ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DOTTORATO DI RICERCA IN

Ingegneria Elettronica, Telecomunicazioni e Tecnologie dell'Informazione

Ciclo 36

Settore Concorsuale: 09/F2 - Telecomunicazioni

Settore Scientifico Disciplinare: ING-INF/03 - Telecomunicazioni

DEEP LEARNING FOR MASSIVE MULTIPLE ACCESS IN 6G

Presentata da: *Muhammad Usman Khan*

Coordinatore Dottorato

Aldo Romani

Supervisore

Marco Chiani

Co-supervisore

Andrea Giorgetti

Esame finale anno 2024

ALMA MATER STUDIORUM
UNIVERSITY OF BOLOGNA

PH.D. PROGRAMME
ELECTRONICS, TELECOMMUNICATIONS, AND
INFORMATION TECHNOLOGIES ENGINEERING
(ETIT)

SSD ING/INF 03

DEEP LEARNING FOR MASSIVE MULTIPLE ACCESS IN 6G

Ph.D. Thesis

Ph.D. candidate
MUHAMMAD USMAN KHAN

Supervisor
Prof. Ing.
MARCO CHIANI

Ph.D. coordinator
Prof. Ing.
ALDO ROMANI

Co-Supervisor
Dr. Ing.
ANDREA GIORGETTI

XXXVI° CYCLE

To my parents,
unwavering pillars of support,
whose love and encouragement
have fueled my PhD journey.

Typeset using LaTeX

Abstract

In recent years, the number of massive Internet of Things (mIoT) has grown tremendously, giving rise to the term massive machine-type communications (mMTC). Cellular Internet of Things (IoT) is an economical solution for connecting devices wirelessly because it reuses existing cellular infrastructure. 3rd Generation Partnership Project (3GPP) has recognized mMTC as one of the use cases of 6G. However, providing massive access to the IoT devices within the constraints of limited system resources has been an ongoing challenge in cellular networks. On the other hand, Deep learning (DL) has emerged as a powerful method for various applications, such as image processing and natural language processing. More recently, DL has been successfully applied to a wide range of wireless communication tasks. Given that, this thesis aims to design massive multiple access protocols using DL algorithms for both cell-based and cell-free networks.

Firstly, a synchronized uplink grant-free (GF) non-orthogonal multiple access (NOMA) scenario is considered in which only a small number of devices out of several devices are active at a given time. In contrast to orthogonal multiple access, NOMA permits sharing of the same time-frequency resource; therefore, can support a massive number of devices. Since there is no grant procedure, the base station (BS) must identify the active users. Consequently, a DL-based solution, comprised of two novel deep neural network (DNN) architectures, is proposed, one for sparsity estimation and the other for identifying the users.

Secondly, an asynchronous GF random access uplink scenario is considered where users are uniformly distributed around the BS. Asynchronous schemes are important for ultra-low-cost IoT devices, as the signaling over-

head is reduced to the minimum. When a user becomes active, it initiates a virtual frame (VF) comprising of slots with each slot duration equal to the packet length. Each active user transmits multiple replicas in the chosen slot to boost performance. As there is no coordination between the BS and active device, the packet detection tasks need to be performed at the BS. For this task, DNN is designed that predicts if the received symbols are preamble or not.

Finally, a cell-free massive MIMO (CF-mMIMO) scenario is considered in which access points (APs) are arranged in a grid form and users are distributed in an area. CF-mMIMO can improve the quality of the service for the users at the end of the cell and can reduce inter-cell interference. Strategic power control and careful pilot assignment are pivotal in mitigating inter-user interference and enhancing network performance. Taking into account, a DNN is designed for joint pilot and data power and pilot assignment that maximizes the minimum user rate.

Artificial intelligence (AI) integration in 6G network holds significant potential for adaptive and efficient network performance. AI can analyze real-time data to predict and manage traffic congestion. AI can strengthen network security by identifying threats and responding proactively. The result is higher efficiency, improved quality of service, and an enhanced user experience.

List of Tables

List of Figures

Contents

List of Acronyms

1G first-generation
2G second-generation
3G third-generation
4G fourth-generation
5G fifth-generation
AI artificial intelligence
AP access point
AUD active users detection
AUE active users enumeration
AUI active users identification
AUEI active users enumeration and identification
ANN artificial neural network
AR augmented reality
AWGN additive white Gaussian noise
AWGN additive white Gaussian noise
B5G beyond 5G
BPSK binary phase shift keying
BS base station
BIHT block iterative hard thresholding
CDF cumulative distribution function
CDMA code division multiple access
CF-mMIMO cell-free massive MIMO
CNN convolutional neural network
CPU central processing unit
CS compressed sensing

D-AUD Deep AUD
DL deep learning
DLPC deep learning power control
DNN deep neural network
DRL deep reinforcement learning
eLU exponential linear unit
eMBB enhanced mobile broad-band
FDMA frequency division multiple access
FLOP floating point operation
gNB Next Generation NodeB
GF grant-free
GLRT generalized likelihood ratio test
HTC human-type communication
i.i.d. independent and identically distributed
IoT Internet of Things
JPCPA joint power control and pilot assignment
JPDCPA joint pilot and data power control and pilot assignment
KPI key performance indicator
LDS low-density signature
LRT likelihood ratio test
LSTM long short-term memory
LTE Long Term Evolution
mIoT massive Internet of Things
MAC medium access control
ML maximum likelihood
MLP multilayer perceptron
MMSE minimum mean square error
MMA massive multiple access
MTC machine-type communication
mMTC massive machine-type communication
mMIMO massive multiple-input multiple-output
NOMA non-orthogonal multiple access
OMA orthogonal multiple access

OFDM Orthogonal Frequency-Division Multiplexing
OFDMA orthogonal frequency division multiple access
mMTC massive machine-type communications
NOMA non-orthogonal multiple access
PDF probability density function
QPSK quadrature phase-shift keying
RA random access
ReLU rectified linear unit
RNN recurrent neural network
ROC receiver operating characteristics
r.v. random variable
SE spectral efficiency
SIC successive interference cancellation
SINR signal-to-interference-plus-noise ratio
SNR signal-to-noise ratio
TDD time-division duplexing
TDMA time division multiple access
URLLC ultra-reliable and low-latency communications
UMa urban macro
VF virtual frame
VR virtual reality

Chapter 1

Introduction

Recent years have witnessed the explosion of wireless Internet of Things (IoT) communications across several application domains, including home appliances, surveillance cameras, smart grid, smart factories, and intelligent transportation systems [1, 2, 3]. The global count of Internet of Things (IoT) devices is projected to nearly double, rising from 15.1 billion in 2020 to surpass 29 billion by 2030 [4]. Owing to the rapid and widespread adoption of IoT across various application domains, the density of connected objects (in terms of devices per unit area) has recently become so large that the terminology massive machine-type communications (mMTC) has been introduced [5, 6, 7] to refer to wireless networking between the devices that are physically located in the same geographic area. Each of these devices is commonly powered by batteries. The device produces data intermittently, with short periods of activity separated by long periods of inactivity. The active period is typically used for transmitting a single message in the form of a short packet. This communication may occur either between devices or from the device to a remote server through the network [8, 9]. In the uplink, i.e., the wireless link from the devices to the base station (BS) or access point (AP), a massive number of devices contend to transmit short data packets over the radio access network, giving rise to term massive multiple access (MMA) [10, 11]. As devices wake up sporadically, unpredictably, and independently, the receiver lacks a priori knowledge of the number and subset of concur-

