appearing in the sequence in position T. Finally, the third rule dictates that a sequence is accepted if the automaton is in a final state when it reaches the end of the sequence (the meaning of the $\texttt{end}(Id, T)$ predicate is that at step T of processing the sequence Id the automaton reads the last symbol in the sequence). Note that via a closed world assumption, any sequence that is not accepted is assumed to be rejected.

The interpreter works in the multivariate case as well, where a single training example consists of multiple sequences. In this case an automaton transitions between states using symbols from different sequences of the same example/instance, which represent domain feature signals. The automaton accepts the example if it reaches the end of at least one of the sequences in the example, while being in a final state. The intuition here is that in the multivariate case an automaton represents a sequential pattern of the features, which need to be observed in a particular order, so as to classify the instance as positive (respectively, for the automaton to accept the instance).

Example 4. Processing Multivariate Instances Consider the multivariate simulation from Example 2 and the automaton M in Figure 3.3. Given the automata interpreter from Table 3.1 it can be seen that M accepts this example. Indeed, M is in state q_0 at time 1 (via the first interpreter rule) and it moves to state q_1, using the second interpreter rule and the facts $\texttt{transition}(\texttt{q}_0, \texttt{x\_coord}(\texttt{g}), \texttt{q}_1)$ from the structural description of M and $\texttt{seq}(\texttt{id}_3, \texttt{x\_coord}(\texttt{g}), 1)$ from the example description (see Example 2). From

these two facts and the second interpreter rule the fact $\texttt{inState}(\texttt{id}_3, \texttt{q}_1, 2)$ may be derived. Subsequently, and in a similar fashion, M moves to state q_2 using the facts $\texttt{transition}(\texttt{q}_1, \texttt{y\_coord}(\texttt{g}), \texttt{q}_2)$ and $\texttt{seq}(\texttt{id}_3, \texttt{y\_coord}(\texttt{g}), 2)$. Finally, M moves to state q_3, again using the second interpreter rule and the facts $\texttt{transition}(\texttt{q}_2, \texttt{strength}(\texttt{b}), \texttt{q}_3)$ and $\texttt{seq}(\texttt{id}_3, \texttt{strength}(\texttt{b}), 3)$, from which the fact $\texttt{inState}(\texttt{id}_3, \texttt{q}_3, 4)$ is derived. Since at time point 4 all sequences in the multivariate example have ended and q_3 is a final state, the third interpreter rule yields the fact $\texttt{accepted}(\texttt{id}_3)$.

Note that the automaton in Figure 3.3 represents a hypothetical, multi-feature sequential pattern that may be observed in a real-life scenario and could allow to discriminate between blocking (positive examples) and non-blocking incidents (negative examples). According to this pattern, an example is classified as interesting if a value of g for the user's x-coordinate is observed, followed by an equal value for her y-coordinate, followed by a value of b for signal strength.

Let us now return to the abductive task that lies at the core of our learning setting. Note that, as shown in action is Example 4, the automata interpreter links the $\texttt{seq}/3$ predicate, which is used to encode the input sequences, with the $\texttt{transition}/3$ predicate, which is used to encode an automaton. This allows to use the interpreter as background knowledge that may be reasoned upon, in order to abductively derive an automaton (a collection of $\texttt{transition}/3$ facts) that explains the input sequences. The latter are treated as observations (as in Definition 2) and the abductive explanations that are derived are subject to a number constraints related to the sequences' acceptance/rejection by the abductively-generated automaton. Definition 3 makes this setting clearer.

Definition 3 (**Abductive Automata Learning**). *Let $\mathcal{A}$ be an abductive task $\mathcal{A} = \langle B, G, A \rangle$ as in Definition 2 where:*

- *B is the interpreter from Table 3.1 in addition to a set of facts representing a training set of labeled input sequences $S^+ \cup S^-$ in logical form, as described in Examples 1 and 2.*
- *G is the set of facts $\{\texttt{accepted}(id_1), \ldots, \texttt{accepted}(id_n)\}$ where id_i is the is of a sequence in S^+ and $n = |S^+|$.*
- *$A = \{\texttt{transition}/3, \texttt{final}/1\}$.*

Note that from the definition of the abductive task $\mathcal{A}$ above, it follows that a solution Δ to this task represents the structural description (i.e. the transition function and the set of final states) of an automaton that accepts all positive sequences in the training set, since from definition 2 we have that the abductive solution Δ has the property of $B \cup \Delta \models G$. Additionally, this automaton rejects all negative sequences in the training set, since via closed world assumption $\texttt{accepted}(id_i)$ should be false for each $id_i \in S^-$.

The abductive learning setting described above is straightforward to represent in ASP and learn the required automaton (i.e. find a solution to the abductive task), by using an off-the-self solver, such as Clingo[1]. This is because abduction naturally fits ASP's generate-and-test problem solving methodology: Generate instances of the abducible predicates and test the candidate solutions against constraints derived from

[1] https://potassco.org/clingo/

$\texttt{inState}(\texttt{Id}, \texttt{q}_0, 0) : \texttt{-}\ \texttt{seq}(\texttt{Id}, \_, \_).$

$\texttt{inState}(\texttt{Id}, \texttt{S}_2, \texttt{T}+1) : \texttt{-}\ \texttt{inState}(\texttt{Id}, \texttt{S}_1, \texttt{T}), \texttt{transition}(\texttt{S}_1, \texttt{X}, \texttt{S}_2), \texttt{seq}(\texttt{Id}, \texttt{X}, \texttt{T}).$

$\texttt{accepted}(\texttt{Id}) : \texttt{-}\ \texttt{inState}(\texttt{Id}, \texttt{S}, \texttt{T}), \texttt{end}(\texttt{Id}, \texttt{T}), \texttt{final}(\texttt{S}).$

$\texttt{state}(\texttt{q}_0).\ \texttt{state}(\texttt{q}_1).\ \texttt{state}(\texttt{q}_2).\ \texttt{start}(\texttt{q}_0).$

$\texttt{positive}(\texttt{SeqId}) : \texttt{-}\ \texttt{label}(\texttt{SeqId}, 1).$

$\texttt{negative}(\texttt{SeqId}) : \texttt{-}\ \texttt{label}(\texttt{SeqId}, 0).$

Generate automata:

$\{\ \texttt{transition}(\texttt{S}_1, \texttt{X}, \texttt{S}_2)\ \} : \texttt{-}\ \texttt{state}(\texttt{S}_1),\ \texttt{state}(\texttt{S}_2),\ \texttt{symbol}(\texttt{X}).$

$\{\ \texttt{final}(\texttt{S})\ \} : \texttt{-}\ \texttt{state}(\texttt{S}).$

Test the performance of the generated automata:

$: \sim\ \texttt{accepted}(\texttt{SeqId}),\ \texttt{negative}(\texttt{SeqId}).\ [1@0, \texttt{SeqId}]$

$: \sim\ \texttt{not accepted}(\texttt{SeqId}),\ \texttt{positive}(\texttt{SeqId}).\ [1@0, \texttt{SeqId}]$

Minimize the size of the learnt automaton as much as possible:

$: \sim\ \texttt{transition}(\texttt{S}_1, \texttt{X}, \texttt{S}_2).\ [1@0, \texttt{S}_1, \texttt{S}_2, \texttt{X}]$

Table 3.2: A complete ASP program for learning automata with at most three states that minimize the training error.

the goals in the abductive task. The generate part of this process is implemented by appropriate choice rules. The test part is implemented via ASP constraints. In particular, using weak constraints allows to realize the learning setting of Definition 1, i.e. learn an automaton with at most N states, for a given N, that minimizes the training error.

Table 3.2 presents an example of a complete program that may be used with Clingo for learning an optimal automaton with at most three states. The choice rules in the "generate" part of the program generate candidate solutions, while the weak constraints in the "test" part of the program are responsible for dismissing sub-optimal automata. Finally, the last weak constraint reduces the size of the learnt automaton as much as possible, by dropping redundant transitions (and states referenced therein). Therefore, the program in Table 3.2 generates automata with at most three states (and it may be adapted to any "state budget" N). Additionally, The learnt automata minimize the training error as defined in Definition 1, since the weak constraints minimize the sum of false positive and false negative mistakes.

Table 3.3 illustrates our abductive, ASP-based automata learning technique on a toy dataset consisting of a single pair of examples, one positive and one negative. The sequences in these examples consist of initial segments (first ten points) of two

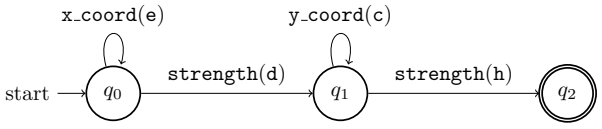
(a) A training set consisting of two examples.

Positive example:	***Negative example:***
x_coords: `eeeedcbbbb`	*x_coords:* `eecdbbbbbb`
y_coords: `aabbbcccdd`	*y_coords:* `aabbbbcccc`
signal strength: `bbbcdghhhhh`	*signal strength:* `bbbcfghhhh`

(b) An optimal answer set generated from the training set and the program in Table 3.2:

$\texttt{transition}(q_0, \texttt{x\_coord}(e), q_0).\texttt{transition}(q_0, \texttt{strength}(d), q_1).\texttt{transition}(q_1, \texttt{strength}(h), q_2).$
$\texttt{transition}(q_1, \texttt{y\_coord}(c), q_1).\texttt{final}(2).$

(c) A graphical representation of the corresponding automaton:



(d) A more compressed automaton learnt from the same training set using extra domain knowledge (see Example 5):

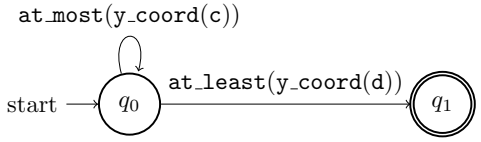


Table 3.3: Automata learning example.

actual The sequences in these examples consist of initial segments (first ten points) of two actual examples from a blockage prediction data set[2], which are deliberately shortened for illustration purposes. Table 3.3 (c) also presents a three-state automaton that is learnt from this training set, which accepts the positive and rejects the negative example.

Combining Automata Learning with Reasoning

An important feature of our proposed ASP-based technique is that it allows to integrate reasoning with arbitrary domain knowledge into the automata learning process. This is important, since it makes it possible to infer, at learning time, relations between the various domain features, which are not directly observable in the input data, and use such relations in the automata construction process. In turn, this may significantly simplify the learning process and further compress the learnt automata, making them more interpretable and accessible to domain experts, while potentially increasing their generalization abilities. We illustrate the case by means of an example.

Example 5. Combining Automata Learning with Reasoning We will use the domain of blockage prediction to provide an example of combining automata learning

[2] `https://viwi-dataset.net`

with reasoning. Since the feature values in the domain are real numbers (discretized into symbols) it is plausible to assume that reasoning with the ordering of these values may have some merit in terms of compressing the learnt automata. To provide some intuition, note that a single transition in a learnt automaton may be executed upon the occurrence of more than one symbols in the alphabet, which, in the representation of the automaton are treated as a disjunction. For example moving from state q_3 to state q_0 in the automaton of Figure 3.2 is possible either after observing a "b" or a "c". Similarly, in the blockage domain, moving from some state q_m to another state q_n may be possible after observing multiple different values for some feature, e.g. "a", "b", or "c" for e.g. the user's (x, y)-coordinates, or the signal strength at some time point, which correspond to three different transitions in the automaton:

$\texttt{transition}(\texttt{q}_\texttt{m}, \texttt{x\_coord}(\texttt{a}), \texttt{q}_\texttt{n})$. $\texttt{transition}(\texttt{q}_\texttt{m}, \texttt{y\_coord}(\texttt{b}), \texttt{q}_\texttt{n})$.
$\texttt{transition}(\texttt{q}_\texttt{m}, \texttt{strength}(\texttt{c}), \texttt{q}_\texttt{n})$.

Since these values ("a","b","c") correspond to numerical intervals (i.e. ranges of values), they are naturally comparable to each other and their lexicographic ordering corresponds to the actual ordering of the corresponding value ranges. Therefore, it is possible to replace $\texttt{x\_coord}(\texttt{a})$, $\texttt{x\_coord}(\texttt{b}), \texttt{x\_coord}(\texttt{c})$ with a single predicate $\texttt{at\_most}(\texttt{c})$ which is true whenever a value that is less than or equal to "c" is observed for the user's x-coordinate (a similar $\texttt{at\_least}/1$ predicate with the obvious meaning may be also defined). The automata learning program in Table 3.2 may be easily modified to take into account this simple reasoning with the ordering of the observed values. Table 3.3 (d) shows a automaton that is learnt with such a modification from the toy training set. Observe that the automaton is simpler and more compressed that the one presented in Table 3.3 (c), while it still correctly discriminates between the positive and the negative examples in the training set.

3.4 Preliminary Results

To perform an empirical assessment of our automata learning technique in the telecommunications domain, while the actual ARIADNE mobility data were under preparation we used a publicly available data set as a surrogate. The ViWi (Vision-aided Wireless communications) data set[3] that we used consists of matlab-generated simulations of blockage incidents involving mobile objects, such as vehicles and static base stations. To derive a first, proof-of-concept evaluation of our technique we used one scenario from the ViWi data set, namely the "co-located camera scenario with blocked view"[4]. Figure 3.4 illustrates the simulation setting.

It is an outdoor scenario depicting a car driving through a city street (across five different trajectories) with two stationary buses. It involves a single Base Station equipped with a mmWave antenna and 3 cameras. The BS is set at 5 meters high and in the middle of the street. The car goes through one of five trajectories, each of which is 90 meters long and has 1000 equally spaced points (0.089 m). Hence, they, all together, create a 5000-point user grid. A blockage incident occurs when a car moves

[3] https://viwi-dataset.net/
[4] https://viwi-dataset.net/scenarios.html

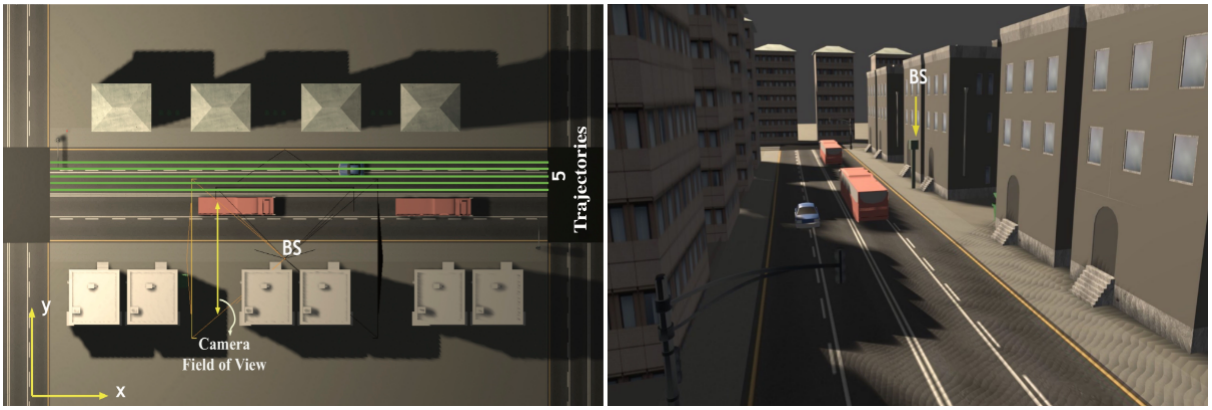


Figure 3.4: An illustration of the "co-located camera scenario with blocked view" from the ViWi data set.

in a way that a bus blocks its signal from the base station. The inclusion of cameras in the simulation is related to the fact that this dataset has been created as a test-bed for methods that learn to detect blockage incidents using computer vision methods, i.e. trained models that detect blockages from video feeds (which are supposed to be provided from mounted traffic inspection cameras). We do not concern ourselves with the vision aspects of the use-case. Instead, we attempted to learn automata that discriminate between blockage/non-blockage incidents using the evolution of the car's position and signal strength over time, which are also provided in the data set.

As in the blockage-related examples provided in the previous sections of this chapter, the training data consists of "routes"/trajectories that lead to a blockage incident and others that do not. An example consist of a trajectory, described by the car's (x, y)-coordinates over time and its signal strength. We generated a training set consisting of 500 positive and 1000 negative examples from the car's trajectories in the data set, where a positive (resp. negative) example represents a (sub-)trajectory where the car gets blocked (resp. non-blocked) at least once over its route. Note that in one such training sub-trajectory several blockage incidents may occur. For instance, in its starting point a training trajectory might already be blocked (i.e. somewhere behind the first bus), it then gets unblocked as it moves away from the bus and gets blocked again as it gets behind the second bus. Each example in the data set consists of three sequences corresponding to three features, the car's x and y coordinates and its signal strength. Each sequence in an example has length 100 (i.e. it sequence references values for the corresponding feature from 100 consecutive time points).

To discretize the numerical data we used the SAX algorithm[5], a state-of-the-art tool for converting numerical sequences into symbolic ones, by associating each value in the original sequence to a symbol that represents a value range/interval. The granularity of the discretization (i.e. the number of different symbols that will be used, and therefore, the length of the value range that corresponds to each symbol) is user-defined. A larger number of symbols represents a discretization that is closer to the original numerical sequence (but also, a symbolic sequence that is more complex and difficult to handle). For this experiment we used an alphabet size (number of different symbols in the discretization process) of 20.

We compared out ASP-based automata learning technique with two state-of-the-

[5] https://jmotif.github.io/sax-vsm_site/morea/algorithm/SAX.html

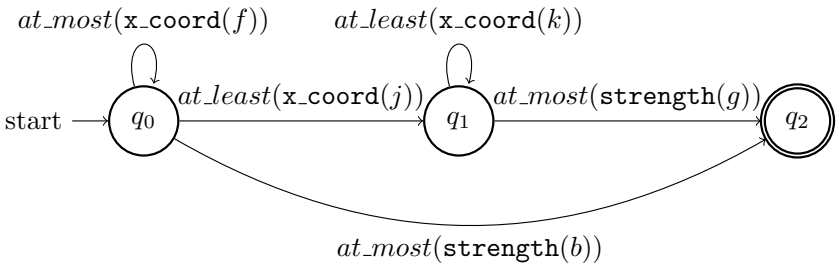


Figure 3.5: An indicative symbolic automaton learnt with Abductive Automata Learning from the ViWi data set. This automaton achieves an average F_1-score of 0.983 on the testing set.

art, widely-used in practice automata learning methods, namely RPNI (Regular Positive/Negative Inference) and EDSM (Evidence-Driven State Merging). Both these algorithms rely on the state-merging techniques for automata learning (see also Section 3.2 for a brief description of the technique), which trades quality/optimality of the learnt automata for efficiency. Both RPNI and EDSM are implemented in the learnlib toolkit[6][7], a highly-optimized toolkit for automata learning, containing implementation for several classical automata induction algorithms, with RPNI and EDSM being the state-of-the-art. An important disadvantage of both RPNI and EDSM is that, similarly to almost all existing automata learning techniques, they cannot handle multi-variate sequences. For this experiment we used each one of the three features in isolation to learn automata with RPNI and EDSM and report the best results, which were obtained by using the signal strength feature for training.

The goal was to learn patterns for blockage incidents with each of the three algorithms being compared. For this comparison we used a five-fold cross-validation process with 80%/20% training/testing splits respectively and measured the F_1-scores of the learnt automata in the testing set, in addition to their size (number of states and transitions in an automaton), as well as training times for each algorithm being compared. Table 3.4 presents the results in the form of average values for each statistic, obtained from the cross-validation process.

	F_1-score on test set	# states	# transitions	Training time (sec)
ASP	**0.988**	**3**	**6**	13.234
RPNI	0.734	17	162	1.332
EDSM	0.702	19	182	**0.987**

Table 3.4: Table to test captions and labels.

Table 3.4 presents the results from the comparison while Figures 3.5 and 3.6 present two indicative automata learnt from the ViWi dataset with the ASP-based abductive automata learning technique and RPNI respectively. The results indicate that our approach clearly outperforms the state-merging-based ones in terms of predictive accuracy, simplicity and interpretability of the learnt patterns. This is due to the fact that the

[6] https://learnlib.de/
[7] https://github.com/LearnLib/learnlib

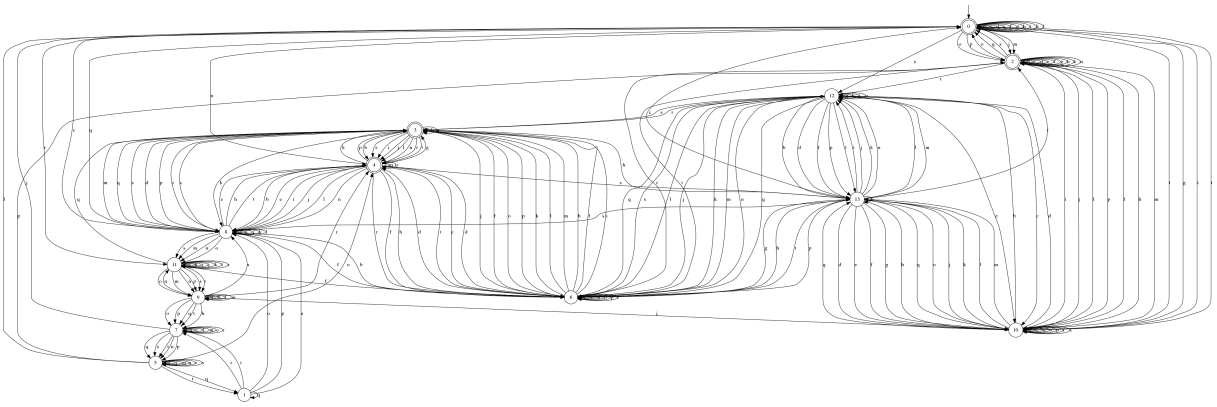


Figure 3.6: An indicative symbolic automaton learnt with RPNI from the ViWi data set. This automaton achieves an average F_1-score of 0.712 on the testing set.

ASP technique can be guided to learn near-optimal automata, in contrast to RPNI and EDSM which learn using sub-optimal heuristics. Additionally, the ASP-based technique incorporates reasoning with domain knowledge into the automata learning process. This makes it possible to learn symbolic automata, which are much more compressive, expressive and interpretable. In turn, the simplicity of the ASP-learnt patterns is reflected to their predictive performance, as compared to that of the state-merging algorithms. The trade-off is that the ASP-based technique is significantly less efficient than its competitors. This is expected, since as mentioned earlier efficiency is the main goal of RPNI and EDSM. Improving the efficiency of our proposed approach using incremental and online learning techniques, while preserving comparable quality of the learnt automata is part of our future work.

3.5 Conclusion and Future Work

There are several points of focus regarding current and future work in ARIADNE. The two immediate goals are (i) to asses the performance of our automata learning technique on the actual ARIADNE mobility data, which have recently been made available and (ii) use the learnt patterns for complex event forecasting with Wayeb, NCSR's forecasting engine outlined in Section 3.2. The latter, in particular, involves the crucial step of modifying our automata learning technique to make the forecasting performance, rather than the current recognition/detection performance, the main training objective that guides the search for high-quality automata patterns. This will allow to learn patterns particularly tailored *for* forecasting complex events, rather that detecting them a posteriori.

Chapter 4

System Level Simulation Model for Proactive Handover and Efficient Resource Allocation in D-band Networks

4.1 Introduction

This chapter focuses on system level simulation and modeling of D-band networks in realistic environments within the concept of ARIADNE. Particularly, a geographic area is demonstrated, where static obstacles are placed in predefined coordinates together with static access points and mobile users. The placement of the access points and the initial placement of the users are defined by stochastic geometry tools. Then, two mobility models have been used for the incorporation of time evolution in the simulations and suitable path loss models are applied. The received power of the users is mapped as function of time and D-band wireless links are identified as LOS, partially or totally blocked. The composite information can be used for various investigations, as for example to detect forthcoming blockage incidents before they actually occur. These blockage forecasts could be exploited for proactive handover, which finally leads to increasing network's reliability, as outage events are prevented. Specifically, the generated data will subsequently be used by AI-based techniques for blockage forecasting and proactive handover, while they will also be used to assess the efficacy of the resource allocation techniques (users-to-access points assignment) in dynamic settings. In general, the parameters of the simulations can be adjusted in order to demonstrate several scenarios under ARIADNE use cases. For example, we can assume that all users in the simulated area move towards a specific point, which is a hotspot (e.g. a data kiosk – Scenario 2.2 – Deliverable 1.1). Also, we could adjust the dimensions of the geographic area and the density of the users and the access points, so that they correspond to an indoor environment (Scenario 2.1 – Deliverable 1.1), to an urban, suburban or rural environment etc. Moreover, we can demonstrate simpler cases regarding the mobility of transmitters (Tx) and receivers (Rx), respectively. Such an example is the case of outdoor backhaul/fronthaul networks of fixed topology (Use Case 1 – Scenarios 1.1 and 1.2 – Deliverable 1.1).

4.2 Deployment of Mobility Models

In this section, we describe in detail the mobility models that will be used for the system level simulations. These are the Pursue Mobility Model and the Obstacle Mobility Model and they are exploited in the simulations, as they give realistic movement features to the users.

4.2.1 Pursue Mobility Model

In this model, the member nodes of a group try to catch a target node (logical center) and finally, the member nodes converge on the target node (Fig. 4.1 [51]). The target node usually moves according to the Random Waypoint model [51]. However, the target node is considered to be a stable point in the cases that we will examine. We make this assumption in order to demonstrate in a realistic way the traffic conditions around a hotspot (such as a bus station, a mall area etc.).

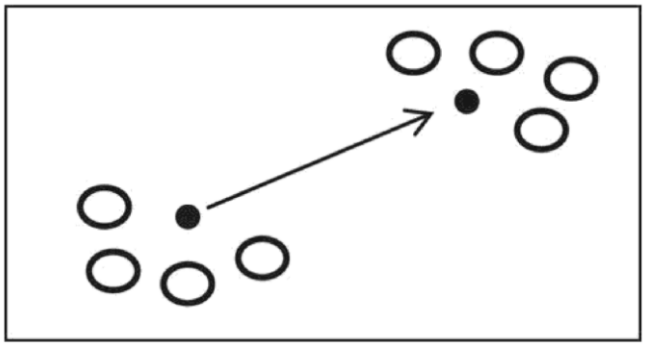


Figure 4.1: Pursue Mobility Model.

4.2.2 Obstacle Mobility Model

In this model, each node moves around the obstacles of the simulation environment following a recursive process in order to reach its destination (Fig. 4.2 [52]). Specifically, if there is an unobstructed line connecting the node with the destination point, the node follows this direct line to get to the desired destination. On the other hand, if the line intersects with an obstacle, the node sets as its next intermediate destination, the vertex of the obstacle's edge that is directly visible and closest to the destination and repeats the same process all over again with starting point its initial position and destination the chosen vertex. This is repeated until an unobstructed direct line is found between the node and the current destination. The whole process is recursively executed until the destination is reached [52].

4.3 Deployed System Level Simulation Model

The system level simulation model which has been developed, consists of the following. Firstly, there is a simulated geographic area, which is considered as a square with dimensions $30m \times 30m$. Then, we assume the presence of mobile users, who are

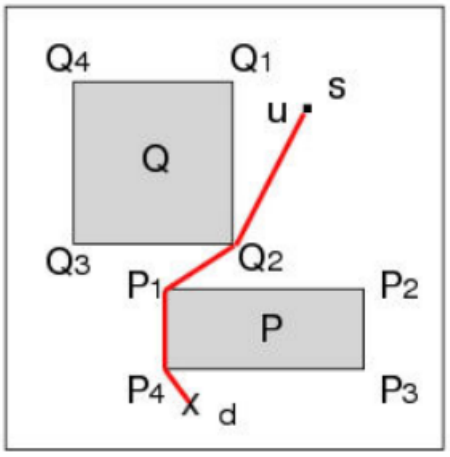


Figure 4.2: Obstacle Mobility Model.

modeled as points coming from a uniform Poisson Point Process (PPP) with density λ_{user}. Secondly, four obstacles are set in the area, which are modeled as rectangular parallelepipeds with dimensions $5m \times 4m \times 5m$. They are located uniformly and symmetrically in the simulated area. Finally, there is a number of access points, which also come from another independent and uniform PPP with density λ_{AP}. They are placed at a height of $2m$ from the ground.

In order to involve time evolution in the simulations and describe the mobility of the users in a realistic way, we exploit the two mobility models that were described in the previous section, namely the Pursue Mobility Model and the Obstacle Mobility Model. In the Pursue Mobility Model, the member nodes of a group of mobile nodes try to catch a $target - point$. Specifically, they converge on this $target - point$, which is assumed to be stable (e.g. a hotspot). In the Obstacle Mobility Model, the movement of a mobile node takes into account the obstacles of the environment. These affect the movement pattern of the nodes. Specifically, a node has to change its trajectory when encounters an obstacle. A simulation session of duration T consists of a number of successive time slots. Each of them has duration dt. The velocity V of a user is constant during a specific time slot and its value is a random variable, which is uniformly distributed over the interval $[0, V_{max}]$. V_{max} is the maximum allowable value of user's velocity. Thus, the displacement dS of a user during a time slot is: $dS = V \times dt$. The direction of dS is determined by the relevant mobility models. Such examples are depicted in Fig. 4.3, 4.4 and 4.5, while the corresponding simulation parameters are shown in Table 4.1. Specifically, the top view of the simulation area is depicted and users' trajectories are represented by successive dots of a specific color for each user. The biggest dot corresponds to the first time slot of the simulation, while the smallest one corresponds to the last time slot.

It is noted that the values of the above parameters can change accordingly, in order to study several ARIADNE use cases and demonstrate their respective scenarios, such as indoor or outdoor environments, heavy traffic conditions, P2P backhaul/fronthaul links, V2V and V2X links etc. Moreover, the use of uniform PPPs (stochastic geometry) for the access points and users' initial place provides us with an analytical tool towards the performance evaluation of D-band networks, which ensures a great degree of randomness in our study, despite the deterministic placement of the obstacles

in the simulation field.

Finally, we have to mention that each mobile user is allocated to the nearest LOS access point during the first time slot of the simulation and remains allocated to the same access point until the end of the simulation, independently of the condition of the link between user and serving access point (LOS or not) during the next time slots. Of course, this policy can be modified depending on the scope of the simulation scenarios.

Table 4.1: Simulation parameters for Fig. 4.3, 4.4, 4.5.

	Figure 4.3	Figure 4.4	Figure 4.5
$\lambda_{user}(m^{-2})$	0.002	0.005	0.015
$\lambda_{AP}(m^{-2})$	0.005	0.01	0.001
$target-point$ coordinates (m)	$(-2,3)$	$(0,0)$	$(-5,-10)$
$dt(sec)$	0.25	0.5	0.1
$T(sec)$	100	150	80
$V_{max}(m/sec)$	0.4	0.5	0.6

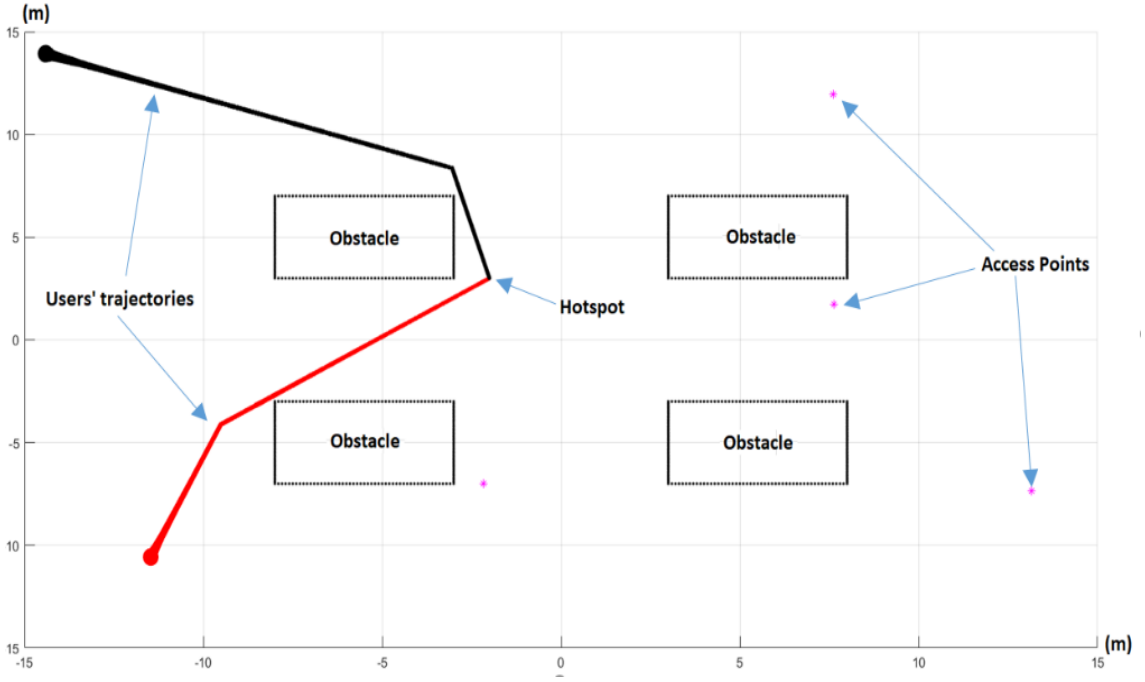


Figure 4.3: Indicative Simulation: $\lambda_{user} = 0.002m^{-2}$, $\lambda_{AP} = 0.005m^{-2}$, $target-point = (-2,3)$, $dt = 0.25sec$, $T = 100sec$ and $V_{max} = 0.4m/sec$.

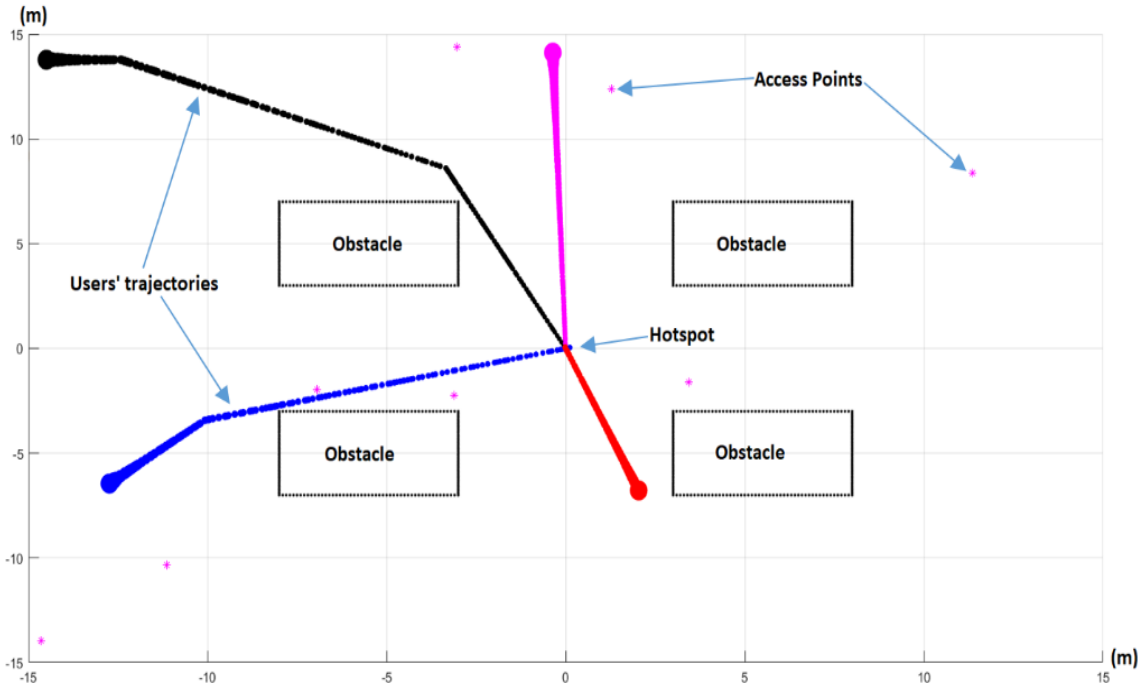


Figure 4.4: Indicative Simulation: $\lambda_{user} = 0.005m^{-2}$, $\lambda_{AP} = 0.01m^{-2}$, $target - point = (0, 0)$, $dt = 0.5sec$, $T = 150sec$ and $V_{max} = 0.5m/sec$.

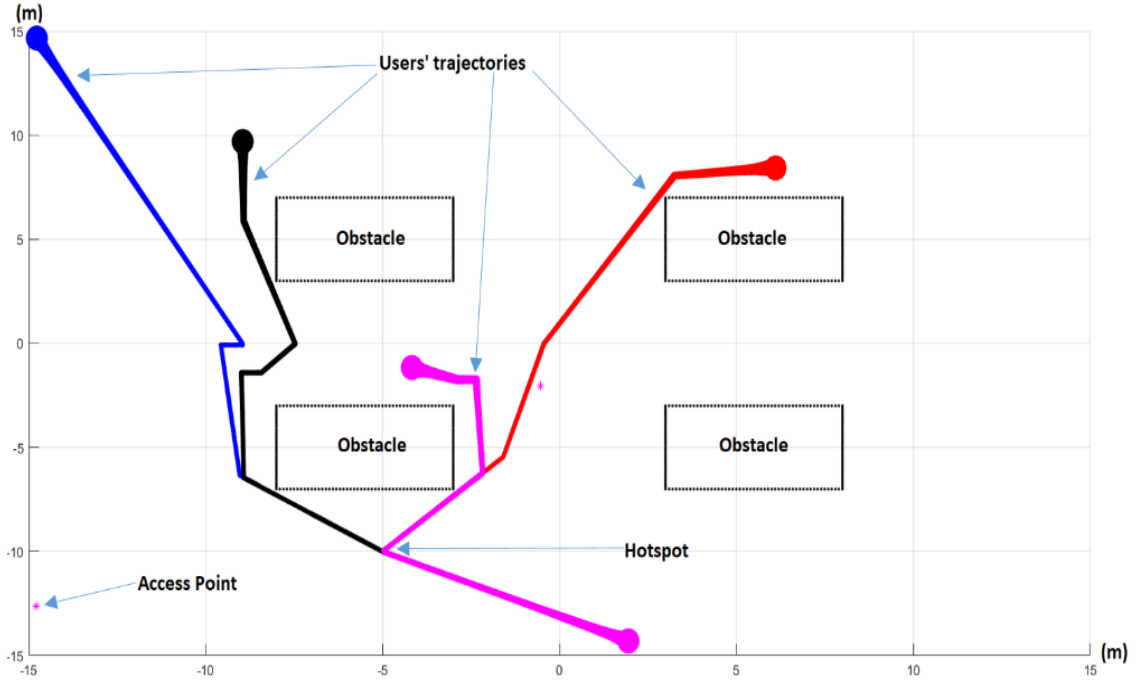


Figure 4.5: Indicative Simulation: $\lambda_{user} = 0.015m^{-2}$, $\lambda_{AP} = 0.001m^{-2}$, $target - point = (-5, -10)$, $dt = 0.1sec$, $T = 80sec$ and $V_{max} = 0.6m/sec$.

4.4 Path Loss Modeling

The next stage is the estimation of the received power of each user from every access point, for every time slot of the simulation. The goal is the mapping of the received power of a user from every access point as function of the time. For that purpose, we make use of Friis equation:

$$P_R = P_T + G_T + G_R + 20log\frac{\lambda}{4\pi \times d_0} - 10n \times log\frac{d}{d_0} \tag{4.1}$$

where P_R is the received power (in dBm), P_T is the transmitted power (in dBm), G_T and G_R are the gains of the Tx and Rx antennas, respectively (in dBi), λ is the free space wavelength (in m), n is the Path Loss Exponent (PLE), d_0 is the reference distance (in m – usually $d_0 = 1m$) and d is the distance between Tx and Rx (in m).

The challenge of using this equation, is the selection of the appropriate value of Path Loss Exponent for every case. Finally, for an urban microcell or a small-cell environment at $142GHz$, we can choose the PLE values to be [53]: $n = 2.1$ for LOS links (Non Blocked Channel) and $n = 3.1$ or 3.6 for Non-LOS links (Partially Blocked Channel). If the wireless channel between a user and an access point is totally blocked, we assume the received power of a user to be equal to the default noise level.

4.5 Partial and Total Blocking of users

A channel is considered as partially blocked, if the line sector between a user and an access point is close to a vertex of an obstacle, that is $r \leq \sqrt{\frac{d\lambda}{4}}$, where r is the distance between the line sector and the vertex. Then,we distinguish two cases:

i) The line sector does not intersect with the obstacle.

ii) The line sector intersects with the obstacle and the intersected sides of the obstacle are adjacent.

In above case i) we choose the PLE to be $n = 3.1$, while in case ii) we set the PLE to be $n = 3.6$. It is noted that the distance $\sqrt{\frac{d\lambda}{4}}$ is the maximum radius of the first Fresnel zone, which corresponds to the middle point of the line sector between the user and the access point. However, in the above approach we assume that condition $r > \sqrt{\frac{d\lambda}{4}}$ has to be verified along the whole line sector in order to consider the channel as LOS. This assumption is simpler than the required condition of Fresnel's Law, where the unobstructed area has to be an ellipsoidal and hence, it is easier to be implemented in the simulations. Nevertheless, it is a stricter condition.

In any other case, where the line sector intersects with an obstacle, the channel is considered as totally blocked, while it is considered as LOS, if it is free of obstacles ($n = 2.1$). All of the above cases are depicted in Fig. 4.6 for a specific time slot.

Finally, we have to mention the following assumption that has been made regarding the wireless channel blocking. Specifically, if the line sector between a user and an access point is close to two vertices of two different obstacles ($r \leq \sqrt{\frac{d\lambda}{4}}$), respectively, then the wireless link is considered to be totally blocked (Fig. 4.7). Therefore, the received power of the user is equal to the default noise level.

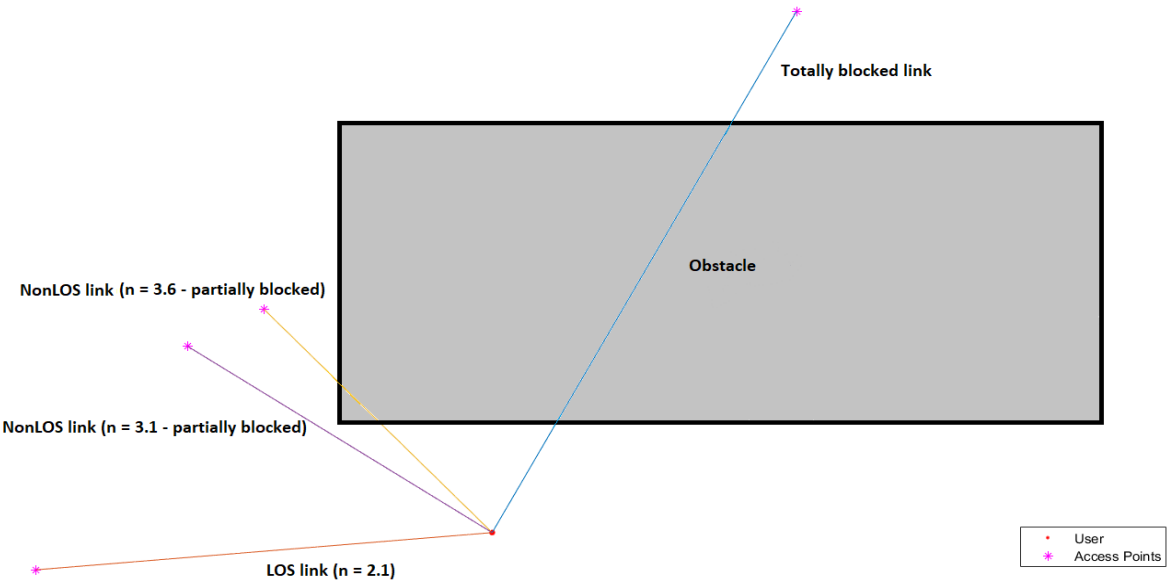


Figure 4.6: LOS, partially blocked and totally blocked wireless links.

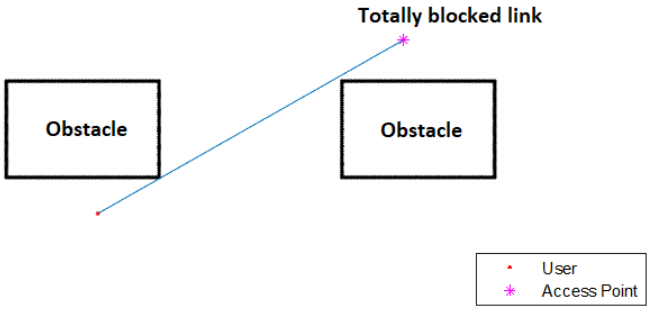


Figure 4.7: Totally blocked wireless link due to the partial blockage caused by two vertices of two different obstacles.

4.6 Received Power Data

An indicative simulated scenario of user's received power is depicted in Fig. 4.8 and Fig. 4.9. Specifically, the top view of the simulated geographic area is depicted in Fig. 4.8, where there are three access points and one user. Time slot duration is $dt = 0.1sec$. The blue line depicts user's trajectory. The dot on the left side of the line corresponds to the initial place of the user at the first time slot of the simulation. The power that the user receives from every access point as function of time, is mapped in Fig. 4.9. The carrier frequency is considered as $f = 142GHz$.

Moreover, in this scenario we assume that the transmitted power is $P_T = 0dBm$ and the gains of the user and the access points antennas are $G_T = G_R = 27dBi$. The received power from Access Point 1 (blue curve) is approximately constant around $-40dBm$ for every time slot of the simulation, as the link between user and Access Point 1 is always LOS. On the other hand, the received power from Access Point 3 (yellow curve) is $-130dBm$ (which is assumed to be the default noise level) for the whole duration of the simulation, as this link is totally blocked for every time slot of the

simulation.