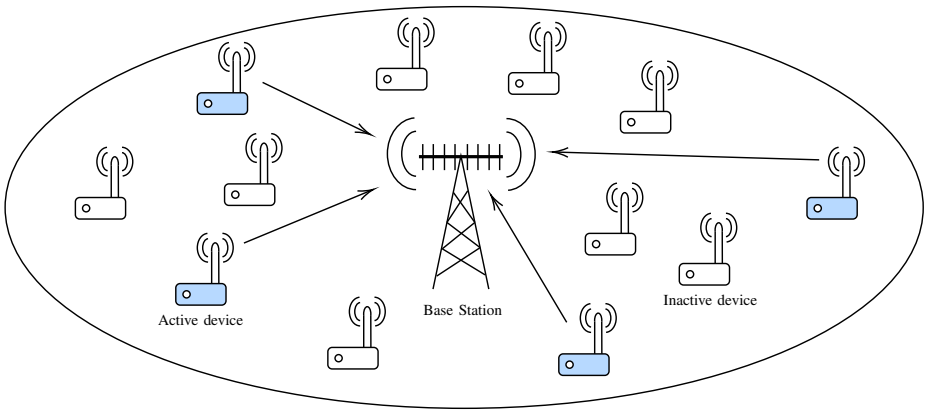


Figure 1.1: A massive multiple access scenario.

rently active devices within a specific time window. A typical MMA scenario is depicted in Fig. 1.1.

Providing multiple access within the constraints of limited system resources has been an ongoing challenge in cellular networks [3]. Over the evolution of cellular technology, various multiple-access techniques have been proposed. In earlier and current wireless networks, orthogonal multiple access (OMA) constitutes the fundamental aspects. For instance, in first-generation (1G) and second-generation (2G) time division multiple access (TDMA) and frequency division multiple access (FDMA) are employed, respectively. While third-generation (3G) systems utilize code division multiple access (CDMA), and fourth-generation (4G) and fifth-generation (5G) systems implement orthogonal frequency division multiple access (OFDMA). To minimize the interference between the adjacent blocks and perform signal detection, these systems divide resource blocks orthogonally across time, frequency, or code domains [12]. Yet, supporting a massive number of devices in beyond 5G (B5G) networks is a non-trivial task and consists of the following challenges.

1. The conventional information-theoretic approach typically concentrates on facilitating a limited number of devices, where the user count is deterministic and pre-known. Extending this traditional multiple-access theory to the more intricate multiple-access scenarios is not a trivial task [13, 14, 15].

2. Grant-based schemes employed in a conventional multiple access scenario allow orthogonal resource allocation due to the small number of devices. However, in MMA applications grant-based access leads to long delays for resource allocation and is inefficient due to control signaling overhead that may even outnumber data [16].

3. Majority of the current IoT networks employ OMA schemes, which simplifies the transceiver design but it often results in a lower overall spectral efficiency (SE). Therefore, implementing OMA schemes in a massive Internet of Things (mIoT) network is inefficient due to underutilization of the radio spectrum [17, 18].

4. To prolong battery life, the IoT devices transmit with a low power which makes signal detection at the BS difficult. As a remedy, IoT devices increase the coverage by employing strategies like re-transmission of the packets and considering low-order modulation, such as binary phase shift keying (BPSK), quadrature phase-shift keying (QPSK) [19]. However, these techniques come at the expense of inefficient utilization of system resources.

5. The broadcast nature of wireless signals introduces the risk of unintended devices intercepting confidential signals, potentially leading to information leakage [20, 21]. Traditionally, security in wireless access relies on cryptography-based encryption techniques. However, due to limited battery and computational capability, the IoT devices cannot employ advanced encryption techniques.

1.1 5G and B5G

The 3rd Generation Partnership Project (3GPP) has classified 5G cellular system services into three distinct categories: enhanced mobile broadband (eMBB), ultra-reliable and low-latency communications (URLLC), and mMTC [22]. Each category has its own set of key performance indicators (KPIs) and requirements to meet the specific needs of different applications.

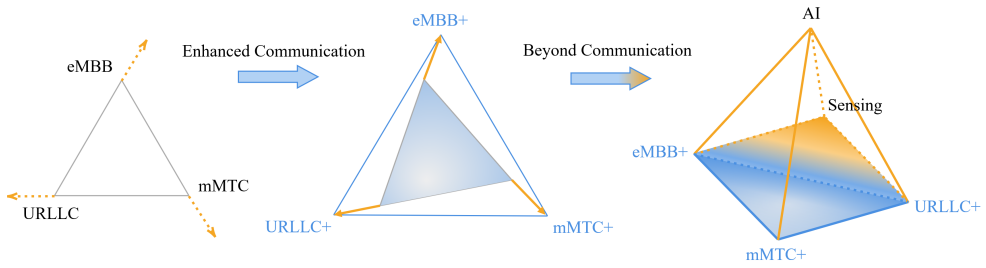


Figure 1.2: The evolution from 5G to B5G.

In 5G mMTC services, the primary focus is on scalability, meaning the network's ability to handle a large number of devices with sporadic traffic, with a minimum target per-device quality of service. Latency and reliability are less critical factors for mMTC applications. URLLC services, on the other hand, demand extremely low latency and high reliability, even when dealing with a smaller number of devices. Scalability is not as critical for URLLC applications. While eMBB specifically targets the provision of exceptional data transfer speeds to support demanding broadband applications like virtual reality (VR) and augmented reality (AR). The 3GPP has established specific requirements for mMTC services in 5G, which are summarized in Table 1.1.

The transition from 5G to 6G will mark a significant shift from the traditional "connected people and things" paradigm to a more advanced "connected intelligence" paradigm. This evolution will involve not only enhancements to the existing communication service classes eMBB, URLLC, and mMTC to eMBB+, URLLC+, and mMTC+, but also introduce a new sensing dimension, with artificial intelligence (AI) playing a pivotal role in unifying all services and applications. In fact, IMT-2030 envisioned expanding on existing use cases like eMBB, URLLC, and mMTC and enabling new ones using the capabilities of AI [23]. Notably, 6G networks will also feature a convergence of services, addressing the need for flexible and adaptable networks that can seamlessly support a wide range of IoT use cases, including industrial applications, vehicle-to-infrastructure communications, and smart city initiatives. This convergence will require a rethinking of the rigid 5G service classification, particularly for applications that demand a balance between

Table 1.1: 5G and B5G IoT key performance indicators (KPIs) and target values [3, 24]

	5G	B5G
Connectivity	$5 \cdot 10^4$ per cell	10^7 per km^2
Battery life	10 years	20 years
Coverage	Ground	Space-air-ground-sea
Latency	1 ms	0.3 ms
Reliability	10^{-4}	10^{-6}

scalability, latency, and reliability. While scalability will remain a primary concern for mMTC services, tighter reliability and latency requirements will emerge to accommodate the growing demands of emerging IoT applications. Additionally, new performance indicators (KPIs) are likely to be introduced in 6G, one of which is the environmental impact of the network, i.e., carbon dioxide (CO_2) footprint. These advancements will pave the way for a more sustainable and intelligent wireless ecosystem that supports a diverse range of applications, enhancing the quality of life for all.

The mIoT networks provide an ideal setting for uncoordinated grant-free (GF) access protocols, where a vast number of devices transmit small data packets at arbitrary times without prior coordination or synchronization. Several examples of GF protocols have been proposed in recent literature, including [25, 26, 27]. These protocols enable machine-type devices to access the channel without any coordination with the BS and other devices. This uncoordinated approach ensures scalability and efficiency for mIoT deployments. The key advantage of GF protocols lies in their simplicity and low overhead on the device side. These approaches simplify device-side implementation but place increased computational demands on the BS. Overall, GF protocols offer a promising solution for enabling mIoT connectivity while maintaining efficient resource utilization and low device-side complexity.

1.2 Deep Learning

Deep learning (DL), a subset of machine learning, has revolutionized the field of AI by enabling machines to learn complex patterns and make intelligent

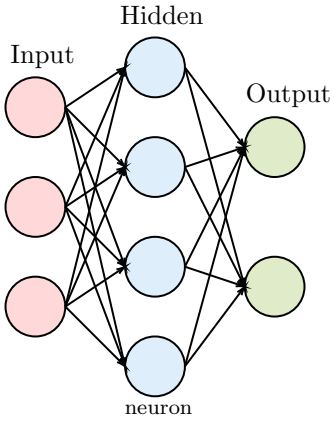


Figure 1.3: Deep Neural Network

decisions, such as speech recognition [28], computer vision [29], and language translation [30]. Depending upon the complexity of the network, it can learn a large number of piecewise smooth functions [31]. DL algorithms are trained on massive datasets. This training process involves adjusting the weights of the network to minimize the error between the network's predictions and the actual data labels. Once trained, deep neural networks (DNNs) can perform complex tasks with low computational cost, as their computations primarily involve multiply–accumulate operations and element-wise nonlinear operations. Among the plethora of DL algorithms, notable ones that have garnered widespread use include DNN, convolutional neural network (CNN), recurrent neural network (RNN), and the groundbreaking Transformer model proposed in [32]. Notably, the Transformer model has revolutionized the field of natural language processing giving rise to large language models, such as Google's BERT and OpenAI's ChatGPT. The subsequent section serves as an introduction to DNN and CNN.

1.2.1 Deep Neural Networks

DNNs, also known as feedforward neural networks or multilayer perceptrons (MLPs), are the foundation of deep learning. These networks aim to approximate a given function f^*. A DNN establishes a mapping $\boldsymbol{y} = f(\boldsymbol{x}; \boldsymbol{\theta})$ and learns the optimal parameter values $\boldsymbol{\theta}$ to achieve the best possible func-

tion approximation [33].

A DNN comprises an input layer, an output layer, and multiple hidden layers as shown in Fig. 1.3. Layers are composed of a node, similar to a neuron, carrying out a sum-of-products operation by multiplying the weights with the inputs and then adding a bias value as follows

$$z = \sum_{i=1}^{m} w_i x_i + b \tag{1.1}$$

where, x, w, and b represent the input, weight, and bias values, respectively. The intermediate result z is then sent through an activation function to introduce non-linearity into the system

$$y = f(z)\,. \tag{1.2}$$

The non-linear operation $f(\cdot)$, known as the activation function, is a crucial component of the DL model. A fully-connected layer composed of many neurons can be conceptualized as [33]

$$\boldsymbol{z} = f(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}) \tag{1.3}$$

where $\boldsymbol{W} \in \mathbb{R}^{\text{out}\times\text{in}}$ represents the weight matrix. The input to the layer is denoted by $\boldsymbol{x} \in \mathbb{R}^{\text{in}\times 1}$, while the bias is represented by $\boldsymbol{b} \in \mathbb{R}^{\text{out}\times 1}$ [33]. The activation function enables the network to learn non-linear relationships between input and output. Some of the most common activation functions are given in Table 1.2.

1.2.2 Convolutional Neural Networks

Convolutional networks are a type of neural network that are designed to efficiently process data that is arranged in a grid-like form, such as images or audio signals [33]. They achieve this by using a mathematical operation

Table 1.2: Common activation functions

	Function
Rectified linear unit (ReLU)	$g(z) = \max(0, z)$
Leaky ReLU	$g(z) = \max(0.01 * z, z)$
Sigmoid	$g(z) = 1/(1 + e^{-z})$
Hyperbolic Tangent	$g(z) = (e^{z} - e^{-z})/(e^{z} + e^{-z})$
Exponential linear unit (eLU)	$g(z) = \begin{cases} \Gamma \cdot (e^{z} - 1), & \text{if } z < 0, \\ z, & \text{otherwise} \end{cases}$

called convolution

$$s_i = (x * w)_i = \sum_{n=-\infty}^{\infty} x_n w_{i-n} \tag{1.4}$$

where x is the input and w is the kernel. In machine learning applications, the data is often represented as a multi-dimensional array; thus, the filters used to extract features are also multi-dimensional arrays. These arrays are known as tensors. The convolution operation is simplified since the values of the tensors being convolved are defined only at a finite set of points. Due to this fact, the convolution can be implemented as a sum over a finite number of array elements instead of an infinite summation as

$$S_{i,j} = (\boldsymbol{I} * \boldsymbol{K}) = \sum_{m} \sum_{n} I_{m,n} K_{i-m,j-n} \tag{1.5}$$

where $S_{i,j}$ represents the element at the ith rows and jth column of $\boldsymbol{S}$. In the convolution process, an odd and square-dimension kernel or filter is traversed across the input. Each convolutional operation involves multiplying the values of the filter with the corresponding values in the input, followed by summing up the results. The outcome of this summation replaces the original value in the input. The kernel is moved to adjacent values, defined by a stride, and convolution operation is applied. The process is iteratively applied to obtain feature maps. The convolution operation leads to a feature map with smaller dimensions than the input. To counteract this, values are

padded to maintain dimension. Based on the input size (N_I), kernel size (F), stride (T), and padding (P), the output size (N_O) can be calculated as

$$N_O = \left\lfloor \frac{N_I - F + 2P}{T} + 1 \right\rfloor . \tag{1.6}$$

The feature maps are passed through an activation function to introduce non-linearity. This step is crucial for enabling the network to learn complex patterns and relationships in the data. The feature maps are modified further using the pooling operation. The pooling operation involves replacing a specific value in the feature map with a derived value based on its magnitude and that of its neighboring values. The max pooling operation is the most popular technique that outputs the maximum value within a rectangular region. Other pooling functions include: computing the average or L_2 norm within the rectangular region [33].