However, the most interesting case is the time evolution of the received power from Access Point 2 (orange curve). During the first time slots of the simulation, the link is LOS. In the middle of the simulation, the link gets partially blocked, as there is an abrupt decrease of the received power. Finally, the specific link becomes totally blocked, as the received power is equal to the default noise level during the last time slots of the simulation.

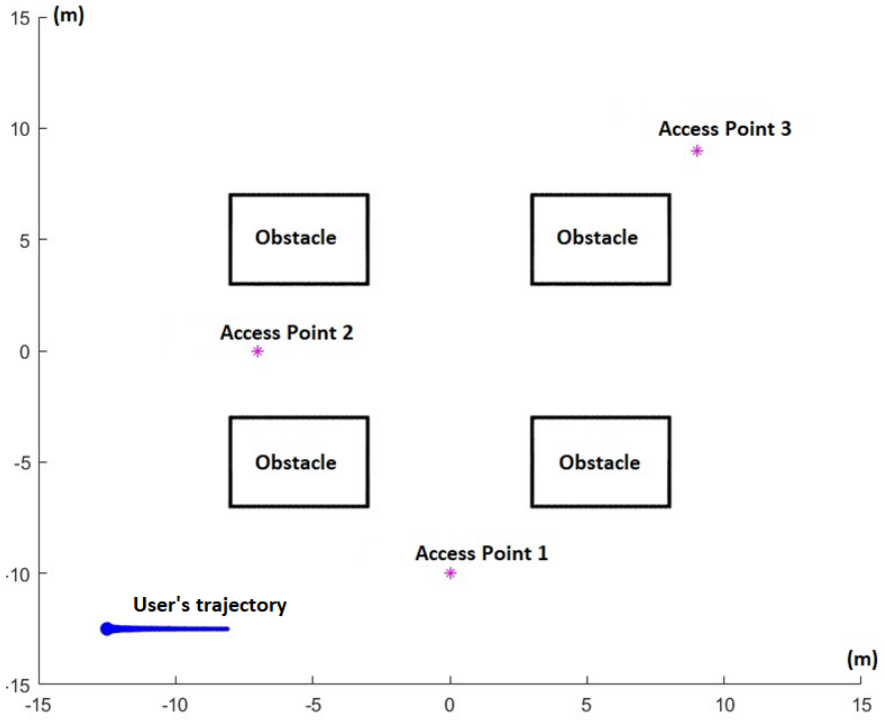


Figure 4.8: Indicative Simulation for three access points and one user.

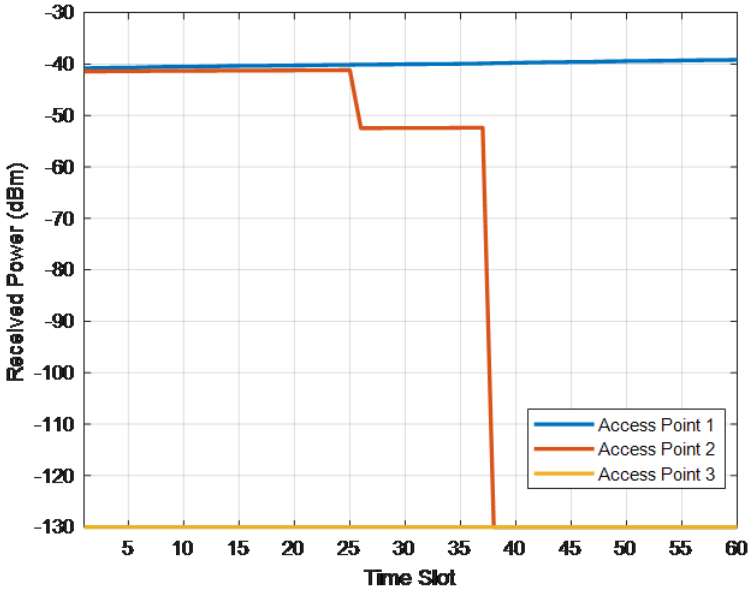


Figure 4.9: Received Power vs Time (three access points, one user).

Another interesting example of a user's received power is the following. Specifically, the top view of the simulated geographic area is depicted in Fig. 4.10, where there are two access points and one user. In this case, the time slot duration is $dt = 0.5sec$. The black line depicts user's trajectory. This line is not straight, as it was in the previous example, because there is an obstacle between the initial place of the user and his destination. The dot on the right side of the line depicts the initial place of the user at the first time slot of the simulation. The power that user receives from every access point as function of time is mapped in Fig. 4.11.

In this scenario, we assume again that the carrier frequency is $f = 142GHz$, the transmitted power is $P_T = 0dBm$ and both gains of the user and the access points antennas are $G_T = G_R = 27dBi$. The received power from Access Point 1 (blue curve) fluctuates around $-40dBm$ for the first time slots of the simulation, as the link between user and the specific access point is LOS during this time interval. However, the wireless link gets totally blocked in an abrupt way during the last time slots of the simulation. Specifically, the received power from this access point becomes equal to $-155dBm$, which is the default noise level in this example. This transition, where partial blockage of the link does not occur, could be explained by the longer duration of the time slots $(dt = 0.5sec)$ compared with the previous simulation (where $dt = 0.1sec$), as well as the geometric features of user's trajectory.

The received power from Access Point 2 (orange curve) is equal to the default noise level during the first time slots of the simulation $(-155dBm)$, as this link is totally blocked during these time slots, but becomes LOS until the end of the simulation. The transition from a totally blocked link to a LOS link is abrupt again for the same reasons (longer time slot and geometric features of user's trajectory).

In conclusion, the transitions from the LOS state to the totally blocked state of a link can happen either in a smooth manner or in a more sharp way. In the case of the smooth transition, the wireless link is partially blocked during some time slots between the LOS and the totally blocked state, while in the case of the abrupt transition, there is not any time slot where the channel is partially blocked.

4.7 Conclusions

In this chapter, we examined proactive handover methods for blockage avoidance that are applied to D-band communications. We have described a system level simulation model, which incorporates the presence of rectangular parallelepiped obstacles in the simulation area, time evolution by using appropriate mobility models for the users and mapping of the received power of a user as function of time. Our main purpose is the simulation and the forecasting of blockage incidents for proactive handover and the efficient allocation of D-band network resources.

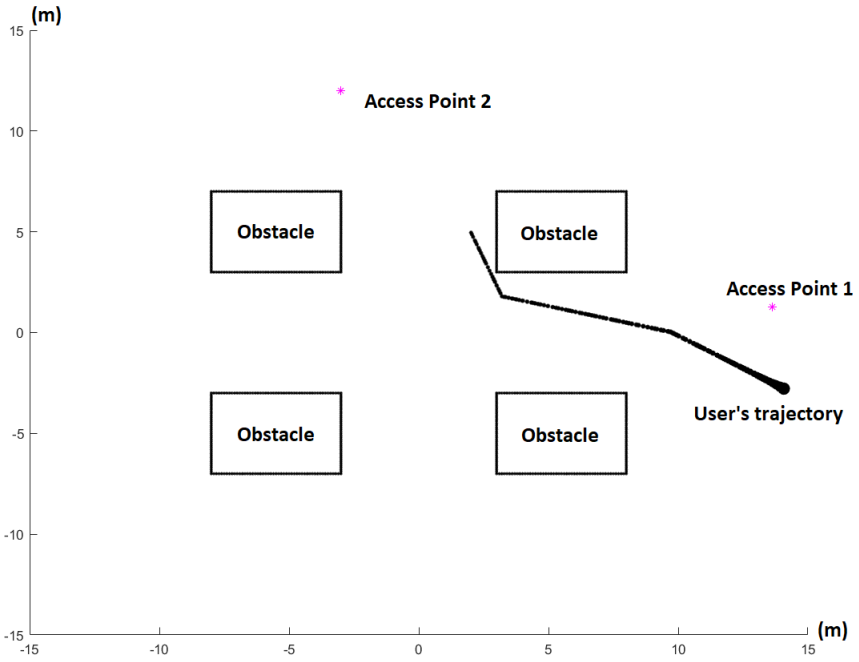


Figure 4.10: Indicative Simulation for two access points and one user.

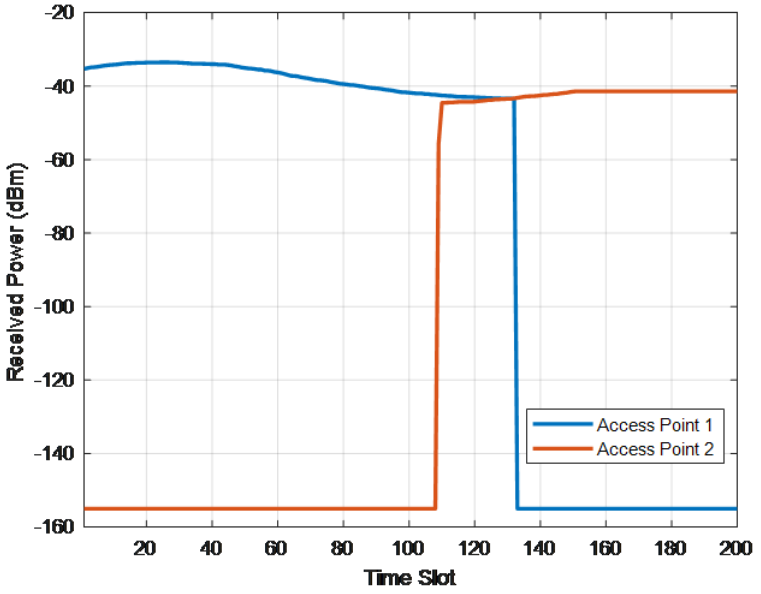


Figure 4.11: Received Power vs Time (two access points, one user).

Chapter 5

Model-Driven Deep Learning Based Channel Estimation and Feedback for High-Frequency Massive Hybrid MIMO Systems

5.1 Introduction

Communication at high frequencies (millimeter-wave, sub-terahertz and terahertz) has been widely recognized as a key technology in future wireless communication systems, since the abundant bandwidth resources can significantly increase the throughput [54, 55]. Moreover, to mitigate the severe propagation losses in the high frequencies, massive multiple-input multiple-output (MIMO) is usually adopted to perform beamforming [56, 57]. However, the fully-digital massive MIMO architecture gives rise to an unaffordable hardware cost and power consumption, where a dedicated radio frequency (RF) chain is required for each antenna. In order to circumvent the technical hurdle and facilitate the deployment of massive MIMO systems at high frequency in practice, the phase shift network (PSN) based hybrid MIMO architecture has been widely adopted to achieve a large array gain with a much smaller number of RF chains [58–60].

To fully capitalize on the large spatial degrees of freedom in massive MIMO systems at high frequency, channel state information (CSI) at the BS is essential, since beamforming, signal detection, and interference alignment heavily rely on accurate CSI at the BS [61]. As for time-division duplexing (TDD) systems, estimating the high-dimensional uplink massive MIMO channels at high frequencies from limited number of RF chains at the BS suffers from an excessively high pilot overhead [62]. As for frequency-division duplexing (FDD) systems, the downlink high-dimensional channel is first obtained at the users using very few RF chains, and is then fed back to the BS. In this case, the prohibitively high channel estimation and feedback overhead problem is more severe [63].

5.1.1 Related Work

To this end, by exploiting the sparsity of massive MIMO channels in the angle-domain and/or delay-domain, several low overhead channel estimation and feedback solutions have been proposed [61–67]. Specifically, by exploiting the temporal correlation of time-varying channels, the authors of [61] proposed a differential channel estimation and feedback scheme for FDD massive MIMO systems with reduced overhead, and a structured compressive sampling matching pursuit (S-CoSaMP) algorithm to acquire a reliable CSI at the BS. In [63], the authors proposed a spatially common sparsity based adaptive channel estimation and feedback scheme for FDD massive MIMO systems, which adapted the training overhead and pilot design to reliably estimate and feed back the downlink CSI with reduced overhead. Moreover, by introducing an enhanced Newtonized orthogonal matching pursuit (eNOMP) algorithm, the authors of [64] proposed an efficient downlink channel reconstruction-based transceiver for FDD massive MIMO systems. However, these schemes [61,63,64] were mainly proposed for low-frequency massive MIMO systems using a fully-digital array.

As for the hybrid MIMO at high frequencies, by exploiting the channel sparsity in both angle and delay domains, a closed-loop sparse channel estimation scheme for TDD systems was proposed in [65], which utilized ESPRIT-type algorithms to acquire super-resolution estimates of the angle of arrivals/departures (AoAs/AoDs) and of the delays of multipath components with low overhead. In [62], two high-resolution channel estimation schemes based on the ESPRIT algorithm were proposed for broadband massive MIMO systems at high frequencies. By exploiting the high frequency channels' sparsity, the authors of [66] proposed a compressive sensing (CS) greedy algorithm based channel estimation solution for reducing the channel estimation overhead. Additionally, by exploiting the 3-D clustered structure exhibited in the virtual AoA-AoD-delay domain, an approximate message passing (AMP) with the nearest neighbor pattern learning algorithm was proposed to estimate the broadband massive MIMO-OFDM channels in [67]. Although the overhead is reduced, the computational complexity of ESPRIT techniques [62, 65] and greedy algorithms [66] can be prohibitively high due to the matrix inversion and singular value decomposition (SVD) operations. Besides, although AMP algorithms based solutions [67] can reduce the computational complexity, they heavily rely on *a priori* models, which would lead to performance degradation since *a priori* models may not always be consistent with the actual systems.

5.1.2 Motivations

Recently, the successful application of deep learning in various fields, particularly in computer science, has gained major attention in the communication community, and has promoted an increasing interest in applying it to address communication and signal processing problems [68–71]. The deep learning based intelligent communication paradigm has attained manifold accomplishments, including channel coding [72], random access [73], beamforming design [74–78] activity and signal detection [79, 80], autoencoder-based end-to-end communication system [81], CSI feedback [82–84], and channel estimation [60, 85, 86], etc. To be specific, pure data-driven deep learning based solutions often employ deep neural networks (DNNs), including fully-connected neural networks and/or convolutional neural networks (CNNs), as a black box to design communication signal processing modules without any *a priori* model information.

Moreover, a large amount of training data samples are required to optimize the neural network through a customized loss function and learning strategy.

In particular, in order to overcome the high-computational complexity and fully exploit the spatial information, the authors of [75] proposed a deep-learning-enabled massive MIMO framework for effective hybrid precoding for high frequencies, in which each selection of precoders for obtaining the optimized decoder is regarded as a mapping relation in the DNN. In addition, the authors of [77] proposed a DNN-based approach for channel sensing and downlink hybrid analog-digital beamforming, which was generalizable for any numbers of users by decomposing the deep learning architecture into multiple parallel independent single-user DNNs. By considering the multiuser channel estimation and feedback problem as a distributed source coding problem, the authors of [78] proposed a joint design of pilots and a novel DNN architecture, which mapped the feedback bits from all the users directly into the precoding matrix at the BS. By exploiting an unsupervised ML model, i.e., an autoencoder, the authors of [76] presented a linear autoencoder-based beamformer and combiner design, which maximizes the achievable rates over a high frequency channel. Moreover, in order to address the overwhelming feedback overhead of FDD massive MIMO systems, the authors of [82] proposed a CS-ReNet framework, where the CSI was first compressed at the users based on CS methods and then reconstructed at the BS using a deep learning-based recovery solver. However, in practical scenarios, there exists various interference and non-linear effects. Therefore, the authors of [84] designed a deep learning-based denoising network, called DNNet, to improve the performance and robustness of channel feedback. Additionally, by exploiting the spatial, temporal, and frequency correlations, the authors of [60] employed a CNN to address the channel estimation problem for massive hybrid MIMO systems at high frequencies. However, the performance of those data-driven approaches heavily depends on the quantity and quality of the training data samples, but good data sets are usually difficult to be obtained in practice and practical issues such as training over-fitting could degrade the capability of the system to generalize. In addition, data-driven approaches lack interpretability and trustability that are major strengths of model-driven signal processing. Moreover, compared with conventional model-based methods, model-driven approaches have better denoising capabilities by utilizing the powerful data processing capabilities and denoising capabilities of neural networks.

Therefore, different from pure data-driven and conventional model-based approaches, model-driven deep learning (MDDL)-based approaches construct the network structure by exploiting *a priori* knowledge from known physical mechanisms, such as well-developed channel models and transmission protocols. Note that the MDDL-based approaches retain some of the advantages of conventional model-based iterative methods, which includes exploiting some *a priori* information to be trained with fewer trainable parameters and with less data samples, e.g., the structured sparsity of the high frequency channels can be exploited. Moreover, they can further retain the learning ability of deep learning methods and avoid the performance degradation caused by the mismatch between the predetermined parameters (based on an assumed model) and the true optimal parameters (based on empirical data samples). By leveraging some *a priori* information, model-driven methods require fewer parameters to be learned and less samples for training as compared to pure data-driven deep learning solutions [87–91]. Specifically, the authors of [88] proposed an MDDL-based downlink channel reconstruction scheme for FDD massive MIMO systems, where a

powerful neural network, named You Only Look Once (YOLO), was introduced to enable a rapid estimation process of the model parameters. Moreover, [91] proposed a novel AMP-based network with deep residual learning, referred to as LampResNet, to estimate the beamspace channel for massive MIMO systems at high frequencies.

5.1.3 Our Contributions

This chapter proposes an MDDL-based channel estimation and feedback scheme for wideband massive hybrid MIMO systems at high frequencies, where the angle-delay domain channels' sparsity is exploited for reducing the overhead. First, we consider the uplink channel estimation for TDD systems. To reduce the uplink pilot overhead for estimating the high-dimensional channels from a limited number of RF chains at the BS, we propose to jointly train the PSN and the channel estimator as an auto-encoder. Particularly, by learning the integrated trainable parameters from data samples and exploiting the channels' structured sparsity from an *a priori* model, the proposed multiple-measurement-vectors learned approximate message passing (MMV-LAMP) network with the devised redundant dictionary can jointly recover multiple subcarriers' channels with significantly enhanced performance. Moreover, we consider the downlink channel estimation and feedback for FDD systems. Similarly, the pilots at the BS and channel estimator at the users can be jointly trained as an encoder and a decoder, respectively. Besides, to further reduce the channel feedback overhead, only the received pilots on part of the subcarriers are fed back to the BS, which can exploit the MMV-LAMP network to reconstruct the spatial-frequency channel matrix. Simulations are conducted to demonstrate the effectiveness of the proposed MDDL-based channel estimation and feedback scheme over the conventional approaches.

The main contributions of this chapter are summarized as follows:

- Operations in the complex domain are well supported by most deep learning frameworks. However, the beamforming/combining matrix is a complex-valued matrix and satisfies the constant modulus constraint due to the RF PSN adopted in the hybrid MIMO architecture. To this end, we design a novel fully-connected channel compression network (CCN) as the encoder to compress the high dimensional channels, and the network parameters are defined as the real-valued phases of the PSN (i.e., beamforming/combining matrix in channel estimation).

- To reliably reconstruct the channels from the compressed measurements, we propose a channel reconstruction network (CRN) based on a developed MMV-LAMP network with the devised redundant dictionary as the decoder, which can exploit the *a priori* model and learn the optimal parameters from data to jointly recover multiple subcarriers' channels with significantly enhanced performance.

- To effectively estimate the channels from compressed feedback signals, a feedback based channel reconstruction network (FCRN) is proposed. The FCRN consists of a feedback reconstruction sub-network (FRSN) and a CRN. The FRSN is based on the MMV-LAMP network and can exploit the delay-domain sparsity of the channels to reliably reconstruct the compressed channel.

- Due to the mismatch between continuous AoAs/AoDs and the limited angle resolution of spatial-angular transform matrix, the resulted power leakage can weaken

the channel sparsity represented in the angle domain. Hence, by quantizing the angles with a finer resolution, we design a redundant dictionary to further improve the sparse channel estimation performance.

- To evaluate the superiority of the proposed solution that jointly trains the pilots and channel estimator, we further consider scenarios with fixed scattering environments. Simulation results verify that, by learning the characteristics of the data samples with fixed scattering, the optimized CCN and MMV-LAMP network can well match the channel environments with improved performance.

Notations: Throughout this chapter, scalar variables are denoted by normal-face letters, while boldface lower and upper-case symbols denote column vectors and matrices, respectively. Superscripts $(\cdot)^{\mathsf{T}}$, $(\cdot)^*$ and $(\cdot)^{\mathsf{H}}$ denote the transpose, conjugate and Hermitian transpose operators, respectively. $\|\mathbf{a}\|_0$ and $\|\mathbf{A}\|_F$ denote the ℓ_0-norm of $\mathbf{a}$ and the Frobenius norm of $\mathbf{A}$, respectively. $[\mathbf{a}]_m$ and $[\mathbf{A}]_{m,n}$ are the m-th element of $\mathbf{a}$ and the m-th row and the n-th colomn element of $\mathbf{A}$, respectively. $\mathbf{A}(m,:)$ and $\mathbf{A}(:,n)$ denote the m-th row vector and the n-th colomn vector of $\mathbf{A}$, respectively. $\mathbf{A}|_{\mathbf{\Omega}}$ denotes a sub-matrix by selecting the rows of $\mathbf{A}$ according to the ordered set $\mathbf{\Omega}$ and $\{\mathbf{\Omega}\}_m$ is the m-th element of the set $\mathbf{\Omega}$. $\mathbf{e}^{\mathsf{j}[\mathbf{\Xi}]}$ denotes a complex matrix with its element being $\left[\mathbf{e}^{\mathsf{j}[\mathbf{\Xi}]}\right]_{m,n} = e^{\mathsf{j}[\mathbf{\Xi}]_{m,n}}$, and $\mathbf{\Xi}$ is a real matrix. Finally, $\partial(\cdot)$ is the first-order partial derivative operation.

5.2 System Model

Consider a massive MIMO system that operates at high frequencies and uses hybrid beamforming, where the BS is equipped with a uniform linear array (ULA) and comprises N_{BS} antennas and N_{RF} RF chains, and the U users have a single-antenna. At the BS, the PSN is employed to connect a large number of antennas with a much fewer number of RF chains (i.e., $N_{\mathsf{BS}} \gg N_{\mathsf{RF}}$), and orthogonal frequency division multiplexing (OFDM) with K subcarriers is adopted to combat the frequency selective fading of the high frequency channels.

5.2.1 Uplink Channel Estimation for TDD Systems

Firstly, we consider the uplink channel estimation for TDD systems. The uplink channel estimation stage includes Q OFDM symbols (i.e., Q time slots) dedicated for channel estimation. For a certain user[1], in order to estimate the k-th subcarrier's channel, the received baseband signal vector $\mathbf{y}'_{\mathsf{UL}}[k,q] \in \mathbb{C}^{N_{\mathsf{RF}}\times 1}$ at the BS in the q-th time slot can be expressed as

$$\mathbf{y}'_{\mathsf{UL}}[k,q] = \mathbf{F}^{\mathsf{H}}_{\mathsf{UL}}[q]\,\mathbf{h}_{\mathsf{UL}}[k]\,x[k,q] + \bar{\mathbf{n}}'_{\mathsf{UL}}[k,q], \tag{5.1}$$

where $1 \le q \le Q$, $1 \le k \le K$, $\mathbf{F}_{\mathsf{UL}}[q] \in \mathbb{C}^{N_{\mathsf{BS}}\times N_{\mathsf{RF}}}$ denotes the uplink combining matrix at the BS, $\mathbf{h}_{\mathsf{UL}}[k] \in \mathbb{C}^{N_{\mathsf{BS}}\times 1}$ is the uplink k-th subcarrier channel, $x[k,q] \in \mathbb{C}$ is the

[1]Consider the uplink multi-user channel estimation, if U users adopt mutually orthogonal pilot signals, the pilot signals associated with different users can be distinguished and then respectively processed.

transmitted pilot symbol, and $\bar{\mathbf{n}}'_{\mathsf{UL}}[k,q] \sim \mathcal{CN}(0,\sigma_n^2\mathbf{I}_{N_{\mathsf{RF}}})$ is the effective noise modeled at the receiver (front-end) level.

Then, the received baseband signal is post-processed by multiplying it by $x^*[k,q]$, i.e.,

$$\mathbf{y}_{\mathsf{UL}}[k,q] = \mathbf{y}'_{\mathsf{UL}}[k,q]\,x^*[k,q] = \mathbf{F}_{\mathsf{UL}}^{\mathsf{H}}[q]\,\mathbf{h}_{\mathsf{UL}}[k] + \mathbf{n}_{\mathsf{UL}}[k,q], \tag{5.2}$$

where we assume that $x[k,q]\,x^*[k,q] = 1$ and $\mathbf{n}_{\mathsf{UL}}[k,q] = \bar{\mathbf{n}}'_{\mathsf{UL}}[k,q]\,x^*[k,q]$. Note that, due to the constant modulus constraint of the adopted fully-connected RF PSN at the BS, the uplink combining matrix $\mathbf{F}_{\mathsf{UL}}[q]$, $\forall q$, can be expressed as $\left[\mathbf{F}_{\mathsf{UL}}[q]\right]_{m,n} = \frac{1}{\sqrt{N_{\mathsf{BS}}}}e^{\mathrm{j}[\mathbf{\Xi}_{\mathsf{UL}}]_{m,n}}$ for $1 \le m \le N_{\mathsf{BS}}$, $1 \le n \le N_{\mathsf{RF}}$, and $[\mathbf{\Xi}_{\mathsf{UL}}]_{m,n}$ denotes the phase value connecting the m-th antenna and the n-th RF chain[2]. By collecting $\mathbf{y}_{\mathsf{UL}}[k,q]$ for $1 \le q \le Q$ together, the aggregate received signals $\mathbf{y}_{\mathsf{UL}}[k] \in \mathbb{C}^{M\times 1}$ ($M = QN_{\mathsf{RF}}$) can be written as

$$\mathbf{y}_{\mathsf{UL}}[k] = \mathbf{F}_{\mathsf{UL}}^{\mathsf{H}}\mathbf{h}_{\mathsf{UL}}[k] + \mathbf{n}_{\mathsf{UL}}[k], \tag{5.3}$$

where $\mathbf{y}_{\mathsf{UL}}[k] = \left[\mathbf{y}_{\mathsf{UL}}^{\mathsf{T}}[k,1],\cdots,\mathbf{y}_{\mathsf{UL}}^{\mathsf{T}}[k,Q]\right]^{\mathsf{T}}$, $\mathbf{F}_{\mathsf{UL}} = \left[\mathbf{F}_{\mathsf{UL}}[1],\cdots,\mathbf{F}_{\mathsf{UL}}[Q]\right] \in \mathbb{C}^{N_{\mathsf{BS}}\times M}$, and $\mathbf{n}_{\mathsf{UL}}[k] = \left[\mathbf{n}_{\mathsf{UL}}^{\mathsf{T}}[k,1],\cdots,\mathbf{n}_{\mathsf{UL}}^{\mathsf{T}}[k,Q]\right]^{\mathsf{T}} \in \mathbb{C}^{M\times 1}$. Finally, by stacking $\mathbf{y}_{\mathsf{UL}}[k]$ from all subcarriers, the received signals $\mathbf{y}_{\mathsf{UL}}[k]$ for $1 \le k \le K$ can be further expressed as

$$\mathbf{Y}_{\mathsf{UL}} = \mathbf{F}_{\mathsf{UL}}^{\mathsf{H}}\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf}} + \mathbf{N}_{\mathsf{UL}}, \tag{5.4}$$

where $\mathbf{Y}_{\mathsf{UL}} = \left[\mathbf{y}_{\mathsf{UL}}[1],\cdots,\mathbf{y}_{\mathsf{UL}}[K]\right] \in \mathbb{C}^{M\times K}$, $\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf}} = \left[\mathbf{h}_{\mathsf{UL}}[1],\cdots,\mathbf{h}_{\mathsf{UL}}[K]\right] \in \mathbb{C}^{N_{\mathsf{BS}}\times K}$ denotes the uplink spatial-frequency domain channel matrix, and have introduced the notation $\mathbf{N}_{\mathsf{UL}} = \left[\mathbf{n}_{\mathsf{UL}}[1],\cdots,\mathbf{n}_{\mathsf{UL}}[K]\right] \in \mathbb{C}^{M\times K}$.

5.2.2 Downlink Channel Estimation and Feedback for FDD Systems

Moreover, we consider the downlink channel estimation and feedback for FDD systems. Specifically, the downlink pilot signals transmitted by the BS can be denoted as $\mathbf{f}_{\mathsf{DL}}[q]\,s[k,q] \in \mathbb{C}^{N_{\mathsf{BS}}\times 1}$ for $1 \le q \le Q$, where $\mathbf{f}_{\mathsf{DL}}[q]$ is the RF pilot signal and $s[k,q]$ is the baseband pilot signal. Mathematically, the received signal in the q-th time slot associated with the k-th subcarrier at the user can be written as

$$y'_{\mathsf{DL}}[k,q] = \mathbf{h}_{\mathsf{DL}}^{\mathsf{T}}[k]\,\mathbf{f}_{\mathsf{DL}}[q]\,s[k,q] + \bar{n}_{\mathsf{DL}}[k,q], \tag{5.5}$$

where $\mathbf{h}_{\mathsf{DL}}[k] \in \mathbb{C}^{N_{\mathsf{BS}}\times 1}$ is the downlink k-th subcarrier's channel, and $\bar{n}_{\mathsf{DL}}[k,q]$ is the complex noise. Similar to (5.2), the received signal can be further post-processed to obtain

$$y_{\mathsf{DL}}[k,q] = y'_{\mathsf{DL}}[k,q]\,s^*[k,q] = \mathbf{h}_{\mathsf{DL}}^{\mathsf{T}}[k]\,\mathbf{f}_{\mathsf{DL}}[q] + n_{\mathsf{DL}}[k,q], \tag{5.6}$$

[2]Note that changing the phase values of the PSN does not require a re-synchronization of the whole system. This is because: i) The phase values of the phase shifts will be changed in the guard interval before each pilot OFDM symbol; ii) The synchronization of frame and symbol can be obtained based on the preambles transmitted before the pilot symbols; iii) When to adjust the phase shifts can be exactly calculated according to the synchronization information and the predefined signal frame structure, and the adjustment of the phase shifts can be controlled according to the system clock.

where we assume that $s[k,q]\,s^*[k,q] = 1$ and $n_{\mathrm{DL}}[k,q] = \bar{n}_{\mathrm{DL}}[k,q]\,s^*[k,q]$. Similarly, due to the constant modulus constraint of the adopted RF PSN, the RF pilot signal $\mathbf{f}_{\mathrm{DL}}[q]$, $\forall q$, can be expressed as $\left[\mathbf{f}_{\mathrm{DL}}[q]\right]_m = \frac{1}{\sqrt{N_{\mathrm{BS}}}} e^{\mathrm{j}[\boldsymbol{\Xi}_{\mathrm{DL}}]_m}$ for $1 \le m \le N_{\mathrm{BS}}$, and $[\boldsymbol{\Xi}_{\mathrm{DL}}]_m$ denotes the phase value connecting the m-th antenna and the activated RF chain. By collecting the received signals from Q time slots, the aggregate received signals can be expressed as

$$\mathbf{y}_{\mathrm{DL}}[k] = \mathbf{F}_{\mathrm{DL}}^{\mathsf{T}} \mathbf{h}_{\mathrm{DL}}[k] + \mathbf{n}_{\mathrm{DL}}[k], \tag{5.7}$$

where $\mathbf{y}_{\mathrm{DL}}[k] = \left[y_{\mathrm{DL}}[k,1], \cdots, y_{\mathrm{DL}}[k,Q]\right]^{\mathsf{T}} \in \mathbb{C}^{Q\times 1}$, $\mathbf{F}_{\mathrm{DL}} = \left[\mathbf{f}_{\mathrm{DL}}[1], \cdots, \mathbf{f}_{\mathrm{DL}}[Q]\right] \in \mathbb{C}^{N_{\mathrm{BS}}\times Q}$, and $\mathbf{n}_{\mathrm{DL}}[k] = \left[n_{\mathrm{DL}}[k,1], \cdots, n_{\mathrm{DL}}[k,Q]\right]^{\mathsf{T}} \in \mathbb{C}^{Q\times 1}$. Similar to (5.4), we can collect $\mathbf{y}_{\mathrm{DL}}[k]$ for $1 \le k \le K$ from K subcarriers to obtain

$$\mathbf{Y}_{\mathrm{DL}} = \mathbf{F}_{\mathrm{DL}}^{\mathsf{T}} \mathbf{H}_{\mathrm{DL}}^{\mathrm{sf}} + \mathbf{N}_{\mathrm{DL}}, \tag{5.8}$$

where $\mathbf{H}_{\mathrm{DL}}^{\mathrm{sf}} = \left[\mathbf{h}_{\mathrm{DL}}[1], \cdots, \mathbf{h}_{\mathrm{DL}}[K]\right] \in \mathbb{C}^{N_{\mathrm{BS}}\times K}$ denotes the downlink spatial-frequency domain channel matrix, $\mathbf{Y}_{\mathrm{DL}} = \left[\mathbf{y}_{\mathrm{DL}}[1], \cdots, \mathbf{y}_{\mathrm{DL}}[K]\right] \in \mathbb{C}^{Q\times K}$, and we introduce the notation $\mathbf{N}_{\mathrm{DL}} = \left[\mathbf{n}_{\mathrm{DL}}[1], \cdots, \mathbf{n}_{\mathrm{DL}}[K]\right] \in \mathbb{C}^{Q\times K}$.

5.2.3 Channel Model

According to typical high frequency channel models [54, 60, 62, 63, 65], the downlink delay-domain continuous channel vector $\mathbf{h}_{\mathrm{DL}}(\tau) \in \mathbb{C}^{N_{\mathrm{BS}}\times 1}$ can be expressed as

$$\mathbf{h}_{\mathrm{DL}}(\tau) = \sqrt{\frac{N_{\mathrm{BS}}}{L}} \sum_{l=1}^{L} \beta_l p(\tau - \tau_l) \mathbf{a}(\varphi_l), \tag{5.9}$$

where $\beta_l \sim \mathcal{CN}(0, \sigma_\alpha^2)$ and τ_l denote the propagation gain and delay corresponding to the l-th path, respectively, $p(\tau)$ is the pulse shaping filter, and φ_l is the angle-of-departure (AoD) of the l-th path at the BS. Moreover, the frequency-domain channel $\mathbf{h}_{\mathrm{DL}}[k]$ at the k-th subcarrier can be expressed as

$$\mathbf{h}_{\mathrm{DL}}[k] = \sqrt{\frac{N_{\mathrm{BS}}}{L}} \sum_{l=1}^{L} \beta_l e^{-\mathrm{j}\frac{2\pi k f_s \tau_l}{K}} \mathbf{a}(\varphi_l), \tag{5.10}$$

where f_s is the system sampling rate.

Since the BS is equipped with an ULA, the corresponding array steering vector $\mathbf{a}(\theta) \in \mathbb{C}^{N_{\mathrm{BS}}\times 1}$ can be written as

$$\mathbf{a}(\theta) = \frac{1}{\sqrt{N_{\mathrm{BS}}}} \left[1, e^{-\mathrm{j}\frac{2\pi d}{\lambda}\sin(\theta)}, \cdots, e^{-\mathrm{j}\frac{2\pi d}{\lambda}(N_{\mathrm{BS}}-1)\sin(\theta)}\right]^{\mathsf{T}}, \tag{5.11}$$

where λ is the carrier wavelength, and d is the adjacent antenna spacing usually satisfying $d = \lambda/2$.

5.3 MDDL-Based TDD Uplink Channel Estimation

In this section, we first propose an improved frame structure design for optimizing the channel estimation duration. Secondly, we propose to jointly train the RF PSN and the channel estimator as an auto-encoder. Finally, we develop an MMV-LAMP network, which can both exploit the structured sparsity from an *a priori* model and adaptively learn the trainable parameters from the data samples.

5.3.1 The Proposed Transmit Frame Structure Design

The proposed frame structure is illustrated in Fig. 5.1, where the cyclic prefix (CP)-OFDM is employed to combat the time dispersive channels and the time-frequency radio resources can be divided into multiple resource elements to convey the pilot signals and payload data. Specifically, a frame comprising T time slots is divided into two phases in the time domain, where the first Q time slots (i.e., pilot phase) are used to transmit pilot signals and the remaining $(T-Q)$ time slots (i.e., data transmission phase) are reserved only for payload data transmission. In the pilot phase, we denote the OFDM's DFT length as $P_L = N_{\text{cp}}$, where N_{cp} is the length of CP. Therefore, the subcarrier spacing is B_s/P_L and each CP-OFDM symbol duration is $(N_{\text{cp}}+P_L)/B_s$, where B_s is the system bandwidth. On the other hand, in the data transmission phase, we consider the OFDM symbol's DFT length is $D_L \gg P_L$ and thus each CP-OFDM symbol duration is $(N_{\text{cp}}+D_L)/B_s$.

5.3.2 The Developed MMV-LAMP Network

In this section, we will detail the developed MMV-LAMP network. Without loss of generality, we consider a typical MMV CS problem

$$\mathbf{Y} = \mathbf{A}\mathbf{X} + \mathbf{N}, \tag{5.12}$$

where $\mathbf{Y} \in \mathbb{C}^{M\times K}$ is a noisy measurement, $\mathbf{A} \in \mathbb{C}^{M\times N}$ is a measurement matrix, $\mathbf{X} \in \mathbb{C}^{N\times K}$ is a sparse matrix whose columns $\{\mathbf{X}(:,i)\}_{i=1}^{K}$ share a common sparsity, and $\mathbf{N} \in \mathbb{C}^{M\times K}$ is the additive white Gaussian noise (AWGN).

To solve the MMV CS problem in (5.12) efficiently, the developed MMV-LAMP network has two features: i) it fully exploits an *a priori* model, i.e., the structured sparsity of $\mathbf{X}$; ii) by integrating the trainable parameters into the unfolded iterations of conventional AMP algorithms, it can adaptively learn and optimize the network from data samples. Specifically, as for the t-th layer $(1 \le t \le T)$ of the developed MMV-LAMP network, the key procedure includes

$$\mathbf{R}_t = \widehat{\mathbf{X}}_{t-1} + \mathbf{B}\mathbf{V}_{t-1}, \tag{5.13a}$$

$$\widehat{\mathbf{X}}_t = \eta(\mathbf{R}_t; \boldsymbol{\theta}, \sigma_t), \tag{5.13b}$$

$$\mathbf{V}_t = \mathbf{Y} - \mathbf{A}\widehat{\mathbf{X}}_t + b_t\mathbf{V}_{t-1}, \tag{5.13c}$$

where $\mathbf{V}_0 = \mathbf{Y}$, $\widehat{\mathbf{X}}_0 = \mathbf{0}$, and

$$\sigma_t = \frac{1}{\sqrt{MK}}\|\mathbf{V}_{t-1}\|_F, \tag{5.14}$$

$$b_t\mathbf{I} = \frac{1}{M}\sum_{j=1}^{N}\frac{\partial\left[\eta(\mathbf{R}_t;\boldsymbol{\theta},\sigma_t)\right]_j}{\partial\left[\mathbf{R}_t(j,:)\right]}. \tag{5.15}$$

Note that, the residual $\mathbf{V}_t$ in (5.13c) includes the "Onsager correction" term $b_t\mathbf{V}_{t-1}$, which is introduced into the conventional AMP algorithms to accelerate the convergence [89]. Moreover, the shrinkage function $\eta(\cdot;\cdot)$ can be expressed as

$$\left[\eta(\mathbf{R}_t;\boldsymbol{\theta},\sigma_t)\right]_j = \frac{\mathbf{r}_{t,j}}{\pi_t\left[1+\exp(\psi_t - \frac{\mathbf{r}_{t,j}^{\mathsf{H}}\mathbf{r}_{t,j}}{2\sigma_t^2\pi_t})\right]}, \tag{5.16}$$

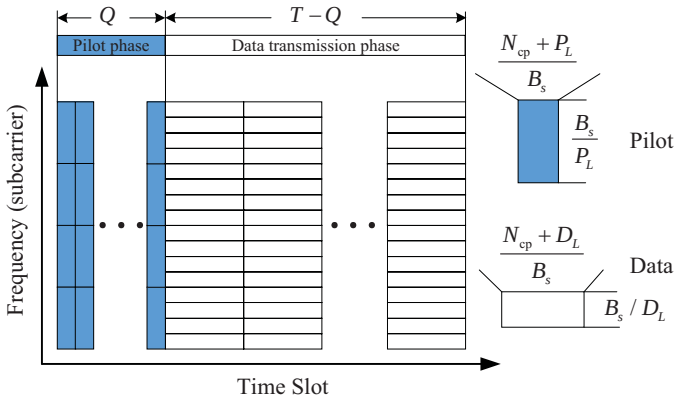


Figure 5.1: The proposed frame structure for the communication transmission.

where $\mathbf{r}_{t,j} = \mathbf{R}_t(j,:)$ denotes the j-th row of the $\mathbf{R}_t$, π_t and ψ_t are respectively given by

$$\pi_t = 1 + \frac{\sigma_t^2}{\theta_1}, \tag{5.17}$$

$$\psi_t = K \log(1 + \frac{\theta_1}{\sigma_t^2}) + \theta_2. \tag{5.18}$$

Note that, different from the learned denoising-based approximate message passing (LDAMP) network [36], where the authors replaced the denoiser module $D_{\hat{\sigma}^l}(\cdot)$ in the DAMP algorithm with the denoising convolutional neural network (DnCNN), we derive the shrinkage function $\eta(\cdot;\cdot)$ in detail, which plays the role of the nonlinear activation function in deep learning. Moreover, instead of processing the element r in the existing MDDL-based scheme [89–92], the developed MMV-LAMP network processes the row vector $\mathbf{r}_j$ for $1 \leq j \leq N$ by fully exploiting the structured sparsity of $\mathbf{X}$ from the *a priori* model. The derivation of the developed MMV-LAMP network is shown in Appendix. Additionally, in order to avoid the performance degradation caused by the mismatch between the continuous angles and the discrete dictionary, we integrate the redundant dictionary into the CRN (i.e., the decoder) to improve the channel estimation performance.

Fig. 5.2 illustrates the t-th layer architecture of the developed MMV-LAMP network. In Fig. 5.2, the inputs are $\widehat{\mathbf{X}}_{t-1} \in \mathbb{C}^{N\times K}$, $\mathbf{V}_{t-1} \in \mathbb{C}^{M\times K}$, and $\mathbf{Y} \in \mathbb{C}^{M\times K}$, where $\widehat{\mathbf{X}}_{t-1}$ and $\mathbf{V}_{t-1}$ are the outputs of the previous $(t-1)$-th layer, and $\mathbf{Y}$ is the noisy measurement in (5.12). Moreover, at the network training stage, we define $\mathbf{B} \in \mathbb{C}^{N\times M}$ and $\boldsymbol{\theta} = \{\theta_1, \theta_2\}$ as the trainable parameters of the MMV-LAMP network, which are identical for all T layers.

5.3.3 MMV-LAMP Network Based Uplink Channel Estimation

The block diagram of the proposed MMV-LAMP network based uplink channel estimation scheme is depicted in Fig. 5.3, which contains the CCN and CRN. The CCN corresponds to the combining matrix $\mathbf{F}_{\text{UL}}^{\text{H}}$ (encoder) in (5.4), and the CRN corresponds to the channel estimator (decoder) at the BS. Specifically, the input and output of the CCN are the uplink spatial-frequency domain channel matrix $\mathbf{H}_{\text{UL}}^{\text{sf}}$ and the noiseless pilot signals received at the BS. Note that the parameters of the CCN $\{\Xi_{\text{UL}}\}$ corresponds to the phase values of the combining matrix $\mathbf{F}_{\text{UL}}^{\text{H}}$ in (5.4). Moreover, the CRN consists of

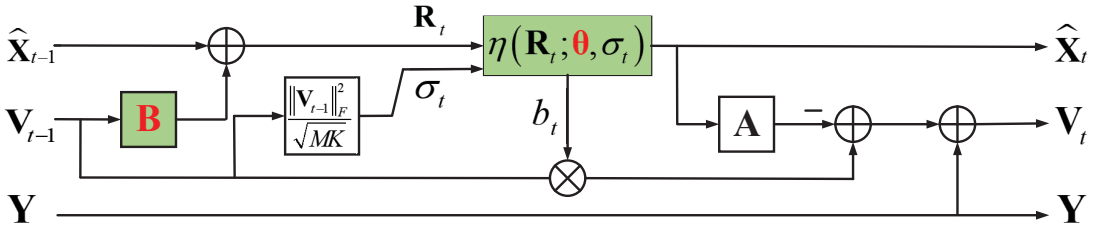


Figure 5.2: The t-th layer architecture of the developed MMV-LAMP network with the trainable parameters $\{\mathbf{B}, \boldsymbol{\theta}\}$.

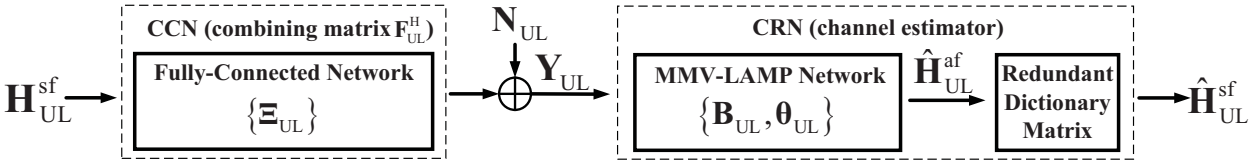


Figure 5.3: The block diagram of the proposed MDDL-based uplink channel estimation solution, which includes a CCN and an MMV-LAMP network based CRN.

T layers and each has the same network structure and trainable parameters $\{\mathbf{B}_{\text{UL}}, \boldsymbol{\theta}_{\text{UL}}\}$ as shown in Fig. 5.2, whose output is the estimated angle-frequency domain channel matrix $\widehat{\mathbf{H}}_{\text{UL}}^{\text{af}}$. At the offline training stage, we jointly train the overall network parameters $\{\boldsymbol{\Xi}_{\text{UL}}, \mathbf{B}_{\text{UL}}, \boldsymbol{\theta}_{\text{UL}}\}$ as an auto-encoder in an end-to-end approach. Finally, the estimated spatial-frequency domain channel matrix $\widehat{\mathbf{H}}_{\text{UL}}^{\text{sf}}$ can be obtained by multiplying a devised redundant dictionary matrix.

Fully-Connected CCN

Consider the formula $\mathbf{Y}_{\text{UL}} = \mathbf{F}_{\text{UL}}^{\text{H}} \mathbf{H}_{\text{UL}}^{\text{sf}} + \mathbf{N}_{\text{UL}}$ in (5.4), in order to mimic the linear compressibility process of high-dimensional channels, the combining matrix $\mathbf{F}_{\text{UL}}$ can be well modeled as a CCN realized by a fully-connected layer without biases and a nonlinear activation function. Note that the combining matrix $\mathbf{F}_{\text{UL}}$ is a complex-valued matrix and satisfies the constant modulus constraint due to the RF PSN adopted in the hybrid MIMO architecture, and the expression of the combining matrix $\mathbf{F}_{\text{UL}}$ is given by

$$\mathbf{F}_{\text{UL}} = \frac{1}{\sqrt{N_{\text{BS}}}} \exp(\mathrm{j}\boldsymbol{\Xi}_{\text{UL}}) = \frac{1}{\sqrt{N_{\text{BS}}}} \left[\cos(\boldsymbol{\Xi}_{\text{UL}}) + \mathrm{j}\sin(\boldsymbol{\Xi}_{\text{UL}})\right], \tag{5.19}$$

where $\mathrm{j} = \sqrt{-1}$ and $[\boldsymbol{\Xi}_{\text{UL}}]_{m,n} \in [0, 2\pi)$. As it is well known that complex-valued outputs are not well supported by most deep learning frameworks (e.g., Tensorflow, Pytorch), it would be difficult to directly train the complex-valued combining matrix $\mathbf{F}_{\text{UL}}$. Hence, for the fully-connected CCN, we choose to train the real-valued phases of PSN $\{\boldsymbol{\Xi}_{\text{UL}}\}$, in other words, we define the real-valued phases of PSN as the real-valued trainable parameters of the fully-connected CCN. Moreover, the structure of the proposed fully-connected CCN is shown in Fig. 5.4, where the trainable parameter of the CCN is $\{\boldsymbol{\Xi}_{\text{UL}}\}$ and the corresponding weight matrix of the CCN is $\exp(\mathrm{j}\boldsymbol{\Xi}_{\text{UL}})/\sqrt{N_{\text{BS}}}$. Hence, the parameters of the fully-connected layer are regarded as the phases of PSN and can be learned at the deep learning training stage.

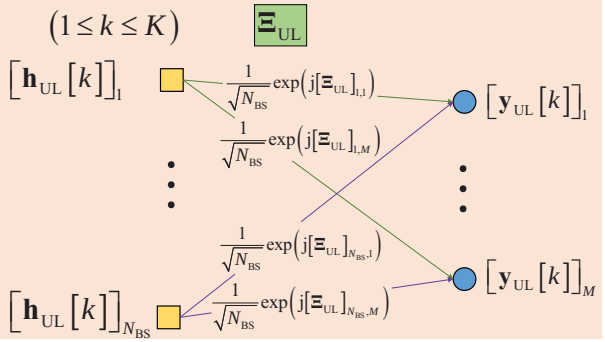


Figure 5.4: The proposed fully-connected CCN with the trainable parameters $\{\Xi_{\mathrm{UL}}\}$, which correspond to the combining matrix $\mathbf{F}_{\mathrm{UL}}^{\mathrm{H}}$.

CRN Based on MMV-LAMP Network

First, we detail the devised redundant dictionary matrix. Specifically, massive MIMO channels are sparse in the angle-domain, and the accuracy of CRN depends heavily on their sparsity, which may be weakened by the power leakage [66]. Therefore, we design a redundant dictionary matrix $\mathbf{D}$ with a finer angular resolution to transform the spatial-frequency domain channel matrix $\mathbf{H}^{\mathrm{sf}}$ into the angle-frequency domain channel matrix $\mathbf{H}^{\mathrm{af}}$, which can be expressed as

$$\mathbf{H}^{\mathrm{sf}} = \mathbf{D}^{\mathrm{H}}\mathbf{H}^{\mathrm{af}}, \tag{5.20}$$

where the redundant dictionary matrix $\mathbf{D} \in \mathbb{C}^{G\times N_{\mathrm{BS}}}$ consists of G column vectors $\mathbf{a}(\phi_g)$ for $1 \le g \le G$, i.e., $\mathbf{D} = \left[\mathbf{a}(\phi_1), \mathbf{a}(\phi_2), \cdots, \mathbf{a}(\phi_G)\right]^{\mathsf{T}}$, with $\mathbf{a}(\phi_g)$ the corresponding array steering vector in (5.11), where the sine function in the array steering vector is defined as $\sin(\phi_g) = -1 + 2(g-1)/G$ by quantifying the range of AoDs into G grids for $g = 1, 2, \cdots, G$. Therefore, estimating $\mathbf{H}_{\mathrm{UL}}^{\mathrm{sf}}$ in (5.4) is equivalent to estimating $\mathbf{H}_{\mathrm{UL}}^{\mathrm{af}}$ represented in the angle-domain redundant dictionary $\mathbf{D}$, i.e.,

$$\mathbf{Y}_{\mathrm{UL}} = \mathbf{F}_{\mathrm{UL}}^{\mathrm{H}}\mathbf{D}^{\mathrm{H}}\mathbf{H}_{\mathrm{UL}}^{\mathrm{af}} + \mathbf{N}_{\mathrm{UL}} = \mathbf{A}_{\mathrm{UL}}\mathbf{H}_{\mathrm{UL}}^{\mathrm{af}} + \mathbf{N}_{\mathrm{UL}}, \tag{5.21}$$

where $\mathbf{A}_{\mathrm{UL}} = \mathbf{F}_{\mathrm{UL}}^{\mathrm{H}}\mathbf{D}^{\mathrm{H}}$ is the effective measurement matrix. It's worth noting that the k-th column of $\mathbf{H}_{\mathrm{UL}}^{\mathrm{af}}$, i.e., $\mathbf{H}_{\mathrm{UL}}^{\mathrm{af}}(:,k)$ is a sparse column vector, meanwhile, $\left\{\mathbf{H}_{\mathrm{UL}}^{\mathrm{af}}(:,k)\right\}_{k=1}^{K}$ share the common sparsity [63]. Consequently, the sparse channel estimation problem can be formulated as an MMV sparse matrix recovery problem in CS. Noth that, given the received signals, the channel matrix $\mathbf{H}_{\mathrm{UL}}^{\mathrm{af}}$ can be estimated by solving the following optimization problem

$$\begin{aligned} &\min_{\mathbf{H}_{\mathrm{UL}}^{\mathrm{af}}} \left(\sum_{k=1}^{K}\left\|\mathbf{H}_{\mathrm{UL}}^{\mathrm{af}}(:,k)\right\|_0^2\right)^{1/2} \\ &\text{s.t.} \left\|\mathbf{Y}_{\mathrm{UL}} - \mathbf{A}_{\mathrm{UL}}\mathbf{H}_{\mathrm{UL}}^{\mathrm{af}}\right\|_F \le \delta, \\ &\text{and } \left\{\mathbf{H}_{\mathrm{UL}}^{\mathrm{af}}(:,k)\right\}_{k=1}^{K} \text{ share the common sparse support set,} \end{aligned} \tag{5.22}$$

where $\left\|\mathbf{H}_{\mathrm{UL}}^{\mathrm{af}}(:,k)\right\|_0$ is the number of non-zero elements of $\mathbf{H}_{\mathrm{UL}}^{\mathrm{af}}(:,k)$ and δ is the error tolerance parameter. By replacing the l_0-norm with the l_1-norm, various CS algorithms

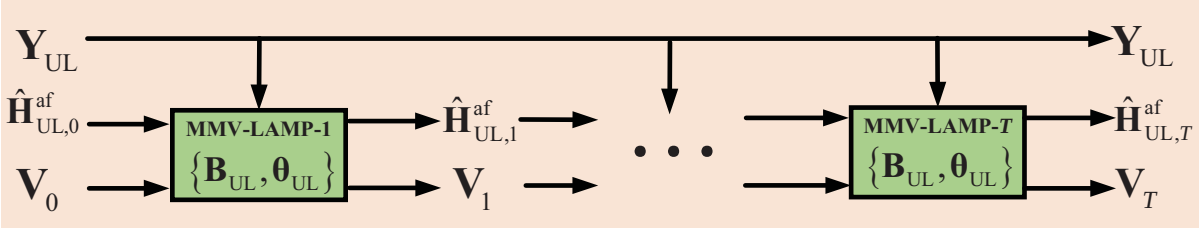


Figure 5.5: The proposed CRN based on MMV-LAMP network with the trainable parameters $\{\mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}$.

Algorithm 1 MMV-LAMP network based CRN

Input: The received signal $\mathbf{Y}_{\mathrm{UL}}$, the measurement matrix $\mathbf{A}_{\mathrm{UL}}$, the number of layers T.

Output: The output of the T-th layer MMV-LAMP network $\widehat{\mathbf{H}}^{\mathrm{sf}}_{\mathrm{UL}} = \mathbf{D}^{\mathrm{H}}\widehat{\mathbf{H}}^{\mathrm{af}}_{\mathrm{UL},T}$.

1: Initialization: $\mathbf{V}_0 = \mathbf{Y}_{\mathrm{UL}}$, $\widehat{\mathbf{H}}^{\mathrm{af}}_{\mathrm{UL},0} = \mathbf{0}$, $\mathbf{B}_{\mathrm{UL}} = \mathbf{A}^{\mathrm{H}}_{\mathrm{UL}}$, $\boldsymbol{\theta}_{\mathrm{UL}} = \{1, 1\}$.
2: **for** $t = 1, 2, \cdots, T$ **do**
3: $\quad \mathbf{R}_t = \widehat{\mathbf{H}}^{\mathrm{af}}_{\mathrm{UL},t-1} + \mathbf{B}_{\mathrm{UL}}\mathbf{V}_{t-1}$
4: $\quad \sigma_t = \frac{1}{\sqrt{QK}}\|\mathbf{V}_{t-1}\|_F$
5: $\quad \widehat{\mathbf{H}}^{\mathrm{af}}_{\mathrm{UL},t} = \eta(\mathbf{R}_t; \boldsymbol{\theta}_{\mathrm{UL}}, \sigma_t)$
6: $\quad b_t = \frac{1}{Q}\sum_{j=1}^{G}\frac{\partial[\eta(\mathbf{R}_t;\boldsymbol{\theta}_{\mathrm{UL}},\sigma_t)]_j}{\partial[\mathbf{R}_t(j,:)]}$
7: $\quad \mathbf{V}_t = \mathbf{Y}_{\mathrm{UL}} - \mathbf{A}_{\mathrm{UL}}\widehat{\mathbf{H}}^{\mathrm{af}}_{\mathrm{UL},t} + b_t\mathbf{V}_{t-1}$
8: **end for**

Figure 5.6: MMV-LAMP network based CRN

can be utilized to solve the problem, such as the simultaneous orthogonal matching pursuit (SOMP) algorithm [93], the MMV-AMP algorithm [94], and the proposed MMV-LAMP algorithm. However, these greedy CS algorithms cannot achieve satisfactory channel estimation accuracy.

To efficiently solve the MMV CS problem in (5.22), we further develop an MMV-LAMP network with T layers as illustrated in Fig. 5.5 and summarized in Algorithm 5.6, which can reconstruct the high-dimensional angle-frequency domain channel matrix $\widehat{\mathbf{H}}^{\mathrm{af}}_{\mathrm{UL}}$ from the low-dimensional received signals $\mathbf{Y}_{\mathrm{UL}}$. Specifically, the input is the received signals $\mathbf{Y}_{\mathrm{UL}}$ and the output of the t-th layer is the estimated angle-frequency domain channel matrix $\widehat{\mathbf{H}}^{\mathrm{af}}_{\mathrm{UL},t}$. In addition, the initial values $\widehat{\mathbf{H}}^{\mathrm{af}}_{\mathrm{UL},0}$ and $\mathbf{V}_0$ are denoted as $\widehat{\mathbf{H}}^{\mathrm{af}}_{\mathrm{UL},0} = \mathbf{0}$ and $\mathbf{V}_0 = \mathbf{Y}_{\mathrm{UL}}$, respectively.

As for the t-th layer, the trainable parameter $\{\mathbf{B}_{\mathrm{UL}}\}$ is denoted as

$$\mathbf{B}_{\mathrm{UL}} = \mathrm{Re}\{\mathbf{B}_{\mathrm{UL}}\} + \mathrm{j}\,\mathrm{Im}\{\mathbf{B}_{\mathrm{UL}}\}, \tag{5.23}$$

where $\mathrm{Re}\{\mathbf{B}_{\mathrm{UL}}\}$ and $\mathrm{Im}\{\mathbf{B}_{\mathrm{UL}}\}$ denote the real and imaginary parts of the trainable parameter $\mathbf{B}_{\mathrm{UL}}$, respectively. In other words, in order to achieve mathematical complex-valued processing in the MMV-LAMP network, we define two real-valued trainable parameters $\mathrm{Re}\{\mathbf{B}_{\mathrm{UL}}\}$ and $\mathrm{Im}\{\mathbf{B}_{\mathrm{UL}}\}$ to form the complex-valued trainable parameters

$\{\mathbf{B}_{\mathsf{UL}}\}$. Finally, the final estimated spatial-frequency domain channel matrix based on the output of the T-th layer is given by

$$\widehat{\mathbf{H}}_{\mathsf{UL}}^{\mathsf{sf}} = \mathbf{D}^{\mathsf{H}}\widehat{\mathbf{H}}_{\mathsf{UL},T}^{\mathsf{af}}. \tag{5.24}$$

Learning Strategy

Inspired by the auto-encoder, we propose a novel layer-by-layer learning strategy to jointly train the CCN (encoder) and CRN (decoder). Specifically, at the offline training stage, we first generate the training data set $\left\{\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf},n}\right\}_{n=1}^{N_{\mathsf{train}}}$ according to (5.10), where N_{train} is the number of channel samples in the training set, and $\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf},n}$ is not only the input of the CCN, but also the corresponding target output. In order to jointly optimize the trainable parameters of the CCN and CRN, i.e., $\{\mathbf{\Xi}_{\mathsf{UL}}, \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}}\}$, we define the normalized mean square error (NMSE) between the target value $\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf},n}$ and the estimated spatial-frequency channel matrix of the t-th layer MMV-LAMP network $\widehat{\mathbf{H}}_{\mathsf{UL},t}^{\mathsf{sf},n} = \mathbf{D}^{\mathsf{H}}\widehat{\mathbf{H}}_{\mathsf{UL},t}^{\mathsf{af},n}$ as the loss function of the t-th layer [3], i.e.,

$$\begin{aligned} &L_{\mathsf{UL},t}\left(\left\{\mathbf{\Xi}_{\mathsf{UL}}, \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}}\right\}_t\right) \\ &= \sum_{n=1}^{N} \frac{\left\|\widehat{\mathbf{H}}_{\mathsf{UL},t}^{\mathsf{sf},n} - \mathbf{H}_{\mathsf{UL}}^{\mathsf{sf},n}\right\|_F^2}{\left\|\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf},n}\right\|_F^2} \\ &= \sum_{n=1}^{N} \frac{\left\|\mathbf{D}^{\mathsf{H}} f_t(\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf},n}, \left\{\mathbf{\Xi}_{\mathsf{UL}}, \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}}\right\}_t) - \mathbf{H}_{\mathsf{UL}}^{\mathsf{sf},n}\right\|_F^2}{\left\|\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf},n}\right\|_F^2}, \end{aligned} \tag{5.25}$$

where N is the number of data samples in each batch of the training set, $\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf},n}$ is the n-th uplink spatial-frequency domain channel sample, and $\widehat{\mathbf{H}}_{\mathsf{UL},t}^{\mathsf{af},n} = f_t(\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf},n}, \{\mathbf{\Xi}_{\mathsf{UL}}, \mathbf{B}_{\mathsf{UL}}, \theta_{\mathsf{UL}}\}_t)$ is the output of the t-th layer MMV-LAMP network. Note that $f_t(\cdot,\cdot)$ indicates the proposed uplink channel estimation solution including the CCN and CRN, where the CRN is iterated t times in the t-th layer, i.e., we have the following definition $f_t(\cdot,\cdot) = f^{\mathsf{CRN}}(\cdots f^{\mathsf{CRN}}(f^{\mathsf{CCN}}(\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf},n}, \mathbf{\Xi}_{\mathsf{UL}}), \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}}), \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}})$.

The proposed novel layer-by-layer learning strategy is summarized in Algorithm 2, and the Adam algorithm with the learning rate 0.001 is adopted [95]. Specifically, for the 1-st layer, the input data is the target data $\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf}}$ and the output is $\widehat{\mathbf{H}}_{\mathsf{UL},1}^{\mathsf{sf}} = \mathbf{D}^{\mathsf{H}} f^{\mathsf{CRN}}(f^{\mathsf{CCN}}(\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf}}, \mathbf{\Xi}_{\mathsf{UL}}), \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}})$, where $f^{\mathsf{CCN}}(\cdot,\cdot)$ and $f^{\mathsf{CRN}}(\cdot,\cdot,\cdot)$ denote the proposed CCN and CRN structure, respectively. Therefore, we aim to optimize the 1-st layer's trainable parameters $\{\mathbf{\Xi}_{\mathsf{UL}}, \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}}\}_1$ by minimizing the loss function of the 1-st layer $L_{\mathsf{UL},1}\left(\{\mathbf{\Xi}_{\mathsf{UL}}, \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}}\}_1\right)$. For the 2-nd layer, the input data is $\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf}}$, but the output is $\widehat{\mathbf{H}}_{\mathsf{UL},2}^{\mathsf{sf}} = \mathbf{D}^{\mathsf{H}} f^{\mathsf{CRN}}(f^{\mathsf{CRN}}(f^{\mathsf{CCN}}(\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf}}, \mathbf{\Xi}_{\mathsf{UL}}), \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}}), \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}})$. The trainable parameters and the loss function of the 2-nd layer are $\{\mathbf{\Xi}_{\mathsf{UL}}, \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}}\}_2$ and $L_{\mathsf{UL},2}\left(\{\mathbf{\Xi}_{\mathsf{UL}}, \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}}\}_2\right)$, respectively. The initial values of the 2-nd layer's trainable parameters $\{\mathbf{\Xi}_{\mathsf{UL}}, \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}}\}_2$ are the obtained parameters $\{\mathbf{\Xi}_{\mathsf{UL}}, \mathbf{B}_{\mathsf{UL}}, \boldsymbol{\theta}_{\mathsf{UL}}\}_1$ from the 1-st layer's training. Similarly, for T-th layer, the input data is still the target data $\mathbf{H}_{\mathsf{UL}}^{\mathsf{sf}}$, and the output is $\widehat{\mathbf{H}}_{\mathsf{UL},T}^{\mathsf{sf}}$,

[3] The ultimate goal of the proposed MDDL-based channel estimation scheme is to obtain better NMSE performance, hence, we choose the NMSE rather than the MSE as the loss function.

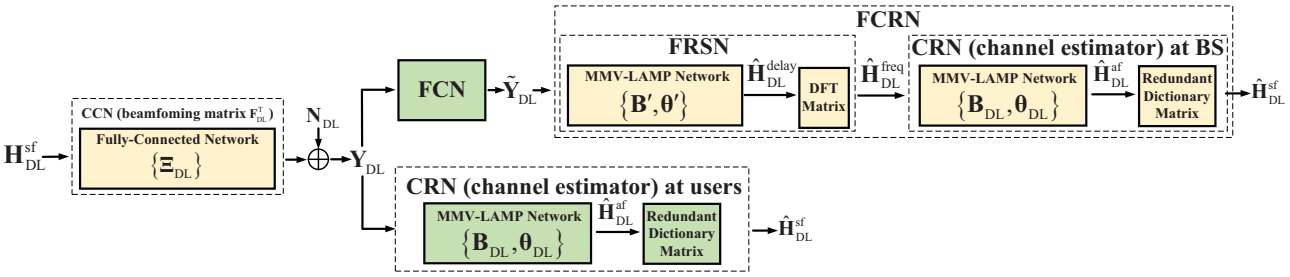


Figure 5.7: The block diagram of the proposed MDDL-based downlink channel estimation and feedback solution, where the green and yellow block diagrams represent that the modules are processed at the users and the BS, respectively.

Algorithm 2 Learning strategy to jointly train CCN's parameters $\{\boldsymbol{\Xi}_{\mathrm{UL}}\}$ and CRN's parameters $\{\mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}$

1: Initialization: $\mathbf{B}_{\mathrm{UL}} = \mathbf{A}_{\mathrm{UL}}^{\mathrm{H}}$, $\boldsymbol{\theta}_{\mathrm{UL}} = \{1, 1\}$, $[\boldsymbol{\Xi}_{\mathrm{UL}}]_{i,j} \in [0, 2\pi)$, $\{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}_0 = \{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}$.
2: **for** $t = 1, 2, \cdots, T$ **do**
3: Initialize $\{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}_t$ as $\{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}_{t-1}$
4: Learn $\{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}_t$ to minimize the loss function of t-th layer $L_{\mathrm{UL},t}(\{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}_t)$
5: **end for**
Output: $\{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\} = \{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}_T$.

Figure 5.8: Learning strategy to jointly train CCN's parameters $\{\boldsymbol{\Xi}_{\mathrm{UL}}\}$ and CRN's parameters $\{\mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}$

where $\widehat{\mathbf{H}}_{\mathrm{UL},T}^{\mathrm{sf}} = \mathbf{D}^{\mathrm{H}} f^{\mathrm{CRN}}(\cdots f^{\mathrm{CRN}}(f^{\mathrm{CCN}}(\mathbf{H}_{\mathrm{UL}}^{\mathrm{sf}}, \boldsymbol{\Xi}_{\mathrm{UL}}), \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}), \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}})$. The trainable parameters and the loss function of the T-th layer are $\{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}_T$ as well as $L_{\mathrm{UL},T}(\{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}_T)$, respectively. Note that, the initial values of the T-th layer's trainable parameters are the obtained parameters $\{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}_{T-1}$ from the $(T-1)$-th layer's training. In other words, the proposed layer-by-layer training strategy combines both the conventional layer-by-layer and all-layer training strategy. Consider the t-th layer training $(1 \leq t \leq T)$, we adopt the all-layer training strategy, i.e., the trainable parameters $\{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}_t$ are jointly optimized; while from the perspective of T times training with the increasing layer number t, it is a modified kind of layer-by-layer training strategy. After the trainable parameters $\{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}_T$ of the T-th layer are optimized, we can obtain the complex-valued PSN and the MMV-LAMP network simultaneously, which can be adopted to design the combining matrix $\mathbf{F}_{\mathrm{UL}}^{\mathrm{H}}$ and the channel estimator at the online channel estimation stage.

5.4 MDDL-Based FDD Downlink Channel Estimation and Feedback

In this section, we first extend the proposed MDDL-based TDD uplink channel estimation scheme to the FDD downlink channel estimation. Moreover, since the up-

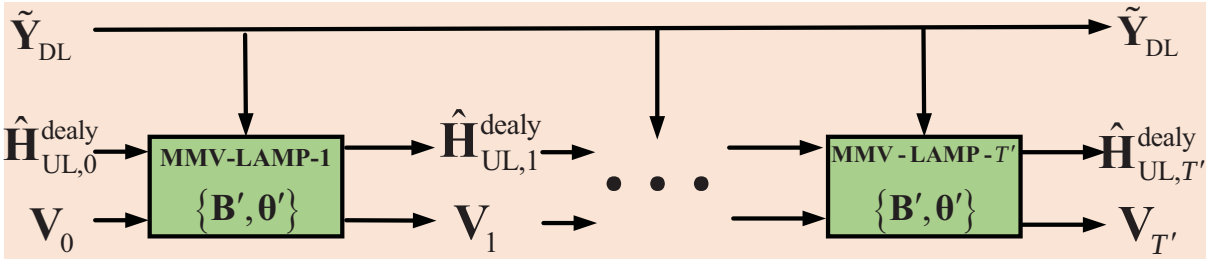


Figure 5.9: The proposed FRSN based on MMV-LAMP network with the trainable parameters $\{\mathbf{B}', \boldsymbol{\theta}'\}$.

link/downlink channel reciprocity does not hold in FDD systems, we further propose an MMV-LAMP network based channel feedback solution, whereby the channels' delay-domain sparsity is exploited for reducing the feedback overhead. As shown in Fig. 5.7, the block diagram of the proposed downlink channel estimation and feedback solution contains a CCN at the BS, a feedback compression network (FCN) at the users, and a FCRN at the BS, where the FCRN further consists of a FRSN and a CRN (this CRN is the same as that in Fig. 5.3).

5.4.1 MMV-LAMP Network Based Downlink Channel Estimation

Similar to Section 5.3, estimating the $\mathbf{H}^{\mathrm{sf}}_{\mathrm{DL}}$ in (5.8) is equivalent to estimating $\mathbf{H}^{\mathrm{af}}_{\mathrm{DL}}$ represented in the angle-domain redundant dictionary $\mathbf{D}$, i.e.,

$$\mathbf{Y}_{\mathrm{DL}} = \mathbf{F}^{\mathsf{T}}_{\mathrm{DL}}\mathbf{D}^{\mathsf{H}}\mathbf{H}^{\mathrm{af}}_{\mathrm{DL}} + \mathbf{N}_{\mathrm{DL}} = \mathbf{A}_{\mathrm{DL}}\mathbf{H}^{\mathrm{af}}_{\mathrm{DL}} + \mathbf{N}_{\mathrm{DL}}, \tag{5.26}$$

where $\mathbf{A}_{\mathrm{DL}} = \mathbf{F}^{\mathsf{T}}_{\mathrm{DL}}\mathbf{D}^{\mathsf{H}} \in \mathbb{C}^{Q\times G}$ is the measurement matrix in CS. We can observe that (5.21) and (5.26) share a similar expression, hence, the proposed MMV-LAMP network for uplink channel estimation scheme at the BS can be used for the downlink channel estimation at the users.

Specifically, at the downlink channel estimation stage, the input of the CCN is the downlink spatial-frequency domain channel matrix $\mathbf{H}^{\mathrm{sf}}_{\mathrm{DL}}$ and the output is the received signals $\mathbf{Y}_{\mathrm{DL}}$ at the user. Similar to Section 5.3, the trainable parameters of the CCN can be regarded as the real-valued phases of PSN, and the corresponding expression of the complex-valued beamforming matrix $\mathbf{F}_{\mathrm{DL}}$ is given by

$$\mathbf{F}_{\mathrm{DL}} = \frac{1}{\sqrt{N_{\mathrm{BS}}}}\mathrm{e}^{\mathrm{j}[\boldsymbol{\Xi}_{\mathrm{DL}}]} = \frac{1}{\sqrt{N_{\mathrm{BS}}}}\left[\cos(\boldsymbol{\Xi}_{\mathrm{DL}}) + \mathrm{j}\sin(\boldsymbol{\Xi}_{\mathrm{DL}})\right]. \tag{5.27}$$

On the other hand, for the CRN at the users, we replace the inputs $\mathbf{Y}_{\mathrm{UL}}$ and $\mathbf{A}_{\mathrm{UL}}$ with $\mathbf{Y}_{\mathrm{DL}}$ and $\mathbf{A}_{\mathrm{DL}}$, respectively, and the other terms are the same as in the Algorithm 5.6. As for the learning strategy, we train the trainable parameters $\{\boldsymbol{\Xi}_{\mathrm{DL}}, \mathbf{B}_{\mathrm{DL}}, \boldsymbol{\theta}_{\mathrm{DL}}\}$ in the downlink channel estimation based on Algorithm 5.8 by replacing the inputs $\mathbf{A}_{\mathrm{UL}}$ and $\{\boldsymbol{\Xi}_{\mathrm{UL}}, \mathbf{B}_{\mathrm{UL}}, \boldsymbol{\theta}_{\mathrm{UL}}\}$ with $\mathbf{A}_{\mathrm{DL}}$ and $\{\boldsymbol{\Xi}_{\mathrm{DL}}, \mathbf{B}_{\mathrm{DL}}, \boldsymbol{\theta}_{\mathrm{DL}}\}$, respectively.

5.4.2 MMV-LAMP Network Based Channel Feedback

Given the received signal $\mathbf{Y}_{\mathrm{DL}}$ in (5.8) at the users, the received signals can be rewritten as

$$\mathbf{Y}^{\mathsf{T}}_{\mathrm{DL}} = \mathbf{H}^{\mathrm{fs}}_{\mathrm{DL}}\mathbf{F}_{\mathrm{DL}} + \mathbf{N}^{\mathsf{T}}_{\mathrm{DL}} = \mathbf{H}^{\mathrm{freq}}_{\mathrm{DL}} + \mathbf{N}^{\mathsf{T}}_{\mathrm{DL}}, \tag{5.28}$$

Algorithm 3 MMV-LAMP network based FRSN

Input: The feedback signal $\widetilde{\mathbf{Y}}_{\mathrm{DL}}$, the measurement matrix $\widetilde{\mathbf{U}}$, the number of layers T'.

Output: The output of the T'-th layer MMV-LAMP network $\widehat{\mathbf{H}}_{\mathrm{DL}}^{\mathrm{freq}} = \mathbf{U}\widehat{\mathbf{H}}_{\mathrm{DL},T'}^{\mathrm{delay}}$.

Initialization: $\mathbf{V}_0 = \widetilde{\mathbf{Y}}_{\mathrm{DL}}$, $\widehat{\mathbf{H}}_{\mathrm{DL},0}^{\mathrm{delay}} = \mathbf{0}$, $\mathbf{B}' = \widetilde{\mathbf{U}}^{\mathrm{H}}$, $\boldsymbol{\theta}' = \{1, 1\}$.

for $t = 1, 2, \cdots, T'$ **do**

$\mathbf{R}_t = \widehat{\mathbf{H}}_{\mathrm{DL},t-1}^{\mathrm{delay}} + \mathbf{B}'\mathbf{V}_{t-1}$

$\sigma_t = \frac{1}{\sqrt{K_c Q}} \|\mathbf{V}_{t-1}\|_F$

$\widehat{\mathbf{H}}_{\mathrm{DL},t}^{\mathrm{delay}} = \eta(\mathbf{R}_t; \boldsymbol{\theta}', \sigma_t)$

$b_t = \frac{1}{K_c} \sum_{j=1}^{K} \frac{\partial[\eta(\mathbf{R}_t; \boldsymbol{\theta}', \sigma_t)]_j}{\partial[\mathbf{R}_t(j,:)]}$

$\mathbf{V}_t = \widetilde{\mathbf{Y}}_{\mathrm{DL}} - \widetilde{\mathbf{U}}\widehat{\mathbf{H}}_{\mathrm{DL},t}^{\mathrm{delay}} + b_t \mathbf{V}_{t-1}$

end for

Figure 5.10: MMV-LAMP network based FRSN

Algorithm 4 Learning strategy to train FRSN's parameters $\{\mathbf{B}', \boldsymbol{\theta}'\}$

Initialization: $\mathbf{B}' = \widetilde{\mathbf{U}}^{\mathrm{H}}$, $\boldsymbol{\theta}' = \{1, 1\}$, $\{\mathbf{B}', \boldsymbol{\theta}'\}_0 = \{\mathbf{B}', \boldsymbol{\theta}'\}$.

for $t = 1, 2, \cdots, T'$ **do**

Initialize $\{\mathbf{B}', \boldsymbol{\theta}'\}_t$ as $\{\mathbf{B}', \boldsymbol{\theta}'\}_{t-1}$

Learn $\{\mathbf{B}', \boldsymbol{\theta}'\}_t$ to minimize $L_t'(\{\mathbf{B}', \boldsymbol{\theta}'\}_t)$

end for

Output: $\{\mathbf{B}', \boldsymbol{\theta}'\} = \{\mathbf{B}', \boldsymbol{\theta}'\}_{T'}$.

Figure 5.11: Learning strategy to train FRSN's parameters $\{\mathbf{B}', \boldsymbol{\theta}'\}$

where $\mathbf{H}_{\mathrm{DL}}^{\mathrm{fs}} = (\mathbf{H}_{\mathrm{DL}}^{\mathrm{sf}})^{\mathsf{T}} \in \mathbb{C}^{K \times N_{\mathrm{BS}}}$ denotes the frequency-spatial domain channel matrix and $\mathbf{H}_{\mathrm{DL}}^{\mathrm{freq}} = \mathbf{H}_{\mathrm{DL}}^{\mathrm{fs}} \mathbf{F}_{\mathrm{DL}} \in \mathbb{C}^{K \times Q}$ is a frequency-domain channel matrix after spatial-domain compression.

In order to accurately acquire the CSI at the BS with reduced feedback overhead, we propose an FCN and an FCRN based on an MMV-LAMP network with two stpdf. In the first step, by exploiting the channels' delay-domain sparsity, we compress the received pilots at the users by only feeding back the received pilot signals on part of K subcarriers. In the second step, the compressed feedback signals received at the BS are regarded as the input of the FRSN to reconstruct the frequency-domain channel matrix $\widehat{\mathbf{H}}_{\mathrm{DL}}^{\mathrm{freq}}$, which is then input to the CRN (can be well trained at the uplink channel estimation stage), so that the spatial-frequency domain channel matrix $\mathbf{H}_{\mathrm{DL}}^{\mathrm{sf}}$ is finally reconstructed at the BS.

Feedback Compression Network at Users

To reduce the channel feedback overhead at the users, we compress the dimensionality of the received pilot signals by exploiting the channels' delay-domain sparsity. Specifically, we transform the frequency-spatial domain channel matrix $\mathbf{H}_{\mathsf{DL}}^{\mathsf{fs}}$ into the delay-spatial domain channel matrix $\mathbf{H}_{\mathsf{DL}}^{\mathsf{ds}}$ via a DFT matrix $\mathbf{U} \in \mathbb{C}^{K\times K}$, and (5.28) can be expressed as

$$\mathbf{Y}_{\mathsf{DL}}^{\mathsf{T}} = \mathbf{U}\mathbf{H}_{\mathsf{DL}}^{\mathsf{ds}}\mathbf{F}_{\mathsf{DL}} + \mathbf{N}_{\mathsf{DL}}^{\mathsf{T}} = \mathbf{U}\mathbf{H}_{\mathsf{DL}}^{\mathsf{delay}} + \mathbf{N}_{\mathsf{DL}}^{\mathsf{T}}, \tag{5.29}$$

where $\mathbf{H}_{\mathsf{DL}}^{\mathsf{delay}} = \mathbf{H}_{\mathsf{DL}}^{\mathsf{ds}}\mathbf{F}_{\mathsf{DL}} \in \mathbb{C}^{K\times Q}$ is a delay-domain channel matrix after spatial-domain compression. The compressed pilot signals fed back to the BS can be expressed as

$$\widetilde{\mathbf{Y}}_{\mathsf{DL}} = \mathbf{Y}_{\mathsf{DL}}^{\mathsf{T}}\Big|_{\mathbf{\Omega}} = \widetilde{\mathbf{U}}\mathbf{H}_{\mathsf{DL}}^{\mathsf{delay}} + \widetilde{\mathbf{N}}_{\mathsf{DL}}, \tag{5.30}$$

where $\widetilde{\mathbf{U}} = \mathbf{U}|_{\mathbf{\Omega}} \in \mathbb{C}^{K_c\times K}$ is a partial DFT matrix, $\widetilde{\mathbf{N}}_{\mathsf{DL}} = \mathbf{N}_{\mathsf{DL}}^{\mathsf{T}}\big|_{\mathbf{\Omega}}$, and the i-th element of the set $\{\mathbf{\Omega}\}_i$ for $1 \le i \le K_c$ is randomly selected without repeating [96].

Feedback Based Channel Reconstruction Network at BS

The FCRN consists of an FRSN and a CRN. Different from the MMV-LAMP network with T layers used in CRN, we just need to train the FRSN based on an MMV-LAMP network with T' layers to obtain the channel matrix $\widehat{\mathbf{H}}_{\mathsf{DL}}^{\mathsf{freq}}$, which is illustrated in Fig. 5.9 and summarized in Algorithm 5.10. Specifically, the input is the feedback pilot signals $\widetilde{\mathbf{Y}}_{\mathsf{DL}}$, and the initial values $\widehat{\mathbf{H}}_{\mathsf{DL},0}^{\mathsf{delay}}$ and $\mathbf{V}_0$ of Algorithm 5.10 are denoted as $\widehat{\mathbf{H}}_{\mathsf{DL},0}^{\mathsf{delay}} = \mathbf{0}$ and $\mathbf{V}_0 = \widetilde{\mathbf{Y}}_{\mathsf{DL}}$, respectively. In FRSN, the measurement matrix is the partial DFT matrix $\widetilde{\mathbf{U}}$ and the trainable parameters are $\{\mathbf{B}', \boldsymbol{\theta}'\}$. After the BS receives the compressed feedback signals $\widetilde{\mathbf{Y}}_{\mathsf{DL}}$ in (5.30), we first exploit the proposed FRSN to reconstruct the channel matrix $\widehat{\mathbf{H}}_{\mathsf{DL}}^{\mathsf{freq}} = \mathbf{U}\widehat{\mathbf{H}}_{\mathsf{DL},T'}^{\mathsf{delay}}$, which is then passed to the CRN to reconstruct $\widehat{\mathbf{H}}_{\mathsf{DL}}^{\mathsf{sf}}$ based on MMV-LAMP network.

Moreover, the training strategy of the trainable parameters $\{\mathbf{B}', \boldsymbol{\theta}'\}$ of the FRSN is summarized in Algorithm 5.11, where the corresponding loss function of t-th layer ($1 \le t \le T'$) is given by

$$\begin{aligned} L_t'\Big(\big\{\mathbf{B}', \boldsymbol{\theta}'\big\}_t\Big) &= \sum_{n=1}^{N'} \frac{\left\|\widehat{\mathbf{H}}_{\mathsf{DL},t}^{\mathsf{freq},n} - \mathbf{H}_{\mathsf{DL}}^{\mathsf{freq},n}\right\|_F^2}{\left\|\mathbf{H}_{\mathsf{DL}}^{\mathsf{freq},n}\right\|_F^2} \\ &= \sum_{n=1}^{N'} \frac{\left\|\mathbf{U}f_t'(\widetilde{\mathbf{Y}}_{\mathsf{DL}}^n, \big\{\mathbf{B}', \boldsymbol{\theta}'\big\}_t) - \mathbf{H}_{\mathsf{DL}}^{\mathsf{freq},n}\right\|_F^2}{\left\|\mathbf{H}_{\mathsf{DL}}^{\mathsf{freq},n}\right\|_F^2}, \end{aligned} \tag{5.31}$$

where N' is the number of data samples in each batch of the training set, and $f'(\cdot,\cdot)_t$ denotes the MMV-LAMP network based FRSN. Specifically, we first generate the training data set $\{\mathbf{H}_{\mathsf{DL}}^{\mathsf{freq},n}\}_{n=1}^{N'}$ according to (5.28). Then, for the t-th layer, we aim to optimize the trainable parameters $\{\mathbf{B}', \boldsymbol{\theta}'\}_t$ by minimizing the loss function of the t-th layer $L_t'(\{\mathbf{B}', \boldsymbol{\theta}'\}_t)$ for $1 \le t \le T'$. Finally, after the trainable parameters $\{\mathbf{B}', \boldsymbol{\theta}'\}_{T'}$ of the T'-th

layer are optimized, we can obtain $\widehat{\mathbf{H}}_{\text{DL}}^{\text{delay}}$ and $\widehat{\mathbf{H}}_{\text{DL}}^{\text{freq}} = \mathbf{U}\widehat{\mathbf{H}}_{\text{DL}}^{\text{delay}}$. Also, we can find that $\mathbf{H}_{\text{DL}}^{\text{freq}}$ in (5.28) can also be expressed as

$$(\mathbf{H}_{\text{DL}}^{\text{freq}})^{\text{T}} = \mathbf{F}_{\text{DL}}^{\text{T}}\mathbf{H}_{\text{DL}}^{\text{sf}} = \mathbf{F}_{\text{DL}}^{\text{T}}\mathbf{D}^{\text{H}}\mathbf{H}_{\text{DL}}^{\text{af}} = \mathbf{A}_{\text{DL}}\mathbf{H}_{\text{DL}}^{\text{af}}, \tag{5.32}$$

which indicates that we can exploit the following CRN to reconstruct the final estimation $\widehat{\mathbf{H}}_{\text{DL}}^{\text{sf}}$.

5.5 Simulation Results

In this section, we provide numerical results to verify the effectiveness of the proposed MDDL-based channel estimation and feedback scheme. First, we elaborate the implementation details and parameters adopted in our simulations setting. Then, since the TDD uplink channel estimation and FDD downlink channel estimation share the same processing mechanism, we take the downlink channel estimation and feedback for FDD systems as examples to evaluate the performance. Finally, we investigate the performance of the proposed MDDL-based scheme in scenarios with fixed scattering environments, which is discussed in more detail next.

5.5.1 Simulation Setup

In our simulations, we consider that the BS is equipped with an ULA with $N_{\text{BS}} = 256$ and $N_{\text{RF}} = 4$ RF chains. The number of OFDM subcarriers in the channel estimation phase is set to $K = 64$. The redundant dictionary with an oversampling ratio G/N_{BS} = 4 is considered, i.e., the quantized angle grids G is set to 1024. In addition, the experiments are performed in PyCharm Community Edition (Python 3.6 environment and Tensorflow 1.13.1) on a computer with dual Intel Xeon 8280 CPU (2.6GHz) and dual Nvidia GeForce GTX 2080Ti GPUs.

The proposed MMV-LAMP network based CRN is composed of $T = 5$ layers, where each layer has the same network structure with the trainable parameters $\{\mathbf{\Xi}_{\text{DL}}, \mathbf{B}_{\text{DL}}, \boldsymbol{\theta}_{\text{DL}}\}$. While the proposed MMV-LAMP based FRSN is composed of $T' = 2$ layers, where the trainable parameters are $\{\mathbf{B}', \boldsymbol{\theta}'\}$. For the training of MMV-LAMP network, we generate a training set including $S_{\text{tr}} = 5000$ spatial-frequency domain channel samples according to the channel model in (5.10), so that the dimension of the input channel samples is $(S_{\text{tr}}, N_{\text{BS}}, K)$. Similarly, the parameters of the validation set and the test set are $S_{\text{va}} = 2000$ and $S_{\text{te}} = 1000$, respectively. We choose the NMSE as the metric for performance evaluation, which is defined as

$$\text{NMSE}(\mathbf{H}, \widehat{\mathbf{H}}) = 10\log_{10}(\mathbb{E}\left[\frac{\left\|\mathbf{H} - \widehat{\mathbf{H}}\right\|_F^2}{\left\|\mathbf{H}\right\|_F^2}\right]). \tag{5.33}$$

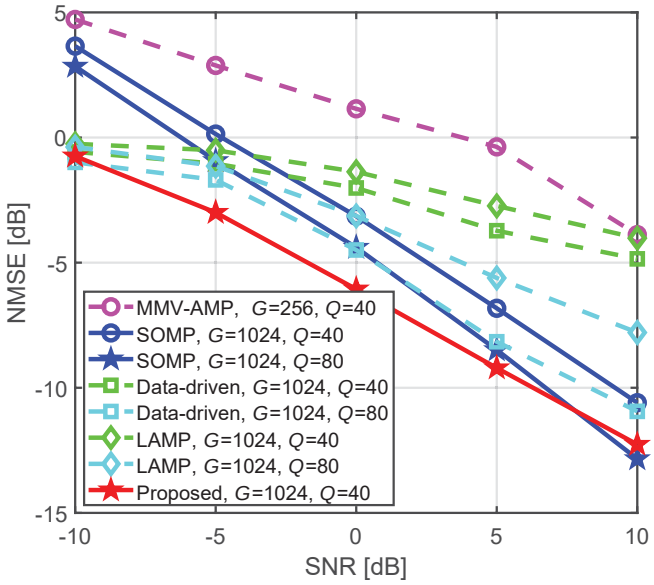


Figure 5.12: NMSE performance comparison of different channel estimation schemes versus SNRs.

5.5.2 MDDL-Based FDD Downlink Channel Estimation

As shown in Fig. 5.12 [4], we plot the NMSE performance NMSE$(\mathbf{H}_{\text{DL}}^{\text{sf}}, \widehat{\mathbf{H}}_{\text{DL}}^{\text{sf}})$ of the different schemes as a function of signal-to-noise ratio (SNR), including the proposed MDDL-based channel estimation scheme using the MMV-LAMP network, the data-driven deep learning based channel estimation scheme [86], the LAMP-based channel estimation scheme [89], and two model-based channel estimation schemes using the MMV-AMP algorithm [94] and the SOMP algorithm [93]. The number of propagation paths is $L = 8$. Note that the AMP-based channel estimation scheme requires the measurement matrix's elements to be independent identically distributed, so we don't consider the redundant dictionary matrix (i.e., $G = N_{\text{BS}} = 256$).

We can observe that the proposed channel estimation scheme outperforms the other channel estimation schemes even with a smaller pilot overhead. This observation indicates that the proposed channel estimation scheme can achieve better NMSE performance while keeping the pilot overhead to a low level. This is because the network architecture (i.e., the cascaded CCN and MMV-LAMP-based CRN) of the MDDL-based approach is constructed based on known physical mechanisms and some *a priori* model knowledge, which can reduce the number of trainable parameters to be learned and can fully take advantage of both model-based algorithms and deep learning methods. We also observe that the proposed channel estimation scheme can significantly improve the NMSE performance in the low SNR regime compared with other channel estimation schemes. Therefore, the proposed scheme can reliably reconstruct the high-dimensional channel with a much reduced pilot overhead.