1.2.3 Training DL-algorithm

The objective of the training is to determine the suitable weights of the DL model that minimizes the loss function. The loss, cost, or objective function is dependent on the problem that we are trying to solve. For a binary classification scenario, where the task involves assigning input to one of two categories, or in a multi-label classification context, where the objective is to classify the input into non-mutually exclusive categories, the binary cross-entropy loss function is employed. It is defined as follows

$$\mathcal{J}(q, \hat{q}) = -q \log \hat{q} - (1 - q) \log(1 - \hat{q}) \tag{1.7}$$

where q is the true label and $\hat{q}$ is the predicted value by the DL algorithm. For a multi-class classification problem, which involves categorizing input into three or more mutually exclusive categories, categorical cross-entropy is

Algorithm 1 Gradient Descent Algorithm

1: Initialize the weights $\boldsymbol{W}$ and bias $\boldsymbol{b}$ of the network randomly.
2: **while** not converged **do**
3: Compute Gradient $\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}}$ and $\frac{\partial \mathcal{J}}{\partial \boldsymbol{b}}$
4: Update weights: $\boldsymbol{W} \leftarrow \boldsymbol{W} - \alpha \frac{\partial \mathcal{J}}{\partial \boldsymbol{W}}$ ▷ α is the learning rate.
5: Update bias: $\boldsymbol{b} \leftarrow \boldsymbol{b} - \alpha \frac{\partial \mathcal{J}}{\partial \boldsymbol{b}}$
6: **end while**
Return: $\boldsymbol{W}$, $\boldsymbol{b}$

employed

$$\mathcal{J}(\boldsymbol{p}, \hat{\boldsymbol{p}}) = -\sum_{i=1}^{M} p_i \log \hat{p}_i \tag{1.8}$$

where $\boldsymbol{p}$ is the one-hot encoded vector of length M, indicating the category position with a value of 1 and all other positions with a value of 0. The DL algorithm outputs $\hat{\boldsymbol{p}}$ providing the probability for each class.

To update the weights of the network, it is supplied with training examples, for which the network generates the outputs commonly known as the forward propagation step. In the backpropagation step, the loss is computed on these examples, and the weights are updated using the gradient descent algorithm, which involves computing gradients of the loss with respect to the weight [33]. The gradient descent algorithm is presented in Algorithm 1. Both steps are repeated until the weight of the model converges, i.e., there is no significant change in the weight values.

1.3 Thesis Structure

The rest of the thesis is organized as follows.

Chapter 2: proposes the implementation of a GF non-orthogonal multiple access (NOMA) scheme to provide services to a large number of devices and to reduce the communication overhead in mMTC scenarios. For NOMA with sparse spreading, a DNN-based approach is proposed for active users detection (AUD) called active users enumeration and identification (AUEI).

It consists of two phases: firstly, a DNN is used to estimate the number of active users; then in the second phase, another DNN identifies them. To speed up the training process of the DNNs, a multi-stage transfer learning technique is proposed. The numerical results show a remarkable performance improvement of AUEI in comparison to previously proposed approaches.

Chapter 3: proposes a DL-based solution for detecting preambles in an asynchronous GF random access uplink scenario, assuming multiple antennas at the BS. In GF random access protocols, a large number of devices activate sporadically and transmit short packets, typically containing a preamble (or a pilot sequence), without any resource allocation from the BS. One of the critical tasks to be accomplished by the BS is thus the preamble-based detection of the transmitted packets. The DL-based approach outperforms the classical correlator-based approach.

Chapter 4: introduce a DNN-based approach for joint power control and pilot assignment, aiming to maximize the minimum user rate, commonly referred to as a max-min problem in a cell-free massive MIMO (CF-mMIMO) network. A custom loss function is designed for training the network. Extensive simulations demonstrate that the proposed method outperforms the existing deep learning power control and random pilot assignment strategies. The model versatility and adaptability are assessed by simulating two different scenarios, namely a urban macro (UMa) and an industrial one. Additionally, the advantage of the proposed approach is demonstrated in terms of energy efficiency by evaluating the per-user average pilot and data transmit power.

Chapter 5: concludes with final remarks and considerations.

References

[1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet Things J.*, vol. 1, pp. 22–32, Feb. 2014.

[2] N. Ahmed, D. De, and I. Hussain, "Internet of Things (IoT) for smart

precision agriculture and farming in rural areas," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4890–4899, 2018.

[3] X. Chen, D. W. K. Ng, W. Yu, E. G. Larsson, N. Al-Dhahir, and R. Schober, "Massive access for 5G and beyond," *IEEE J. Sel. Areas Commun.*, vol. 39, pp. 615–637, Mar. 2021.

[4] L. S. Vailshery, "Number of Internet of Things (IoT) connected devices worldwide from 2019 to 2023, with forecasts from 2022 to 2030." Available: https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/, July 2023.

[5] H. Shariatmadari, R. Ratasuk, S. Iraji, A. Laya, T. Taleb, R. J antti, and A. Ghosh, "Machine-type communications: Current status and future perspectives toward 5G systems," *IEEE Commun Mag*, vol. 53, pp. 10–17, Sept. 2015.

[6] C. Bockelmann, N. Pratas, H. Nikopour, K. Au, T. Svensson, C. Stefanovic, P. Popovski, and A. Dekorsy, "Massive machine-type communications in 5G: physical and MAC-layer solutions," *IEEE Commun. Mag.*, vol. 54, pp. 59–65, Sept. 2016.

[7] C. Bockelmann, N. K. Pratas, G. Wunder, S. Saur, M. Navarro, D. Gregoratti, G. Vivier, E. De Carvalho, Y. Ji, C. Stefanović, P. Popovski, Q. Wang, M. Schellmann, E. Kosmatos, P. Demestichas, M. Raceala-Motoc, P. Jung, S. Stanczak, and A. Dekorsy, "Towards massive connectivity support for scalable mMTC communications in 5G networks," *IEEE Access*, vol. 6, pp. 28969–28992, May 2018.

[8] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Computer Networks*, vol. 54, pp. 2787–2805, Oct. 2010.

[9] E. Paolini, L. Valentini, M. U. Khan, F. Babich, M. Comisso, and V. Tralli, *6G Wireless Systems: Enabling Technologies*, vol. 09 of *CNIT Technical Reports*, ch. Massive Multiple Access for 6G, pp. 137–158. Texmat, Roma, Italy, 2022.

[10] S.-Y. Lien, K.-C. Chen, and Y. Lin, "Toward ubiquitous massive accesses in 3GPP machine-to-machine communications," *IEEE Commun. Mag.*, vol. 49, pp. 66–74, Apr. 2011.

[11] Y. Wu, X. Gao, S. Zhou, W. Yang, Y. Polyanskiy, and G. Caire, "Massive access for future wireless communication systems," *IEEE Wireless Commun.*, vol. 27, pp. 148–156, Aug. 2020.

[12] Y. Cai, Z. Qin, F. Cui, G. Y. Li, and J. A. McCann, "Modulation and multiple access for 5G networks," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 629–646, 2018.

[13] Y. Watanabe and K. Kamoi, "A formulation of the channel capacity of multiple-access channel," *IEEE Trans. Inf. Theory*, vol. 55, pp. 2083–2096, May 2009.

[14] H. H. Permuter, T. Weissman, and J. Chen, "Capacity region of the finite-state multiple-access channel with and without feedback," *IEEE Trans. Inf. Theory*, vol. 55, pp. 2455–2477, June 2009.