To clearly present the percentage of the reduced pilot overhead compared to state-of-the-art algorithms, we compare the proposed scheme with four state-of-the-art algorithms at SNR=0 dB and SNR=5 dB, as shown in Table 5.1. We can observe that

[4] This simulated results for $\{M = 40, Q = 10, N_{\text{RF}} = 4\}$ are equivalent to the uplink channel estimation results for $\{M = 40, Q = 40, N_{\text{RF}} = 1\}$, $\{M = 40, Q = 20, N_{\text{RF}} = 2\}$, and $\{M = 40, Q = 5, N_{\text{RF}} = 8\}$, as long as $M = QN_{\text{RF}}$.

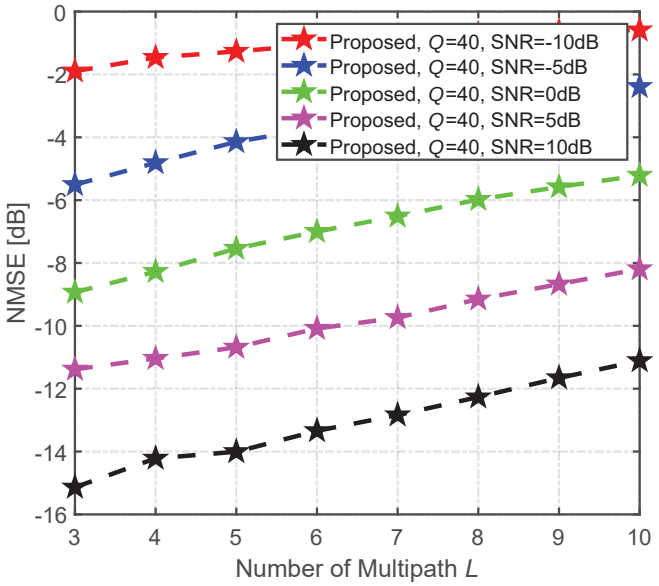


Figure 5.13: NMSE performance comparison of the proposed scheme versus the number of multipath L.

Table 5.1: NMSE in dB

Channel Estimation Schemes	SNR=0dB		SNR=5dB	
	Q=40	Q=80	Q=40	Q=80
MMV-AMP	1.15	-0.31	-0.37	-7.62
LAMP	-1.37	-3.13	-2.73	-5.61
Data Driven Deep Learning	-2.02	-4.49	-3.72	-8.16
SOMP	-3.14	-4.39	-6.82	-8.48
Proposed	**-6.06**		**-9.21**	

the proposed scheme with $Q = 40$ outperforms the MMV-AMP algorithm, the LAMP network, and data driven deep learning methods with $Q = 80$ in the whole range of SNR. Therefore, we conclude that the proposed scheme can reduce the pilot overhead by at least 50% while achieving the same or even better channel estimation NMSE performance.

We further investigate the robustness of the proposed channel estimation scheme as a function of the number of multipath L in Fig. 5.13. Note that the proposed MMV-LAMP based CRN is trained during the offline training stage, which is based on the channel samples with $L = 8$ multipath components. However, at the online estimation stage, we observe that the proposed scheme can be robustly adopted to estimate multipath channels with $L \neq 8$, without having to retrain the entire network architecture. In Fig. 5.14, we show the NMSE performance of the proposed scheme when it is trained based for different SNR values. For example, the setup denoted by "Proposed, SNR=-10dB, $G = 1024$, $Q = 40$" means that the proposed scheme was trained at the SNR=-10dB, but tested in the whole SNR range, and the setup denoted by "Proposed,

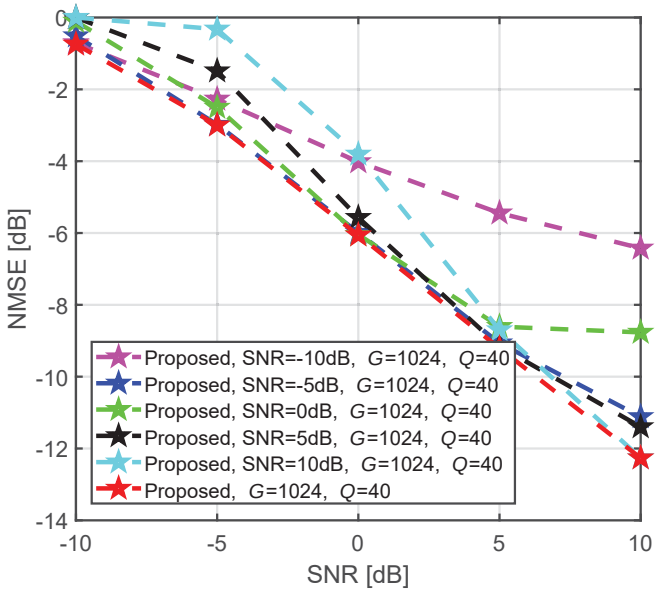


Figure 5.14: NMSE performance comparison of the proposed scheme trained at the different SNRs.

$G = 1024$, $Q = 40$" means that the proposed scheme was trained and tested for the same values of SNR. We can observe that the proposed scheme trained at SNR=-5dB is robust in the low SNR regime, and the NMSE performance is deteriorated just at SNR=10dB. Therefore, the proposed MMV-LAMP based channel estimator enjoys a better robustness and generalization capability to different channel conditions.

As for the number of antennas N_{BS}, we further investigated the channel estimation NMSE performance with $N_{\text{BS}} = 128$ in Fig. 5.15. Specifically, since the dimension of the beamforming/combining matrix is related to the number of antennas, changing the number of antennas requires retraining the CCN and CRN. In order to meet the same compression ratio, i.e., $Q/N_{\text{BS}} = 40/256$, the adopted pilot overhead is $Q = 20$. We can observe that the proposed channel estimation scheme outperforms the other channel estimation schemes in this case.

As for the different numbers of subcarriers K, as shown in Fig.5.16, we have investigated the channel estimation NMSE performance with $K = 64, 128, 256, 512$. Specifically, since the proposed channel estimation scheme was trained under the number of subcarriers $K = 64$, we divide the subcarriers evenly into multiple sub-groups (each sub-group has 64 subcarriers) to directly apply the trained channel estimation network. For instance, for the number of subcarriers $K = 128$, we divide the subcarriers into $128/64 = 2$ sub-groups, i.e., the subcarrier indices of each sub-group are respectively $\{1, 3, 5, \cdots, 127\}$ and $\{2, 4, 6, \cdots, 128\}$. We can observe that the proposed scheme trained with 64 subcarriers can be effectively applied to the case $K =$ 128, 256, and 512, which further proves the robustness of the proposed scheme.

As mentioned above, we design a redundant dictionary matrix to combat the power leakage problem by quantizing the angles with a finer resolution. Therefore, in Fig. 5.17, we also compare the performance of the proposed MMV-LAMP based channel estimator without using redundant dictionary matrix, i.e., $G = N_{\text{BS}} = 256$, to demonstrate the effectiveness of the redundant dictionary matrix. We can find that the proposed MMV-LAMP based and the SOMP-based channel estimators can improve the

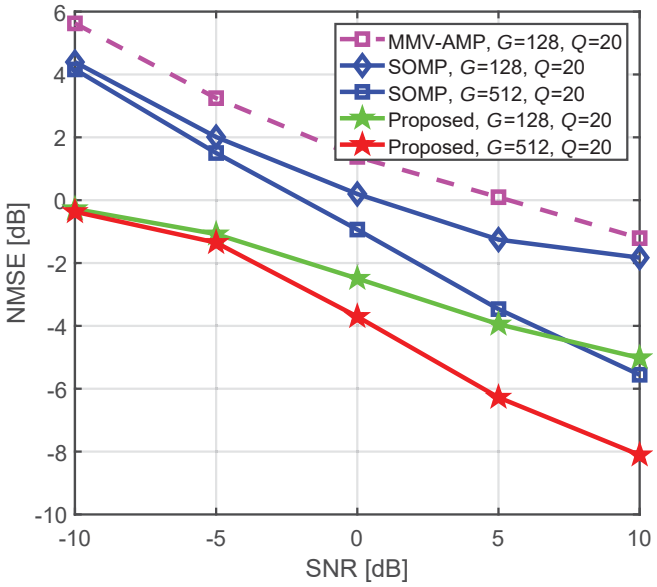


Figure 5.15: NMSE performance comparison of different channel estimation schemes versus SNRs.

sparse channel estimation performance by utilizing the redundant dictionary matrix to cope with the power leakage problem.

Note that the effectiveness of the proposed MMV-LAMP based channel estimator for OFDM systems is based on a training dataset based on multi-carrier channel samples, which consist of the channels of all subcarriers from different channel realizations. To verify the training effectiveness of multi-carrier channel samples, we further compare the performance of the proposed scheme based on single carrier channel samples, as shown in Fig. 5.18. We can observe that the proposed MMV-LAMP based channel estimator trained by multi-carrier channel samples can exhibit more excellent performance.

We further investigate the computational complexity. In the case of offline training, specifically, the computation complexity is not a major concern, because the required time is usually not strictly limited. Moreover. we discuss the computational complexity of the different channel estimation schemes as follows.

- As for the data driven deep learning based channel estimation scheme, its main stpdf in testing stage include: i) two fully-connected operations with computational complexity $\mathcal{O}(2MG)$; ii) N_{re} convolutional operations with computational complexity $\mathcal{O}(G\beta^2 \sum_{i=1}^{N_{\text{re}}} n_{i-1}n_i)$, where β is the side length of the convolutional filters, n_{i-1} and n_i denote the numbers of input and output feature maps of the i-th convolutional layer $(1 \leq i \leq N_{\text{re}})$, respectively. Therefore, the computational complexity of the data driven deep learning based channel estimation is $\mathcal{O}(2MG + G\beta^2 \sum_{i=1}^{N_{\text{re}}} n_{i-1}n_i)$.

- The proposed channel estimation scheme is developed from the MMV-AMP algorithm, which mainly requires matrix multiplication operations, Therefore, the MMV-AMP algorithm, the LAMP network, and the proposed MMV-LAMP network

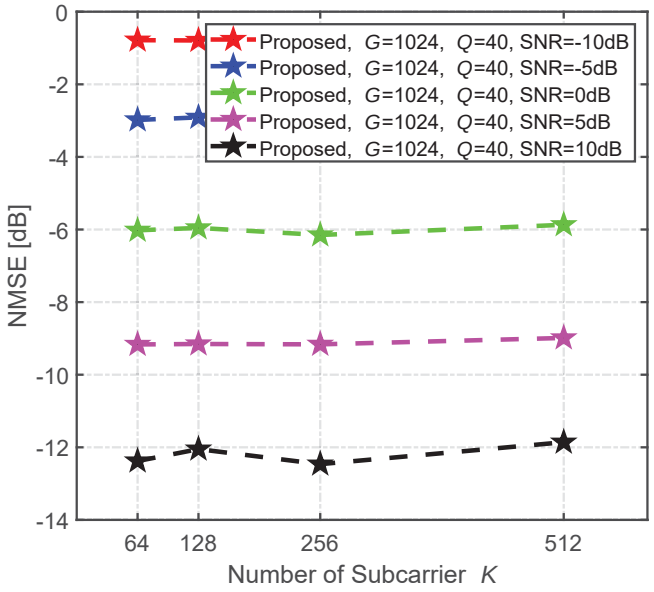


Figure 5.16: NMSE performance comparison of the proposed scheme versus the number of subcarrier K.

share similar computational complexities, i.e., $\mathcal{O}(MN_{\text{BS}}K + TMGK)$ in the case of K subcarriers.

- As for the SOMP algorithm, we denote the number of iterations as I and its main stpdf include: i) correlation operation with computational complexity $\mathcal{O}(MG^2KI)$; ii) project subspace operation with computational complexity $\mathcal{O}(\frac{1}{4}I^2(I+1)^2+\frac{1}{3}MI(I+1)(2I+1)+\frac{1}{2}MKI(I+1))$; iii) update residual operation with computational complexity $\mathcal{O}(\frac{1}{2}\rho N_{\text{BS}}KI(I+1))$. Therefore, the computational complexity of the SOMP algorithm is $\mathcal{O}(MG^2KI+\frac{1}{4}I^2(I+1)^2+\frac{1}{3}MI(I+1)(2I+1)+\frac{1}{2}MKI(I+1)+\frac{1}{2}\rho N_{\text{BS}}KI(I+1))$.

Finally, we provide the computational complexity analysis of the different channel estimation schemes as shown in Table 5.2.

5.5.3 Channel Estimation Under Non-Ideal Hardware Constraints

In practical systems, the phase of each combining and beamforming matrix coefficient is not a continuous value. Therefore, we further investigate the performance of the proposed MDDL-based channel estimation scheme under the constraint of PSN with a finite phase resolution. Specifically, after the offline training, the CCN including the continuous complex-valued combining matrix $\mathbf{F}_{\text{UL}} = \frac{1}{\sqrt{N_{\text{BS}}}}\exp\left[\text{j}(\mathbf{\Xi}_{\text{UL}})\right]$ in (5.19) and the beamforming matrix $\mathbf{F}_{\text{DL}} = \frac{1}{\sqrt{N_{\text{BS}}}}\exp\left[\text{j}(\mathbf{\Xi}_{\text{DL}})\right]$ in (5.27) are quantized to the elements in the set $\mathbf{\Delta}$ according to the minimum Euclidean distance criterion. That is to say, after quantization, we have $\{\mathbf{\Xi}_{\text{UL}}, \mathbf{\Xi}_{\text{DL}}\} \in \mathbf{\Delta}$, and $\mathbf{\Delta}$ is the quantized phase set of the PSN whose resolution is B^{ps}, given by

$$\mathbf{\Delta} = \left\{0, \frac{2\pi}{2^{B^{\text{ps}}}}, 2\cdot\frac{2\pi}{2^{B^{\text{ps}}}}, \cdots, 2\pi - \frac{2\pi}{2^{B^{\text{ps}}}}\right\}. \tag{5.34}$$

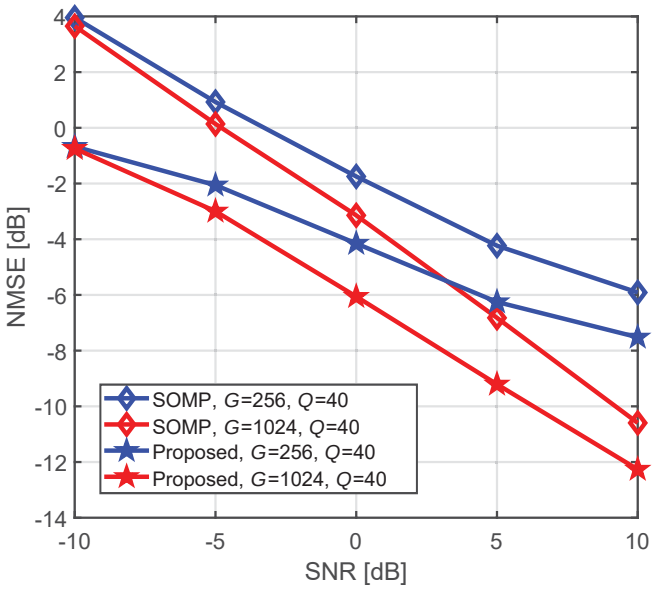


Figure 5.17: NMSE performance comparison of the proposed MDDL-based channel estimation scheme versus SNRs, where the effectiveness of the devised redundant dictionary matrix.

The network architecture and training strategy of the CRN remains unchanged[5]. As shown in Fig. 5.19, we plot the NMSE performance NMSE$(\mathbf{H}_{\text{DL}}^{\text{sf}}, \widehat{\mathbf{H}}_{\text{DL}}^{\text{sf}})$ of the proposed channel estimation scheme as a function of the SNR, where the resolution of the PSN is set to $B^{\text{ps}} = 2, 3$ bits. We observe that the proposed MMV-LAMP based channel estimator works well in the case of 3-bit quantization and suffers from a little loss in the case of 2-bit quantization. This observation further demonstrates the robustness of the proposed MMV-LAMP based channel estimator under non-ideal PSN with limited resolution.

Moreover, considering the limited resolution of the analog-to-digital converter (ADC) at the BS, the downlink received signals are first quantized by the ADC in the time domain, so the received frequency-domain pilot signals after time-domain quantization can be expressed as

$$\mathbf{Y}_{\text{DL}}^{\text{quan}} = \lambda^{\text{quan}}(\mathbf{Y}_{\text{DL}}\mathbf{U})\mathbf{U}^{\text{H}}, \tag{5.35}$$

where $\mathbf{Y}_{\text{DL}}\mathbf{U}$ and $\lambda^{\text{quan}}(\mathbf{Y}_{\text{DL}}\mathbf{U})$ are the received time-domain signals before the ADC and after the ADC, respectively, and $\lambda^{\text{quan}}(\cdot)$ is the complex-valued quantization function. This quantization function is applied to the received signals element-wise, and the real and imaginary parts are quantized separately. Here, we consider a uniform codebook for quantization as

$$C = \left\{ -\frac{2^{B^{\text{adc}}} - 1}{2}, \cdots, \frac{2^{B^{\text{adc}}} - 1}{2} \right\}, \tag{5.36}$$

[5]Since the quantized phase shifters will result in non-differential gradients, it is not feasible to directly use the Adam algorithm. To avoid this challenge, we consider a simple method that the offline training is performed by assuming the infinite precision phase resolution. At the online test stage, the combining/beamforming matrix are quantized according to the minimun Euclidean distance criterion. Note that the authors of [76] proposed a sub-optimum quantization method to solve this issue, which may be integrated into the proposed scheme for better performance in our future work..

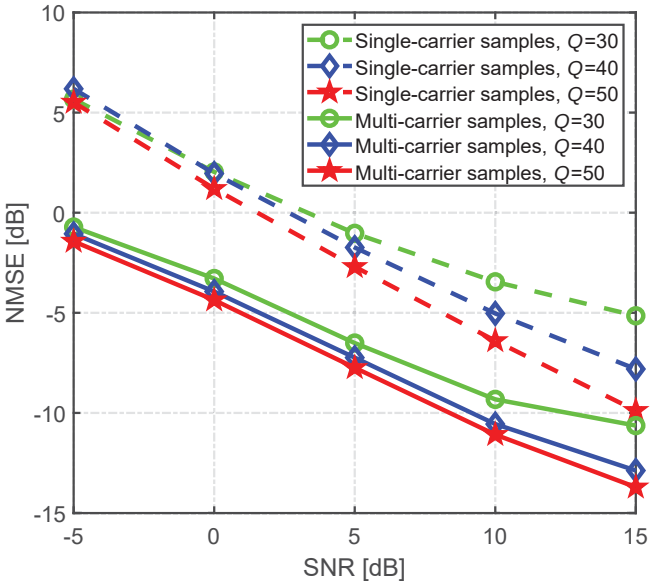


Figure 5.18: NMSE performance comparison of the proposed MDDL-based channel estimation scheme versus SNRs, where the effectiveness of the proposed scheme using multi-carrier channel samples to train.

where B^{adc} is the number of quantization bits, $= [y_{\max} - y_{\min}] / 2^{B^{\text{adc}}}$, $y_{\max}$ and $y_{\min}$ are the maximum and the minimum real value of both the real and imaginary parts of $\mathbf{Y}_{\text{DL}}\mathbf{U}$, respectively. Specifically, we first train the CCN's parameters $\{\Xi_{\text{DL}}\}$ and CRN's parameters $\{\mathbf{B}_{\text{DL}}, \boldsymbol{\theta}_{\text{DL}}\}$ based on the ADC with infinite resolution, then we adopt the above quantization method to obtain the quantized frequency-domain signal $\mathbf{Y}_{\text{DL}}^{\text{quan}}$. Finally, we input the quantized $\mathbf{Y}_{\text{DL}}^{\text{quan}}$ directly into the previously trained CRN to reconstruct the channel matrix $\widehat{\mathbf{H}}_{\text{DL}}^{\text{sf}}$.

As shown in Fig. 5.20, we plot the NMSE performance NMSE$(\mathbf{H}_{\text{DL}}^{\text{sf}}, \widehat{\mathbf{H}}_{\text{DL}}^{\text{sf}})$ of the proposed channel estimation scheme as a function of the SNR, where the number of quantization bits is set to $B^{\text{adc}} = 2, 3$. We observe that the performance of the MMV-AMP-based and the SOMP-based channel estimation schemes degrade. However, the proposed channel estimation scheme can still work even in the low SNR regime. Considering that the SNR is usually low in most systems that operate at high frequencies at the channel estimation stage, the proposed scheme is effective in estimating the channels with non-ideal ADC at the receiver.

5.5.4 MDDL-Based FDD Downlink Channel Feedback

In this section, we investigate the channel reconstruction performance NMSE$(\mathbf{H}_{\text{DL}}^{\text{sf}}, \widehat{\mathbf{H}}_{\text{DL}}^{\text{sf}})$ of the proposed channel feedback scheme. Specifically, as shown in Fig. 5.4, the CCN at the BS is the same as that used at the downlink channel estimation stage, then the noisy frequency-domain received signal $\mathbf{Y}_{\text{DL}}^{\text{T}}$ is obtained at the user. Moreover, in order to compress the feedback overhead, the user only feeds $\mathbf{Y}_{\text{DL}}^{\text{T}}$ on K_c of K subcarriers back to the BS. Finally, the BS exploits the proposed MMV-LAMP based FRSN and the CRN to recover the spatial-frequency domain channel matrix $\widehat{\mathbf{H}}_{\text{DL}}^{\text{sf}}$. In addition, we define the feedback compression radio as $\rho = K_c/K$. As shown in Fig. 5.21, we can observe that the proposed MMV-LAMP based FCRN (including the FRSN and the

Table 5.2: Computational Complexity of Different Channel Estimation Schemes

Schemes	Complexity
Data driven deep learning	$\mathcal{O}(2MG + G\beta^2 \sum_{i=1}^{N_{\text{re}}} n_{i-1}n_i)$
SOMP	$\mathcal{O}(MG^2KI + \frac{1}{4}I^2(I+1)^2 + \frac{1}{3}MI(I+1)(2I+1) + \frac{1}{2}MKI(I+1) + \frac{1}{2}\rho N_{\text{BS}}KI(I+1))$
LAMP	$\mathcal{O}(MN_{\text{BS}}K + TMGK)$
MMV-AMP	$\mathcal{O}(MN_{\text{BS}}K + TMGK)$
Proposed	$\mathcal{O}(MN_{\text{BS}}K + TMGK)$

following CRN) with $\rho = 0.25$, i.e., $K_c = 16$, even outperforms the SOMP-based channel feedback scheme with $\rho = 0.5$, i.e., $K_c = 32$. Therefore, the effectiveness of the proposed MDDL-based channel feedback scheme is verified.

5.5.5 Channel Estimation Based on Fixed Scattering Environments

In this section, the TDD uplink channel estimation NMSE performance NMSE$(\mathbf{H}_{\text{UL}}^{\text{sf}}, \widehat{\mathbf{H}}_{\text{UL}}^{\text{sf}})$ is investigated. To evaluate the superiority of the proposed learning strategy that jointly trains the pilots and channel estimator, we consider the scenario with fixed scattering environments. The fixed scattering environment is shown in Fig. 5.22, in which the positions of the BS, the user, and the scatterers are marked by blue, red, and green, respectively. The red solid line represents the line-of-sight (LoS) link between the BS and the user, and the black dotted line indicates the non-line-of-sight (NLoS) link via the scatterer. Note that for the sake of simplifying the channel generation, we only consider a single-bounce NLoS channel model. As for the array setting, we take the BS side as an example, i.e., the bold blue solid line represents the antenna array of the BS, whose normal direction is marked as the black arrow. The antenna array setting at users is the same as the BS.

Next, we describe how to generate the channel samples based on the fixed scattering environment. Specifically, as shown in Fig. 5.22, the scenario consists of a BS and a number of users geographically distributed in a certain outdoor environment, in which the scatterers are randomly distributed. For the given fixed scattering environment, we can generate the channel samples based on the channel parameters (including the AoAs/AoDs, path loss), which can be calculated based on the geometric characteristics between the BS and the user. More specifically,

- **AoAs/AoDs:** As for the uplink channel estimation, the AoA ϕ_{BS} at the BS is the angle relative to the horizontal axis. For the user, the AoD ϕ_{UE} at the user follows

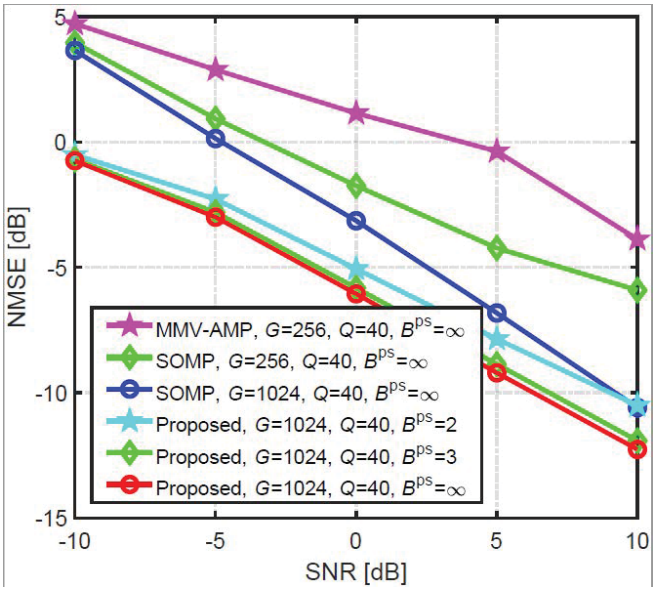


Figure 5.19: NMSE performance comparison of the proposed MDDL-based channel estimation scheme versus SNRs, where phase shift quantization.

the uniform distribution $[0, 2\pi)$, which is defined as the angle between the normal direction of the user array and the horizontal axis.

- **Path loss:** Taking the NLoS link as an example, the large-scale fading gain G_l can be modeled based on the free-space path loss of Friis' formula as

$$G_l = 20\log_{10}(\frac{4\pi d_{l,1}}{\lambda_c}) + 20\log_{10}(\frac{4\pi d_{l,2}}{\lambda_c}) + G_s, \tag{5.37}$$

where $d_{l,1}$ ($d_{l,2}$) denotes the communication distance between the user and the l-th scatterer (the l-th scatterer and the BS), λ_c is the carrier wavelength, and G_s denotes the path loss via the l-th scatterer.

Based on the fixed scattering environment discussed above, we generate 10000 channel samples, named as fixed scattering environment (FSE) data set, by considering a randomly distributed user location and normal direction of the user array. In simulations, we divide the 10000 channel samples into 8000, 1000, and 1000, corresponding to the training, validation, and test sets, respectively.

As shown in Fig. 5.23, the uplink channel estimation performance NMSE$(\mathbf{H}_{\text{UL}}^{\text{sf}}, \widehat{\mathbf{H}}_{\text{UL}}^{\text{sf}})$ of the different schemes is shown as a function of the SNR, including the proposed scheme trained by using the FSE data set, the proposed scheme trained in Section V-B (i.e., not trained with the FSE data set), and the SOMP algorithm using random combining matrix, where we consider $G = 1024$, $Q = 40$. Note that, at the NMSE performance evaluation phase, the input channel samples for these schemes come from the test set of the FSE data set. We can observe that the proposed channel estimation scheme (including the combining matrix $\mathbf{F}_{\text{UL}}^{\text{H}}$ and the MMV-LAMP network based channel estimator) trained with FSE data set can effectively learn the channel environment characteristics using less pilot overhead. Moreover, we can utilize the trained combining matrix $\mathbf{F}_{\text{UL}}^{\text{H}}$ in the uplink as the beamforming $\mathbf{F}_{\text{DL}}^{\text{T}}$ in the downlink channel estimation for improving the receive SNR at the users as verified in Fig. 5.24.

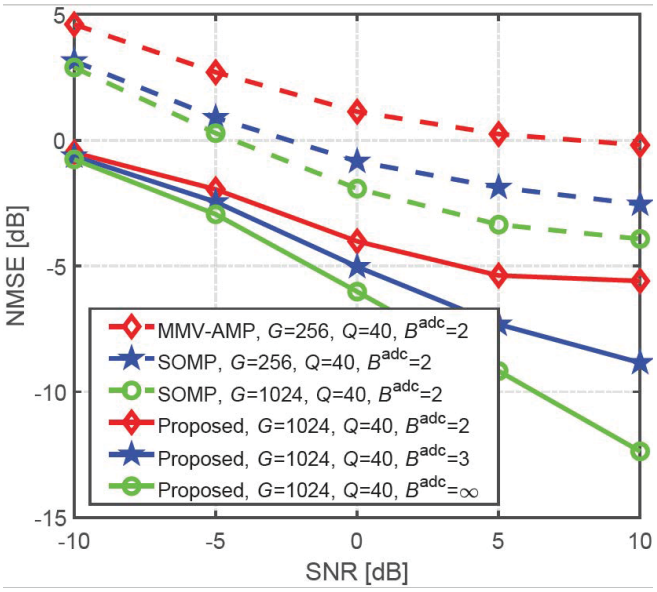


Figure 5.20: NMSE performance comparison of the proposed MDDL-based channel estimation scheme versus SNRs, where analog-to-digital converter (ADC) quantization.

Fig. 5.24 depicts the received SNR distributions at the users using two different CCN designs in the downlink channel estimation phase. Specifically, "CCN trained by FSE Data Set" indicates that the BS adopts the beamforming matrix $\mathbf{F}_{\text{DL}}^{\text{T}}$ or CCN being the same as the combining matrix $\mathbf{F}_{\text{UL}}^{\text{H}}$ trained in Fig. 5.23, while "CCN with random phases" indicates that the phases of the beamforming matrix $\mathbf{F}_{\text{DL}}^{\text{T}}$ adopted by the BS are random. In Fig. 5.24, under the fixed scattering environment, we can observe that more users can achieve a larger received SNR by exploiting the proposed CCN trained with the FSE data set than the CCN with random phases.

5.6 Conclusions

In this chapter, we have proposed an MDDL-based channel estimation and feedback scheme for wideband massive hybrid MIMO systems at high frequencies, where the angle-delay domain channels' sparsity is exploited for reduced overhead. First, we have considered the uplink channel estimation for TDD systems. To reduce the uplink pilot overhead for estimating the high-dimensional channels from a limited number of RF chains at the BS, we have proposed to jointly train the PSN and the channel estimator as an auto-encoder. Specifically, by exploiting channels' structured sparsity from an *a priori* model and learning the integrated trainable parameters from the data samples, the MMV-LAMP network with the devised redundant dictionary has been proposed to jointly recover multiple subcarriers' channels with significantly enhanced performance. Moreover, we have considered the downlink channel estimation and feedback for FDD systems. Similarly, the pilots at the BS and channel estimator at the users can be jointly trained as an encoder and a decoder, respectively. To further reduce the channel feedback overhead, only the received pilots on part of the subcarriers are fed back to the BS, which can exploit the proposed MMV-LAMP network to reconstruct the spatial-

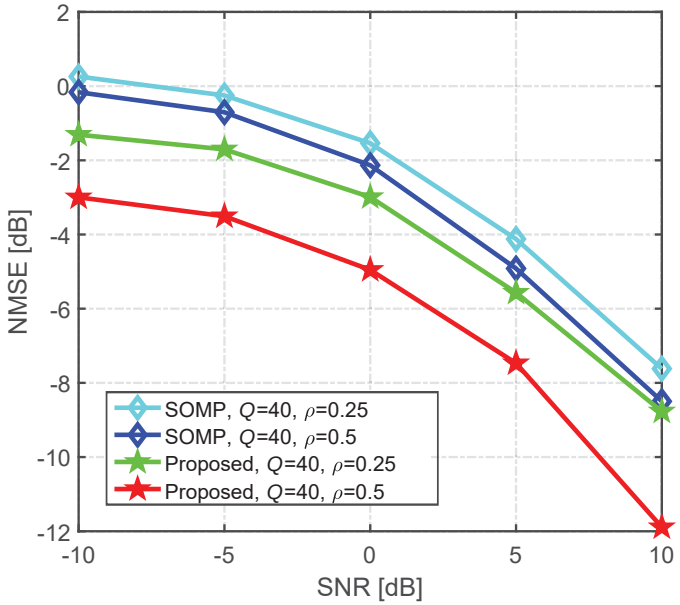


Figure 5.21: Channel reconstruction NMSE performance comparison of the proposed MDDL-based channel feedback scheme versus SNRs.

frequency channel matrix. Further, we consider to generate the data samples from the scenarios with fixed scattering environments, and optimize the combining/beamforming matrix and MMV-LAMP network by learning and perceiving the characteristics of the channel environments for the improved performance. Simulation results have verified that the proposed MDDL-based solution can achieve a significant improvement in channel estimation and feedback performance than the conventional schemes.

Appendix

First, we introduce the AMP algorithm. We consider a typical single-measurement-vector (SMV) CS problem

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}, \tag{5.38}$$

where $\mathbf{y} \in^{M\times 1}$ is a noisy measurement, $\mathbf{A} \in \mathbb{C}^{M\times N}$ represents the measurement matrix with $M \ll N$, $\mathbf{x} \in \mathbb{C}^{N\times 1}$ denotes the sparse vector, and $\mathbf{n} \in \mathbb{C}^{M\times 1}$ is the AWGN. To reconstruct the sparse vector $\mathbf{x}$, the AMP algorithm can be given as (see equations (10)-(12) in [89])

$$\mathbf{r}_t = \hat{\mathbf{x}}_{t-1} + \mathbf{A}^{\mathsf{H}}\mathbf{v}_{t-1}, \tag{5.39a}$$
$$\hat{\mathbf{x}}_t = \eta\left(\mathbf{r}_t; \boldsymbol{\theta}_t, \sigma_t\right), \tag{5.39b}$$
$$\mathbf{v}_t = \mathbf{y} - \mathbf{A}\hat{\mathbf{x}}_t + b_t\mathbf{v}_{t-1}, \tag{5.39c}$$

where $\mathbf{v}_0 = \mathbf{y}$, $\hat{\mathbf{x}}_0 = \mathbf{0}$. And

$$\sigma_t = \frac{1}{\sqrt{M}}\|\mathbf{v}_{t-1}\|_2, \tag{5.40}$$
$$b_t = \frac{1}{M}\sum_{j=1}^{N}\frac{\partial\left[\eta\left(\mathbf{r}_t; \boldsymbol{\theta}_t, \sigma_t\right)\right]_j}{\partial r_j}, \tag{5.41}$$

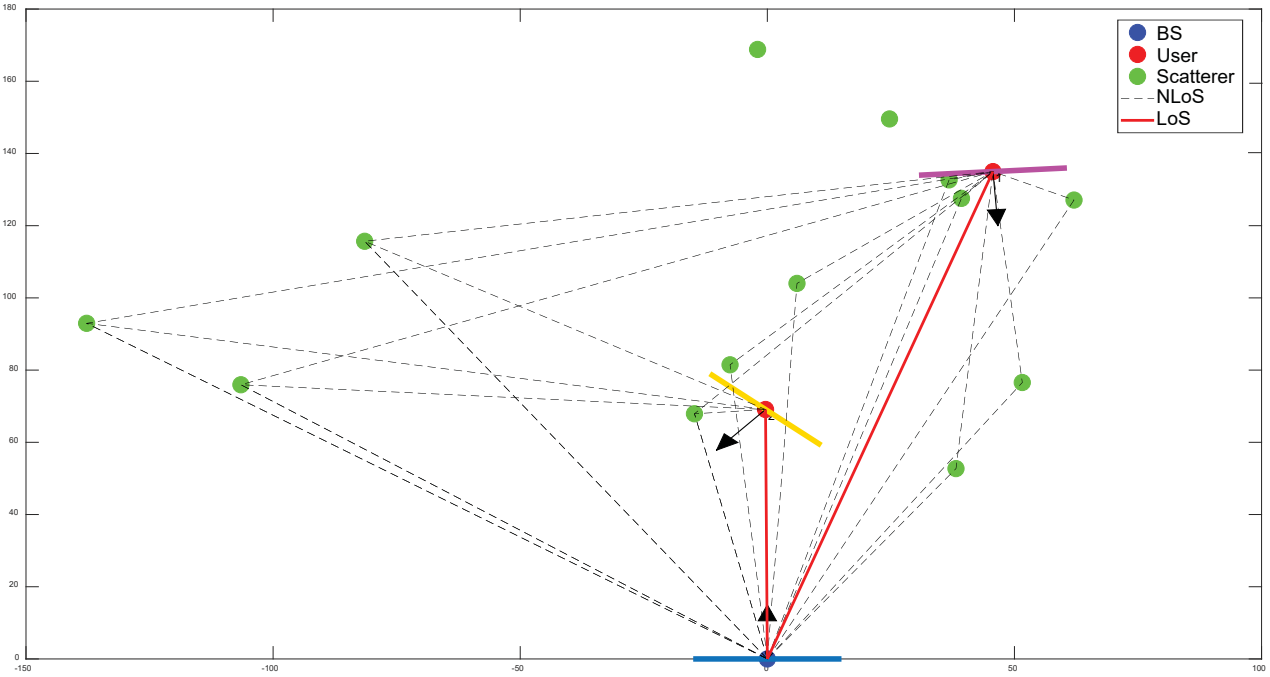


Figure 5.22: The schematic diagram of the fixed scattering environment.

where $\left[\eta\left(\mathbf{r}_t;\theta_t,\sigma_t\right)\right]_j = \eta\left(\left[\mathbf{r}_t\right]_j;\theta_t,\sigma_t\right)$, for $1 \leq j \leq N$. Moreover, we consider the shrinkage function $\eta\left(\cdot;\cdot\right)$ that corresponds to MSE-optimal denoisers under zreo-mean Bernoulli-Gaussian (BG) priors. That is, $\hat{x} = \mathsf{E}\left\{x|\, r\right\}$, where x has the BG prior

$$p\left(x;\gamma,\phi\right) = \left(1-\gamma\right)\delta\left(x\right) + \gamma\mathcal{N}\left(x;0,\phi\right), \tag{5.42}$$

and r is an AWGN-corrupted measurement of x, i.e.,

$$r = x + e \text{ for } e \sim \mathcal{N}\left(0,\sigma^2\right). \tag{5.43}$$

The MSE-optimal denoiser is then (see equation (39) in [89])

$$\hat{x} = \frac{r}{(1+\frac{\sigma^2}{\phi})(1+\frac{1-\gamma}{\gamma}\frac{\mathcal{N}\left(r;0,\sigma^2\right)}{\mathcal{N}\left(r;0,\sigma^2+\phi\right)})}. \tag{5.44}$$

Then, to turn (5.44) into a learnable shrinkage function, i.e., the LAMP algorithm [89], we set $\theta_1 = \phi$ and $\theta_2 = \log\frac{1-\gamma}{\gamma}$ and then simplify (5.44) as (see equation (40) in [89])

$$\left[\eta(\mathbf{r};\boldsymbol{\theta},\sigma)\right]_j = \frac{r_j}{(1+\frac{\sigma^2}{\theta_1})(1+\sqrt{1+\frac{\theta_1}{\sigma^2}}\exp[\theta_2 - \frac{r_j^2}{2\sigma^2(1+\frac{\sigma^2}{\theta_1})}])}, \tag{5.45}$$

where the trainable parameters include θ_1 and θ_2, i.e., $\boldsymbol{\theta} = [\theta_1,\theta_2]$.

Finnaly, for the developed MMV-LAMP algorithm, we consider a typical MMV CS problem

$$\mathbf{Y} = \mathbf{A}\mathbf{X} + \mathbf{N}, \tag{5.46}$$

where $\mathbf{Y} \in \mathbb{C}^{M\times K}$ is a noisy measurement, $\mathbf{A} \in \mathbb{C}^{M\times N}$ is a measurement matrix, $\mathbf{X} \in \mathbb{C}^{N\times K}$ denotes the sparse matrix satisfying that its columns $\left\{\mathbf{X}\left(:,i\right)\right\}_{i=1}^{K}$ share the

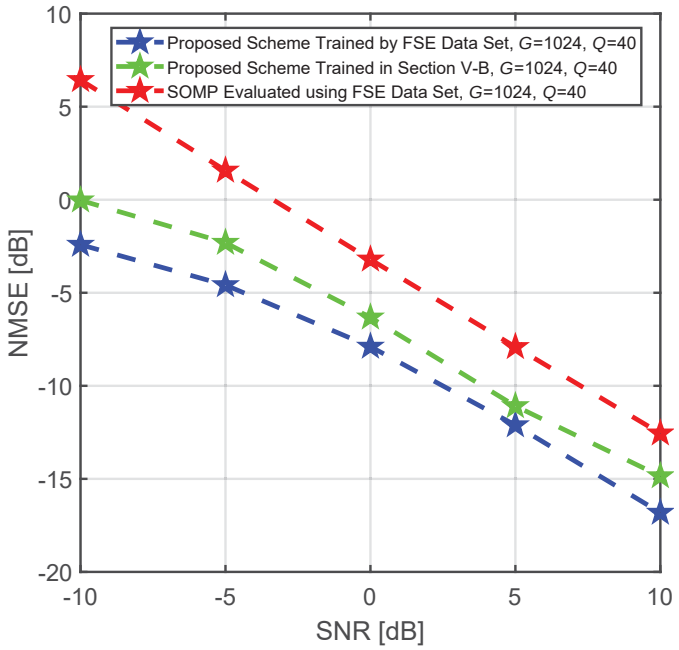


Figure 5.23: NMSE performance comparison of different channel estimation schemes versus SNRs.

common sparsity, and $\mathbf{N} \in \mathbb{C}^{M\times K}$ is the AWGN. According to [97], we developed the MMV-LAMP algorithm as

$$\mathbf{R}_t = \widehat{\mathbf{X}}_{t-1} + \mathbf{B}\mathbf{V}_{t-1}, \tag{5.47a}$$

$$\widehat{\mathbf{X}}_t = \eta\left(\mathbf{R}_t; \boldsymbol{\theta}, \sigma_t\right), \tag{5.47b}$$

$$\mathbf{V}_t = \mathbf{Y} - \mathbf{A}\widehat{\mathbf{X}}_t + b_t\mathbf{V}_{t-1}, \tag{5.47c}$$

where $\mathbf{V}_0 = \mathbf{Y}$, $\widehat{\mathbf{X}}_0 = \mathbf{0}$, and

$$\sigma_t = \frac{1}{\sqrt{MK}}\|\mathbf{V}_{t-1}\|_F, \tag{5.48}$$

$$b_t\mathbf{I} = \frac{1}{M}\sum_{j=1}^{N}\frac{\partial\left[\eta\left(\mathbf{R}_t; \boldsymbol{\theta}, \sigma_t\right)\right]_j}{\partial\left[\mathbf{R}_t\left(j,:\right)\right]}. \tag{5.49}$$

Similarly, we consider the zero-mean BG prior and the MSE-optimal denoiser is then (see equations (24)-(26) in [97])

$$\hat{\mathbf{x}} = \frac{\mathbf{r}}{(1+\frac{\sigma^2}{\phi})(1+\frac{1-\gamma}{\gamma}\frac{\mathcal{N}\left(\mathbf{r};0,\sigma^2\mathbf{I}_K\right)}{\mathcal{N}\left(\mathbf{r};0,\left(\sigma^2+\phi\right)\mathbf{I}_K\right)})}, \tag{5.50}$$

where $\hat{\mathbf{x}} \in \mathbb{C}^{K\times 1}$. We set $\theta_1 = \phi$ and $\theta_2 = \log\frac{1-\gamma}{\gamma}$ and then the corresponding shrinkage function $\eta\left(\cdot;\cdot\right)$ can be expressed as

$$[\eta(\mathbf{R}_t; \boldsymbol{\theta}, \sigma_t)]_j = \frac{\mathbf{r}_{t,j}}{\pi_t[1+\exp(\psi_t - \frac{\mathbf{r}_{t,j}^{\mathsf{H}}\mathbf{r}_{t,j}}{2\sigma_t^2\pi_t})]}, \tag{5.51}$$

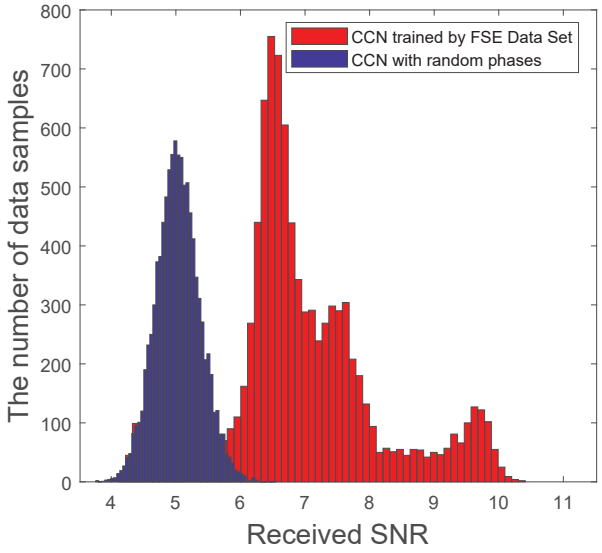


Figure 5.24: The received SNR distributions at the user side using different CCNs.

where $\mathbf{r}_{t,j} = \mathbf{R}_t(j,:)$ denotes the j-th row of the $\mathbf{R}_t$, π_t and ψ_t are respectively given by

$$\pi_t = 1 + \frac{\sigma_t^2}{\theta_1}, \tag{5.52}$$

$$\psi_t = K \log(1 + \frac{\theta_1}{\sigma_t^2}) + \theta_2. \tag{5.53}$$

The trainable parameters of the developed MMV-LAMP network include $\mathbf{B}$ and $\boldsymbol{\theta}$, where $\boldsymbol{\theta} = [\theta_1, \theta_2]$.

Chapter 6

A Machine Learning Tutorial

6.1 Introduction

As a response to the spectrum scarcity problem that was created due to the aggressive proliferation of wireless devices and quality-of-service (QoS) and quality-of-experience (QoE) hungry services, which are expected to support a broad range of diverse multi-scale and multi-environment applications, sixth-generation (6G) wireless networks adopt higher frequency bands, such as terahertz (THz) that ranges from 0.1 to 10 THz) [98–101]. In more detail and according to the IEEE 802.15.3d standard [102, 103], THz wireless communications are recognized as the pillar technological enabler of a varied set of use cases stretching from in-body nano-scale, to indoor and outdoor wireless personal/local area and fronthaul/backhaul networks. Nano-scale applications require compact transceiver designs and self-organized ad-hoc network topologies. On the other hand, macro-scale applications demand flexibility, sustainability, adaptability in an ever changing heterogeneous environment, and security. Moreover, supporting high data-rate that may reach 1 Tb/s, and energy-efficient massive connectivity are only some of the key demands. To address the aforementioned requirements, artificial intelligence (AI), in combination with novel structures capable of altering the wireless environment, have been regarded as complementary pillars to 6G wireless THz systems.

AI is expected to enable a series of new features in next-generation networks, including, but not limited to, self-aggregation, context awareness, self-configuration as well as opportunistic deployment [104]. In addition, integrating AI in wireless networks is envisioned to bring a revolutionary transformation of conventional cognitive radio systems into intelligent platforms by unlocking the full potential of radio signals and exploiting new degrees-of-freedom (DoF) [105, 106]. Identifying this opportunity, a significant amount of researchers turned their eyes on AI-empowered wireless systems and specifically in machine learning (ML) algorithms (see e.g., [107–110] and references therein). In more detail, in [107], the authors reviewed the thirty-year history of ML highlighting its application in heterogeneous networks, cognitive radios, device-to-device communications and Internet-of-Things (IoT). Likewise, in [108], big data analysis is combined with ML in order to predict the requirements of mobile users and enhance the performance of "social network-aware wireless." Moreover, in [109], a brief survey was conducted that summarizes the basic physical layer (PHY) authentication schemes that employ ML in fifth generation (5G)-based IoT. Finally, in [110], the au-

thors provided a systematic and comprehensive review of the ML approaches that was employed to address a number of nano-scale biomedical challenges, including once that refer to molecular and nano-scale THz communications.

The methodologies, which are presented in the aforementioned contributions, are tightly connected to the communication technology characteristics and building blocks of radio and microwave communication systems and networks, which are inherently different from higher frequency bands that can support higher data rates, and propagation environments with conventional and unconventional structures, like RIS. Additionally, in order to quantify the AI-approaches efficiency, new key performance indicators (KPIs) need to be defined. Motivated by this, this chapter focuses on reporting the role of AI in THz wireless networks. In particular, we first identify the THz wireless systems particularities that require the adoption of AI. Building upon this, we present a brief survey that summarizes the contributions in this area and focus on indicative AI approaches that are expected to play an important role in different layers of the THz wireless networks. Finally, possible future research directions are provided.

Notations: In this paper, matrices are denoted in bold, capital letters, while vectors in bold, lower case letters. The base-10 logarithm of x is given by $\log(x)$. Additionally, $\frac{\partial f(x_1,\cdots,x_N)}{\partial x_i}$ stands for the the partial derivative of $\partial f(x_1,\cdots,x_N)$ with respect to x_i, with $i \in [1, N]$. The operator $\max(x_1, x_2, \cdots, x_N)$ yields the numerically largest of the x_i, while $\min(x_1, x_2, \cdots, x_N)$ returns the numerically smallest of the x_i. Moreover, $\sqrt{x}$ represents the square root of x. The index of the value of $\mathbf{x}$ that maximizes and minimizes $f(\mathbf{x})$ are respectively given by $\arg\max f(\mathbf{x})$ and $\arg\min f(\mathbf{x})$. The expected value of $f(x)$ is represented as $\mathbb{E}[\cdot]$.

6.2 The role of ML in THz wireless systems and networks

Together with the promise of supporting high data-rate massive connectivity, THz wireless systems and networks come with several challenges. In particular, these challenges can be summarized as:

- Due to the high transmission frequency, i.e. the small wavelength, in the THz band, we can design high-directional antennas (with gains that may surpass 30 dBi) with unprecedented low beamwidths, which may be less than 4^o[1]. These antennas are used to counterbalance the high channel attenuation by establishing high-directional links. On the one hand, high directionality creates additional DoFs, which, if they are appropriately exploited, they can enhance both dynamic spectrum access and network densification; thus, boost its connectivity capabilities. In this direction, new approaches to support intelligent interference monitoring and cognitive access are required. On the other hand, high directionality comes with the requirement of extremely accurate beam alignment between the fixed and moving communication nodes. To address this, beam tracking and channel estimation approaches of high latency need to be developed, which account for the latency requirement.

[1] It is worth noting that in mmW wireless systems, antennas with beamwidths that are in the range of 10^o are employed.

- Molecular absorption causes frequency- and distance-dependent path loss, which creates frequency windows that are unsuitable for establishing communication links [111]. As a consequence, despite the high bandwidth availability in the THz band, windowed transmission with time varying loss and per-window adaptive bandwidth as well as power usage is expected to be employed. This characteristic is expected to influence both beamforming design as well as resource allocation and user association.

- The large penetration loss in the THz band, which may surpass 40 or even 50 dB [112–116], renders questionable the establishment of the non-LoS links. As a result, blockage avoidance schemes are needed. Note that in lower frequency bands, such as mmW, the penetration loss is in the range of 20 to 30 dB [117–119].

- To exploit the spatial dimension of THz radio resources, support MU-connectivity as well as increase the link capacity in heterogeneous environments of moving nodes, suitable beamforming and mobility management designs that predict the number and motion of UEs need to be designed. Moreover, in mobile scenarios, accurate channel state information (CSI) is needed. In lower-frequency systems, this is achieved by performing frequent channel estimation. However, in THz wireless systems, due to the small transmission wavelength, the channel can be affected by slight variations in the micrometer scale [120]. As a result, the frequency of channel estimation is expected to be extremely high and the corresponding overhead unaffortable. This motivates the design of prediction-based channel estimation and beam tracking approaches.

- From the hardware point of view, high-frequency transceivers suffer from a number of hardware imperfections. In particular, the EVM of THz transceiver may even reach 0.4 [121]. It is questionable that conventional mitigation approaches would be able to limit the impact of hardware imperfections. As a result, smarter signal detection approaches need to be developed. Of note, as stated in [122–124], in lower-frequency communication systems including mmW, the EVM does not exceed 0.17.

- THz wireless systems are expected to support a large variety of applications with diverse set of requirements as well as UEs equipped with antennas of different gains. To guarantee high availability and association rate that tends to 100%, association schemes that take into account all together the nature of THz resource block, the UEs' transceivers capabilities, as well as the applications requirements, need to be presented. Moreover, to ensure uninterrupted connectivity, these approaches should be highly adaptable to network topology changes. In other words, they should be able to predict network topologies changes and pre-actively perform hand-overs.

- Conventional routing strategies account neither the communication nodes distance nor their memory limitations. However, in THz mobile scenarios, where both the transmission range is limited to some decades of meter and the device memory is comparable to the packet length, routing may become a complex optimization problem.

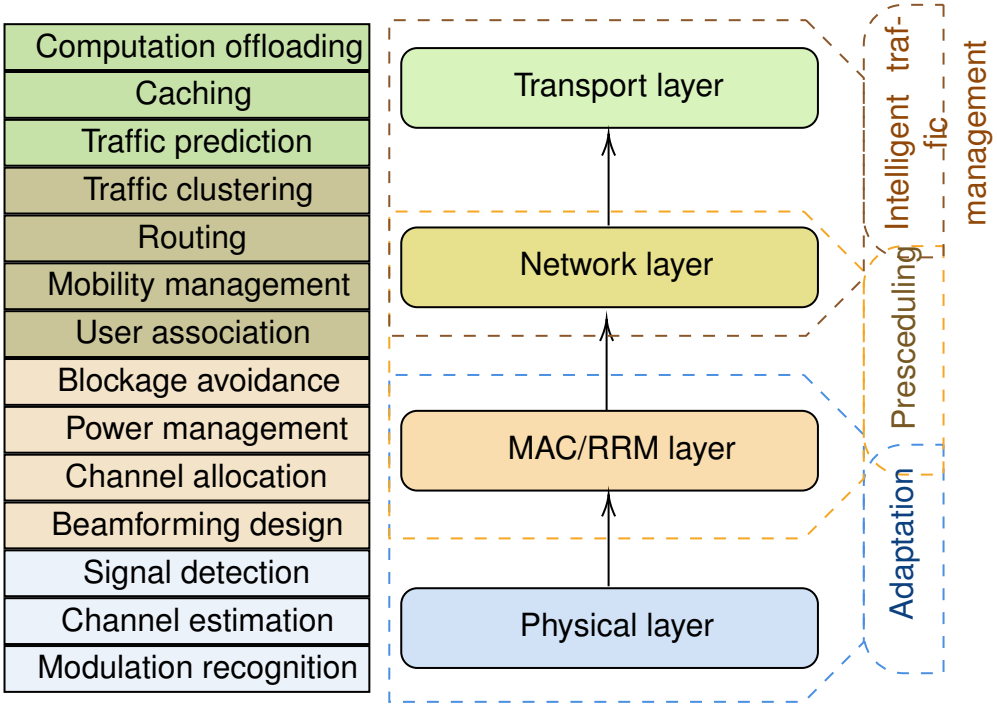


Figure 6.1: ML-based applications to different layers of THz wireless systems and networks.

- Finally, for several realistic scenarios, the aggregated data-rate of the fronthaul is expected to reach $1\ \mathrm{Tb/s}$, which is comparable with the backhaul's achievable data-rate. This may cause service latency due to data congestion in network nodes. To avoid this intelligent traffic prediction and caching strategies need to created to pre-actively bring the future-requested content near the end-user.

The rest of this section is focused on explaining the role of ML in THz wireless systems and networks by presenting the current state-of-the-art. As illustrated in Fig. 6.1 and in order to provide a comprehensive understanding of the need to formulate ML problems and devise such solutions, we classify the ML problems into four categories, namely i) PHY, ii) MAC and RRM, and iii) Network. Of note, this classification is in line with the open system interconnection (OSI) model. For each category, we identify the communications and networks problems for which ML solutions have been proposed as well as the needs of utilizing them. Finally, this section provides a review on the ML problems and solutions that have been employed. Apparently, some of the aforementioned challenges have been also discussed in lower-frequency systems and networks. For the sake of completeness, in our literature review, we have also included contributions that although refer to lower frequency, can find application to THz wireless systems and networks.

6.2.1 PHY layer

The additional DoFs, which have been brought by the ultra-wide band THz channels as well as their spatial nature, allow us to establish high data-rate links with limited

transmission power. Moreover, advances in the fields of communication-components designs, fueled by new artificial materials, such as reconfigurable intelligent surfaces (RISs), created a controllable wireless propagation environment; thus, offered new opportunities in simplifying the PHY layer processes and further increased the systems DoFs [125]. Finally, the use of direct conversion architectures (DCAs) in both transceivers created the need to utilize digital signal processing (DSP) algorithms in order to "decouple" the system's reliability from the hardware imperfection related degradation. The computational complexity of these algorithms increases as the modulation order of the transmission signal increases. The aforementioned factors create a rather dynamic high-dimensional complex environment with processes that are hard or even impossible to be analytically expressed. Motivated by this, the objectives of employing ML in the PHY layer is to provide adaptability to an ever changing wireless environment consisted of heterogeneous components (transceivers, RIS, active and passive relays, etc.) and to countermeasure the impact of transceivers hardware imperfections with increasing neither the communication nor the computation overheads.

In this direction, several researcher have focused on presenting ML-related solutions for automatic modulation recognition (AMR) [126–132], *channel estimation* [133–140], and *signal detection* [141–145]. In more detail, AMR has been identified as an important task for several wireless systems, since it enables dynamic spectrum access, interference monitoring, radio fault self-detection as well as other civil, government, and military applications. Moreover, it is considered as a key enabler of intelligent/cognitive nano- and macro-scale receivers (RXs). Fast AMR can significantly improve the spectrum utilization efficiency [146]. However, it is a very challenging tasks, since it depends to the variation of the wireless channel. This aspires the introduction of intelligence in this task. In this sense, in [126], the authors employed convolutional NNs (CNNs) and recurrent NNs (RNNs) in order to perform AMR. As inputs, the algorithms used the in-phase and quadrature (I/Q) components of the unknown received signal and classified it to one of the following modulation schemes: binary frequency shift keying (BFSK), differential quadrature phase shift keying (DQPSK), 16 quadrature amplitude modulation (16-QAM), quaternary pulse amplitude modulation (4PAM), minimum shift keying (MSK), Gaussian minimum shift keying (GMSK). Moreover, in [127], a deep NN (DNN) approach was discussed for identifying the received signal modulation in coherent RXs. The DNN is comprised of two auto-encoders and an output perceptron layer. To train and verify the ML network, two datasets of numerous amplitude histograms are used. After training, the network is capable of accurately extracting the modulation format of the signal after receiving a number of symbols.

Similarly, in [128], the authors presented a DNN approach that is based on RNNs with short memory and is capable of to exploit the temporal and spatial correlation of the received samples in order to accurately extract the their modulation type. In this contribution, as an input, the ML algorithm requires a predetermined number of samples. Meanwhile, in [129], the authors reported a extreme supervised-learning ML algorithm capable of accurately and time-efficiently estimating the modulation type of the received samples. The disadvantage of the aforementioned approaches is that the training period is large and if the characteristics of channel changes, for example due to a RIS reconfiguration, a partial blockage phenomenon or the user equipment movement, the algorithm need to be re-trained.

Additionally, in [130], a semi-supervised deep convolutional generative adversarial network (GAN) was presented that consists of a pair of GANs that collaboratively cre-

ate a powerful modulator discriminator. The ML network receives as inputs the I/Q components of a number of received signal samples and matches them to a set of modulation formats. The main disadvantage of this approach is its high-computational resource demands as well as its sensitivity to received signal distribution variations. To deal with the data distribution variation, Bu et al. [131] introduced an adversarial transfer learning architecture that exploits transfer learning and is capable of achieving accuracy comparable to the ones of supervised learning approaches. However, this approach demands careful handling of former knowledge since there may exist differences between wireless environments. In [131], the ML algorithm uses as an input the (I/Q) components of the received signal. Finally, in [132], an expectation maximization (EM) algorithm was employed in order to perform modulation mode detection and systematically differentiate between pulse- and carrier-based modulations. The presented results revealed the existence of a unique Pareto-optimal point for both the SNR and the classification threshold, where the error probability is minimized.

Another important task in PHY layer is *channel estimation and tracking*. In particular, in order to employ high directional beaforming in THz wireless systems, it is necessary to acquire channel information for all the transmitter (TX) and RX antenna pairs. The conventional approach that is supported by several standards, including IEEE 802.11ad and IEEE 802.11ay [147–149], depends on creating a set of transmission and reception beamforming vector pairs and scanning between them in order to identify the optimal one. During this process, the access point (AP) sends synchronization signals using all its possible beamforming vectors, while the user equipment (UE) performs energy detection in all possible reception directions. In the end of this process, the UE determines the optimal AP beamforming vector and feed it back to the AP. Next, the roles of the AP and UE interchange in order to allow the AP to identity the optimal UE beamforming vector and feed it back to it. Notice that in the second phase, the AP locks its RX to its optimal beamforming vector. Then, channel estimation is performed using the optimal beamforming pair and a classical DSP technique (e.g., minimum least square error, minimum mean square error, etc.). Let us assume that the AP and the UE respectively have L_A and L_U available beamforming vectors and that N_a and N_e received signal samples are needed for energy detection and channel estimation. As a consequence, the latency and power consumption due to channel estimation is proportionally to $L_A L_U N_a + L_U N_a + N_e + 2$. This indicates that as the number of the antennas at TX and RX increases, the number of beamforming pairs also increases; thus, the training overhead significantly increase and the conventional channel estimation approach becomes more complex as well as time and power inefficient.

To better understand the importance of this challenge in THz wireless systems, let us refer to an indicative example. Assume that we would like to support a virtual reality (VR) application for which the transmission distance between the AP and the UE is $20\ \mathrm{m}$, a data-rate of $20\ \mathrm{Gb/s}$ is required with an uncoded bit error rate (BER) in the order of 10^{-6}. Furthermore, a false-alarm probability that is lower than 1% is required. This indicates that $N_a \geq 100$. Let us assume that the transmission bandwidth is $10\ \mathrm{GHz}$, the transmission power is set to $10\ \mathrm{dBm}$, while the receives mixer convention and miscellaneous losses are respectively 8 and $5\ \mathrm{dB}$. Additionally, the RX's low noise amplifier (LNA) gain is $25\ \mathrm{dB}$, whereas the RX's mixer and LNA noise figures are respectively 6 and $1\ \mathrm{dB}$. If the transmission frequency is $287.28\ \mathrm{GHz}$, both the TX and RX need to be equipped with antennas of $35\ \mathrm{dBi}$ gain. Such antennas have a beamwidth of 3.6^o;

hence, $L_A = L_U = 100$. Also, by assuming that $N_e = 100$, the channel estimation latency becomes equal to approximately 1 ms. Of note, VR requires a latency that is lower than 1 μs.

To address the aforementioned problem, researchers turned their attention to regression and clustering ML-based methods. Specifically, in [133], the authors employed a DNN-based algorithm to predict the user's channel in a sub-THz multiple-input multiple-output (MIMO) vehicular communications system. The ML algorithm of [133] takes as input the signals received by a predetermined number of APs and outputs a vector containing the estimated channel coefficients. Similarly, in [134], the authors reported a deep learning compressed sensing (DLCS) algorithm for channel estimation scheme in multi-user (MU) massive MIMO sub-THz systems. The DLCS is a supervised learning algorithm that takes as input a simulation-based generated received signal vector as well as real measurements and performs two functionalities, i.e. beamspace channel amplitude estimation; and ii) channel reconstruction. The results indicate that this approach can achieve a minimum mean square error that is comparable with the one of the orthogonal matching pursuit scheme. Likewise, in [135], a manifold learning-extreme learning machine was presented for estimating high-directional channels. In more detail, manifold learning was employed for dimentionality reduction, whereas the extreme learning algorithm with one-shot training was employed for channel state information estimation. Moreover, in [136], the authors presented a Gaussian process based ML (GPML) algorithm to predict the channel in an unmanned aerial vehicle (UAV) aided coordinated multiple point (CoMP) communication system. The main idea was to provide real-time predictions of the location of the UAV and reconstruct the line-of-sight (LOS) channel between the AP and the UAV. Similarly, in [137], a sparse Bayesian learning algorithm was introduced to estimate the propagation parameters of the wireless system. Meanwhile, in [138], the authors presented a neural network (NN)-based algorithm for channel prediction and showed that, after sufficient training, it can faithfully reproduce the channel state.

Likewise, in [139], a NN-based algorithm, which consists of three hidden layers and one fully-connected layer, was reported to obtain the beam distribution vector and reproduce the channel state. The algorithm uses as inputs the (I/Q) components of the received signal. It is trained offline using simulations and is able to achieve similar performance in terms of total training time slots and the spectral efficiency with previously proposed approaches, like adaptive compressed sensing, hierarchical search, and multi-path decomposition. In order for this approach to work properly, the simulation and real world data should follow the same distribution. Also, the propagation parameters of both types of data should coincide. Furthermore, in [140], the authors presented a deep denoising NN assisted compressive sensing broadband channel estimation algorithm that exploits the relation of angular-delay domain MIMO channels in sub-THz RIS-assisted wireless systems. In particular, the algorithm takes as inputs the received signals at a number of active elements of the RIS and forward them to a compressive sensing block, which feeds the NN. The proposed approach outperformed the well-known simultaneous orthogonal match pursuit (SOMP) algorithm in terms of normalized mean square error (NMSE). The main disadvantage of deep denoising NN approach is that it is not adaptable to changes in the propagation environment characteristics. Furthermore, in [150], Li et al. reported a deep leaning architecture for channel estimation in RIS-assisted THz MIMO systems. The idea behind this approach was to convert the channel estimation problem into the sparse recovery problem by

exploiting the space nature of THz wireless channels. In this direction, the algorithm uses received signals that carry pilot symbols for training the (I/Q) components. In the operation phase, the (I/Q) components of the received signal are used as inputs. Finally, in [151], the authors employed k-nearest neighbors (kNN) and support vector classifiers (SVC) to estimate the angle-of-arrival (AoA) in hybrid beamforming wireless systems.

Signal detection is another PHY layer task. Conventional approaches require accurate estimation of both the channel model and the impact of hardware imperfections in order to design suitable equalizers and detectors. However, as the wireless environment becomes more complex and the influence of hardware imperfections become more severe, due to the high number of transmission and reception antennas, data detection becomes more challenging. This fact motivates the study of ML-based solution for end-to-end signal detection, in which no channel and hardware imperfections equalization is required. In this sense, in [141], the authors employed a DNN architecture for data detection, whereas, in [142], the effectiveness of deep learning for end-to-end signal detection was reported. Similarly, in [143], the authors presented a deep learning assisted approach for beam index modulation detection in high-frequency massive MIMO systems. Additionally, in [144], the authors demonstrated the use of deep learning assisted soft-demodulator in multi-set space-time shift keying millimeter wave (mmW) wireless systems. To cancel the impact of hardware imperfections without employing equalization units, a supervised ML signal detection approach was presented in [145].

To sum up, PHY-specific ML algorithms usually employ as inputs I/Q components of the received signal samples. In case of modulation recognition, due to the lack of unlabeled data for training, unsupervised learning approaches are adopted. On the other hand, in channel estimation and signal detection, pilot signals can be exploited to train the ML algorithm. As a consequence, for channel estimation, supervised learning approaches are the usual choice. In particular, as observed in Table 6.1, both supervised and unsupervised learning approaches were applied that return classification, regression, dimensionality reduction and density estimation rules. Interestingly, it is observed that for signal detection only DNN-based algorithms have been reported. Finally, it is worth noting that the main requirement for the algorithm selection in most cases was to provide high adaptation to the THz wireless system. It is worth mentioning that, for the sake of competence, Table 6.1 includes some contributions that although do not refer to THz wireless systems, they can be straightforwardly applied to them. For example, [126, 129] and [130] can be applied to any system that employ I/Q modulation and demodulations approaches, whereas [127] is suitable for the ones that use coherent RXs. Of note, according to [98], coherence RXs are a usual choice in THz wireless fiber extenders. Similarly, since an inherent characteristic of THz channels is the high temporal correlation, the ML methodology presented in [128] is expected to find application in THz wireless systems. Additionally, despite the fact that the ML algorithms in [133–137, 139, 140, 143, 144] were applied to mmW systems and exploit the spatio-temporal and directional characteristics of the channels in this band, the same approach can be used in THz systems that have the similar particularities. Finally, the ML approach presented in [145] can be applied in any MIMO regardless the operating frequency.

Table 6.1: ML algorithm types applied in PHY

	NN	CNN	RNN	DNN	GAN	DLCS	GPML	Bayesian learning	SOMP	kNN	EM
AMR											
[126]	–	✓	✓	–	–	–	–	–	–	–	–
[127]	–	–	–	✓	–	–	–	–	–	–	–
[128]	–	–	–	✓	–	–	–	–	–	–	–
[129]	✓	–	–	–	–	–	–	–	–	–	–
[130]	–	–	–	–	✓	–	–	–	–	–	–
[131]	–	–	–	–	✓	–	–	–	–	–	–
[132]	–	–	–	–	–	–	–	–	–	–	✓
Channel estimation and beam tracking											
[133]	–	–	–	✓	–	–	–	–	–	–	–
[134]	–	–	–	–	–	✓	–	–	–	–	–
[135]	✓	–	–	–	–	–	–	–	–	–	–
[136]	–	–	–	–	–	–	✓	–	–	–	–
[137]	–	–	–	–	–	–	–	✓	–	–	–
[138]	✓	–	–	–	–	–	–	–	–	–	–
[139]	✓	–	–	–	–	–	–	–	–	–	–
[140]	–	–	–	–	–	–	–	–	✓	–	–
[150]	–	–	–	✓	–	–	–	–	–	–	–
[151]	–	–	–	✓	–	–	–	–	–	✓	–
Signal detection											
[141]	–	–	–	✓	–	–	–	–	–	–	–
[142]	–	–	–	✓	–	–	–	–	–	–	–
[143]	–	–	–	✓	–	–	–	–	–	–	–
[144]	–	–	–	✓	–	–	–	–	–	–	–
[145]	–	–	–	✓	–	–	–	–	–	–	–

6.2.2 MAC and RRM layer

MAC and RRM layers are responsible of providing uninterrupted high quality of experience (QoE) to mutliple end users. In contrast to lower-frequency communications, where omni-directional or quasi-omni-directional links are established, in THz wireless systems both the AP/BS and the UE employ beamforming. As a result, an additional to time and frequency resource, i.e. space, is created. The optimal exploitation of the tertiary nature of the channel require the design of beamforming approaches[2] and channel allocation strategies. Moreover, to satisfy the end user's data rate demands and to support non-orthogonal multiple access (NOMA), power management policies are required. Finally, to address the impact of blockage, environmental awareness need to be obtained by the wireless THz network and proactive blockage avoidance mechanisms are a necessity. Motivated by the above, the rest of this section discuss ML-based beamforming designs, channel allocation strategies, power management schemes and blockage avoidance mechanisms.

Beamforming design is a crucial task for MIMO and MU-MIMO THz wireless systems. However, conventional beamforming approaches strongly rely on accurate channel estimation; as a result, their complexity is relatively high. To simplify the beamfoming design process, a great amount of research effort was put on investigating ML-based approaches. In more detail, in [153], the authors presented a DNN for determining the optimal beamforming vectors that maximizes the sum rate in a two-user multiple-input single-output (MISO) wireless system. The ML algorithm uses as inputs the real and imaginary part of the complex MISO channel coefficients as well as the transmitted power. For training the ML model, the cross-entropy is used as a cost function, which evaluates the errors by calculating the difference between probability distributions labeled and model outputted data. Three DNN architectures to approximate the hybrid beamformer's singular value decomposition, with varying levels of complexity were discussed in [154]. The architectures take as inputs the real and imaginary components of the MIMO channel coefficients. The first architecture predicts a predetermined number of the most important singular values and vectors of a given channel matrix by employing a single DNN. The second architecture employs k DNNs. Each of them returns the largest singular value and corresponding right and left singular vectors of the MIMO channel matrix. The third architecture is suitable for channel matrices of rank-1 and outputs a predetermined number of singular values and vectors by recursively using the same DNN. The architectures are trained through comparison of the extracted channel matrices to the channel matrices extracted by the ML-models. The proposed ML-based architectures were shown, by means of Monte Carlo simulations, to improve the system's date rate by up to $50 - 70\%$ compared to conventional approaches.

The main disadvantage of the ML-architectures presented in [154] is that they require a large number of estimations of channel matrices for training, which may generate an unaffortable latency in a fast changing environment. To counterbalance this, in [155], an unsupervised K-means algorithm is employed, which exploits the electric-field response of each antenna element in order to design beam codebooks that opti-

[2]It is worth noting that in lower-frequency systems, beamforming design has been usually considered to be part of the PHY layer. However, according to IEEE 802.11.3ay [152], since beamforming manages the spatial resources of THz systems, although it is conducted at the baseband, part of it belongs to the PHY, while the other to the MAC layer.

mize the average received power gain of UEs that are located within a cluster. Although this approach does not require large training periods, it is sensitive to drastical changes of the wireless environment, which may arise due to users or scatterers/blockers movement. Meanwhile, in [156], the authors formulated an interference managing problem by means of coordinated beamforming in ultra-dense networks that aims at "almost real-time" sum rate maximization. To solve the aforementioned problem, a Q-learning based ML algorithm was introduced, which only require large scale channel fading parameters and achieves similar results to the ones of the corresponding analytical approach that demands full channel state information. Note that the Q-learning algorithm in [156] takes as input the state of the system, i.e. a logarithmic transformation of the second-norm of the MIMO channel matrix. Likewise, in [157], the authors reported a centralized deep learning based algorithm for coordinated beamforming vector design in high-mobility and high-directional wireless systems. This algorithm uses as inputs the I/Q components of a virtual omni-directional received signal and extracts a prediction for the optimal beamforming vectors. To train the deep-learning algorithm pilot symbols are employed, which are exchanged between the UE and basestations (BSs) within the coherence time. This approach provided high-adaptation with the cost of creating important overhead due to the message exchange need from/to BSs to/from a central/cloud processing unit. In [158], a supervised NN was employed, which is based on singular value decomposition, to design hybrid beamformers in massive MIMO systems that are capable of mitigating the impact of limited resolution phase shifters. The proposed approach is executed in a single BS and achieve a higher spectral efficiency compared with unsupervised learning ones. However, the performance enhancement demands a relatively high training period.

To address this inconvenience, in [159], the authors presented a reinforcement learning algorithm to jointly-design the analog and digital layer vectors of a hybrid beamformer in large-antenna wireless systems. The algorithms take as input the achievable data rate and returns the phase shifts of each antenna element. The disadvantage of this algorithm is that it is unable to achieve the same performance as the corresponding supervised ML one. Moreover, in [160], the authors studied the use of extreme learning machine for jointly optimizing transmit and receive hybrid beamforming in MU-MIMO wireless systems. The algorithm requires as inputs the real and imaginary parts of the MIMO channel coefficients and returns the an optimal beamforming vectors estimation. As a training cost function the difference between the targeted and achievable SNR is used. In [161], the extreme learning and NNs were employed in order to extract the transmit and receive beamforming vectors in full-duplex massive MIMO systems.

In [162], a CNN with quantized weights (Q-CNN) algorithm was utilized as a solution to the problem of jointly designing transmit and receive hybrid beamforming vectors. As input, the real and imaginary components of the MIMO channel matrix is used. Q-CNN has limited memory and low-overhead demands; hence, it is suitable for deployment in mobile devices. Furthermore, in [163], three NN-based approaches for designing hybrid beamforming schemes were reported. The first one is based on mapping various hybrid beamformers to NNs and thus transforming the beamforming codeword design non-convex optimization problem into a NN training one. The second approach is an extension of the first one that aims at optimizing the beam vectors for the case of MU access. In comparison to the aforementioned approach, the third one takes into account the hardware limitations, namely low-resolution phase shifters

and analog-to-digital converters (ADCs). Simulation results revealed that the proposed approaches outperform analytical ones in terms of BER. All the aforementioned approaches in [163] require as input the MIMO channel matrix. In [164], the authors presented a support vector machine (SVM) algorithm for analog beam selection in hybrid beamforming MIMO systems that uses as inputs the complex coefficients of the channel samples. This approach provides near-optimal uplink sum rates with reduced complexity compared to conventional strategies. On the other, it requires sufficient training data that leads to high training periods, when the characteristics of the wireless channel changes.

To countermeasure the aforementioned problem, unsupervised learning approaches were employed in several works [165–167]. In more detail, in [165], an autoencoding-based SVD methodology was used in order to estimating the optimal beamforming codes at the TX and RX, while, in [166], Lin et al. introduced a deep NN architecture for beamforming design that outperforms several previously presented deep learning approaches. Additionally, in [167], a ML-based clustering strategy with feature selection was employed to design three dimensional (3D) beamforming. In particular, the algorithm has three steps. In the first step, it uses pre-collected data in order to obtain a set of eigenbeams, while, in the second one, the aforementioned sets are used to estimate the channel state information, which in the third step is feed to a Rosenbrock search engine. The two first steps are executed offline, whereas, the third one is an online process. As the number of antenna elements increases, the size of the eigenbeam vector set also increases; thus, the practicality of this approach may be questionable in massive MIMO systems. All the aforementioned ML algorithms employ as input the MIMO channel matrix.

ML was also employed to optimize the beamforming vectors of relaying and reflected assisted high-directional THz links. For example, in [168], Li et al. presented a cross-entropy hybrid beamforming vector estimation deep reinforcement learning-based scheme for unmanned aerial vehicle (UAV)-assisted massive MIMO network. The deep reinforcement learning method was employed in order to minimize the AP transmission power by jointly optimizing the AP and RIS active and passive beamforming vectors. Finally, in [169], Liu et al. used Q-learning in order to jointly designing the movement of the UAV, phase shifts of the RIS, power allocation policy from the UAV to mobile UEs, as well as determining the dynamic decoding order of a NOMA scheme, in a RIS-UAV assisted THz wireless system. Both the ML algorithms utilized in [168] and [169] use as input the the estimated channel coefficient matrix.

Another challenging and important task in THz wireless networks is *channel allocation*. Of note, in this band, the radio resource block (RRB) has three dimensions, i.e., time, frequency, and space. As a result, the additional DoF, namely space, creates a more complex resource allocation problem. Aspired by this fact, several contributions studied the use ML approaches in order to design suitable resource allocation policies. Indicative examples are in [170–175]. In particular, in [170], a centralized NN was employed to return the channel allocation strategy that minimizes the co-channel interference in an ultra-dense wireless network. The NN takes as input a binary matrix that contains the user-channel association and estimates the up-link SINR. In [172], a centralized supervised cluster-based ML interference management channel allocation that takes into account the time-varying network load was introduced. As inputs to the algorithm, the RRB allocation data, the acknowledgement (ACK) and the negative acknowledgement (NACK) data collected from the network are used. The algorithm

outputs an estimation of the interference intensity.

To deal with the ever changing topology and time-varying channel conditions of ultra-dense mobile wireless networks, in [171], a deep Q-learning model was presented that uses quantized local AP and UE channel state information to cooperatively allocate the channels in a downlink scenario. The algorithm is self-adaptive and does not require any training. Additionally, in [173], a DNN was used that takes as inputs the channel state information and returns the spatial resource block (i.e. beam) allocation in massive MIMO wireless systems. Moreover, in [174], a deep learning approach was proposed for channel and power allocation in MIMO-NOMA wireless systems that aims at maximizing the sum data rate and energy efficiency of the overall network. The approach uses as inputs the channel vectors, precoding matrix and the power allocation factors. Finally, in [175], a feedforward NN that takes as inputs the uplink channel state information and returns a channel allocation strategy in a rank and power constrained massive MIMO wireless system, was employed.

For multi-antenna THz transceivers, the hardware complexity and *power management* become a burden toward practical implementation [176]. To lighten the power management process, several researchers turned their eyes on ML. For instance, in [119], the K-means algorithm was employed to cluster the users of a NOMA-THz wireless network and to maximize the energy efficiency by optimizing the power allocation. The K-means algorithm in [119] requires as inputs the number of clusters and set of users as well as the channel vectors and outputs the UE-AP association matrix. Moreover, in [177], Kwon et al. reported a self-adaptive DRL deterministic policy gradient-based power control of BS and proactive cache allocation toward BSs in distributed Internet-of-vehicle (IoV) networks. In [177], the system's state that is the input of the DRL, takes into account the available and total buffer capacity of each BS, the average e quality state of the provisioned video at each UE. Meanwhile, in [178], a transfer learning approach that is based on the Q-learning algorithm, was used to allocate power in MU cellular networks, in which each cell has different user densities. Similarly, Q-learning was employed in [179] to develop a self-organized power allocation strategy in mmW networks. Finally, in [180], Zhang et al. presented a semi-supervised learning and DNN for sub-channel and power allocation in directional NOMA wireless THz networks. The algorithm requires as inputs the set of users and their channel vectors as well as a predetermined number of clusters.

Another burden that THz wireless system face is *blockage*. Sudden blockage of the THz LOS path cause communication interruptions; thus, creates a detrimental impact on the system's reliability. Further, re-connections to the same or other BS/AP demands high beam training overhead, which in turn result to high latency. To avoid this, some contributions discuss the use of ML in order to predict dynamic (moving) obstacles position and their probability to block the LOS between the AP/BS and UE in order to proactively hand-over users to other AP/BS. Towards this direction, in [181], a reinforcement learning algorithm was used to create a proactive hand-off blockage avoidance strategy. The state of the system that is defined by the beam index of each AP/BS at every time step is used as an input to the reinforcement learning agent. Moreover, in [182], NN and CoMP clustering was employed to predict the channel state and avoid blockage. In the scenario under investigation, the authors consider dynamic blockers (i.e. cars) that perform deterministic motion. The presented algorithm use as inputs the UE location as well as the system's clock time. Similarly, in [183], stochastic gradient descent (SGD) was employed to design outage-minimization robust directional CoMP

systems by selecting communication paths that minimize the blockage probability. In this direction, the algorithm uses as inputs an initial estimation of the beamforming vector as well as the channel state information and returns the optimized beamforming vector. Finally, in [184], a DNN was employed to provide environmental awareness to an RIS-assisted wireless sub-THz network that performs beam switching between direct and RIS-assisted connectivity in order to avoid blockage. The inputs of the algorithm in [184] are the network topology and the links line-of-sight conditions.

According to Table 6.2, supervised, unsupervised, and reinforcement learning were employed to solve different problems in MAC layer. In particular, for beamforming design, where the main requirement is adaptation to the ever changing propagation environment, unsupervised learning was used to cluster UEs, supervised learning was applied to design appropriate codebooks, and reinforcement learning was employed for beam refinement and fast adaptation. As a consequence, reinforcement learning approaches are attractive for mobile and non-deterministic varying wireless environments. On the other hand, supervised learning approaches are more suitable for static environments or environments that change in deterministic way. A key requirement that supervised learning approaches have is the need to be training through a set of channel or received signal vectors that are accompanied by an achievable performance indicator. The indicator can be the data-rate, outage probability or any other KPI of interest. Likewise, unsupervised clustering and supervised learning were used for channel allocation, which was performed based on the UE communication demands. Due to lack of extensive training data sets and need of adaptation, unsupervised and reinforcement learning approaches were employed for power management. Finally, supervised and reinforcement learning were used to provide proactive policies for blockage avoidance based on statistical or instantaneous information, respectively. Of note, some of the contributions in Table 6.2 refer to wireless systems that employ the same technological enablers as THz communications without specifying the operating frequency band. Indicative examples are [153, 154, 156, 158, 162, 166, 169] that discuss ML approaches for beamforming design in analog and hybrid beamforming systems, as well as [170–173, 175], which present ML-based channel allocation approaches for ultra-dense networks in which the communication channels are high directional. Apparently, the aforementioned approaches are suitable for THz wireless deployments.

Table 6.2: ML algorithm types applied in MAC

	NN	DNN	k-Means	Q-learning	DRL	Q-CNN	Autoencoder	SGD
Beamforming design								
[153]	–	✓	–	–	–	–	–	–
[154]	–	✓	–	–	–	–	–	–
[155]	–	–	✓	–	–	–	–	–
[156]	–	–	–	✓	–	–	–	–
[157]	–	✓	–	–	–	–	–	–
[158]	✓	–	–	–	–	–	–	–
[159]	–	–	–	–	✓	–	–	–
[160]	✓	–	–	–	–	–	–	–
[161]	✓	–	–	–	–	–	–	–
[162]	–		–	–	–	✓	–	–
[163]	✓	–	–	–	–	–	–	–
[164]	✓	–	–	–	–	–	–	–
[165]	–	–	–	–	–	–	✓	–
[166]	–	✓	–	–	–	–	–	–
[167]	–	–	✓	–	–	–	–	–
[168]	–	–	–	–	✓	–	–	–
[169]	–	–	–	✓	–	–	–	–
Channel allocation								
[170]	✓	–	–	–	–	–	–	–
[172]	–	–	✓	–	–	–	–	–
[171]	–	–	–	✓	–	–	–	–
[173]	–	✓	–	–	–	–	–	–
[174]	–	✓	–	–	–	–	–	–
[175]	✓	–	–	–	–	–	–	–

Power management								
[119]	–	–	✓	–	–	–	–	–
[177]	–	–	–	–	✓	–	–	–
[178]	–	–	–	✓	–	–	–	–
[179]	–	–	–	✓	–	–	–	–
[119]	–	✓	–	–	–	–	–	–
Blockage avoidance								
[119]	–	✓	–	–	–	–	–	–
[181]	–	–	–	✓	–	–	–	–
[182]	✓	–	–	–	–	–	–	–
[183]	–	–	–	–	–	–	–	✓
[184]	–	✓	–	–	–	–	–	–

6.2.3 Network layer

The ultra-wideband extremely directional nature of the sub-THz and THz links in combination with the non-uniform UE spatial distribution may lead to inefficient user association, when the classical minimum-distance criterion is employed. Networks operating in such frequencies can be considered noise- and blockage-limited, due to the fact that high path and penetration losses attenuate the interference [98, 185]. Hence, user association metrics designed for interference limited homogenous systems are not well suited to sub-THz and THz networks [186]. As a result, *user association* should be designed to meet the dominant requirements of throughput and guarantee low blockage probability. Another challenge that user association schemes need to face is the user orientation, which is observed to have a detrimental effect on the performance of THz wireless systems [187, 188].

Scanning the technical literature, we can identify several contributions that employ ML for user association in sub-THz and THz wireless networks [189–197]. In more detail, in [189], the authors employed multi-label classification ML that takes as input both topological as well as network characteristics and returns a user association policy that satisfy users' latency demands. Meanwhile, in [190], the authors presented an online deep reinforcement learning (DRL) based algorithm for heterogeneous networks, where multiple parallel DNNs generate user association solutions and shared memory is used to tore the best association scheme. Moreover, in [191], Khan et al. introduces a federate learning approach to jointly minimize the latency and the effect of model accuracy losses due to channel uncertainties.The inputs of the ML algorithm presented in [191] are the device association and the resource block matrices, while the output is the resource alocation matrix. Likewise, in [192], a deep gradient reinforcement learning based policy was presented as a solution to the joint user association

and resource allocation problem in mobile edge computing. The reinforcement learning agent of [192] takes as input the system state that is described by the current backhaul and resource block usage.

In [193], two clustering approaches, namely least standard deviation user clustering and redistribution of BSs load-based clustering were presented that take into account the characteristics of both radio frequency (RF) and THz as well as the traffic load across the network in order to provide appropriate associations in RF and THz heterogeneous networks. Furthermore, in [194], a transfer learning methodology was employed for inter-operator spectrum sharing in mmW cellular networks. The aforementioned methodology takes as input the network topology, the association matrix, the coordination matrix, the effective channels and outputs approximate the achievable data-rate. In [195], an asynchronous distributed DNN based scheme, which takes as inputs the channel coefficient matrix, was reported as a solution to the joint user association and power minimization problem. In [196], Elsayed et al. reported a transfer Q-learning based strategy for joint user-cell association and selection of number of beams for the purpose of maximizing the aggregate network capacity in NOMA-mmW networks. Finally, in [197], the authors exploited distributed deep reinforcement learning (DDRL) and the asynchronous actor critic algorithm (A3C) to design a low complexity algorithm that returns a suboptimal solution for the vehicle-cell association problem in mmW. The DDRL takes as input the current state of the network that is described by a set of a predetermined number of channel observation, the current achievable and required data rates.

After associating UEs to APs, uninterrupted connectivity needs to be guaranteed. However, the network is continuously undergoing change; thus, its management should be adaptive as well. This is where the conventional heuristic based exploration of state space needs to be extended to support UE mobility in an online manner. Aspired by this, several contributions presented ML-based *mobility management* solutions that aim at accurately tracking the UE and proactively steering the AP and UE beams [198, 199] as well as performing hand-overs between beams and APs/BSs [200,201]. In particular, in [198], a RNN with a modified cost function that takes as input the observed received signal as well as the previous AoA estimation, was employed to track the AoA in a mmW network. The proposed approach was shown to outperform the corresponding Kalman-based one in terms of accuracy. Moreover, in [199], a long short term memory (LSTM) structure was designed to prevent the user position in order to proactively perform beam steering in mmW vehicular networks. The structure uses as input the estimated by the BS channel vector. Additionally, in [201], the authors employed kNN to predict handover decisions without involving time-consuming target selection and beam training processes in mmW vehicle-to-infrastructure (V2I) wireless topologies. Finally, in [202], centralized Q-learning was employed that takes into account the current received signal strength in order to provide real-time controlling capabilities to the hand-over process between neighbor BS in directional wireless systems.

Another challenging task of THz wireless networks is *routing*. The limited transmission range in combination with the transmission power constraints and memory limitations of mobile devices render conventional routing strategies that are employed in lower-frequency networks unsuitable for THz ones. Motivated by this, in [203], the authors presented a reinforcement learning routing algorithm and compared it with NN and decision tree-based solutions. The results showed that the reinforcing learning approach not only provides on-line routing optimization suggestions but also outperforms

the NN and decision tree ones. To the best of the authors knowledge, the aforementioned contribution is the only published one that discuss ML-based routing policies in high-frequency wireless networks.

To create a fully automated THz wireless network or even integrate THz technologies into current cellular networks, one of the essential problems that a network manager need to solve is *traffic clustering*. An accurate traffic clustering allows the detection of suspicious data and can aid in the identification of security gaps. However, as the diversity of the data increases, due to their generation for different type of sources, e.g., sensors, artificial/virtual reality devices, robotics, etc, traffic clustering may become a difficult task. Additionally, labeled samples are usually scarce and difficult to obtain.

To address the aforementioned challenges, several researchers turned their eye to ML. In particular, in [204], Noorbehbahani et al. presented a semi-supervised method for traffic classification, which is based on x-means clustering algorithm and a label propagation technique. This approach takes as input network traffic flow data. It was tested in real-data and achieved 95% accuracy using a limited labeled data. Similarly, in [205], the authors reported a semi-supervised classification method that exploits both labeled and unlabeled samples. This method combines offline particle swarm optimization (PSO) to cluster the labeled and unlabeled samples of the dataset with a mapping approach that enables matching clusters to applications.