[15] Y. Polyanskiy, H. V. Poor, and S. Verdu, "Channel coding rate in the finite blocklength regime," *IEEE Trans. Inf. Theory*, vol. 56, pp. 2307–2359, May 2010.

[16] M. Hasan, E. Hossain, and D. Niyato, "Random access for machine-to-machine communication in LTE-advanced networks: issues and approaches," *IEEE Commun. Mag.*, vol. 51, pp. 86–93, June 2013.

[17] N. H. Nguyen, B. Berscheid, and H. H. Nguyen, "Fast-OFDM with index modulation for NB-IoT," *IEEE Commun. Lett.*, vol. 23, pp. 1157–1160, July 2019.

[18] M. Zeng, A. Yadav, O. A. Dobre, G. I. Tsiropoulos, and H. V. Poor, "On the sum rate of MIMO-NOMA and MIMO-OMA systems," *IEEE Wireless Commun. Lett.*, vol. 6, pp. 534–537, Aug. 2017.

[19] J. Xu, J. Yao, L. Wang, Z. Ming, K. Wu, and L. Chen, "Narrowband Internet of Things: Evolutions, technologies, and open issues," *IEEE Internet Things J.*, vol. 5, pp. 1449–1462, June 2018.

[20] H.-M. Wang, Q. Yang, Z. Ding, and H. V. Poor, "Secure short-packet communications for mission-critical IoT applications," *IEEE Trans. Wireless Commun.*, vol. 18, pp. 2565–2578, May 2019.

[21] X. Chen and Y. Zhang, "Mode selection in MU-MIMO downlink networks: A physical-layer security perspective," *IEEE Syst. J.*, vol. 11, pp. 1128–1136, June 2017.

[22] P. Popovski, J. J. Nielsen, C. Stefanovic, E. d. Carvalho, E. Strom, K. F. Trillingsgaard, A.-S. Bana, D. M. Kim, R. Kotaba, J. Park, and R. B. Sorensen, "Wireless access for ultra-reliable low-latency communication: Principles and building blocks," *IEEE Netw.*, vol. 32, pp. 16–23, Mar. 2018.

[23] International Telecommunication Union, "Framework and overall objectives of the future development of IMT for 2030 and beyond," Tech. Rep. M.2160-0, November 2023.

[24] L. Zhang, Y.-C. Liang, and D. Niyato, "6G visions: Mobile ultra-broadband, super internet-of-things, and artificial intelligence," *China Commun.*, vol. 16, pp. 1–14, Aug. 2019.

[25] L. Liu and W. Yu, "Massive connectivity with massive MIMO—part i: Device activity detection and channel estimation," *IEEE Trans. Signal Process.*, vol. 66, pp. 2933–2946, June 2018.

[26] J. H. Sørensen, E. De Carvalho, Č. Stefanović, and P. Popovski, "Coded pilot random access for massive MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 17, pp. 8035–8046, Dec. 2018.

[27] H. Han, Y. Li, W. Zhai, and L. Qian, "A grant-free random access scheme for M2M communication in massive MIMO systems," *IEEE Internet Things J.*, vol. 7, pp. 3602–3613, Apr. 2020.

[28] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, pp. 82–97, Nov. 2012.

[29] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," in *Proc. 31st AAAI Conf. on Artificial Intelligence*, (San Francisco, California, USA), p. 4278–4284, Feb. 2017.

[30] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst. (NeurIPS)*, vol. 2, (Montreal, Canada), pp. 3104–3112, Dec. 2014.

[31] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, pp. 359–366, Mar. 1989.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[33] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

Chapter 2

Enumeration and Identification of Active Users for GF NOMA

2.1 Introduction

In recent years, mMTC has gained a lot of attention due to applications such as smart grid and metering, smart factories, autonomous driving, and public health [1],[2]. In cellular scenarios, mMTC has to provide connectivity between BSs and a very large number of devices [3].

In a conventional multiple-access scenario consisting of a relatively small number of human-type users, the BS assigns radio resources in a coordinated fashion to each user. On the contrary, in mMTC scenario, the resource allocation approach will yield tremendous control signaling overhead which may be large in comparison to the size of the data, making the protocol highly inefficient.

To cope with these limitations, GF-based approaches have been proposed. In GF random access, signalling overhead and latency are reduced as the active devices transmit data without a grant procedure. In contrast to orthogonal multiple access, NOMA permits sharing of the same time-frequency resources, therefore, it can support a massive number of devices in a limited radio spectrum. In the code domain NOMA, each user is assigned a sparse spreading sequence, known to the BS. The length of the spreading sequences

is kept low to efficiently utilize the radio spectrum. Due to a large number of users, the sequences are non-orthogonal. Despite this, decoding is possible in mMTC because the number of active devices at any given time is a small fraction of the total number of devices. Since there is no previous coordination or grant procedure, the BS must identify the active users to be able to decode them by their respective spreading sequences. Thus, the first crucial step is active user detection. Due to the sparseness of the users' activation pattern, compressed sensing (CS)-based techniques have been proposed in NOMA to identify them [4, 5, 6]. In [7], the authors proposed a low-complexity algorithm for active users detection using pilot sequences with a massive number of antennas at the BS. A receiver which works independently of parameters such as signal-to-noise ratio (SNR) and user activity ratio in a NOMA setting is proposed in [8]. However, it has been shown that the performance of CS-based detection schemes degrade considerably as the sparsity level (number of active devices) increases [6]. Moreover, CS-based algorithms fail to consider time constraint [9]. For instance, the number of iterations of block iterative hard thresholding (BIHT) presented in [10] depends on the sparsity level, i.e., the algorithm will take more time to converge as the sparsity level increases.

To overcome some of these issues, DL methods could be used instead of CS. Indeed, it has been shown that a DNN can learn a large number of piecewise smooth functions [11], and since then DL methods have been successfully proposed in various fields, such as speech recognition [12], computer vision [13], and language translation [14]. DL techniques find several applications in the wireless communication domain as well [9, 15, 16, 17]. In contrast to CS solutions, DL requires a large amount of data for training, but once the algorithm is trained the complexity becomes low. Indeed, in the operational mode, DL involves multiply-accumulate and element-wise nonlinear evaluations, which are far less computationally expensive than the CS-based techniques [9, 18]. Thus, some studies have been carried out to identify active users in NOMA scenarios using DL algorithms [3, 18]. Specifically, a RNN has been proposed for both AUD and channel estimation considering a NOMA scenario with sparse spreading sequences in [3]. Another approach

that deals with AUD using a DNN architecture with residual connections has been proposed in [18]. The existing DNN-based algorithms for AUD can be divided into three categories: i) assuming the number of active users is perfectly known [19]; ii) without preliminary estimation of the number of active users [3, 20, 21]; iii) estimating this number through thresholding-based algorithms [18]. Assuming perfect knowledge of the number of active users is unrealistic. Also, sparsity estimation by thresholding-based algorithms is not an easy task, as the threshold level would depend on several system parameters in an unknown way, leading to poor results when compared with the other categories [3].