In [206], Wang et al. applied K-means algorithm that takes into account the correlations of network domain background information and transforms them into pair-wise must-link constraints that are incorporated in the process of clustering. Experimental results highlighted that incorporating constraints in the clustering process can significantly improve the overall accuracy. Furthermore, in [207], Liu et al. employed feature selection to identify optimal feature sets and log transformation to improve the accuracy of K-means based network traffic classification. Another use of K-means was presented in [208], where the authors used it to detect networks cyber-attacks. In particular, the K-means in [208] takes as input network traffic-related data and identifies irregularities. Moreover, a network traffic feature selection scheme that provides accurate suspicious flow detection was reported in [209], whereas, in [210], random forest was employed to perform the same task. Also, Bayesian ML was used to [211], which take as input transport control protocol traffic flaws in order to identify internet traffic. Although, this approach does not require access to packet content, it demands a significant set of training data. To deal with the lack of training data, in [212], the authors presented an unsupervised random forest clustering methodology for automatic network traffic categorization. Meanwhile, in [213], the performance of EM and K-means based algorithms for network traffic clustering were compared and it was shown that K-means outperforms EM in terms of accuracy. In addition, in [214], the authors evaluated and compared the effectiveness of a number of supervised, unsupervised and feature selection algorithms in real-time traffic classification problem, which takes as input network's statistical characteristics. Naive Bayes, Bayesian networks, multilayer perception, decision trees, K-means, and best-first search were among the algorithms that their performance were quantified. The results revealed that in terms of accuracy decision trees outperforms all the aforementioned algorithms. Finally, in [215], the authors combined LSTM with CNN in order to develop a two-layer convolution LSTM mechanism capable of accurately clustering traffic generated by different application types.

Table 6.3 connects the published contributions to the ML algorithms that was applied

in the network layer. Note that some of the contributions in these table does not explicitly refer to THz networks, however, the presented algorithms can find applications to THz wireless systems, due to system and network topologies commonalities. For example in [190], the ML approach can be used in any heterogeneous network that consists of macro-, pico- and femto-cells. Notice that femto cells can be established in the THz band. Similarly, the contributions in [191, 192, 195, 204, 205, 207] and [208] can be applied in any femto-cell network that support high data traffic demands, such as the THz wireless networks, independently from the operation frequency. Moreover, [194, 196–199, 201, 209–212] and [214] refer to mmW networks that support data-rates in the order of $100\ \mathrm{Gb/s}$ in $60\ \mathrm{GHz}$. In such networks, both the AP and UEs employ high-gain antennas. Notice that THz wireless network are also designed to support data-rates in the order of $100\ \mathrm{Gb/s}$ and establish high directional links in order to ensure an acceptable transmission range. Thus, the aforementioned contributions can be adopted to THz wireless networks.

From Table 6.3, we observe that based on the network nature, i.e. fixed or mobile topology, supervised and reinforcement learning approaches are respectively employed for user association. Additionally, when the network had prior knowledge of the UE possible direction supervised learning approaches were employed for mobility management. On the other hand, in problems in which the UE motion is stochastic, reinforcement learning mechanisms were adopted. For routing, where accuracy plays an important role and no instantaneous adaptation is required, supervised learning was used. Finally, for traffic clustering both supervised and unsupervised learning were employed. In more detail, unsupervised learning seems an attractive approach when searching for data irregularities.

Table 6.3: ML algorithm types applied in network layer

	NN	DNN	kNN	Decision trees	Random Forest	Naive Bayes	Bayesian Network	k-Means	x-Means	Feature selection	EM	Multi-layer Perception	DRL	Q-learning	A3C
User association															
[190]	–	–	–	–	–	–	–	–	–	–	–	–	✓	–	–
[191]	✓	–	–	–	–	–	–	–	–	–	–	–	–	–	–
[192]	–	–	–	–	–	–	–	–	–	–	–	–	✓	–	–
[193]	–	–	–	–	–	–	–	✓	–	–	–	–	–	–	–
[194]	–	–	–	–	–	–	–	–	–	–	–	–	–	✓	–
[195]	–	✓	–	–	–	–	–	–	–	–	–	–	–	–	–
[196]	–	–	–	–	–	–	–	–	–	–	–	–	–	✓	–
[197]	–	–	–	–	–	–	–	–	–	–	–	–	✓	–	✓
Mobility management															
[198]	✓	–	–	–	–	–	–	–	–	–	–	–	–	–	–
[199]	✓	–	–	–	–	–	–	–	–	–	–	–	–	–	–
[201]	–	–	✓	–	–	–	–	–	–	–	–	–	–	–	–
[202]	–	–	–	–	–	–	–	–	–	–	–	–	–	✓	–

Routing															
[203]	✓	–	–	✓	–	–	–	–	–	–	–	–	–	–	–
Traffic clustering															
[204]	–	–	–	–	–	–	–	–	✓	–	–	–	–	–	–
[205]	✓	–	–	–	–	–	–	–	–	–	–	–	–	–	–
[206]	–	–	–	–	–	–	–	✓	–	–	–	–	–	–	–
[207]	–	–	–	–	–	–	–	✓	–	–	–	–	–	–	–
[208]	–	–	–	–	–	–	–	✓	–	–	–	–	–	–	–
[209]	–	–	–	–	–	–	–	–	–	✓	–	–	–	–	–
[210]	–	–	–	–	✓	–	–	–	–	–	–	–	–	–	–
[211]	–	–	–	–	–	–	✓	–	–	–	–	–	–	–	–
[212]	–	–	–	–	✓	–	–	–	–	–	–	–	–	–	–
[213]	–	–	–	–	–	–	–	✓	–	–	✓	–	–	–	–
[214]	–	–	–	✓	–	✓	✓	✓	–	–	–	✓	–	–	–
[215]	✓	–	–	–	–	–	–	–	–	–	–	–	–	–	–

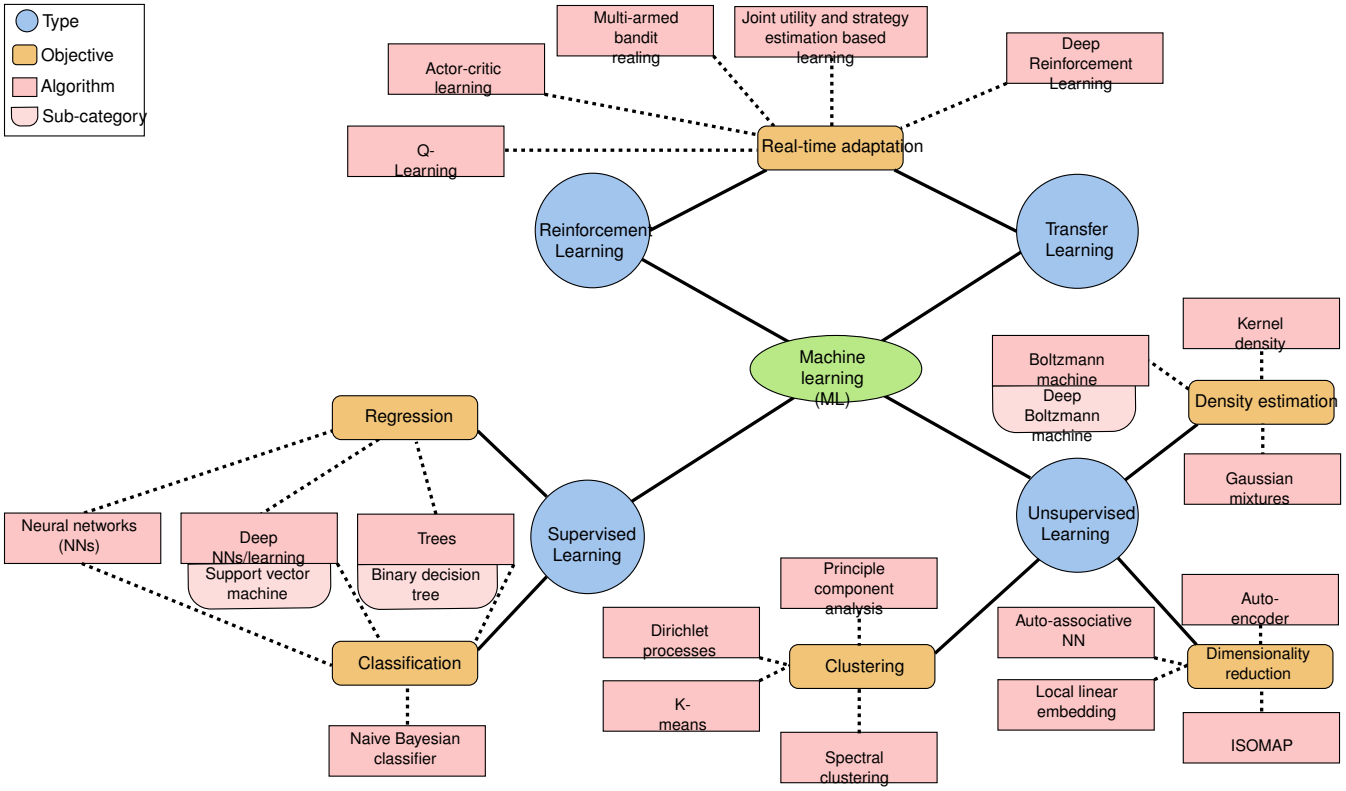


Figure 6.2: Types of ML algorithms.

6.3 A Methodology to Select a Suitable ML Algorithm

As illustrated in Fig. 6.2, ML can be classified into four categories, namely i) supervised, ii) unsupervised, iii) reinforcement, and iv) transfer learning. In what follows, we report the main features of each category and revisit indicative ML algorithms emphasizing their operation, training process, advantages and disadvantages.

6.3.1 Supervised learning

Supervised learning focuses on extracting a mapping function between the input and output values based on a labeled dataset. As a consequence, it can be applied as a solution to regression and classification problems. In more detail, in this paper, the following supervised learning algorithms are discussed.

- *NNs*: are computing machines inspired by biological NN. In more detail and as illustrated in Fig. 6.3, they consist of three types of layers, namely input, hidden, and output. Each layer has a certain number of nodes that are called neurons and process their input signal. A neuron in layer k implements a linear or non-linear manipulation, called *activation function*, of the input data and forwards its output to a number of edges, which connects neurons that belong to layer k with the ones of layer $k+1$. In more detail, let

$$\mathbf{x}_k = \left[x_1^k, x_2^k, \cdots, x_l^k\right]^T \tag{6.1}$$

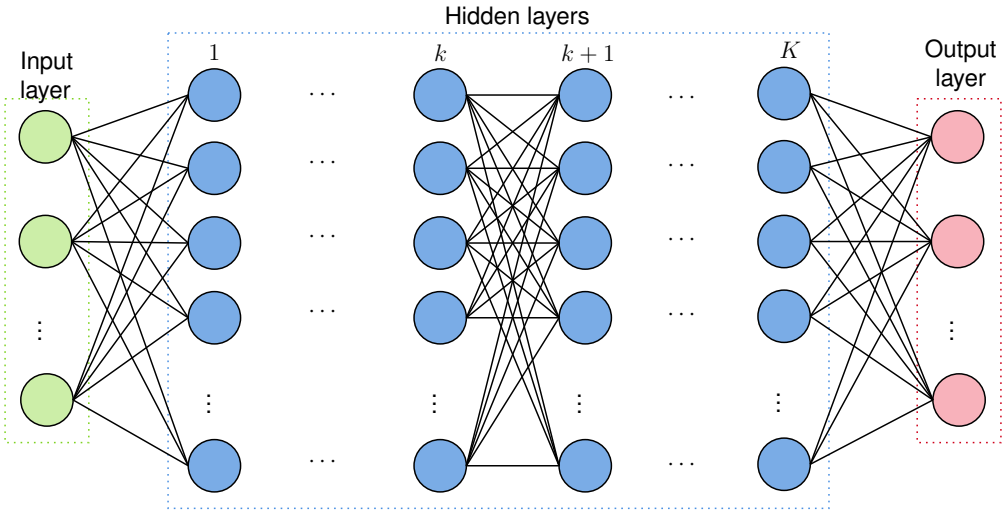


Figure 6.3: General NN structure.

being the input vector of the $k-$th layer of the NN, and

$$\mathbf{w}_{k,n} = \left[w_1^{(k,n)}, w_2^{(k,n)}, \cdots, w_l^{(k,n)}\right]^T \tag{6.2}$$

being the weight vector of the $n-$th node of the $k-$th layer, then the output of this node would be

$$y_{k,n} = \sum_{i=1}^{l} w_i^{(k,n)} f(x_i^k) + b, \tag{6.3}$$

or equivalently

$$y_{k,n} = \left(\mathbf{w}_{k,n}\right)^T f(\mathbf{x}_k) + b, \tag{6.4}$$

where $f(\cdot)$ stands for a mapping function and b is a constant. Of note, if $f(\mathbf{x}_k) = \mathbf{x}_k$, then the mapping is linear; otherwise, the mapping is characterized as non-linear and usually returns a high-dimensional representation of $\mathbf{x}_k$.

The learning process aims at finding the optimal parameters $\mathbf{w}_{k,n}$ so that

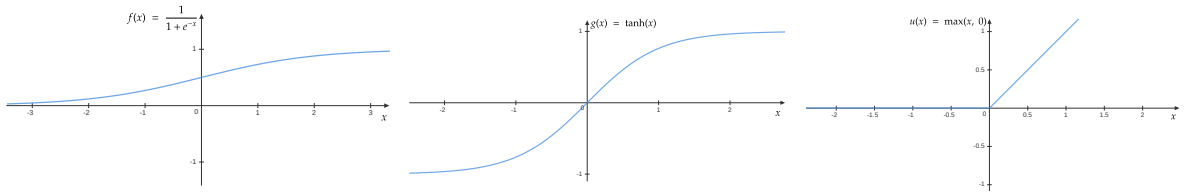
$$\hat{y}_{k,n} = f\left(\mathbf{x}_k; \mathbf{w}_{k,n}\right) \tag{6.5}$$

to be as close as possible to the target $y_{k,n}$. To achieve this a cost/error function $\mathcal{J}\left(\mathbf{w}_{k,n}\right)$ is defined and minimized, i.e.,

$$\frac{\partial \mathcal{J}\left(\mathbf{w}_{k,n}\right)}{\partial \mathbf{w}_{k,n}} = \mathbf{0}. \tag{6.6}$$

The analytical differentiation of (6.6) is usually impossible; thus, numerical optimization methods are applied. The most commonly used methods are gradient descent as well as single and batch perceptron training [216].

- *DNNs*: are NNs with multiple hidden layers that, as depicted in Fig. 6.4, commonly employs $\tanh$, sigmoid or rectified as an activation function. A DNN is segmented



(a) Sigmoid function (b) $\tanh(x)$ function (c) rectified function

Figure 6.4: Three indicative examples of commonly used activation functions.

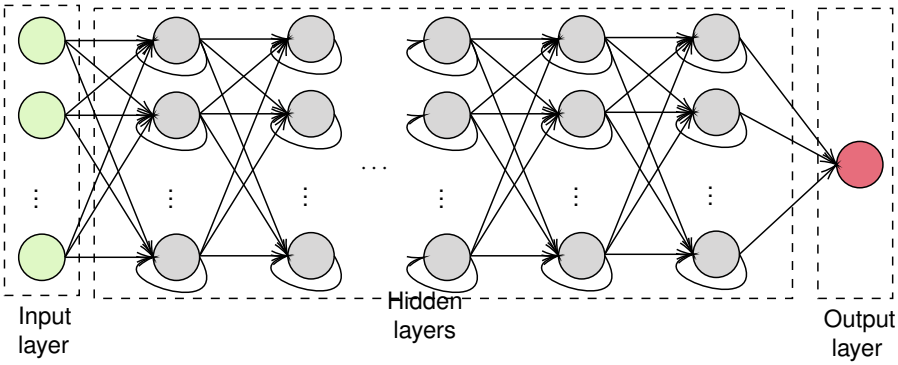


Figure 6.5: RNN structure.

into two phases, i.e. training and execution. Training phase employs labeled data in order to extract the weights of all the activation functions of the DNN. Usually, the SGD with back-propagation algorithm is employed for this task. In general, as the number of hidden layers increases, the number of training data that is requires increases; however, the classification or regression accuracy also increases. In the execution phase, the DNN returns proper decisions based on its inputs, even when the input values have not been within the training data set. As a result, the main challenge of DNN is to optimally select its weights [217].

The special types of DNNs have been extensively used in THz wireless systems and networks, namely CNN, and RNN.

RNN can be used for regression and classification. In contrast to conventional DNNs and as illustrated in Fig. 6.5, it allows back-propagation by connecting neurons of layer k, with the ones of previous layers. In other words, it creates a memory that enables future inputs to be inherited by previous layers [218]. As a result, fewer tensor operations in comparison to the corresponding DNN need to be implemented, which is translated into lower computational complexity and training latency. Building upon this advantage, RNNs have been widely used for a large variety of applications ranging from automation modulation recognition, where channel correlation was discovered by exploiting the recurrent property, to traffic prediction, in which the data spatial-temporal correlation may play an important role.

Finally, *CNNs* have been employed as solutions to several THz wireless networks problems from AMR to traffic prediction. Their objective is to identify local correlations within the data and exploit them in order to reduce the number of parameters as we move from the input to the output through the hidden layers. In this type of networks, a hidden layer may play the role of a convolution, a recti-

fier linear unit (RELU), a pooling, or a flattening layer [219]. Convolution layers are used to extract the distinguished feature of each sample, while RELUs impose decision boundaries. Likewise, pooling layers are responsible for spatial dimensions down-sampling. Last, flattening is used to reorganize the values of high-dimensional matrices into vectors.

- *SVM*: can be employed as a solution for both high-dimensional regression and classification problems by mapping the original feature space into a higher-dimensional one, in which their discriminability is increased [220]. In other words, SVM aims at creating a space in which the minimum distance between nearest points are maximized. In this direction, let us describe the new space as a linear transformation of the original one, which can be described according to the following kernel function:

$$\mathbf{b}^t\hat{\mathbf{x}} + c = 0, \tag{6.7}$$

where $\mathbf{b}$ and c are SVM optimization parameters, while $\hat{\mathbf{x}}$ are the labeled sample that belongs to the set of $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M]$, with the lowest separation distance. Note that $\mathbf{x}_m$, with $m = 1, 2, \cdots, M$, contains the N features of the $n-$th labeled sample. Then, their separation of the training samples can be expressed as

$$s_m = l_m \left(\mathbf{b}^t\mathbf{x}_m + c\right), \tag{6.8}$$

where l_m is the label of the $m-$th class. As a consequence, the optimization problem that describes SVM can be formulated as

$$\begin{aligned} & \max_{\mathbf{b},c} \min_{m=1,2,\cdots,M} \frac{l_m s_m}{\left|\left|\mathbf{b}\right|\right|} \\ \text{s. t.} \quad & C_1 : s_m \geq \min_{m=1,2,\cdots,M} s_m, \quad m = 1, 2, \cdots, M \\ & C_2 : \left|\left|\mathbf{b}\right|\right| = 1 \end{aligned} \tag{6.9}$$

This problem may return a non-linear classification or regression of the original space. Another weakness is that training an SVM model is computationally expensive especially as the training data size increases. In practice, it can take long time to train an SVM model as the number of dimensions (features) increase in a dataset and the problem is exacerbated with increase in datapoints beyond a few hundreds of thousands. SVM has been extensively used for traffic clustering. The main challenge is the optimal selection of the kernel function. This function may be a linear, a polynomial, a radial, or an NN one. To device a suitable kernel function, we usually apply inner product operations between input samples over the Hilbert space in order to extract feature mappings.

- *KNN*: A widely-used algorithm for classification is KNN. KNN consists of three steps, namely i) distance calculation, ii) neighbor identification, and iii) label voting. To provide a comprehensive understanding of KNN, let us define a set of N training samples as $\mathcal{T} = \left\{(\mathbf{x}_1, l_1), (\mathbf{x}_2, l_2), \cdots, (\mathbf{x}_N, l_N), \right\}$, with $\mathbf{x}_i = \left[x_{i,1}, x_{i,2}, \cdots, x_{i,M}\right]$, $i = 1, 2, \cdots, N$, being the samples of a class with label l_i, while $x_{i,m}$, $m = 1, 2, \cdots, M$ are M discrete features. Likewise, let us also define an unlabeled sample as $\tilde{\mathbf{x}} = [\tilde{x}_1, \tilde{x}_2, \cdots, \tilde{x}_M]$, where $\tilde{x}_m$ with $m = 1, 2, \cdots, M$

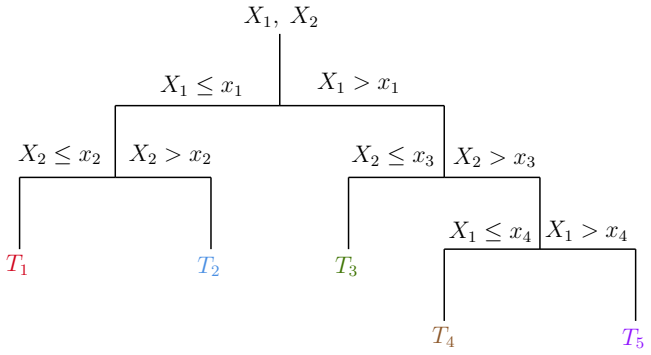


Figure 6.6: An indicative example of a decision tree.

standing for the $m-$th feature of the unlabeled sample $\tilde{\mathbf{x}}$. During the first step, the Euclidean of the Manhattan distance between $\tilde{\mathbf{x}}$ and $\mathbf{x}_i$ is evaluated for all $i = 1, 2, \cdots, N$, according to

$$d_i = \begin{cases} \sqrt{\sum_{m=1}^{M} \left(\tilde{x}_m - x_{i,m}\right)^2}, & \text{Euclidean distance} \\ \sum_{m=1}^{M} \left|\tilde{x}_m - x_{i,m}\right|, & \text{Manhattan distance} \end{cases} . \tag{6.10}$$

In the second step, the K most similar labeled samples, i.e., with the lowest distance from $\tilde{\mathbf{x}}$, are identified. These samples are called "*K nearest neighbors*". In the final step, a majority rule is applied which classify $\tilde{\mathbf{x}}$ to the class in which the majority of the K nearest neighbors belongs to.

In THz wireless systems, KNN has been employed for channel estimation and beam tracking as well as mobility management purposes. Its main challenge is to appropriately select K. On the one hand, a large K can aim at counterbalancing the negative impact of noise. On the other hand, it may fuzzify the boundary of each class. This calls for heuristic approaches that returns approximations for K.

- *Decision trees*: are considered one the most attractive ML approach for both regression and clustering, due to their simplicity and intelligibility. They are defined by recursively segmenting the input space in order to create a local model for each one of the resulting regions. To provide a comprehensive understanding of decision trees operation, we consider an indicative tree that is depicted in Fig. 6.6. We represent the target values by the tree's leaves, while branches stand for observations. In more detail, the first node checks whether the observation X_1 is lower or higher than the threshold x_1. If $X_1 \leq x_1$, then, we check whether the observation X_2 is lower or higher than another threshold x_2. If both $X_1 \leq x_1$ and $X_2 \leq x_2$, the decision tree returns the target value T_1. On the other hand, if $X_1 \leq x_1$ and $X_2 < x_2$, the target value T_2 is returned. Similarly, if $X_1 > x_1$, the decision tree checks whether X_2 is lower than the threshold x_3. If a positive answer is returned, the target value is set to T_3, otherwise, it checks whether X_1 is lower or higher of x_4. If it is lower, the target value T_4 is returned; otherwise, T_5 is returned.

In general, the decision tree model can be analytically expressed as

$$g(x) = \sum_{m=1}^{M} r_m \delta\left(\mathbf{X} \in \mathbf{R}_m\right), \tag{6.11}$$

where $\delta\left(\cdot\right)$ is an indicator function that is defined as

$$\delta\left(\mathbf{X} \in \mathbf{Y}\right) = \begin{cases} 1, & \mathbf{X} \in \mathbf{Y} \\ 0, & \text{otherwise} \end{cases}. \tag{6.12}$$

Moreover, $\mathbf{R}_m$ stands for the $m-$th decision region, and r_m represents the mean response of this region. Finally, M represents the total number of regions. From (6.11), it becomes evident that training a decision tree network can be translated into finding the optimal partitioning, i.e., the optimal regions $\mathbf{R}_m$ with $m \in [1, M]$. This is usually an NP hard optimization problem and its solution require the implementation of greedy algorithms.

Although decision trees are easy to implement, they come with some fundamental limitations. In particular, they are have lower accuracy in comparison with NNs and DNNs. This is due to the greedy nature of the training process. Another disadvantage of decision trees is that their sensitive to changes to the input data. In other words, even small changes to the inputs may greatly affect the structure of the tree. In more detail, due to the hierarchical nature of the training process, errors that are caused at the top layers of the decision tree affect the rest of its structure.

- *Random forests*[3]: improve the accuracy of decision trees by averaging several estimations. In more detail and as illustrated in Fig. 6.7, instead of training a single tree, random forest methodology is based on training N different trees using different sets of data, which are randomly chosen. The outputs of the N trees are averaged; hence, the random forest model can be described as

$$g(x) = \frac{1}{N} \sum_{n=1}^{N} g_n(x), \tag{6.13}$$

with $g_n\left(\cdot\right)$ standing for the $n-$th tree model.

The main challenge of random forests is to guarantee that the trees operate as uncorrelated predictors. To achieve this, the training data is randomly divided into subsets, where each subset is used to train a tree". It is very confusing to say "input variable subsets". Basically, one of the ideas behind the random splits of training data into subsets is that it makes the model resilient to outliers and overfitting. For example, if there is an outlier in one or more subsets, the model's accuracy is not skewed due to it, as this is an ensemble model and the outcome reflects joint decision of all trees, and since different trees have "seen" different distributions in data (due to random subsetting), it is expected to handle overfitting better. This indicates that there exists a trade-off between the accuracy

[3]It is worth-noting that random forest is implementing a technique which is called bagging.

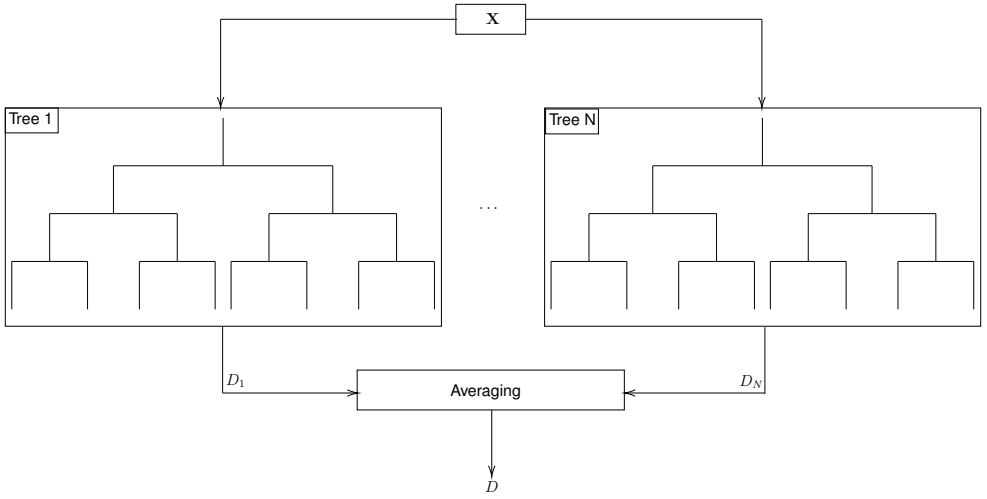


Figure 6.7: An indicative example of a random forest.

and training latency/overhead. In more detail, as the number of trees increases and thus the accuracy of the random forest improves, the training set need to be lengthen. Therefore, both the training latency and the overhead in the network increases. Another disadvantage of random forests is the interpretability of the model is not as simple in comparison with singular or non-ensemble model such as a decision tree.

- *Naive Bayesian classifier*: aims at choosing the class that maximizes the posteriori probability of occurrence. In particular, let us define the vector $\mathbf{x} \in \{1, \cdots, R\}^S$, where R stands for the number of values for each feature, while D represents the number of features. Naive Bayesian classifier assigns a class conditional probability, $p\left(C_t \,|\mathbf{x}\right)$ for each possible class C_t with $t \in [1, T]$. Of note, T stands for the number of different classes. By applying the Bayes' theorem, we can express $p\left(C_t \,|\mathbf{x}\right)$ as

$$p\left(C_t \,|\mathbf{x}\right) = \frac{p\left(C_t\right) p\left(\mathbf{x} \,|C_t\right)}{p\left(\mathbf{x}\right)}. \tag{6.14}$$

Moreover, by "naively" assuming that, for a given class label C_t, the features are conditionally independent, $p\left(\mathbf{x} \,|C_t\right)$ can be obtained as

$$p\left(\mathbf{x} \,|C_t\right) = \prod_{s=1}^{S} p\left(x_s \,|C_t\right). \tag{6.15}$$

Here, conditional independence means that the algorithm treats all features as equally important and statistically independent of each other. This may apparently seem counter-intuitive as several features may indeed have some form of correlation. However, this "naive" assumption can often lead to good predictive accuracy due to the emphasis on evidence observed in the form of conditional probability of features, for a given outcome class of the predicted (target) variable. The independence assumption weakens the explainability of the predictions made by the Naive Bayes classifier but at the same time, the algorithm is very efficient to train because probabilities in (6.15) can be measured in a single data scan of the training dataset.

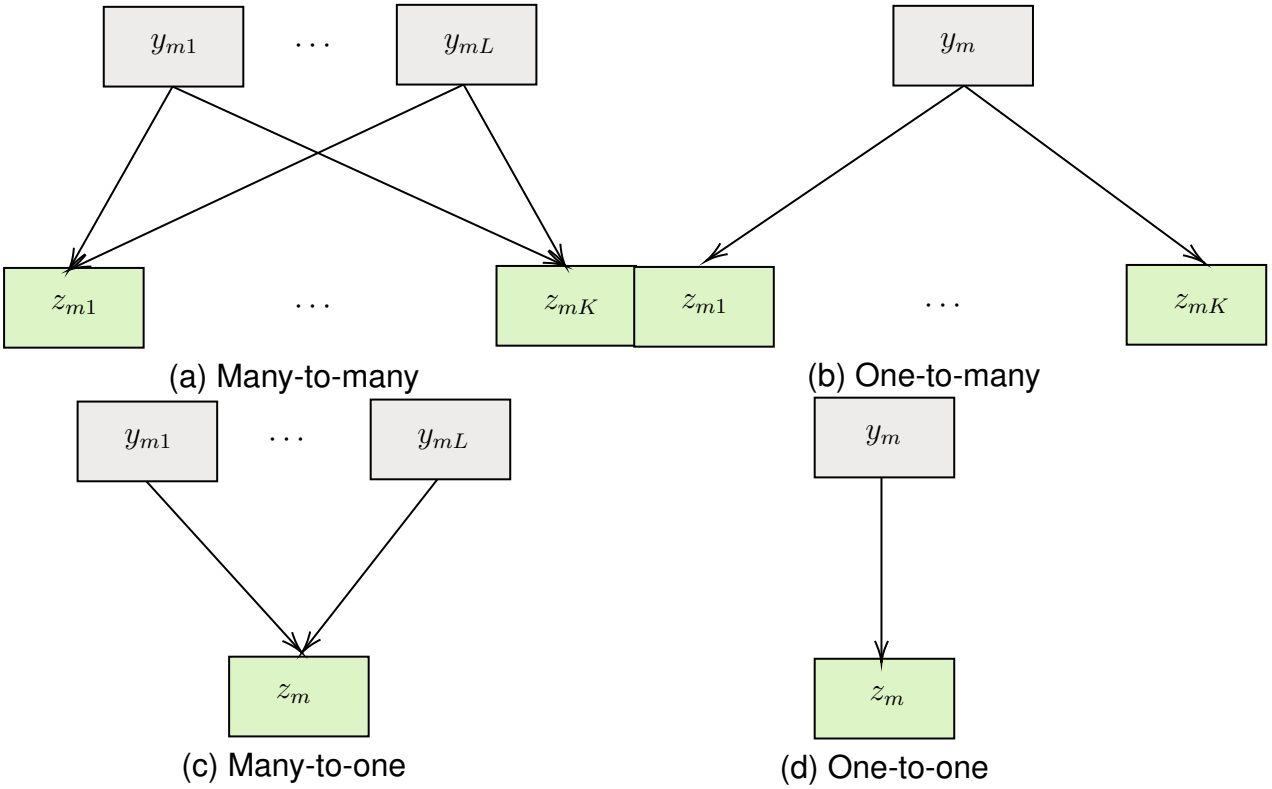


Figure 6.8: Indicative hidden variable models.

Based on (6.14) and (6.15), a class label can be assigned according to

$$\tilde{c} = \arg \max_{t \in [1,T]} p\left(C_t\right) \prod_{s=1}^{S} p\left(x_s \left| C_t \right.\right). \tag{6.16}$$

Note that based on the type of each feature, $\mathcal{U}_{s,t} = \{x_s \left| C_t \right.\}$ may follow a Gaussian, Bernoulli, multinoulli well-defined distribution. Likewise, from (6.16), it becomes evident that the training problem is converted to a maximum likelihood one, which may generate overfitting issues and compromise the accuracy of the naive Bayes model.

6.3.2 Unsupervised learning

Supervised learning highly depends on the existence of labeled datasets for training. However, in several practical scenario, no such datasets are available. In this case, *unsupervised learning* can be applied. Unsupervised learning aims at extracting data unknown features and identify the relationship between them and the system response. In more detail, unsupervised learning algorithms search for four types of relationships, namely: (i) many-to-many; (ii) one-to-many; (iii) many-to-one; and (iv) one-to-one. This is graphically presented in Fig. 6.8, where $y_{m1}, \cdots, y_{mL}$ denotes L hidden variables and $z_{m1}, \cdots, z_{mK}$ are K known ones, with $K >> L$. These types of relationships can be used for clustering, density estimation, and dimensionality reduction. In more detail,

in THz wireless systems and networks the following clustering approaches have been employed.

- *EM*: is a low-complexity iterative algorithm that aims at identifying maximum likelihood estimates of parameters by means of iteration between two phases, namely E and M. During the E phase, it infers the missing values, for a given set of parameters, while, in the M phase, it optimizes the parameters for a fixes "filled in" data set. In more detail, in the $i-$th iteration, at the E phase, EM computes an auxiliary function for the expectation of the log-likelihood using the parameters estimations of the $i-1$ iteration, which can be expressed as

$$\mathcal{F}\left(\theta_{i-1}\left|\theta_i\right.\right) = \mathbb{E}_{\mathbf{Z}|\mathbf{Y},\theta_i}\left[\log\left(p\left(\mathbf{Y},\mathbf{Z}\left|\theta_{i-1}\right.\right)\right)\right], \tag{6.17}$$

  where $\mathbf{Y}$ and $\mathbf{Z}$ are respectively the set of known and hidden variables, whereas, θ_i is the set of the unknown parameter at the $i-$th iteration. In the M phase, the parameters that maximize the expected log-likelihood are determined based on the following formula:

$$\tilde{\theta}_{\mathbf{i}} = \arg\max_{\theta_i} \mathcal{F}\left(\theta_{i-1}\left|\theta_i\right.\right). \tag{6.18}$$

  This process stops when parameters convergence is achieved.

  The EM approach can be used to parameter estimation problems that are based on popular statistics models, like mixture Gaussian, hidden Markov, etc. However, it has an important disadvantage. It cannot guarantee convergence to a global optimum and not to a local one. As a result, it usually achieves poor performance in high-dimensional problems.

- *K-means*: The objective of K-means is to partition M unlabeled samples into K clusters, such as each sample to belong to exactly one cluster, based on their similarity in terms of distance. In order to achieve this a two step approach is followed, according to which, each training sample is assigned to one of the K clusters based on its distance from the cluster center[4], i.e., as a solution to the following optimization problem:

$$C^* = \arg\min_{\mathbf{c}} \sum_{l=1}^{K} \sum_{\mathbf{x}\in\mathbf{c}_k} \left|\left|\mathbf{x}-\mu_k\right|\right|^2, \tag{6.19}$$

  where C^* is the optimal cluster segmentation, $\mathbf{x}$ is the set of samples and μ_k is the mean of the samples that belongs in the cluster $\mathbf{c}_k$. Then, the cluster center is updated, based on the new samples that are included or the ones that was removed from each cluster. This process is reaped until convergence is achieved. A graphical representation of the K-means algorithm is depicted in Fig. 6.9.

  From (6.19), it becomes evident that the clustering optimization problem is a NP-hard one. As a result, a heuristic algorithm needs to be employed in order to solve it. However, such algorithms cannot guarantee convergence in a global

[4]Note that as the cluster of the center, most K-means implementations use the mean of the samples that belong in the same cluster.

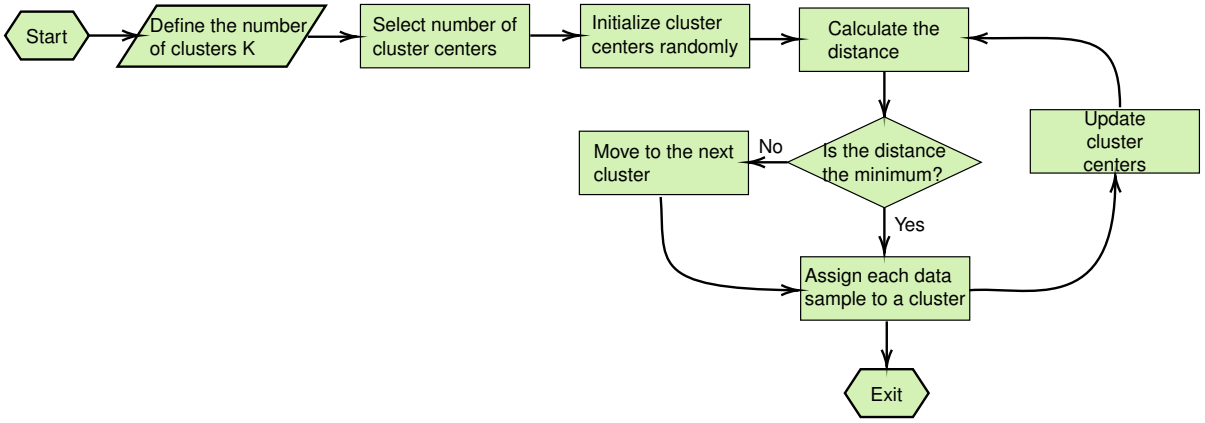


Figure 6.9: K-means algorithm.

optimum. Its result is tightly connected to the initial cluster selections as well as their centers. Despite this disadvantage, it has been used as a solution to a wide range of problems spanning from beamfoming design to caching.

Feature selection or dimensionality reduction can be seen as a preprocessing phase of ML, since it enables the elimination of correlated features by means of feature transformation. In this direction, the following feature selection/dimensionality reduction approaches have been employed in THz wireless systems and networks:

- *Principle component analysis (PCA)*: implements an orthogonal transformation in order to convert potential correlated features of a dataset into uncorrelated ones that are called principle components. The operation pillar of PCA is based on the dogma that the first principle component has larger variance than the second, which in turn has larger than the third, and so on. As the variance decreases, the amount of the encaptulated information of the original features decreases, given that the original feature has a considerable correlation. Motivated by this, PCA aims at solving the following maximization problem:

$$\mathbf{v}^* = \max_{\mathbf{v}} \frac{1}{N} \sum_{n=1}^{N} \left(\mathbf{y}_n^t \mathbf{v}\right)^2, \tag{6.20}$$

or equivalently

$$\mathbf{v}^* = \max_{\mathbf{v}} \mathbf{v}_n^t \mathbf{G} \mathbf{v}, \tag{6.21}$$

where $\mathbf{G}$ stands for the covariance matrix of the training dataset that can be expressed as

$$\mathbf{G} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{y}_n \mathbf{y}_n^t, \tag{6.22}$$

while $\mathbf{y}_n$ represent the $n-$th training dataset, and $\mathbf{v}$ is a unit vector.

Notice that the solution of (6.21), is the eigenvectors $\mathbf{v}_1, \cdots, \mathbf{v}_K$ of $\mathbf{G}$, with $K < M$, where M the number of the original features. As a result, the dimensionality reduction can be mathematically written as

$$\mathbf{z}_n = \left[\mathbf{v}_1, \cdots, \mathbf{v}_K\right]^t \mathbf{y}_n. \tag{6.23}$$

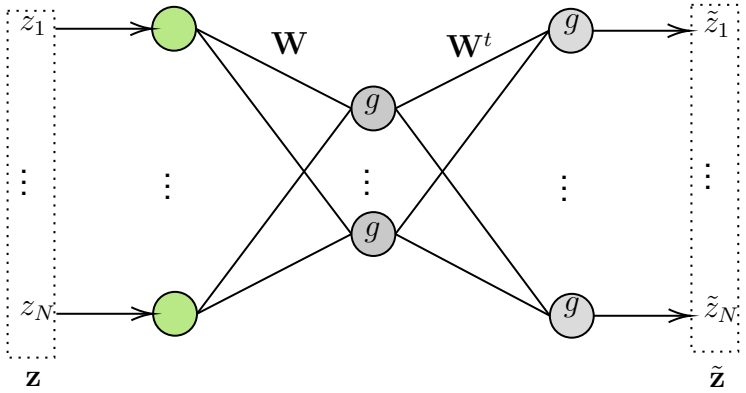


Figure 6.10: The autoencoder's structure.

- *Auto-encoder*: is a feed-forward NN that it is trained to predict its inputs. As a consequence, the number of inputs is the same as the one of the outputs. As depicted in Fig. 6.10, the auto-encoder is a three-layer network that can be described as [221]

$$\tilde{\mathbf{z}} = \mathbf{g}\left(\mathbf{W}^t\left(\mathbf{W}\mathbf{z} + \mathbf{c}\right) + \mathbf{b}\right), \tag{6.24}$$

  where $\mathbf{W}$, $\mathbf{b}$, and $\mathbf{c}$ are auto-encoder's parameters, while $\mathbf{g}$ stands for an activation function vector. Finally, $\mathbf{z}$ and $\tilde{\mathbf{z}}$ are $N-$dimensional vectors that contains the auto-encoder's inputs and outputs.

  The first layer of the auto-encoder is a bottleneck one that is responsible for preventing the system from learning a trivial identity mapping. The connection weights between the first and the second as well as the second and the third layer are shared, i.e., $\mathbf{W}$ and $\mathbf{W}^t$. Note that the objective of the training phase of the auto-encoder is to select a suitable $\mathbf{W}$ in order to minimize the input-output error. This is usually performed by feeding inputs and outputs in the training algorithm. Finally, the hidden nodes are used to capture the most relevant dataset aspects.

  In comparison with PCA, auto-encoder is capable of performing not only linear but also non-linear transformations. However, since a greedy algorithm is usually employed for its training, it is also sensitive to fitting errors. As a result, an important task for using auto-encoders is to appropriately select the activation function to be employed.

- *ISOMAP*: also called manifold learning, is a non-linear dimensionality reduction approach that is build upon the principle of preserving the geodesic distances[5] of the lower-dimension [222]. In more detail, its implementation follows four stages. In the first stage, the neighbors of each points are determined. In particular, for each pair of points i, j, the input space distance $d_X(i,j)$, is calculated. Points that have an input distance lower than a predetermined fixed radius, ϵ, are considered neighboring points. Building upon the first stage, the second one generates neighborhood graph that connects each point with its neighbors. Then, in the third

[5]Note that the geodestic distance is the distance between two points following the available/possible path that connects them.

stage, the shortest path between two nodes of the neighborhood graph is evaluated, according to the Dijkstra's or Floyd-Warshall algorithm [223]. Towards this direction, the geodesic distance, $d_G(i,j$, between all pair of point on the manifold is calculated as

$$d_G\left(i,j\right) = \min\left(d_a\left(i,j\right), d_a\left(i,k\right) + d_a\left(k,i\right)\right) \tag{6.25}$$

for each $k \in [1,N]$ were N stands for the total number of points. Moreover, $d_a\left(m,n\right)$ is an auxiliary variable that can be defined as

$$d_a\left(i,j\right) = \begin{cases} d_X\left(i,j\right), & \text{for } (i,j) \in \mathcal{A} \\ \infty & \text{for } (i,j) \notin \mathcal{A} \end{cases}, \tag{6.26}$$

with $\mathcal{A}$ being the set of neighboring points. Finally, in the forth stage, the lower-dimensional embedding, y_i, is extracted by minimizing the embedding cost function

$$J_c(j) = \sum_{i=1}^{N}\left(\left|\left|y_i - y_j\right|\right| - d_G\left(i,j\right)\right)^2. \tag{6.27}$$

This approach finds several applications in identifying non-linear correlated hidden variables, such as traffic clustering. However, it comes with an important disadvantage. In general, it is topologically unstable [224]; thus, it should only be applied after extensive pre-processing of the data [225].

For density estimation the Boltzmann machine is usually used. *Boltzmann machines*: are employed to discover hidden features, which denote complex regulations in the training dataset. As a result, they can be used to extract the stochastic dynamics of datasets. Regarding its structure, as presented in Fig. 6.11, it can be seen as a network, in which its units are bidirectionally symmetrically connected to each other with fixed weights and return stochastic binary decisions, i.e., 0 or 1. A unit can be a visible or a hidden node of the network. Notice that we can interact only with visible units. A unit that returns a state 0 indicates that the system rejects a hypothesis, while in state 1, it accepts it. The weight on a connection stands for a pairwise constraint between two hypotheses. In particular, positive weights refers to hypotheses that support each other, i.e., if one is accepted, the other should also be accepted, while negative ones indicate that only one of the two hypothesis can be accepted. The objective of a Boltzmann machine is to minimize its global state, which is defined as [226]

$$E = -\sum_{i<j} w_{ij} s_i s_j + \sum_i \theta_i s_i, \tag{6.28}$$

where w_{ij} is the i,j connection weight, s_i and s_j respectively stand for the $i-$th and $j-$th units states, and θ_i represents a threshold. To achieve this, we usually employ heuristic algorithms. As a consequence, Boltzmann machines suffer from performance degradation when the network is scaled up in size.

Finally, a special ML framework that can be used for both supervised and unsupervised learning is *GAN* [227]. GANs are usually used to generate new data that have

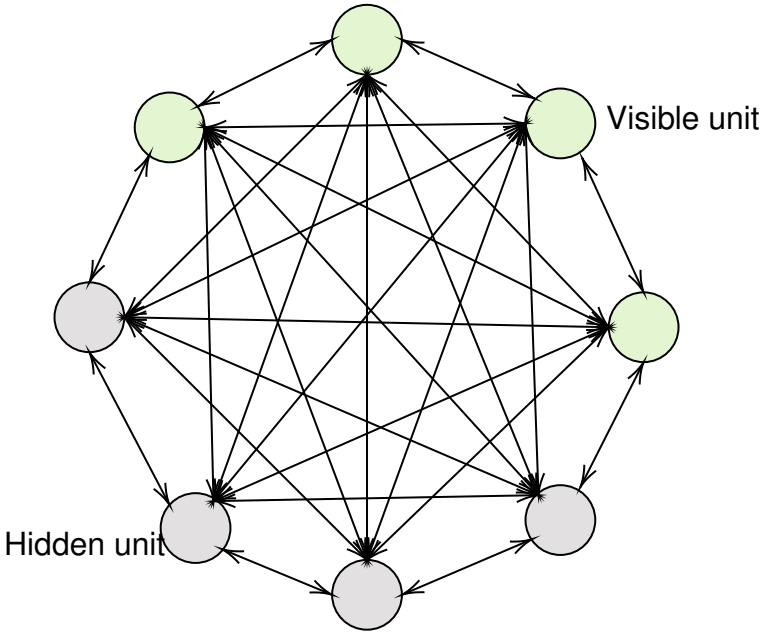


Figure 6.11: The Boltzmann machine's structure.

the same statistics as the training ones [228]. As shown in Fig. 6.12, they consist of two networks, a *generator* and a *discriminator*. The generator produces new samples after providing an estimation of the dataset distribution, while the discriminator compares the generated samples distribution to the one that arises by the unlabeled data. The generator's distribution, $p_Z(z)$, over data z is defined by introducing a prior on input samples distortion, which is distributed according to $p_N(n)$, and a mapping to the data space, which is represented by $H\left(n\,|\theta\right)$, where H describes a multi-layer preceptor (MLP) with parameters θ. The discriminator that follows utilizes another MLP, which we denote $D\left(z\,|\theta_D\right)$ with parameter θ_D, that outputs a scalar that indicates the probability that the training samples and the data generated by H are labeled correctly. In this direction, the generator is trained in order to minimize the term

$$\mathcal{F}_H = \log\left(1 - D\left(H\left(n\,|\theta\right)\right)\right), \tag{6.29}$$

whereas the discriminator aims at maximizing the term

$$\mathcal{F}_G = \log\left(H(z)\right). \tag{6.30}$$

In other words, a two-players min-max game is formulated, which can be solved by employing iterative numerical methods.

In THz wireless systems, GANs have been applied as the solution to AMR problems, due to their capability to predict the different versions of the received signal, when a specific symbol is transmitted. However, it comes with some disadvantages. First of all, the training phase may be unstable, if not considerable amount of time is not spend in this phase. Moreover, to deal with training instability, visual examination may be needed in each step; this creates an important workload to the ML designer. Finally, it has no density estimation capabilities. This indicates that it cannot be used for anomaly detection.

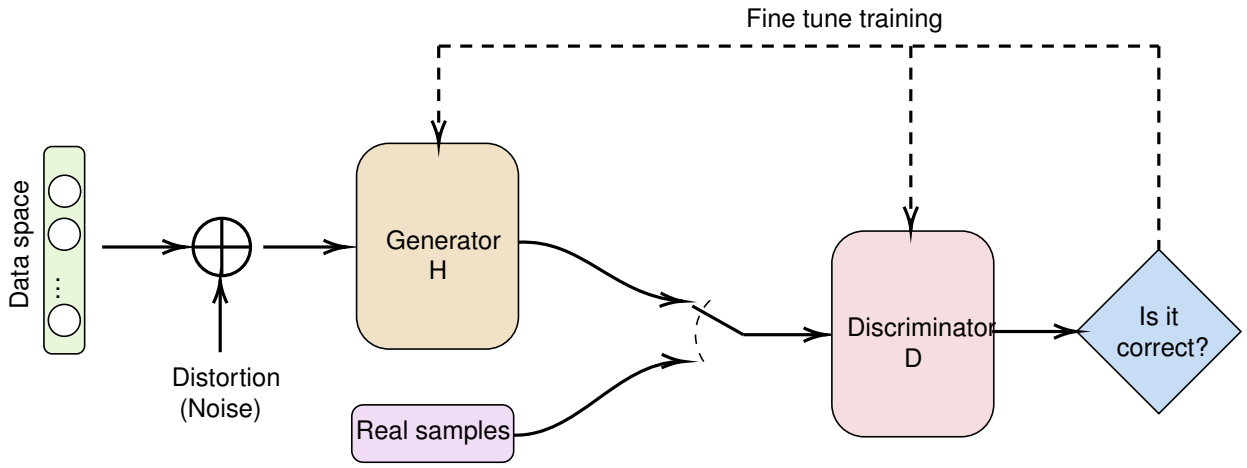


Figure 6.12: The GAN's structure.

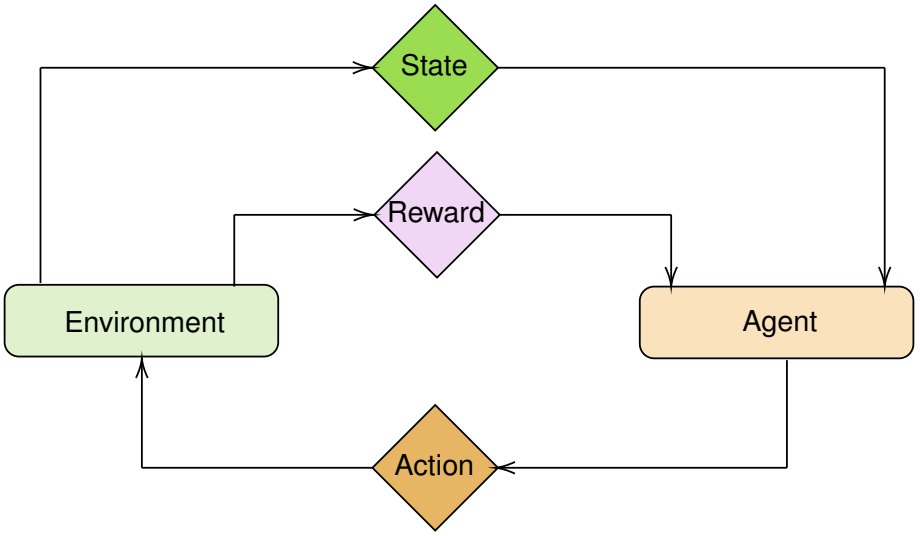


Figure 6.13: Reinforcement learning structure.

6.3.3 Reinforcement and transfer learning

This section is devoted to reporting the reinforcement and transfer learning approaches that have been employed in THz wireless systems and networks with emphasis to their operation principles, applications and challenges. In this direction, in Section 6.3.3, reinforcement learning approaches are documented, while, in Section 6.3.3, the transfer learning ones are reported.

Reinforcement learning

As illustrated in Fig. 6.13, the fundamental idea of reinforcement learning is to resemble the trail and error process by employing an agent that continuously interacts with the environment [229]. In more detail, an agent sense the environment state and applies an action that affect the environment. As a response, the environment returns a quantified reward. Of note, the environment stage is influenced by two factors, namely i) the environment itself; and ii) the agent's action. Similarly, the award is evaluated based on

its impact to the environment and the action of the reinforcement learning method. As a result, reinforcement learning approaches allows real or almost-real time interactions to environmental changes. This characteristic is a key requirement in several system and network operation processes in all the OSI layers. Thus, they have extensively adopted as solutions to a wide range of problems, such as beamforming design, power management, blockage avoidance, user association, mobility management, caching, and computational offloading. Likewise, in contrast to conventional optimization approaches that focuses on immediate reward maximization, reinforcement learning aims at long-term reward. This is achieved by taking into account in the optimization process both the immediate and the future reward; thus, allowing intelligent prediction of the future system's state. The following reinforcement learning algorithms have been applied in THz wireless systems and networks:

- *Q-learning*: or as alternatively called *temporal-difference learning*, is a commonly adopted model-free reinforcement learning approach capable of directly acquiring knowledge from raw experience without requiring either an environmental model or a delayed reward system. Its interaction with the environment is based on a state-action value function that is called Q-function, which is continuously updated in order to achieve maximization by means of selecting an appropriate action, i.e.,

$$A^* = \arg\max_{A \in \mathcal{A}} Q\left(S, A\right), \tag{6.31}$$

  where S stands for the system state, A represents the selected action among the available ones that are included in the set $\mathcal{A}$. Moreover, $Q\left(\cdot, \cdot\right)$ is the Q-function, which is updated, according to [230], as

$$\hat{Q}\left(S, A\right) = \left(1 - c\right) Q\left(S, A\right) + c\left(r + d \max_{A \in \mathcal{A}} Q\left(S^{'}, A\right)\right). \tag{6.32}$$

  In (6.32), c and d respectively denote the updated weight and the discount factor, while r is a constant.

- *Deep reinforcement learning*: or *deep Q-learning* is usually applied in problems in which the dimensions of the state and action spaces are quite large. In these scenarios, the use of a Q-function as a table that contains values for each state and action is deemed impractical. To address this issue, we train a NN with parameters θ, which is responsible for the estimation of the Q values through a Q-function approximation, i.e., $Q_N\left(S, A; \theta\right) \approx Q\left(S, A\right)$. The training phase aims at minimizing in each step i a loss function that can be expressed as

$$L_i\left(\theta_i\right) = \mathbb{E}_{S,A,r,S^{'}|\rho_s}\left[\left(T\left(S, S^{'}, A, A^{'}, r; \theta_i, \theta_{i-1}\right)\right)^2\right], \tag{6.33}$$

  where

$$T\left(S, S^{'}, A, A^{'}, r; \theta_i, \theta_{i-1}\right) = r + d \max_{A^{'}} Q_N\left(S^{'}, A^{'}; \theta_{i-1}\right) - Q_N\left(S, A; \theta_i\right) \tag{6.34}$$

  stands for the temporal difference, while ρ_s represents the system's behavior distribution.

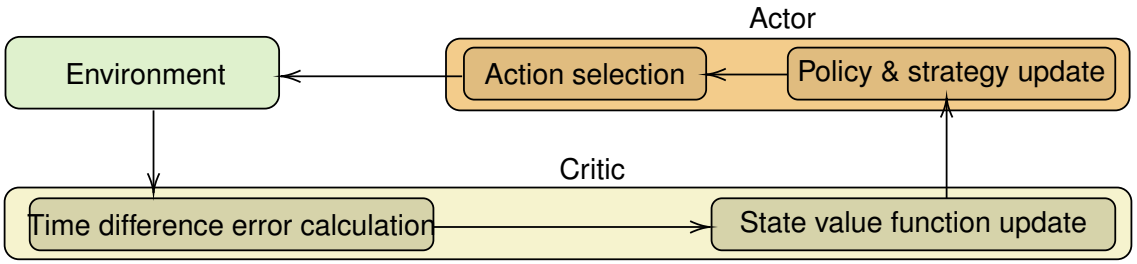


Figure 6.14: A3C structure.

- *A3C*: is another reinforcement learning approach that, as presented in Fig. 6.14, is composed by three units, namely *actor*, *critic*, and *environment*. The actor makes an initial action selection A from the set of available actions $\mathcal{A}$, based on a current policy or strategy. Next, the critic computes the new state value, which was extracted due to the environment variation, and updates the a time difference error (TDE). The new TDE is fed to the actor, which create a revised policy and/or strategy. The policy update is usually based on a Boltzmann distribution. The A3C is going to converge to an optimal state, after revisiting each action for each state by infinite times [231].

Reinforcement learning faces an important challenge. In more detail, the methodology to design the optimal state, action, reward/cost in different that enables convergence into optimal system performance depends on the scenario under investigation. As a consequence, in low-dimension state-action spaces in which all the space-action pairs can be documented into Q-value tables and explored by the reinforcement learning algorithm, a near-optimal solution can be rapidly identified. However, as the state-action space dimension increase, the reinforcement learning algorithm performance degrades, since several state-action pairs may remain unexplored. To counterbalance this, deep reinforcement learning is adopted. However, this approach comes with the training latency of NNs.

Transfer learning

To avoid training latency, the concept of *transfer learning* was born, according to which knowledge from a specific domain can be used to speed up the learning process. In more detail, the aforementioned knowledge can be represented as Q-values of Q-learning and A3C algorithms, or NN weights in deep reinforcement learning approaches. The Q values and weights may have been learned by an agent in a former and similar environment [232].

Transfer layer have enabled several operation in wireless THz systems that vary from beamforming design to computational offloading. However, it comes with an important drawback. If the difference between former and current tasks and environments are important, the knowledge that is transferred will cause a negative impact to the system performance [233].

6.3.4 Conclusions

This section provides a systematic methodology that can be used to select the appropriate family of algorithms in order to solve a ML problem. In this direction, the first step is to categorize the problem under investigation. This can be achieved by examining the type of the input data and the expected outcomes. In more details, if labeled input data are provided, a supervised learning ML strategy can be selected. On the other hand, if the input data are unlabeled, unsupervised ML algorithms should be employed. Finally, if no data exists and the model need to interact with the environment, reinforcement or transfer learning algorithms should be applied, based on the absence or existence of simulation data.

Next, the outputs should be examined in order to identify the category of the problem that we want to solve. In particular, if the output data of the ML algorithm is a number, the problem is a regression one. Regression is usually used for AMR, beam training, signal detection, beam tracking, beamforming design, blockage avoidance, mobility management, and traffic prediction. Additionally, if the output of the model is a class and the number of the expected classes is predefined, then the problem is a classification one. An indicative example of a classification problem is user association. In both cases of regression and classification, NNs, DNN, naive Bayes, decision trees or random forests will be applied. To choose between the aforementioned algorithms, we should first examine the variation of the input data. If they have a small variation and low-latency or interpretability are key requirements, decision trees will most likely be used. This is the reason why decision trees are attractive approaches for routing problems. On the other hand, if they have small variation but the latency and interpretability existence are not the main requirements, a random forest will be applied. On the contrary, if the input data have a relatively large variation, naive Bayes or NNs/DNNs will be employed based on whether they follow or not a well-known distribution.

On the other hand, if the objective of the problem is to categorize data into an initially unknown-number of classes, it's a clustering problem. Indicative examples of such problems are traffic clustering, caching, and computational offloading. These problems can be solved by employing k-Means, k-Median, EM and hierarchical clustering. k-Means and k-Medians are usually selected in high-dimensional problems, while EM and hierarchical clustering in low-dimensional problems.

Another objective of ML problem is to improve the system and/or network performance. Such problems belong to the optimization category and are usually multivariate ones. In THz wireless systems and networks, several researchers have employed gradient descent and reinforcement/transfer learning algorithms, either to predict the optimal operation point or to correct it. Similarly, recommendation problems are the ones that return options based on the history of actions and are usually solved by employing reinforcement/transfer learning algorithms. Reinforcement/transfer learning provides high-adaptability to environmental changes and requires no training. Therefore, they are suitable for problems that require fast adaptation, like channel allocation, power management, blockage avoidance and user association in mobile THz wireless networks, as well as computational offloading.

Finally, if the goal is to obtain insight from data for pattern recognition or anomaly detection, then dimensional reduction or feature selection algorithms can be applied, such as PCA, auto-encoder or ISOMAP. PCA is usually applied if the data are linearly correlated. On the other hand, auto-encoder and ISOMAP achieve acceptable per-

formance in non-linear correlated datasets. These algorithms can find application in beamforming design and traffic clustering.

Chapter 7

Strategies for Deployment of AI/ML Solutions in Beyond 5G Applications

7.1 Introduction

B5G systems are expected to satisfy an ever increasing data connectivity, data-rate and throughput demands. AI/ML methods are positioned to play a central role to enable the functional and non functional demands expected in these systems. AI/ML would be integrated in various layers of the network management stack. In this chapter, we present an overview of deployment strategies, part of which are published in recent journal article [234], that must be considered before AI and/or ML based models are operationalized. Here, operationalization refers to the deployment of a well-trained and tested AI/ML model in production to help automate decision making in real time. This stage is usually preceded by staging, where the models are observed in production settings and monitored but do not drive the decision making process.

Deployment of AI/ML models needs to take into consideration non-functional concerns more than mere application of the model to get predictions. We sketch out two parallel formations have to be put in place for deployment. First, the performance in terms of accuracy and response time is to be continually monitored, and secondly, for recurring and adaptive problems, models may need to be updated or retrained at regular intervals, on new training data that may have become available over time. It should be noted that this data may become available on the periphery or edges of the network, and may need to be transmitted to central node or processed directly at the edge.

On the one hand, the AI/ML model deployment shares some commonalities with how the software components in an information technology (IT) system are updated e.g., following blue green deployment mechanism, but also some major differences exist, in that the deployment of AI/ML models usually follows a “primary-secondary” (sometimes referred to as “primary-challenger”) dual models. The primary model is the one being used for automated decision making, whereas the challenger is used as a stand-by if at any point in time, the functional performance (e.g., accuracy) of the primary drops significantly lower than the secondary model’s performance. This also helps as a counter measure to drift or shift of concept phenomenon that is observed in ML systems when the data gradually exhibits different distributions unlike what the models are originally trained on. Finally, the models may need to be scaled up when the overall management function faces an increased demand for connectivity, hand-

over or adaptation scenarios. For instance, a large amount of UEs approach an area which is governed by a few APs, or the data-rate demand increases significantly due to a major event. In such scenarios, horizontal scaling of the model plays an effective role to continue to deliver QoS levels expected by the users and hence, infrastructure resources need to be managed for automatic scaling of models on end device, base station or the relevant layer of network architecture that may be hosted partially or fully in cloud.

The layout of this chapter is as follows. Firstly, we distinguish the centralized and distributed ML deployments. We then present a short scenario to highlight how ITU-T standardization guideline can be used to deploy ML pipelines in future networks. We further present a short summary of popular methods and synergies among emerging paradigms like distributed ML, cloud computing and federated learning that enable complex scenarios for training and deployment of AI/ML models for scalable consumption and upgrade. Lastly, we present an initial assessment of the promising role of online optimizations that can serve to solve recurring and adaptivity requirements such as for UE-to-AP assignments in a dynamic on the fly fashion within an evolving network.

7.2 Centralized and Distributed ML Deployments

When AI/ML models are to be consumed in a distributed setting, it becomes important to consider topological and resource-constrained aspects of various components of the wireless networks. Some of these are more central and resource-rich such as the APs, while other elements such as the RIS, passive elements such as metasurface reflectors, or consumer devices such as UEs or IoT sensor devices are distributed and may find themselves at edges of the network while also being resource-constrained. Hence, future wireless networks present a distributed system which can benefit from various development and deployment capabilities seen in the broader field of Distributed ML, where the models may be developed, updated or deployed at any of the cited entities.

7.3 Deployment Units and Deployment Enabling Paradigms

In B5G/6G networks, AI is expected to be used across the network components, from the core to the terminals (UEs), and at all communication layers, from the physical layer (L1) to the application layer (L7). With this broad perspective in mind, the long-term management and sustainability of AI/ML capability necessitates the identification of a deployment unit. The concept of pipelines arises as a modular, flexible, and reusable implementation unit for ML and its associated data processing requirements, using some well-established methods from the field of data science.

7.3.1 ITU-T standardization guideline for deploying ML in future networks

The ITU-T standardization body's Focus Group FG-ML5G have recognized this potential of ML and extract, transform, load (ETL) pipelines and developed a technical

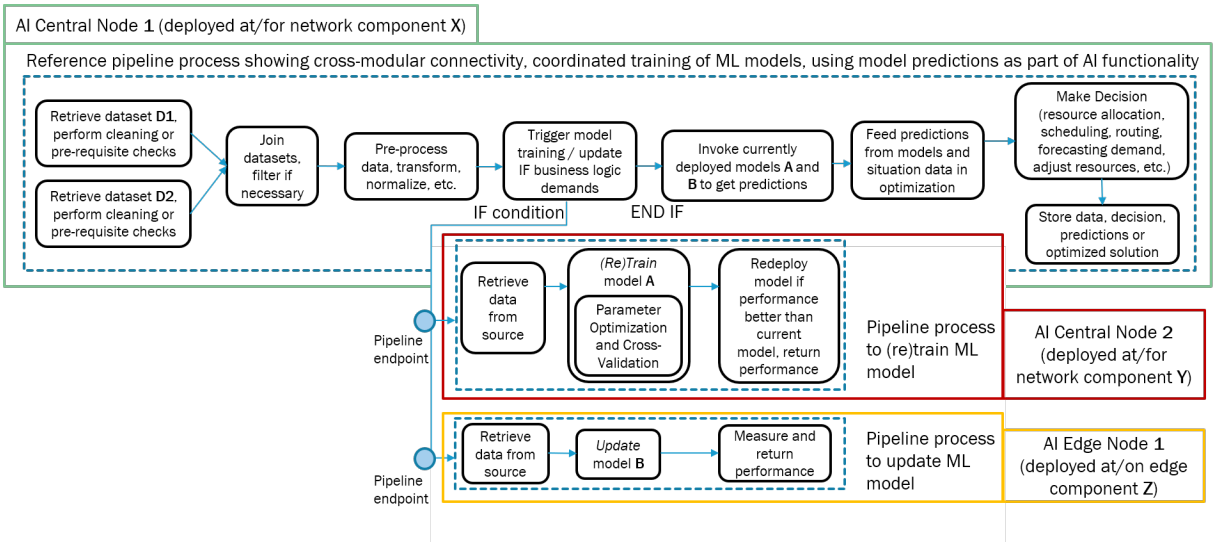


Figure 7.1: Reference pipeline processes illustrating cross layer deployment, connectivity and coordination

specification codenamed Y.3172, entitled “Architectural framework for ML in future networks including IMT-2020” [235]. This specification provides guidelines on training, deployment and orchestration aspects centered around the notion of ML/ETL pipelines that are bundled as cloud computing containers. Harnessing cloud computing techniques, the pipeline containers may be exposed as REST web-services and deployed at the core or edges of the network. In this way, complex interactions can be realized among distributed components of the network from whom data can be collected widely and frequently, while complex data processing tasks such as model training or updates, can be triggered on infrastructure nodes that possess better resource capacity. The objective of this synergy with cloud computing is to deliver the expected QoS for connectivity, adaptation and other low-latency requirements of THz systems. In Fig. 7.1 we depict this concept diagrammatically.

As exemplified in Fig. 7.1, a pipeline process is a set of operations, arranged as nodes, which ingest data, transform or pre-process it, and may trigger local or remote training, retraining or updating of ML models (by invoking externally deployed pipeline’s endpoint). It may also conditionally employ predictions from ML models for newly arriving data to adapt connectivity in an evolving network environment, or interfacing with optimization components in real-time. Predictions may be a classification score, regression value, or a decision artefact e.g., a schedule, resource allocation plan or an advice, which is returned to the invoking component to take automated action. Outcomes can also be stored in a data sink for later reuse. Pipeline processes may be invoked on demand or scheduled fashion either offline or online scenarios. Such pipeline-based distributed AI/ML pipelines also integrate disparate B5G network functions that need to cooperate or coordinate with each other to achieve cross-layer decision making.

As recognized and advocated in Y.3172 and related specifications by ITU-T FG-5GML as well its predecessor focus group on autonomous networks (FG-AN), enabling AI/ML in future wireless networks would require synergies with cloud computing paradigm. This synergy is essential to realize effective deployment, adoption and continuous upgrade of AI/Ml models by utilizing compute and storage capabilities of the cloud in scalable fashion. These specifications also highlight the challenges and op-

portunities by exploiting pipelines to loosely interface data from different upstream and downstream layers of the network. This fusion of data can be exploited for instance in UE-to-AP assignment predictions, which usually consume link level properties of the network including location, topology, resource requirements of UEs and resource capacity of APs, to optimally assign a UE to the best AP having minimal LoS blockages or interference. With the overlay of interacting pipelines, such connectivity can be effectively established by consuming data upstream from lower layers of the network and vice versa. Such interactive constellations to fuse and merge data from different network components and layers holds promising opportunities for AI/ML to tackle hard problems under more realistic settings such as by enabling cross-layer collaboration.

The emerging paradigm of edge computing raises similar cross-layer deployment challenges. The fast-growing Internet-of-Things (IoT) industry is expected to adopt edge computing and would arguably be one of the biggest beneficiaries of B5G based communications. The primary challenge is to find effective mechanisms to process massive volumes of data created by a large number of sensors or end devices. However, adopting edge computing in relevance to machine learning demands resolution of standard issues such as training models on partial or local data, deploying and updating collaborative processes at the network's core and edge - all of which remain topics of active research and investigations within the field of distributed machine learning.

7.3.2 Distributed ML

Distributed ML aims at establishing mechanisms to train ML models in a collaborative fashion, with the objective to harness distributed compute infrastructure. Some components of this infrastructure may be small devices with limited system and communication resources, while others may be resource-rich, such as cloud-based virtual machines or containers that can be scaled horizontally and vertically.

In [236], some ideas were presented to train a deep neural network (DNN) model comprising a hierarchy of distributed compute nodes. These include end devices, edge nodes and a cloud node. A DNN is trained and maintained at each layer. The end devices and edge nodes have fewer neural network (NN) layers, while the network in the cloud has more NN layers. Using this architecture, a multi-sensor multi-camera surveillance application deployed at various end nodes is able to efficiently perform inference with required accuracy. To limit the communication latency between the end device (or edge node) and the cloud, aggregate data is sent to the cloud node for inference only when the end devices cannot reach a high degree of classification confidence. In comparison to the alternative option when all raw data from end (sensor) devices were to be sent to the central cloud node, the presented approach is 20 times faster. While this strategy seems to hold good promise for sensor fusion based applications, its applicability in IoT-based or communication-intensive systems remains disputed because the models at end devices are rather limited NN models and may need to rely on the central NN model more frequently.

7.3.3 Federated Learning

Federated learning makes use of decentralized infrastructure to train a shared machine learning model with the help of potentially resource restricted end devices distributed at

network edges. These nodes collaborate with a centralized node that serves as a core. In federated learning, end devices or edge nodes train and maintain local machine learning models utilizing the device's regularly available (or created) batches of data. The intricacy and capability of local models are limited by the end device's resources.

Infrequently, a summary of local model's parameters e.g. weights or coefficients are securely transmitted to the core node which updates a consolidated machine learning model. Hence, the consolidated model is a result of joint but disparate training conducted at potentially millions of end nodes. This approach retains data and the inference (application of prediction model) only at the end nodes so the communication overhead to transfer all raw data to the core is avoided, while also preserving the privacy and confidentiality of local data.

In [237], Google researchers introduced federated learning for a query suggesting application that is installed on mobile applications. The work presents proposals to train a miniature version of tensor flow based model on mobile devices that consumes minimum energy and minimally interferes with the user experience. Additionally, several technical challenges to achieve federated learning are highlighted. A fundamental challenge is to update the shared model. In [238], the same team presented their solution in the form of a federated averaging algorithm that combines local stochastic gradient descent (SGD) based updates of local model with a server that performs model averaging on the shared model. However, to ensure the integrity of the consolidated (shared) model, these updates are governed through a Secure Aggregation protocol that only performs cryptographically-secured model updates after a sufficient number (hundreds or thousands) of end devices share their updates i.e. an aggregate data structure must be assembled first. This helps i) to prevent local phenomenon such as concept drift which happens when distributions of data start exhibiting gradual or abrupt differences, ii) to tackle corner cases such as anomalous data collection or existence of outliers, which can negatively skew the parameters of the consolidated model and iii) to preserve privacy of the local model parameters which are shared in cryptographically secured format [239]. The updated shared model is made available to selected end devices, which test the model on locally available data, after which the shared model is updated on all end devices.

The cited solutions are applicable in many other domains as well, e.g., IoT where heterogeneous sensor devices can federate, autonomous driving where a large fleet of vehicles can federate or network industries where multiple sensors or devices communicate e.g., railways, airlines, gas networks and power grids. However, future B5G networks with dense topologies containing a potentially large number of UEs and APs located within a small geographical area, may especially find various use cases for federated learning. To summarize, federated learning can be beneficial in many industry verticals, where general-purpose privacy-aware pattern recognition takes precedence over personalized pattern recognition. However, federated learning has yet to mature in terms of widely available deployment and orchestration tooling that can be applied in a variety of domains and applications.

7.4 Online Optimizations in B5G

Another promising but under-explored approach for a deploy-able solution to a changing problems lies in online optimizations. The basic concept is that, given a recurring

optimization problem such as the joint UE-to-AP association and resource allocation problem, the problem instance is fed to a solver which solves it in a continuous fashion, while taking into consideration any trigger events that may change the problem instance being solved.

This approach offers several benefits. Firstly, a best solution can be fetched within a desired time period after a change event is triggered, e.g., after new UEs appear in the network or locations of UE(s) change due to user mobility. Secondly, the network assignments can be consolidated for optimal resource usage at frequent intervals, in what can be referred to as 'resource consolidation'. Thirdly, a set of best solutions retrieved from the solver at regular intervals or after a change event is triggered, can be labelled (see for reference: ARIADNE deliverable D4.1, chapter 4) to generate training datasets for ML. These situation-rich training datasets can be used in parallel to update the ML model from time to time, where the ML model aims to predict the assignment and resource allocation as a secondary solution.

Hence, online optimizations can provide just-in-time solutions as well as frequent snapshots of the network, which are leveraged for (re)training of the ML predictive models in a hybrid scheme, until the ML model achieves the desired predictive quality and stability - at which point, online predictions can replace online optimizations.

7.5 Conclusion

This chapter summarized deployment strategies for the adoption of ML based solutions in future B5G wireless networks. The chapter presented enabling paradigms, technologies and also presented some challenges and obstacles that could be tackled with the centralized and distributed deployments. Certain principles and limitations observed in the state of art were highlighted as these provide guidelines in selecting appropriate deployment strategies. Last but not least, promising research directions like online optimizations were briefly addressed as a potential dual solution to just-in-time solving of recurrent optimization problems as well to generate and retrain ML models post deployment.

Summary of Conclusions

This deliverable presented work that extended the initial results shared in the deliverable D4.1 by addressing machine learning based network intelligence for problems of interest in ARIADNE. It focused on optimization of radio elements placement, beamforming for mobile UEs, channel estimation and feedbacks in massive MIMO systems as well as complex event recognition techniques that can assist in forecasting complex events in an evolving network. Moreover, system level modeling was described which was used to generate synthetic data to be exploited by AI/ML frameworks and tools. Additionally, machine learning algorithms and deployment strategies were analyzed within the light of state of the art, where emerging synergies of enabling paradigms were highlighted.

To summarize, **Chapter 1** introduced the formulation of radio placement optimization as a multi-agent reinforcement learning (MARL) problem, where the agents represent base stations and the RISs, while the radio receivers face blockages. Moreover, the implementation of the multi-agent environment was described together with the implementation of the multi-agent policy as neural network. **Chapter 2** introduced the innovative approach of dynamic beamforming optimization using deep reinforcement learning (DRL), including a BS-RIS-UE scenario. This is the first work of its kind that used IMPALA framework for distributed DRL to provide beamforming optimization in a MISO multi-UE case as well as in the BS-RIS-UE scenario. **Chapter 3** presented an empirical assessment of automata learning technique in the telecommunications domain using publicly available data and compared the preliminary results of current recognition/detection performance for forecasting complex events. The performance demonstrated superiority of proposed technique in terms of predictive accuracy, simplicity and comprehensibility of learned automata. **Chapter 4** described the system level modeling with mobile users that can simulate various ARIADNE scenarios and test-cases. A first set of simulation data included received power, which is planned to be used as input to complex event forecasting techniques towards proactive handover for blockage avoidance. Subsequently, in **Chapter 5** an MDDL-based channel estimation and feedback scheme was presented for wideband high frequency massive hybrid MIMO systems, achieving reduced overhead. First, for the uplink channel estimation of TDD systems, joint training of the phase shift network and the channel estimator was proposed as an auto-encoder. Specifically, an MMV-LAMP network was applied to recover multiple subcarriers' channels. Moreover, downlink channel estimation and feedback of FDD systems were considered where the pilots at the BS and channel estimator at the users can be jointly trained as an encoder and a decoder, respectively. Only the received pilots on part of the subcarriers were fed back to the BS, while the proposed MMV-LAMP network efficiently reconstructed the spatial-frequency channel matrix. The simulation results verified that the proposed MDDL-based solution outperforms conventional schemes. Lastly, **Chapter 6** and **Chapter 7** focused on critical operations and techniques based on AI/ML methods that are enabling innovative applications and services in B5G wireless networks, ranging from PHY to the network layer. Successful transformation and integration of these layers with AI/ML will heavily rely on their formulation as ML problems and thus, a comprehensive review was presented on how to select the appropriate family of algorithms in order to solve the problems depending on the type of input data and the expected outcomes. Finally, some deployment strategies have been highlighted that stem from the non-functional

and operational concerns to manage AI/ML pipelines, which collaboratively solve static, dynamic and recurrent problems in future networks, and can be deployed in centralized or distributed constellations.

All the above activities are in-progress in terms of theoretical studies, AI/ML algorithms implementation, simulation platforms and tools. The work on beamforming optimization will continue based on deep learning methods for channel estimation as well as joint optimization with RIS phase shifting. To that end, channel coefficients will be generated by ray-tracing simulations. Radio placement optimization results based on reinforcement learning, will become available for RIS-aided networks. Moreover, system level simulation data will be generated to be used as input to complex event forecasting methods, while metaheuristics optimization together with ML modeling work will be continued for resource allocation optimizations. New simulation scenarios will be investigated to include RISs. Proactive handover, user-AP assignment, capacity outage are some of the problems that will be studied in order to evaluate the benefits of RISs and AI/ML techniques. Finally, within WP4 activities, the partners will collectively select the test cases and the tools that will be used for the system level software demonstration in T5.3.

Bibliography

[1] E. C. A. Correia, "Attention, please! a survey of neural attention models in deep learning," *arXiv:2103.16775 [cs.LG]*, Mar. 2021.

[2] K. C. e. a. D. Bahdanau, "Neural machine translation by jointly learning to align and translate," in *the 3rd International Conference on Learning Representations*, May 2015.

[3] N. H. e. a. V. Mnih, "Recurrent models of visual attention," in *the 27th International conference on Neural Information Processing Systems*, Dec 2014, pp. 2204–2212.

[4] J. B. e. a. K. Xu, "Show, attend and tell: Neural image caption generation with visual attention," in *the 32nd International Conference on Machine Learning*, Jul 2015, pp. 2048–2057.

[5] N. S. e. a. A. Vaswani, "Attention is all you need," in *the 31st International Conference on Neural Information Processing Systems*, Dec 2017, pp. 6000–6010.

[6] L. E. e. a. W. Shang, "Agent-centric representations for multi-agent reinforcement learning," *arXiv:2104.09402 [cs.LG]*, Apr. 2021.

[7] G. R. MacCartney and T. S. Rappaport, "Millimeter-wave base station diversity for 5g coordinated multipoint (comp) applications," *IEEE Trans. Wireless Commun.*, vol. 18, no. 7, p. 3395–3410, Jul. 2019.

[8] R. Sutton and A. Barto, *Reinforcement Learning— An Introduciton, 2nd ed.* Cambridge, MA, USA: The MIT Press, 2018.

[9] V. Konda and J. Tsitsiklis, "On actor-critic algorithms," *SIAM Journal on Control and Optimization*, pp. 1143–1166, 2003.

[10] e. a. L. Espeholt, "Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures," in *the 35th International Conference on Machine Learning*, 2018, pp. 1407–1416.

[11] V. Lifschitz, *Answer set programming*. Springer, 2019.

[12] G. Cugola and A. Margara, "Processing flows of information: From data stream to complex event processing," *ACM Comput. Surv.*, vol. 44, no. 3, pp. 15:1–15:62, 2012.

[13] N. Giatrakos, E. Alevizos, A. Artikis, A. Deligiannakis, and M. N. Garofalakis, "Complex event recognition in the big data era: a survey," *VLDB J.*, vol. 29, no. 1, pp. 313–352, 2020.

[14] E. Alevizos, A. Artikis, and G. Paliouras, "Complex event forecasting with prediction suffix trees," *VLDB Journal*, vol. to appear, 2021.

[15] Y. Li and T. Ge, "Imminence monitoring of critical events: A representation learning approach," in *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*, G. Li, Z. Li, S. Idreos, and D. Srivastava, Eds. ACM, 2021, pp. 1103–1115. [Online]. Available: https://doi.org/10.1145/3448016.3452804

[16] R. Bruns, J. Dunkel, and N. Offel, "Learning of complex event processing rules with genetic programming," *Expert Syst. Appl.*, vol. 129, pp. 186–199, 2019. [Online]. Available: https://doi.org/10.1016/j.eswa.2019.04.007

[17] G. Dafé, A. Veloso, M. J. Zaki, and W. M. Jr., "Learning sequential classifiers from long and noisy discrete-event sequences efficiently," *Data Min. Knowl. Discov.*, vol. 29, no. 6, pp. 1685–1708, 2015. [Online]. Available: https://doi.org/10.1007/s10618-014-0391-9

[18] O. Lee and J. E. Jung, "Sequence clustering-based automated rule generation for adaptive complex event processing," *Future Gener. Comput. Syst.*, vol. 66, pp. 100–109, 2017. [Online]. Available: https://doi.org/10.1016/j.future.2016.02.011

[19] A. Margara, G. Cugola, and G. Tamburrelli, "Learning from the past: automated rule generation for complex event processing," in *The 8th ACM International Conference on Distributed Event-Based Systems, DEBS '14, Mumbai, India, May 26-29, 2014*, U. Bellur and R. Kothari, Eds. ACM, 2014, pp. 47–58. [Online]. Available: https://doi.org/10.1145/2611286.2611289

[20] N. Katzouris, G. PALIOURAS, and A. ARTIKIS, "Online learning probabilistic event calculus theories in answer set programming," *Theory and Practice of Logic Programming*, pp. 1–25, 2021.

[21] N. Katzouris, E. Michelioudakis, A. Artikis, and G. Paliouras, "Online learning of weighted relational rules for complex event recognition," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2018, pp. 396–413.

[22] N. Katzouris and A. Artikis, "Woled: A tool for online learning weighted answer set rules for temporal reasoning under uncertainty," in *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, vol. 17, no. 1, 2020, pp. 790–799.

[23] M. Law, A. Russo, and K. Broda, "Inductive learning of answer set programs from noisy examples," *arXiv preprint arXiv:1808.08441*, 2018.

[24] D. C. Montgomery, C. L. Jennings, and M. Kulahci, *Introduction to time series analysis and forecasting*. John Wiley & Sons, 2015.

[25] R. Begleiter, R. El-Yaniv, and G. Yona, "On prediction using variable order markov models," *J. Artif. Intell. Res.*, vol. 22, pp. 385–421, 2004.

[26] D. Ron, Y. Singer, and N. Tishby, "The power of amnesia: Learning probabilistic automata with variable memory length," *Machine Learning*, vol. 25, no. 2-3, pp. 117–149, 1996.

[27] J. G. Cleary and I. H. Witten, "Data compression using adaptive coding and partial string matching," *IEEE Trans. Communications*, vol. 32, no. 4, pp. 396–402, 1984.

[28] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens, "The context-tree weighting method: basic properties," *IEEE Trans. Information Theory*, vol. 41, no. 3, pp. 653–664, 1995.

[29] R. Vilalta and S. Ma, "Predicting rare events in temporal domains," in *ICDM*. IEEE Computer Society, 2002, pp. 474–481.

[30] S. Laxman, V. Tankasali, and R. W. White, "Stream prediction using a generative model based on frequent episodes in event sequences," in *KDD*. ACM, 2008, pp. 453–461.

[31] C. Zhou, B. Cule, and B. Goethals, "A pattern based predictor for event streams," *Expert Syst. Appl.*, vol. 42, no. 23, pp. 9294–9306, 2015.

[32] C. Cho, Y. Wu, S. Yen, Y. Zheng, and A. L. P. Chen, "On-line rule matching for event prediction," *VLDB J.*, vol. 20, no. 3, pp. 303–334, 2011.

[33] A. E. Márquez-Chamorro, M. Resinas, and A. Ruiz-Cortés, "Predictive monitoring of business processes: A survey," *IEEE Trans. Services Computing*, vol. 11, no. 6, pp. 962–977, 2018.

[34] L. J. Fülöp, Á. Beszédes, G. Toth, H. Demeter, L. Vidács, and L. Farkas, "Predictive complex event processing: a conceptual framework for combining complex event processing and predictive analytics," in *BCI*. ACM, 2012, pp. 26–31.

[35] M. Christ, J. Krumeich, and A. W. Kempa-Liehr, "Integrating predictive analytics into complex event processing by using conditional density estimations," in *EDOC Workshops*. IEEE Computer Society, 2016, pp. 1–8.

[36] Y. Engel and O. Etzion, "Towards proactive event-driven computing," in *DEBS*. ACM, 2011, pp. 125–136.

[37] Z. Li, X. Ding, and T. Liu, "Constructing narrative event evolutionary graph for script event prediction," in *IJCAI*. ijcai.org, 2018, pp. 4201–4207.

[38] B. Chang, Y. Park, D. Park, S. Kim, and J. Kang, "Content-aware hierarchical point-of-interest embedding model for successive POI recommendation," in *IJCAI*. ijcai.org, 2018, pp. 3301–3307.

[39] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "Statistical and machine learning forecasting methods: Concerns and ways forward," *PloS one*, vol. 13, no. 3, p. e0194889, 2018.

[40] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, no. 5, pp. 206–215, 2019.

[41] D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju, "Fooling LIME and SHAP: adversarial attacks on post hoc explanation methods," in *AIES '20: AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, February 7-8, 2020*, A. N. Markham, J. Powles, T. Walsh, and A. L. Washington, Eds. ACM, 2020, pp. 180–186. [Online]. Available: https://doi.org/10.1145/3375627.3375830

[42] A. Jung and P. H. J. Nardelli, "An information-theoretic approach to personalized explainable machine learning," *IEEE Signal Process. Lett.*, vol. 27, pp. 825–829, 2020. [Online]. Available: https://doi.org/10.1109/LSP.2020.2993176

[43] W. Van Der Aalst, *Process mining: discovery, conformance and enhancement of business processes*. Springer, 2011.

[44] A. Grez, C. Riveros, and M. Ugarte, "A formal framework for complex event processing," in *ICDT*, ser. LIPIcs, vol. 127. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2019, pp. 5:1–5:18.

[45] L. D'Antoni and M. Veanes, "The power of symbolic automata and transducers," in *CAV (1)*, ser. Lecture Notes in Computer Science, vol. 10426. Springer, 2017, pp. 47–67.

[46] W. Daelemans, "Colin de la higuera: Grammatical inference: learning automata and grammars - cambridge university press, 2010, iv + 417 pages," *Mach. Transl.*, vol. 24, no. 3-4, pp. 291–293, 2010. [Online]. Available: https://doi.org/10.1007/s10590-011-9086-9

[47] W. Wieczorek, *Grammatical Inference - Algorithms, Routines and Applications*, ser. Studies in Computational Intelligence. Springer, 2017, vol. 673. [Online]. Available: https://doi.org/10.1007/978-3-319-46801-3

[48] M. Hasanbeig, N. Y. Jeppu, A. Abate, T. Melham, and D. Kroening, "Deepsynth: Automata synthesis for automatic task segmentation in deep reinforcement learning," in *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 2021, pp. 7647–7656. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/16935

[49] D. Furelos-Blanco, M. Law, A. Jonsson, K. Broda, and A. Russo, "Induction and exploitation of subgoal automata for reinforcement learning," *J. Artif. Intell. Res.*, vol. 70, pp. 1031–1116, 2021. [Online]. Available: https://doi.org/10.1613/jair.1.12372

[50] M. Shvo, A. C. Li, R. T. Icarte, and S. A. McIlraith, "Interpretable sequence classification via discrete optimization," in *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative*

Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021. AAAI Press, 2021, pp. 9647–9656. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/17161

[51] N. I. U. Khaleel Ahmad and G. C. Deka, *Opportunistic Networks - Mobility Models, Protocols, Security, and Privacy*. CRC Press-Taylor & Francis Group, 2018.

[52] Papageorgiou, Christos and Birkos, Konstantinos and Dagiuklas, Tasos and Kotsopoulos, Stavros, "An obstacle-aware human mobility model for ad hoc networks," in *2009 IEEE International Symposium on Modeling, Analysis Simulation of Computer and Telecommunication Systems*, 2009.

[53] Xing, Yunchou and Rappaport, Theodore S., "ICC 2021 - IEEE International Conference on Communications," in *Propagation Measurements and Path Loss Models for sub-THz in Urban Microcells*, 2021.