In this chapter, it is assumed that at the beginning of the transmission the BS is unaware of the number of active users. The main contributions of this chapter are summarized as follows:

- A new solution to active users detection is proposed, which comprises of two novel DNN architectures, one for sparsity estimation called active users enumeration (AUE), and the other one for identifying the active users called active users identification (AUI);
- The proposed solution is compared with previous approaches to assess the performance improvement;
- False alarm rate is reported to completely characterize the performance of the proposed model as it has never been analyzed in the literature on AUD to the best of my knowledge;
- A multi-stage transfer learning approach is investigated to reduce the training time of the DNNs.

The rest of the chapter is organized as follows. The system model along with the concept of spreading sequences and multiple measurements is reported in Section 2.2. In Section 2.3, the DNN architecture for AUD and sparsity estimation is explained. Section 2.4 contains the simulation settings and results. Section 2.5 concludes the study.

Boldface uppercase, boldface lowercase, and lowercase letters are used to denote matrices, vectors, and scalars respectively. Also, $\text{abs}(v)$ and $\arg(v)$

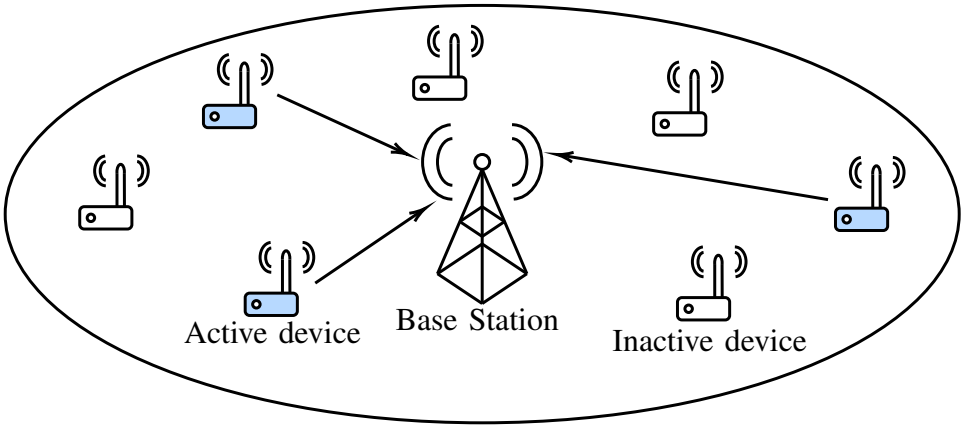


Figure 2.1: The depiction of GF-NOMA uplink communication scenario where only a few devices are active. The active devices are highlighted in blue color.

denote the magnitude and argument of the complex number v, respectively. The operator diag($\boldsymbol{v}$) outputs a diagonal matrix with entries of the vector $\boldsymbol{v}$ along the diagonal, and $\| \, . \, \|_p$ represents the p-norm.

2.2 System Model

A synchronized uplink GF NOMA system scenario is considered as in [3, 18], in which N machine-type devices can transmit to the BS (see Fig. 2.1), both machine-type devices and BS are equipped with a single antenna, and each device is assigned a preconfigured sequence (or codeword), known by the BS. A small number of devices K are active at a given time, with $1 \leq K \leq K_{\max}$ and $K_{\max} \ll N$, where $K_{\max}$ is a system parameter representing the maximum number of active users under consideration. The symbols generated by each active device are spread with its device-specific non-orthogonal codeword. Then, the samples are transmitted through parallel frequency-flat channels, e.g., by Orthogonal Frequency-Division Multiplexing (OFDM).

For instance, if the ith device wants to send at time t a symbol $s_i^{(t)} \in \mathbb{C}$, it encodes it into $\boldsymbol{q}_i^{(t)} = \boldsymbol{c}_i^{(t)} s_i^{(t)} \in \mathbb{C}^S$, where $\boldsymbol{c}_i^{(t)} = [c_{i,1}^{(t)}, \cdots, c_{i,S}^{(t)}]^T \in \mathbb{C}^S$ is the codeword of length S associated with the ith device. The S elements of $\boldsymbol{q}_i^{(t)}$ are sent over S parallel additive white Gaussian noise (AWGN) channels with gains $\boldsymbol{h}_i^{(t)} = [h_{i,1}^{(t)}, h_{i,2}^{(t)}, \ldots, h_{i,S}^{(t)}]^T$. Overall, the received vector at the BS

at time t can be written as

$$\boldsymbol{y}^{(t)} = \sum_{i=1}^{N} \delta_i \, \mathrm{diag}(\boldsymbol{c}_i^{(t)}) \boldsymbol{h}_i^{(t)} s_i^{(t)} + \boldsymbol{n}^{(t)} \tag{2.1}$$

where $\boldsymbol{n}$ denotes the complex Gaussian noise vector $\boldsymbol{n} \sim \mathcal{CN}(0, \sigma_n^2 \boldsymbol{I})$. The device indicator $\delta_i \in \{0, 1\}$ indicates the activity status of the ith device, with $\delta_i = 0/1$ for inactive/active devices, respectively.

To minimize the interuser interference, low-density signature (LDS) codewords are employed, i.e., each codeword has only a small number n_S of non-zero values [22]. Similarly to [3, 18, 22], n_S positions are randomly picked to generate a codeword, and then the non-zero entries are generated as independent and identically distributed (i.i.d.) according to a complex Gaussian distribution $\mathcal{CN}(0, \sigma_w^2)$.

Assuming the devices transmit N_d consecutive symbols, the received measurements can be arranged in a vector as follows

$$\tilde{\boldsymbol{y}} = \begin{bmatrix} \boldsymbol{\Phi}_1 \cdots \boldsymbol{\Phi}_N \end{bmatrix} \begin{bmatrix} \delta_1 \boldsymbol{x}_1 \\ \vdots \\ \delta_N \boldsymbol{x}_N \end{bmatrix} + \begin{bmatrix} \boldsymbol{n}^{(1)} \\ \vdots \\ \boldsymbol{n}^{(N_d)} \end{bmatrix} \tag{2.2}$$

where $\boldsymbol{\Phi}_i = \mathrm{diag}[(\boldsymbol{c}_i^{(1)})^T \cdots (\boldsymbol{c}_i^{(N_d)})^T]$ are the codebook matrices of dimension $(N_d \cdot S) \cdot (N_d \cdot S)$, $\boldsymbol{c}_i^{(t)}$ are the randomly generated codewords, and $\boldsymbol{x}_i = [(s_i^{(1)} \boldsymbol{h}_i^{(1)})^T \cdots (s_i^{(N_d)} \boldsymbol{h}_i^{(N_d)})^T]^T \in \mathbb{C}^{N_d \cdot S \cdot 1}$ denote the composite channel vectors and data symbols. For example, consider the case where only the 2nd and 4th users are active. Then, (2.2) reduces to

$$\tilde{\boldsymbol{y}} = \begin{bmatrix} \boldsymbol{\Phi}_2 \boldsymbol{\Phi}_4 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_2 \\ \boldsymbol{x}_4 \end{bmatrix} + \begin{bmatrix} \boldsymbol{n}^{(1)} \\ \vdots \\ \boldsymbol{n}^{(N_d)} \end{bmatrix} . \tag{2.3}$$

Assuming a maximum number of active users $K_{\max}$, AUD can be formu-

lated as the support identification problem

$$\hat{\Omega} = \underset{\Omega,\, |\Omega| \leq K_{\max}}{\arg\min} \parallel \tilde{\boldsymbol{y}} - \boldsymbol{\Phi}_{\Omega} \boldsymbol{x}_{\Omega} \parallel_2 \tag{2.4}$$

where Ω are the subsets of $\{1, 2, \ldots, N\}$, and the $\hat{\Omega}$ contains the indexes of the estimated active users.

One possible approach to solve (2.4) consists of applying CS-based techniques, which however could be challenging for real-time applications [6, 10, 23, 24, 25]. On the contrary, once a DNN is trained, estimating $\hat{\Omega}$ will be less computationally expensive with respect to CS-based approaches. In the next section, the proposed DL approach is discussed.

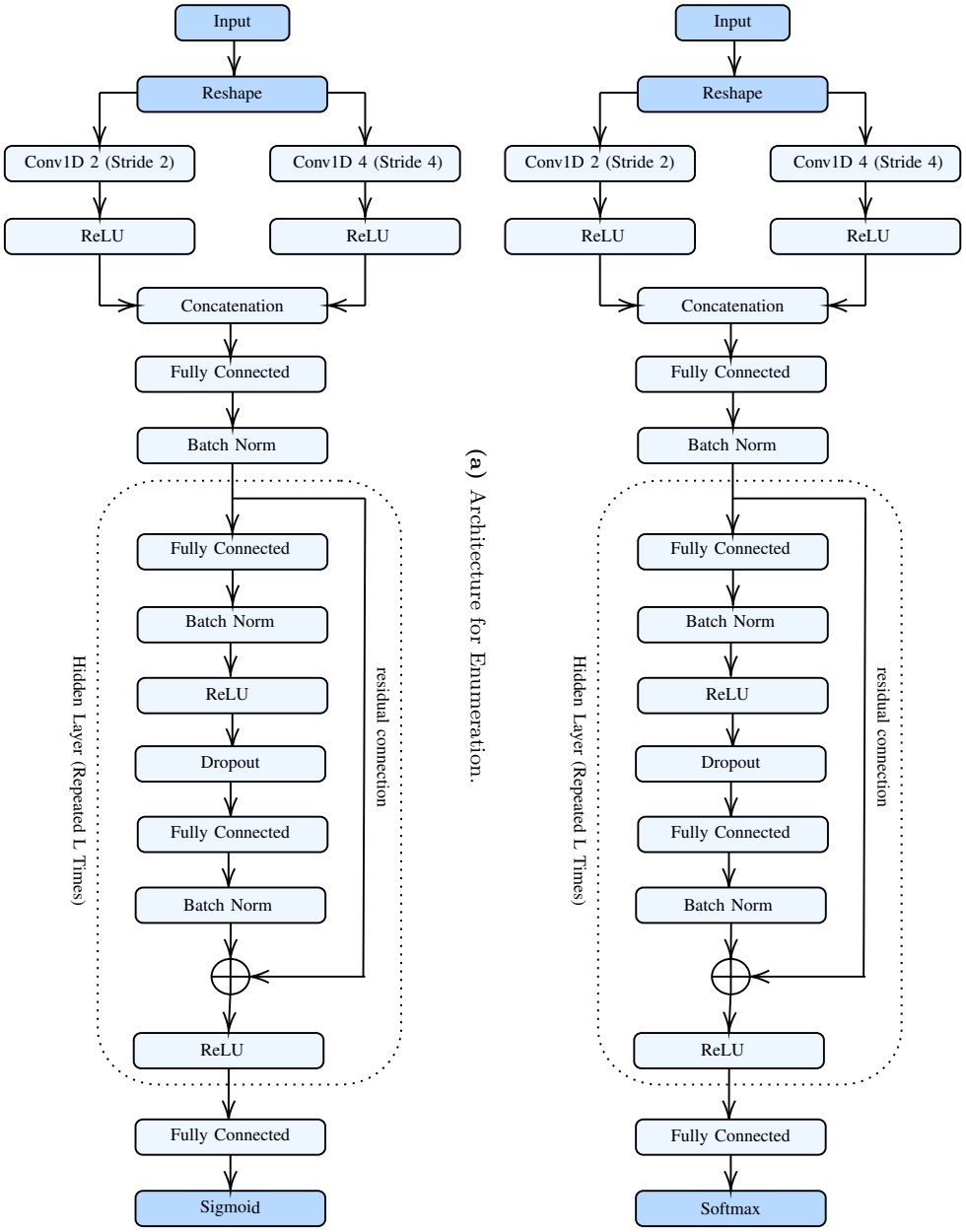
2.3 Deep Learning-based AUD

Different approaches based on DNNs have been proposed in the literature for AUD, all employing thresholding-based algorithms for determining the number of active users [18, 26]. Here a different solution composed of two separate DNN architectures is presented, one for active users enumeration and the other for active users identification. To the best of my knowledge, this is the first work which utilizes a DNN-based architecture for enumerating the active users in a NOMA scenario. The task of the AUE network is to output the number of active users, while a set of AUI networks, each trained for a different sparsity level, identifies the active users. More precisely, the former learns the mapping between the received vector $\tilde{\boldsymbol{y}}$ and the estimated number of active users $\hat{K}$, while the latter learns the mapping between the received vector $\tilde{\boldsymbol{y}}$ and $\hat{\Omega}$ for the cardinality $|\hat{\Omega}| = \hat{K}$.

The networks provide the result as follows

$$\begin{aligned} \hat{K} &= f(\tilde{\boldsymbol{y}}; \Psi) \\ \hat{\Omega} &= g_{\hat{K}}(\tilde{\boldsymbol{y}}; \Theta_{\hat{K}}) \end{aligned} \tag{2.5}$$

where Ψ and Θ_k are the sets of weights and biases associated with the enumeration DNN and the identification DNN for sparsity $k \in \{1, 2, \ldots, K_{\max}\}$,



(a) Architecture for Enumeration. **(b)** Architecture for Identification.

Figure 2.2: Architecture of the proposed DNNs.

respectively.