[54] R. W. H. Jr., N. González Prelcic, S. Rangan, W. Roh, and A. M. Sayeed, "An overview of signal processing techniques for millimeter wave MIMO systems," *IEEE J. Sel. Top. Signal Process.*, vol. 10, no. 3, pp. 436–453, 2016. [Online]. Available: https://doi.org/10.1109/JSTSP.2016.2523924

[55] Z. Gao, L. Dai, D. Mi, Z. Wang, M. A. Imran, and M. Z. Shakir, "Mmwave massive-mimo-based wireless backhaul for the 5g ultra-dense network," *IEEE Wirel. Commun.*, vol. 22, no. 5, pp. 13–21, 2015. [Online]. Available: https://doi.org/10.1109/MWC.2015.7306533

[56] L. Lu, G. Y. Li, A. L. Swindlehurst, A. E. Ashikhmin, and R. Zhang, "An overview of massive MIMO: benefits and challenges," *IEEE J. Sel. Top. Signal Process.*, vol. 8, no. 5, pp. 742–758, 2014. [Online]. Available: https://doi.org/10.1109/JSTSP.2014.2317671

[57] S. Han, C. I, Z. Xu, and C. Rowell, "Large-scale antenna systems with hybrid analog and digital beamforming for millimeter wave 5g," *IEEE Commun. Mag.*, vol. 53, no. 1, pp. 186–194, 2015. [Online]. Available: https://doi.org/10.1109/MCOM.2015.7010533

[58] F. Sohrabi and W. Yu, "Hybrid digital and analog beamforming design for large-scale antenna arrays," *IEEE J. Sel. Top. Signal Process.*, vol. 10, no. 3, pp. 501–513, 2016. [Online]. Available: https://doi.org/10.1109/JSTSP.2016.2520912

[59] J. Mao, Z. Gao, Y. Wu, and M. Alouini, "Over-sampling codebook-based hybrid minimum sum-mean-square-error precoding for millimeter-wave 3d-mimo," *IEEE Wirel. Commun. Lett.*, vol. 7, no. 6, pp. 938–941, 2018. [Online]. Available: https://doi.org/10.1109/LWC.2018.2839723

[60] P. Dong, H. Zhang, G. Y. Li, I. S. Gaspar, and N. Naderializadeh, "Deep cnn-based channel estimation for mmwave massive MIMO systems," *IEEE J. Sel. Top. Signal Process.*, vol. 13, no. 5, pp. 989–1000, 2019. [Online]. Available: https://doi.org/10.1109/JSTSP.2019.2925975

[61] W. Shen, L. Dai, Y. Shi, B. Shim, and Z. Wang, "Joint channel training and feedback for FDD massive MIMO systems," *IEEE Trans. Veh. Technol.*, vol. 65, no. 10, pp. 8762–8767, 2016. [Online]. Available: https://doi.org/10.1109/TVT.2015.2508033

[62] W. Ma, C. Qi, and G. Y. Li, "High-resolution channel estimation for frequency-selective mmwave massive MIMO systems," *IEEE Trans. Wirel. Commun.*, vol. 19, no. 5, pp. 3517–3529, 2020. [Online]. Available: https://doi.org/10.1109/TWC.2020.2974728

[63] Z. Gao, L. Dai, Z. Wang, and S. Chen, "Spatially common sparsity based adaptive channel estimation and feedback for FDD massive MIMO," *IEEE Trans. Signal Process.*, vol. 63, no. 23, pp. 6169–6183, 2015. [Online]. Available: https://doi.org/10.1109/TSP.2015.2463260

[64] Y. Han, Q. Liu, C. Wen, S. Jin, and K. Wong, "FDD massive MIMO based on efficient downlink channel reconstruction," *IEEE Trans. Commun.*, vol. 67, no. 6, pp. 4020–4034, 2019. [Online]. Available: https://doi.org/10.1109/TCOMM.2019.2900625

[65] A. Liao, Z. Gao, H. Wang, S. Chen, M. Alouini, and H. Yin, "Closed-loop sparse channel estimation for wideband millimeter-wave full-dimensional MIMO systems," *IEEE Trans. Commun.*, vol. 67, no. 12, pp. 8329–8345, 2019. [Online]. Available: https://doi.org/10.1109/TCOMM.2019.2942911

[66] Z. Wan, Z. Gao, B. Shim, K. Yang, G. Mao, and M. Alouini, "Compressive sensing based channel estimation for millimeter-wave full-dimensional MIMO with lens-array," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 2337–2342, 2020. [Online]. Available: https://doi.org/10.1109/TVT.2019.2962242

[67] X. Lin, S. Wu, C. Jiang, L. Kuang, J. Yan, and L. Hanzo, "Estimation of broadband multiuser millimeter wave massive MIMO-OFDM channels by exploiting their sparse structure," *IEEE Trans. Wirel. Commun.*, vol. 17, no. 6, pp. 3959–3973, 2018. [Online]. Available: https://doi.org/10.1109/TWC.2018.2818142

[68] Z. Qin, H. Ye, G. Y. Li, and B. F. Juang, "Deep learning in physical layer communications," *IEEE Wirel. Commun.*, vol. 26, no. 2, pp. 93–99, 2019. [Online]. Available: https://doi.org/10.1109/MWC.2019.1800601

[69] H. Huang, S. Guo, G. Gui, Z. Yang, J. Zhang, H. Sari, and F. Adachi, "Deep learning for physical-layer 5g wireless techniques: Opportunities, challenges and solutions," *IEEE Wirel. Commun.*, vol. 27, no. 1, pp. 214–222, 2020. [Online]. Available: https://doi.org/10.1109/MWC.2019.1900027

[70] Z. Qin, X. Zhou, L. Zhang, Y. Gao, Y. Liang, and G. Y. Li, "20 years of evolution from cognitive to intelligent communications," *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 1, pp. 6–20, 2020. [Online]. Available: https://doi.org/10.1109/TCCN.2019.2949279

[71] A. Zappone, M. D. Renzo, and M. Debbah, "Wireless networks design in the era of deep learning: Model-based, ai-based, or both?" *IEEE*

Trans. Commun., vol. 67, no. 10, pp. 7331–7376, 2019. [Online]. Available: https://doi.org/10.1109/TCOMM.2019.2924010

[72] Y. He, J. Zhang, S. Jin, C. Wen, and G. Y. Li, "Model-driven DNN decoder for turbo codes: Design, simulation, and experimental results," *IEEE Trans. Commun.*, vol. 68, no. 10, pp. 6127–6140, 2020. [Online]. Available: https://doi.org/10.1109/TCOMM.2020.3010964

[73] Y. Cui, S. Li, and W. Zhang, "Jointly sparse signal recovery and support recovery via deep learning with applications in mimo-based grant-free random access," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 3, pp. 788–803, 2021. [Online]. Available: https://doi.org/10.1109/JSAC.2020.3018802

[74] C. Qi, Y. Wang, and G. Y. Li, "Deep learning for beam training in millimeter wave massive mimo systems," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2020.

[75] H. Huang, Y. Song, J. Yang, G. Gui, and F. Adachi, "Deep-learning-based millimeter-wave massive MIMO for hybrid precoding," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 3027–3032, 2019. [Online]. Available: https://doi.org/10.1109/TVT.2019.2893928

[76] T. Peken, R. Tandon, and T. Bose, "Unsupervised mmwave beamforming via autoencoders," in *2020 IEEE International Conference on Communications, ICC 2020, Dublin, Ireland, June 7-11, 2020*. IEEE, 2020, pp. 1–6. [Online]. Available: https://doi.org/10.1109/ICC40277.2020.9149222

[77] K. M. Attiah, F. Sohrabi, and W. Yu, "Deep learning approach to channel sensing and hybrid precoding for TDD massive MIMO systems," in *IEEE Globecom Workshops, GLOBECOM Workshops 2020, Virtual Event, Taiwan, December 7-11, 2020*. IEEE, 2020, pp. 1–6. [Online]. Available: https://doi.org/10.1109/GCWkshps50303.2020.9367586

[78] F. Sohrabi, K. M. Attiah, and W. Yu, "Deep learning for distributed channel feedback and multiuser precoding in FDD massive MIMO," *IEEE Trans. Wirel. Commun.*, vol. 20, no. 7, pp. 4044–4057, 2021. [Online]. Available: https://doi.org/10.1109/TWC.2021.3055202

[79] Y. Qiang, X. Shao, and X. Chen, "A model-driven deep learning algorithm for joint activity detection and channel estimation," *IEEE Commun. Lett.*, vol. 24, no. 11, pp. 2508–2512, 2020. [Online]. Available: https://doi.org/10.1109/LCOMM.2020.3011571

[80] H. He, C. Wen, S. Jin, and G. Y. Li, "Model-driven deep learning for MIMO detection," *IEEE Trans. Signal Process.*, vol. 68, pp. 1702–1715, 2020. [Online]. Available: https://doi.org/10.1109/TSP.2020.2976585

[81] H. Ye, L. Liang, G. Y. Li, and B. Juang, "Deep learning-based end-to-end wireless communication systems with conditional gans as unknown channels," *IEEE Trans. Wirel. Commun.*, vol. 19, no. 5, pp. 3133–3143, 2020. [Online]. Available: https://doi.org/10.1109/TWC.2020.2970707

[82] P. Liang, J. Fan, W. Shen, Z. Qin, and G. Y. Li, "Deep learning and compressive sensing-based CSI feedback in FDD massive MIMO systems," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 9217–9222, 2020. [Online]. Available: https://doi.org/10.1109/TVT.2020.3004842

[83] J. Guo, C. Wen, S. Jin, and G. Y. Li, "Convolutional neural network-based multiple-rate compressive sensing for massive MIMO CSI feedback: Design, simulation, and analysis," *IEEE Trans. Wirel. Commun.*, vol. 19, no. 4, pp. 2827–2840, 2020. [Online]. Available: https://doi.org/10.1109/TWC.2020.2968430

[84] H. Ye, F. Gao, J. Qian, H. Wang, and G. Y. Li, "Deep learning-based denoise network for CSI feedback in FDD massive MIMO systems," *IEEE Commun. Lett.*, vol. 24, no. 8, pp. 1742–1746, 2020. [Online]. Available: https://doi.org/10.1109/LCOMM.2020.2989499

[85] W. Ma, C. Qi, Z. Zhang, and J. Cheng, "Sparse channel estimation and hybrid precoding using deep learning for millimeter wave massive MIMO," *IEEE Trans. Commun.*, vol. 68, no. 5, pp. 2838–2849, 2020. [Online]. Available: https://doi.org/10.1109/TCOMM.2020.2974457

[86] X. Ma and Z. Gao, "Data-driven deep learning to design pilot and channel estimator for massive MIMO," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5677–5682, 2020. [Online]. Available: https://doi.org/10.1109/TVT.2020.2980905

[87] A. Zappone, M. D. Renzo, M. Debbah, T. L. Thanh, and X. Qian, "Model-aided wireless artificial intelligence: Embedding expert knowledge in deep neural networks for wireless system optimization," *IEEE Veh. Technol. Mag.*, vol. 14, no. 3, pp. 60–69, 2019. [Online]. Available: https://doi.org/10.1109/MVT.2019.2921627

[88] Y. Han, M. Li, S. Jin, C. Wen, and X. Ma, "Deep learning-based FDD non-stationary massive MIMO downlink channel reconstruction," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 9, pp. 1980–1993, 2020. [Online]. Available: https://doi.org/10.1109/JSAC.2020.3000836

[89] M. Borgerding, P. Schniter, and S. Rangan, "Amp-inspired deep networks for sparse linear inverse problems," *IEEE Trans. Signal Process.*, vol. 65, no. 16, pp. 4293–4308, 2017. [Online]. Available: https://doi.org/10.1109/TSP.2017.2708040

[90] H. He, S. Jin, C. Wen, F. Gao, G. Y. Li, and Z. Xu, "Model-driven deep learning for physical layer communications," *IEEE Wirel. Commun.*, vol. 26, no. 5, pp. 77–83, 2019. [Online]. Available: https://doi.org/10.1109/MWC.2019.1800447

[91] Y. Wei, M. Zhao, M. Zhao, M. Lei, and Q. Yu, "An amp-based network with deep residual learning for mmwave beamspace channel estimation," *IEEE Wirel. Commun. Lett.*, vol. 8, no. 4, pp. 1289–1292, 2019. [Online]. Available: https://doi.org/10.1109/LWC.2019.2916786

[92] H. He, C. Wen, S. Jin, and G. Y. Li, "Deep learning-based channel estimation for beamspace mmwave massive MIMO systems," *IEEE Wirel.*

Commun. Lett., vol. 7, no. 5, pp. 852–855, 2018. [Online]. Available: https://doi.org/10.1109/LWC.2018.2832128

[93] Z. Gao, L. Dai, S. Han, C. I, Z. Wang, and L. Hanzo, "Compressive sensing techniques for next-generation wireless communications," *IEEE Wirel. Commun.*, vol. 25, no. 3, pp. 144–153, 2018. [Online]. Available: https://doi.org/10.1109/MWC.2017.1700147

[94] M. Ke, Z. Gao, Y. Wu, X. Gao, and R. Schober, "Compressive sensing-based adaptive active user detection and channel estimation: Massive access meets massive MIMO," *IEEE Trans. Signal Process.*, vol. 68, pp. 764–779, 2020. [Online]. Available: https://doi.org/10.1109/TSP.2020.2967175

[95] D. P. Kingma and B. Jimmy, "Adam: A method for stochastic optimization," *arxiv:1412.6980*, 2014.

[96] Z. Gao, L. Dai, and Z. Wang, "Structured compressive sensing based superimposed pilot design in downlink large-scale mimo systems," *Electronics Letters*, vol. 50, no. 12, pp. 896–898, 2014.

[97] L. Liu and W. Yu, "Massive connectivity with massive MIMO - part I: device activity detection and channel estimation," *IEEE Trans. Signal Process.*, vol. 66, no. 11, pp. 2933–2946, 2018. [Online]. Available: https://doi.org/10.1109/TSP.2018.2818082

[98] A.-A. A. Boulogeorgos, A. Alexiou, T. Merkle, C. Schubert, R. Elschner, A. Katsiotis, P. Stavrianos, D. Kritharidis, P.-K. Chartsias, J. Kokkoniemi, M. Juntti, J. Lehtomaki, A. Teixeira, and F. Rodrigues, "Terahertz technologies to deliver optical network quality of experience in wireless systems beyond 5g," *IEEE Commun. Mag.*, vol. 56, no. 6, pp. 144–151, Jun. 2018.

[99] A.-A. A. Boulogeorgos and A. Alexiou, "Outage probability analysis of THz relaying systems," in *IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE, aug 2020.

[100] ——, *Next Generation Wireless Terahertz Communication Networks*. CRC Press, 2020, ch. Antenna misalignment and blockage in THz communications.

[101] A.-A. A. Boulogeorgos, A. Alexiou, D. Kritharidis, and et. al., "Wireless terahertz system architectures for networks beyond 5G," TERRANOVA CONSORTIUM, White paper 1.0, Jul. 2018.

[102] IEEE Standard for Information technology, "IEEE standard for information technology– local and metropolitan area networks– specific requirements– part 15.3: Amendment 2: Millimeter-wave-based alternative physical layer extension," 2009.

[103] IEEE Standard for High Data Rate Wireless Multi-Media Networks, "IEEE standard for high data rate wireless multi-media networks–amendment 2: 100 gb/s wireless switched point-to-point physical layer," 2017.

[104] S. Dang, O. Amin, B. Shihada, and M.-S. Alouini, "What should 6G be?" *Nature Electronics*, vol. 3, no. 1, pp. 20–29, 2020.

[105] K. B. Letaief, W. Chen, Y. Shi, J. Zhang, and Y.-J. A. Zhang, "The roadmap to 6g: AI empowered wireless networks," *IEEE Commun. Mag.*, vol. 57, no. 8, pp. 84–90, Aug. 2019.

[106] W. Saad, M. Bennis, and M. Chen, "A vision of 6g wireless systems: Applications, trends, technologies, and open research problems," *IEEE Network*, vol. 34, no. 3, pp. 134–142, 2020.

[107] J. Wang, C. Jiang, H. Zhang, Y. Ren, K.-C. Chen, and L. Hanzo, "Thirty years of machine learning: The road to pareto-optimal wireless networks," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1472–1514, 2020.

[108] Y. Liu, S. Bi, Z. Shi, and L. Hanzo, "When machine learning meets big data: A wireless communication perspective," *IEEE Veh. Technol. Mag.*, vol. 15, no. 1, pp. 63–72, mar 2020.

[109] J.-R. Jiang, "Short survey on physical layer authentication by machine-learning for 5G-based internet of things," in *3rd IEEE International Conference on Knowledge Innovation and Invention (ICKII)*. Kaohsiung, Taiwan: IEEE, Aug. 2020, pp. 41–44.

[110] A. A. Boulogeorgos, S. E. Trevlakis, S. A. Tegos, V. K. Papanikolaou, and G. K. Karagiannidis, "Machine learning in nano-scale biomedical engineering," *IEEE Trans. Mol. Biol. Multi-Scale Commun.*, vol. 7, no. 1, pp. 10–39, Mar. 2021.

[111] A.-A. A. Boulogeorgos, E. N. Papasotiriou, and A. Alexiou, "A distance and bandwidth dependent adaptive modulation scheme for THz communications," in *19th IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Kalamata, Greece, Jul. 2018.

[112] J. Kokkoniemi, J. Lehtomäki, and M. Juntti, "Measurements on penetration loss in terahertz band," in *10th European Conference on Antennas and Propagation (EuCAP)*, 2016, pp. 1–5.

[113] V. Petrov, J. M. Eckhardt, D. Moltchanov, Y. Koucheryavy, and T. Kurner, "Measurements of reflection and penetration losses in low terahertz band vehicular communications," in *14th European Conference on Antennas and Propagation (EuCAP)*, 2020, pp. 1–5.

[114] G. Stratidakis, G. D. Ntouni, A. A. Boulogeorgos, D. Kritharidis, and A. Alexiou, "A low-overhead hierarchical beam-tracking algorithm for thz wireless systems," in *European Conference on Networks and Communications (EuCNC)*, 2020, pp. 74–78.

[115] G. Stratidakis, E. N. Papasotiriou, H. Konstantinis, A.-A. A. Boulogeorgos, and A. Alexiou, "Relay-based blockage and antenna misalignment mitigation in thz wireless communications," in *2nd 6G Wireless Summit (6G SUMMIT)*, 2020, pp. 1–4.

[116] G. Stratidakis, A.-A. A. Boulogeorgos, and A. Alexiou, "A cooperative localization-aided tracking algorithm for thz wireless systems," in *IEEE Wireless Communications and Networking Conference (WCNC)*, 2019, pp. 1–7.

[117] R. Zhu, Y. E. Wang, Q. Xu, Y. Liu, and Y. D. Li, "Millimeter-wave to microwave mimo relays (m4r) for 5g building penetration communications," in *IEEE Radio and Wireless Symposium (RWS)*, 2018, pp. 206–208.

[118] S. K. Mahapatra, S. K. Mohapatra, S. Behera, and L. Kanoje, "An experimental analysis of penetration loss around buildings of an institution," in *International Conference on Green Computing and Internet of Things (ICGCIoT)*, 2015, pp. 75–76.

[119] H. Zhang, H. Zhang, W. Liu, K. Long, J. Dong, and V. C. M. Leung, "Energy efficient user clustering, hybrid precoding and power optimization in terahertz MIMO-NOMA systems," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 9, pp. 2074–2085, sep 2020.

[120] H. Sarieddeen, M.-S. Alouini, and T. Y. Al-Naffouri, "An overview of signal processing techniques for terahertz communications," 2021.

[121] S. Koenig, D. Lopez-Diaz, J. Antes, F. Boes, R. Henneberger, A. Leuther, A. Tessmann, R. Schmogrow, D. Hillerkuss, R. Palmer, T. Zwick, C. Koos, W. Freude, O. Ambacher, J. Leuthold, and I. Kallfass, "Wireless sub-thz communication system with high data rate," *Nature Photonics*, vol. 7, no. 12, pp. 977–981, 2013.

[122] T. Schenk, *RF Imperfections in High-Rate Wireless Systems*. The Netherlands: Springer, 2008.

[123] A.-A. A. Boulogeorgos, "Interference mitigation techniques in modern wireless communication systems," Ph.D. dissertation, Aristotle University of Thessaloniki, Thessaloniki, Greece, Sep. 2016.

[124] A.-A. A. Boulogeorgos and A. Alexiou, "How much do hardware imperfections affect the performance of reconfigurable intelligent surface-assisted systems?" *IEEE Open Journal of the Communications Society*, vol. 1, pp. 1185–1195, 2020.

[125] A.-A. A. Boulogeorgos, N. Chatzidiamantis, H. G. Sandalidis, A. Alexiou, and M. D. Renzo, "Cascaded composite turbulence and misalignment: Statistical characterization and applications to reconfigurable intelligent surface-empowered wireless systems," 2021.

[126] C. Yang, Z. He, Y. Peng, Y. Wang, and J. Yang, "Deep learning aided method for automatic modulation recognition," *IEEE Access*, vol. 7, pp. 109 063–109 068, Aug. 2019.

[127] F. N. Khan, K. Zhong, W. H. Al-Arashi, C. Yu, C. Lu, and A. P. T. Lau, "Modulation format identification in coherent receivers using deep machine learning," *IEEE Photon. Technol. Lett.*, vol. 28, no. 17, pp. 1886–1889, Sep. 2016.

[128] Y. Wu, X. Li, and J. Fang, "A deep learning approach for modulation recognition via exploiting temporal correlations," in *IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. Kalamata, GR: IEEE, Jun. 2018.

[129] S. I. H. Shah, S. Alam, S. A. Ghauri, A. Hussain, and F. A. Ansari, "A novel hybrid cuckoo search- extreme learning machine approach for modulation classification," *IEEE Access*, vol. 7, pp. 90 525–90 537, Jul. 2019.

[130] M. Li, G. Liu, S. Li, and Y. Wu, "Radio classify generative adversarial networks: A semi-supervised method for modulation recognition," in *IEEE 18th International Conference on Communication Technology (ICCT)*. Chongqing, China: IEEE, Oct. 2018.

[131] K. Bu, Y. He, X. Jing, and J. Han, "Adversarial transfer learning for deep learning based automatic modulation classification," *IEEE Signal Process. Lett.*, vol. 27, pp. 880–884, May 2020.

[132] M. O. Iqbal, M. M. Ur Rahman, M. A. Imran, A. Alomainy, and Q. H. Abbasi, "Modulation mode detection and classification for in vivo nano-scale communication systems operating in terahertz band," *IEEE Transactions on NanoBioscience*, vol. 18, no. 1, pp. 10–17, 2019.

[133] S. Moon, H. Kim, and I. Hwang, "Deep learning-based channel estimation and tracking for millimeter-wave vehicular communications," *J. Commun. Networks*, vol. 22, no. 3, pp. 177–184, Jun. 2020.

[134] W. Ma, C. Qi, Z. Zhang, and J. Cheng, "Sparse channel estimation and hybrid precoding using deep learning for millimeter wave massive MIMO," *IEEE Trans. Commun.*, vol. 68, no. 5, pp. 2838–2849, May 2020.

[135] Z. Mai, Y. Chen, and L. Du, "A novel blind mmWave channel estimation algorithm based on ML-ELM," *IEEE Commun. Lett.*, pp. 1–1, 2021.

[136] J. Wang, R. Han, L. Bai, T. Zhang, J. Liu, and J. Choi, "Coordinated beamforming for UAV-aided millimeter-wave communications using GPML-based channel estimation," *IEEE Trans. on Cogn. Commun. Netw.*, pp. 1–1, 2021.

[137] F. Zhu, A. Liu, and V. K. N. Lau, "Channel estimation and localization for mmWave systems: A sparse bayesian learning approach," in *IEEE International Conference on Communications (ICC)*. Shanghai, China: IEEE, May 2019.

[138] K. Satyanarayana, M. El-Hajjar, A. A. M. Mourad, and L. Hanzo, "Multi-user full duplex transceiver design for mmWave systems using learning-aided channel prediction," *IEEE Access*, vol. 7, pp. 66 068–66 083, May 2019.

[139] W. Ma, C. Qi, and G. Y. Li, "Machine learning for beam alignment in millimeter wave massive MIMO," *IEEE Wireless Commun. Lett.*, vol. 9, no. 6, pp. 875–878, Jun. 2020.

[140] S. Liu, Z. Gao, J. Zhang, M. D. Renzo, and M.-S. Alouini, "Deep denoising neural network assisted compressive channel estimation for mmWave intelligent reflecting surfaces," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 9223–9228, Aug. 2020.

[141] N. Samuel, T. Diskin, and A. Wiesel, "Learning to detect," *IEEE Trans. Signal Process.*, vol. 67, no. 10, pp. 2554–2564, May 2019.

[142] F. A. Aoudia and J. Hoydis, "Model-free training of end-to-end communication systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 11, pp. 2503–2516, Nov. 2019.

[143] S. Katla, L. Xiang, Y. Zhang, M. El-Hajjar, A. A. M. Mourad, and L. Hanzo, "Deep learning assisted detection for index modulation aided mmWave systems," *IEEE Access*, vol. 8, pp. 202 738–202 754, Nov. 2020.

[144] K. Satyanarayana, M. El-Hajjar, A. A. M. Mourad, P. Pietraski, and L. Hanzo, "Soft-decoding for multi-set space-time shift-keying mmWave systems: A deep learning approach," *IEEE Access*, vol. 8, pp. 49 584–49 595, Feb. 2020.

[145] Y.-S. Jeon, S.-N. Hong, and N. Lee, "Supervised-learning-aided communication framework for MIMO systems with low-resolution ADCs," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 7299–7313, Aug. 2018.

[146] Q. Mao, F. Hu, and Q. Hao, "Deep learning for intelligent wireless networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 2595–2621, Jul. 2018.

[147] Y. Ghasempour, C. R. C. M. da Silva, C. Cordeiro, and E. W. Knightly, "IEEE 802.11ay: Next-generation 60 GHz communication for 100 gb/s wi-fi," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 186–192, Dec. 2017.

[148] C. R. C. M. D. Silva, J. Kosloff, C. Chen, A. Lomayev, and C. Cordeiro, "Beamforming training for IEEE 802.11 ay millimeter wave systems," in *Information Theory and Applications Workshop (ITA)*. San Diego, CA, USA: IEEE, Feb. 2018.

[149] A.-A. A. Boulogeorgos and A. Alexiou, "Performance evaluation of the initial access procedure in wireless THz systems," in *16th International Symposium on Wireless Communication Systems (ISWCS)*. Oulu, Finland: IEEE, Aug. 2019.

[150] Z. Li, Z. Chen, X. Ma, and W. Chen, "Channel estimation for intelligent reflecting surface enabled terahertz MIMO systems: A deep learning perspective," in *IEEE/CIC International Conference on Communications in China (ICCC Workshops)*. IEEE, Aug. 2020.

[151] C. Anton-Haro and X. Mestre, "Learning and data-driven beam selection for mmWave communications: An angle of arrival-based approach," *IEEE Access*, vol. 7, pp. 20 404–20 415, Jan. 2019.

[152] A.-A. A. Boulogeorgos, G. Stratidakis, E. Papasotirou, J. Lehtomaki, J. Kokkoniemi, M. S. Mushtaq, M. Hatami, J.-C. Point, and A. Alexiou, "D4.2-THz driven MAC layer design and caching overlay method," 2017.

[153] H. J. Kwon, J. H. Lee, and W. Choi, "Machine learning-based beamforming in two-user MISO interference channels," in *International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. IEEE, Feb. 2019.

[154] T. Peken, S. Adiga, R. Tandon, and T. Bose, "Deep learning for SVD and hybrid beamforming," *IEEE Trans. Wireless Commun.*, vol. 19, no. 10, pp. 6621–6642, Oct. 2020.

[155] J. Mo, B. L. Ng, S. Chang, P. Huang, M. N. Kulkarni, A. Alammouri, J. C. Zhang, J. Lee, and W.-J. Choi, "Beam codebook design for 5g mmWave terminals," *IEEE Access*, vol. 7, pp. 98 387–98 404, Jul. 2019.

[156] C. Sun, Z. Shi, and F. Jiang, "A machine learning approach for beamforming in ultra dense network considering selfish and altruistic strategy," *IEEE Access*, vol. 8, pp. 6304–6315, Jan. 2020.

[157] A. Alkhateeb, S. Alex, P. Varkey, Y. Li, Q. Qu, and D. Tujkovic, "Deep learning coordinated beamforming for highly-mobile millimeter wave systems," *IEEE Access*, vol. 6, pp. 37 328–37 348, Jun. 2018.

[158] M. S. Aljumaily and H. Li, "Machine learning aided hybrid beamforming in massive-MIMO millimeter wave systems," in *IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*. IEEE, Nov. 2019.

[159] E. M. Lizarraga, G. N. Maggio, and A. A. Dowhuszko, "Hybrid beamforming algorithm using reinforcement learning for millimeter wave wireless systems," in *XVIII Workshop on Information Processing and Control (RPIC)*. Salvador da Bahia, Argentina: IEEE, Sep. 2019.

[160] S. Huang, Y. Ye, and M. Xiao, "Hybrid beamforming for millimeter wave multi-user MIMO systems using learning machine," *IEEE Wireless Commun. Lett.*, vol. 9, no. 11, pp. 1914–1918, Nov. 2020.

[161] ——, "Learning based hybrid beamforming design for full-duplex millimeter wave systems," *IEEE Trans. on Cogn. Commun. Netw.*, pp. 1–1, Aug. 2020.

[162] A. M. Elbir and K. V. Mishra, "Robust hybrid beamforming with quantized deep neural networks," in *IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*. Pittsburgh, PA, USA: IEEE, Oct. 2019.

[163] 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 Trans. Veh. Technol.*, vol. 69, no. 10, pp. 11 353–11 368, Oct. 2020.

[164] Y. Long, Z. Chen, J. Fang, and C. Tellambura, "Data-driven-based analog beam selection for hybrid beamforming under mm-wave channels," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 2, pp. 340–352, May 2018.

[165] T. Peken, R. Tandon, and T. Bose, "Unsupervised mmWave beamforming via autoencoders," in *IEEE International Conference on Communications (ICC)*. IEEE, Jun. 2020.

[166] C.-H. Lin, Y.-T. Lee, W.-H. Chung, S.-C. Lin, and T.-S. Lee, "Unsupervised ResNet-inspired beamforming design using deep unfolding technique," in *IEEE Global Communications Conference (GLOBECOM)*. Taipei, Taiwan: IEEE, Dec. 2020.

[167] W.-C. Kao, S.-Q. Zhan, and T.-S. Lee, "AI-aided 3-d beamforming for millimeter wave communications," in *International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*. Ishigaki, Okinawa, Japan: IEEE, Nov. 2018.

[168] L. Li, H. Ren, Q. Cheng, K. Xue, W. Chen, M. Debbah, and Z. Han, "Millimeter-wave networking in the sky: A machine learning and mean field game approach for joint beamforming and beam-steering," *IEEE Trans. Wireless Commun.*, vol. 19, no. 10, pp. 6393–6408, Oct. 2020.

[169] X. Liu, Y. Liu, and Y. Chen, "Machine learning empowered trajectory and passive beamforming design in UAV-RIS wireless networks," *IEEE J. Sel. Areas Commun.*, pp. 1–1, Dec. 2020.

[170] J. Cao, T. Peng, X. Liu, W. Dong, R. Duan, Y. Yuan, W. Wang, and S. Cui, "Resource allocation for ultradense networks with machine-learning-based interference graph construction," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 2137–2151, Mar. 2020.

[171] J. Jang and H. J. Yang, "Deep reinforcement learning-based resource allocation and power control in small cells with limited information exchange," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 13 768–13 783, Nov. 2020.

[172] T. Peng, J. Cao, X. Liu, W. Dong, R. Duan, Y. Yuan, and W. Wang, "A data-driven and load-aware interference management approach for ultra-dense networks," *IEEE Access*, vol. 7, pp. 129 514–129 528, Sep. 2019.

[173] H. P. Tauqir and A. Habib, "Deep learning based beam allocation in switched-beam multiuser massive MIMO systems," in *Second International Conference on Latest trends in Electrical Engineering and Computing Technologies (INTELLECT)*. Karachi, Pakistan: IEEE, Nov. 2019.

[174] H. Huang, Y. Yang, Z. Ding, H. Wang, H. Sari, and F. Adachi, "Deep learning-based sum data rate and energy efficiency optimization for MIMO-NOMA systems," *IEEE Trans. Wireless Commun.*, vol. 19, no. 8, pp. 5373–5388, Aug. 2020.

[175] I. Ahmed and H. Khammari, "Joint machine learning based resource allocation and hybrid beamforming design for massive MIMO systems," in *IEEE Globecom Workshops (GC Wkshps)*. Abu Dhabi, United Arab Emirates: IEEE, Dec. 2018.

[176] S. Han, C. lin I, Z. Xu, and C. Rowell, "Large-scale antenna systems with hybrid analog and digital beamforming for millimeter wave 5G," *IEEE Commun. Mag.*, vol. 53, no. 1, pp. 186–194, Jan. 2015.

[177] D. Kwon, J. Kim, D. A. Mohaisen, and W. Lee, "Self-adaptive power control with deep reinforcement learning for millimeter-wave internet-of-vehicles video caching," *J. Commun. Networks*, vol. 22, no. 4, pp. 326–337, Aug. 2020.

[178] F. Meng, P. Chen, and L. Wu, "Power allocation in multi-user cellular networks with deep q learning approach," in *IEEE International Conference on Communications (ICC)*. Shanghai, China: IEEE, May 2019.

[179] R. Amiri and H. Mehrpouyan, "Self-organizing mm wave networks: A power allocation scheme based on machine learning," in *11th Global Symposium on Millimeter Waves (GSMM)*. Boulder, CO, USA: IEEE, May 2018.

[180] H. Zhang, H. Zhang, K. Long, and G. K. Karagiannidis, "Deep learning based radio resource management in NOMA networks: User association, subchannel and power allocation," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2406–2415, Oct. 2020.

[181] A. Alkhateeb, I. Beltagy, and S. Alex, "MACHINE LEARNING FOR RELIABLE MMWAVE SYSTEMS: BLOCKAGE PREDICTION AND PROACTIVE HANDOFF," in *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. Anaheim, CA, USA: IEEE, Nov. 2018.

[182] J. Khan and L. Jacob, "Learning based CoMP clustering for URLLC in millimeter wave 5g networks with blockages," in *IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*. Goa, India: IEEE, Dec. 2019.

[183] H. Iimori, G. T. F. de Abreu, O. Taghizadeh, R.-A. Stoica, T. Hara, and K. Ishibashi, "Stochastic learning robust beamforming for millimeter-wave systems with path blockage," *IEEE Wireless Commun. Lett.*, vol. 9, no. 9, pp. 1557–1561, Sep. 2020.

[184] C. Jia, H. Gao, N. Chen, and Y. He, "Machine learning empowered beam management for intelligent reflecting surface assisted MmWave networks," *China Commun.*, vol. 17, no. 10, pp. 100–114, Oct. 2020.

[185] E. N. Papasotiriou, J. Kokkoniemi, A.-A. A. Boulogeorgos, J. Lehtomaki, A. Alexiou, and M. Juntti, "A new look to 275 to 400 GHz band: Channel model and performance evaluation," in *IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. Bologna, Italy: IEEE, Sep. 2018.

[186] A.-A. A. Boulogeorgos, S. K. Goudos, and A. Alexiou, "Users association in ultra dense THz networks," in *IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. Kalamata, Greece: IEEE, Jun. 2018.

[187] A.-A. A. Boulogeorgos, E. N. Papasotiriou, and A. Alexiou, "Analytical performance assessment of THz wireless systems," *IEEE Access*, vol. 7, pp. 11 436–11 453, Jan. 2019.

[188] A.-A. A. Boulogeorgos and A. Alexiou, "Error analysis of mixed THz-RF wireless systems," *IEEE Commun. Lett.*, vol. 24, no. 2, pp. 277–281, Feb. 2020.

[189] R. Liu, M. Lee, G. Yu, and G. Y. Li, "User association for millimeter-wave networks: A machine learning approach," *IEEE Trans. Commun.*, vol. 68, no. 7, pp. 4162–4174, Jul. 2020.

[190] Z. Li, M. Chen, K. Wang, C. Pan, N. Huang, and Y. Hu, "Parallel deep reinforcement learning based online user association optimization in heterogeneous networks," in *IEEE International Conference on Communications Workshops (ICC Workshops)*. Dublin, Ireland: IEEE, Jun. 2020.

[191] L. U. Khan, U. Majeed, and C. S. Hong, "Federated learning for cellular networks: Joint user association and resource allocation," in *21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*. Daegu, Korea (South): IEEE, Sep. 2020.

[192] P.-Y. Chou, W.-Y. Chen, C.-Y. Wang, R.-H. Hwang, and W.-T. Chen, "Deep reinforcement learning for MEC streaming with joint user association and resource management," in *IEEE International Conference on Communications (ICC)*. Dublin, Ireland: IEEE, Jun. 2020.

[193] N. Hassan, M. T. Hossan, and H. Tabassum, "User association in coexisting RF and TeraHertz networks in 6g," in *IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*. London, ON, Canada: IEEE, Aug. 2020.

[194] H. S. Ghadikolaei, H. Ghauch, G. Fodor, M. Skoglund, and C. Fischione, "A hybrid model-based and data-driven approach to spectrum sharing in mmWave cellular networks," *IEEE Trans. on Cogn. Commun. Netw.*, vol. 6, no. 4, pp. 1269–1282, Dec. 2020.

[195] H. Zhang, H. Zhang, W. Huangfu, W. Liu, J. Dong, K. Long, and A. Nallanathan, "Distributed DNN based user association and resource optimization in mmWave networks," in *IEEE Global Communications Conference (GLOBECOM)*. Waikoloa, HI, USA: IEEE, Dec. 2019.

[196] M. Elsayed, M. Erol-Kantarci, and H. Yanikomeroglu, "Transfer reinforcement learning for 5g-NR mm-wave networks," *IEEE Trans. Wireless Commun.*, pp. 1–1, 2020.

[197] H. Khan, A. Elgabli, S. Samarakoon, M. Bennis, and C. S. Hong, "Reinforcement learning-based vehicle-cell association algorithm for highly mobile millimeter wave communication," *IEEE Trans. on Cogn. Commun. Netw.*, vol. 5, no. 4, pp. 1073–1085, Dec. 2019.

[198] D. Burghal, N. A. Abbasi, and A. F. Molisch, "A machine learning solution for beam tracking in mmWave systems," in *53rd Asilomar Conference on Signals, Systems, and Computers*. Pacific Grove, CA, USA: IEEE, Nov. 2019.

[199] Y. Guo, Z. Wang, M. Li, and Q. Liu, "Machine learning based mmWave channel tracking in vehicular scenario," in *IEEE International Conference on Communications Workshops (ICC Workshops)*. Shanghai, China: IEEE, May 2019.

[200] Z. Ali, M. Miozzo, L. Giupponi, P. Dini, S. Denic, and S. Vassaki, "Recurrent neural networks for handover management in next-generation self-organized networks," in *IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*. London, United Kingdom: IEEE, Aug. 2020.

[201] L. Yan, H. Ding, L. Zhang, J. Liu, X. Fang, Y. Fang, M. Xiao, and X. Huang, "Machine learning-based handovers for sub-6 GHz and mmWave integrated vehicular networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4873–4885, oct 2019.

[202] V. Yajnanarayana, H. Ryden, and L. Hevizi, "5g handover using reinforcement learning," in *IEEE 3rd 5G World Forum (5GWF)*. Bangalore, India: IEEE, Sep. 2020.

[203] C.-C. Wang, X. Yao, W.-L. Wang, and J. M. Jornet, "Multi-hop deflection routing algorithm based on reinforcement learning for energy-harvesting nanonetworks," *IEEE Trans. Mobile Comput.*, pp. 1–1, 2020.

[204] F. Noorbehbahani and S. Mansoori, "A new semi-supervised method for network traffic classification based on x-means clustering and label propagation," in *8th International Conference on Computer and Knowledge Engineering (ICCKE)*. Mashhad, Iran: IEEE, Oct. 2018.

[205] L. Bin and T. Hao, "An application traffic classification method based on semi-supervised clustering," in *2nd International Symposium on Information Engineering and Electronic Commerce*. Ternopil, Ukraine: IEEE, Jul. 2010.

[206] Y. Wang, Y. Xiang, J. Zhang, and S. Yu, "A novel semi-supervised approach for network traffic clustering," in *5th International Conference on Network and System Security*. Milan, Italy: IEEE, Sep. 2011.

[207] Y. Liu, W. Li, and Y. Li, "Network traffic classification using k-means clustering," in *Second International Multi-Symposiums on Computer and Computational Sciences (IMSCCS 2007)*. Iowa City, IA, USA: IEEE, Aug. 2007.

[208] R. Kumari, Sheetanshu, M. K. Singh, R. Jha, and N. Singh, "Anomaly detection in network traffic using k-mean clustering," in *3rd International Conference on Recent Advances in Information Technology (RAIT)*. Dhanbad, India: IEEE, Mar. 2016.

[209] L. Su, Y. Yao, N. Li, J. Liu, Z. Lu, and B. Liu, "Hierarchical clustering based network traffic data reduction for improving suspicious flow detection," in *17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. New York, NY, USA: IEEE, Aug. 2018.

[210] Y. Wang, Y. Xiang, and J. Zhang, "Network traffic clustering using random forest proximities," in *IEEE International Conference on Communications (ICC)*. Budapest, Hungary: IEEE, Jun. 2013.

[211] T. Auld, A. W. Moore, and S. F. Gull, "Bayesian neural networks for internet traffic classification," *IEEE Trans. Neural Netw.*, vol. 18, no. 1, pp. 223–239, Jan. 2007.

[212] F. Kruber, J. Wurst, and M. Botsch, "An unsupervised random forest clustering technique for automatic traffic scenario categorization," in *21st International Conference on Intelligent Transportation Systems (ITSC)*. Maui, HI, USA: IEEE, Nov. 2018.

[213] H. Singh, "Performance analysis of unsupervised machine learning techniques for network traffic classification," in *Fifth International Conference on Advanced Computing & Communication Technologies*. IEEE, Feb. 2015.

[214] Y. Wang and S.-Z. Yu, "Machine learned real-time traffic classifiers," in *Second International Symposium on Intelligent Information Technology Application*. Shanghai, China: IEEE, Dec. 2008.

[215] C. Zhang, H. Zhang, J. Qiao, D. Yuan, and M. Zhang, "Deep transfer learning for intelligent cellular traffic prediction based on cross-domain big data," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1389–1401, Jun. 2019.

[216] D. Graupe, *Principles of Artificial Neural Networks*. WORLD SCIENTIFIC, jun 2013.

[217] L. Zhang, Y.-C. Liang, and D. Niyato, "6G visions: Mobile ultra-broadband, super internet-of-things, and artificial intelligence," *China Commun.*, vol. 16, no. 8, pp. 1–14, Aug. 2019.

[218] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," *ArXiV*, May 2015.

[219] A. A. Boulogeorgos, S. E. Trevlakis, S. A. Tegos, V. K. Papanikolaou, and G. K. Karagiannidis, "Machine learning in nano-scale biomedical engineering," *IEEE Trans. Mol. Biol. Multi-Scale Commun.*, pp. 1–1, 2020.

[220] R. Gholami and N. Fakhari, "Support vector machine: Principles, parameters, and applications," in *Handbook of Neural Computation*. Elsevier, 2017, pp. 515–535.

[221] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, p. 3371–3408, Dec. 2010.

[222] J. B. Tenenbaum, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, dec 2000.

[223] T. H. D. C. Cormen, C. E. M. Leiserson, R. L. M. Rivest, and C. C. U. Stein, *Introduction to Algorithms*. MIT Press Ltd, 2009. [Online]. Available: https://www.ebook.de/de/product/8474892/thomas_h_dartmouth_college_cormen_charles_e_mit_leiserson_ronald_l_mit_rivest_clifford_columbia_university_stein_introduction_to_algorithms.html

[224] E. Schwartz, B. Merker, E. Wolfson, and A. Shaw, "Applications of computer graphics and image processing to 2d and 3d modeling of the functional architecture of visual cortex," *IEEE Comput. Graph. Appl.*, vol. 8, no. 4, pp. 13–23, Jul. 1988.

[225] M. Balasubramanian, "The isomap algorithm and topological stability," *Science*, vol. 295, no. 5552, pp. 7a–7, Jan. 2002.

[226] S. Alla and S. K. Adari, "Boltzmann machines," in *Beginning Anomaly Detection Using Python-Based Deep Learning*. Apress, 2019, pp. 179–212.

[227] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *arXiv preprint arXiv:1406.2661*, 2014.

[228] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29. Curran Associates, Inc., 2016. [Online]. Available: https://proceedings.neurips.cc/paper/2016/file/cc7e2b878868cbae992d1fb743995d8f-Paper.pdf

[229] B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2042–2062, Jun. 2018.

[230] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, May 1992.

[231] S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvári, *Machine Learning*, vol. 38, no. 3, pp. 287–308, 2000.

[232] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *J. Mach. Learn. Res.*, vol. 10, p. 1633–1685, Dec. 2009.

[233] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, "Application of machine learning in wireless networks: Key techniques and open issues," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3072–3108, 2019.

[234] A.-A. A. Boulogeorgos, E. Yaqub, M. Di Renzo, A. Alexiou, R. Desai, and R. Klinkenberg, "Machine learning: A catalyst for thz wireless networks," *Frontiers in Communications and Networks*, p. 37, 2021.

[235] I.-T. F. G. on Machine Learning for 5G (FG-ML5G), "Architectural framework for machine learning in future networks including imt-2020 (y.3172)," 2019. [Online]. Available: https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.3172-201906-I!!PDF-E&type=items

[236] B. M. S. Teerapittayanon and H. T. Kung, "Distributed deep neural networks over the cloud, the edge and end devices," in *37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 328–339.

[237] B. McMahan and D. Ramage, "Federated learning: Collaborative machine learning without centralized training data," 2017.

[238] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and*

Statistics, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. Fort Lauderdale, FL, USA: PMLR, 20–22 Apr 2017, pp. 1273–1282. [Online]. Available: http://proceedings.mlr.press/v54/mcmahan17a.html

[239] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 1175–1191. [Online]. Available: https://doi.org/10.1145/3133956.3133982