2.3.1 DNNs Architecture

The received vector obtained through (2.2) has complex elements. To work with common DNNs, which assume real numbers as input, the complex elements are split into the magnitude and phase parts. More precisely, for a received vector $\tilde{\boldsymbol{y}} = [y_1, \ldots, y_m]^T \in \mathbb{C}^m$ then the input to the DNNs would be

$$\hat{\boldsymbol{y}} = [\text{abs}(y_1), \arg(y_1), \ldots, \text{abs}(y_m), \arg(y_m)]^T .$$

Fig. 2.2 shows the architecture for the AUE and AUI. Both DNNs consist of convolutional layers, fully-connected layers, batch normalization layers, dropout layer, and activation layers. The difference between the AUE and AUI is in the output layer, which is a softmax for the AUE, and a sigmoid layer for the AUI. These output layers are described precisely below. The input to the DNNs $\hat{\boldsymbol{y}}$ is reshaped to a 2-D feature map $(N_d, 2S)$ using the reshape layer, where the first dimension corresponds to the channels analogous to the channels in a colour image. The 1-D convolution operation is performed using filters of size 2 and 4, with a stride equal to the filter size. Here, valid convolution is performed, i.e., the output is only considered when the filter is fully contained in the feature map and the output feature map is reduced according to the input feature map, filter size and stride [27]. The output feature maps from the convolutional layers are passed through a ReLU activation function. The output from the activation function is reduced to 1-D and then concatenated through the concatenation layer. The rationale behind using convolutional layers is to reduce the computational complexity and to extract the features shared among N_d multiple measurements. The fully-connected layers, defined in (1.3), consist of α neurons, except for the last one. In fact, the last fully-connected layer dimension must agree with the output layer dimension, so it contains $K_{\max}$ and N neurons for the enumeration and the identification DNNs, respectively. The fully-connected layers employ a linear activation function.

Instead of a single training example, DNNs are trained on a batch of training examples called a mini-batch, $\boldsymbol{B} = [\boldsymbol{a}^{(1)}, \ldots, \boldsymbol{a}^{(n)}]$. The batch normalization layer normalizes the mini-batch $\boldsymbol{B}$ to zero mean and unit variance and then scales it using the trainable parameters γ and β

$$z_i^{(k)} = \frac{\gamma(a_i^{(k)} - \mu_i)}{\sqrt{\sigma_i^2}} + \beta \qquad i = 1, \ldots, \alpha \tag{2.6}$$

where μ_i and σ_i^2 are estimates of the mean and variance of the ith element of the vector, respectively, obtained by moving average [28]. The activation layers are introduced so that the DNNs can learn non-linear functions. ReLU is a common choice as an activation function in the hidden layers for numerous DNN architectures [27, 29] and ReLU can be mathematically described as

$$a = \max(0, z) \tag{2.7}$$

where the operation is to be considered element-wise.

A DNN consists of many hidden layers, and it becomes challenging to train due to the vanishing/exploding gradient problem [30]. Therefore, the residual connections scheme is adopted, proposed in [31]. Residual connections directly pass information from the previous layer to the next layer as depicted in Fig. 2.2.

The output layer of the AUE has dimension equal to the maximum sparsity level $K_{\max}$. The softmax layer takes as input a vector and normalizes it to a probability distribution

$$\hat{p}_i = \frac{e^{z_i}}{\sum_{j=1}^{K_{max}} e^{z_j}} \tag{2.8}$$

where z_i and z_j are the ith and jth element of $\boldsymbol{z}$, while $\hat{p}_i$ is the ith element of $\hat{\boldsymbol{p}}$. The final estimate is

$$\hat{K} = \arg\max_i \hat{p}_i \,. \tag{2.9}$$

For active user identification $K_{\max}$ neural networks $g_1(\cdot), g_2(\cdot), \ldots, g_{K_{\max}}(\cdot)$ are used, as defined in (2.5). The architecture of all the $K_{\max}$ AUI networks remains the same as shown in Fig. 2.2b, only the dataset used for training each AUI is different. For instance, for training $g_k(\cdot)$, a dataset comprising of k active users is considered. Here, a dropout layer is employed to avoid overfitting of the model on the training dataset. In this layer, during the training phase, a fraction of the input and output connections from the neurons are dropped [28]. To identify active users, a sigmoid layer with N outputs is adopted as output, one per user. Each output is calculated as

$$\hat{q}_i = \frac{1}{1 + e^{-z_i}} \qquad i = 1, \ldots, N \tag{2.10}$$

where $\hat{q}_i$ represents the likelihood of user i being active. In previous approaches, a comparison with a threshold was proposed to decide which users were active, but these methods suffer from difficulties in finding a suitable threshold value. In this approach, summarized in Algorithm 2, the foundation is built upon $\hat{K}$ derived from the AUE network. Subsequently, AUI network is employed which is trained for $\hat{K}$ active users. With this network, using $\tilde{\boldsymbol{y}}$ as input, the $\hat{K}$ users with the largest likelihoods are considered as active

$$\hat{\Omega} = \underset{\Omega, |\Omega| = \hat{K}}{\arg\max} \sum_{i \in \Omega} \hat{q}_i \,. \tag{2.11}$$

2.3.2 DNNs Training

Sparsity estimation can be seen as a multi-class classification, in which the input $\tilde{\boldsymbol{y}}$ is categorized in one of the categories ranging from 1 to $K_{\max}$. To this aim, a categorical cross-entropy loss is employed. The true label vector is indicated as $\boldsymbol{p} = [p_1, p_2, \ldots, p_{K_{\max}}]$. If the number of active users is k, it will be $p_k = 1$ and $p_j = 0 \, \forall j \neq k$. For instance, if the number of active users is 2, then $\boldsymbol{p} = [0, 1, 0, \ldots, 0]$. The categorical cross-entropy $\mathcal{J}_S(\boldsymbol{p}, \hat{\boldsymbol{p}})$ loss is

Algorithm 2 Deep learning-based AUEI

Input: $\tilde{\boldsymbol{y}}$, $K_{\max}$

Output: $\hat{\Omega}$

1: Pass $\tilde{\boldsymbol{y}}$ through the enumeration DNN to obtain $\hat{\boldsymbol{p}}$

2: $\hat{K} \leftarrow \underset{i\in\{1,\ldots,K_{\max}\}}{\arg\max}\ \hat{p}_i$

3: $\hat{\Omega} \leftarrow g_{\hat{K}}(\tilde{\boldsymbol{y}}, \boldsymbol{\Theta}_{\hat{K}})$

Return: $\hat{\Omega}$

defined as

$$\mathcal{J}_S(\boldsymbol{p}, \hat{\boldsymbol{p}}) = -\sum_{i=1}^{K_{\max}} p_i \log \hat{p}_i = -\log \hat{p}_K \,. \tag{2.12}$$

User activity identification can be seen as a multi-label classification problem, in which $\hat{K}$ out of N users are selected. To this aim, a binary cross-entropy loss is employed. The true label vector is indicated as $\boldsymbol{q} = [q_1, q_2, \ldots, q_N]$ where each element represents the user as active ($q_i = 1$) or inactive ($q_i = 0$). For instance, if $\Omega = \{2, 4\}$ then $\boldsymbol{q} = [0, 1, 0, 1, \ldots, 0]$. The binary cross-entropy loss is defined as

$$\mathcal{J}_A(\boldsymbol{q}, \hat{\boldsymbol{q}}) = -\sum_{i=1}^{N} (q_i \log \hat{q}_i + (1 - q_i) \log(1 - \hat{q}_i)) \,. \tag{2.13}$$

In order to determine the parameters Ψ and Θ_k in (2.5), the loss functions $\mathcal{J}_S(\boldsymbol{p}, \hat{\boldsymbol{p}})$ and $\mathcal{J}_A(\boldsymbol{q}, \hat{\boldsymbol{q}})$ are needed to be minimized for enumeration and identification tasks, respectively. For that purpose, the well-known Adam optimizer is employed [32].

With the proposed approach, $K_{\max}$ AUI networks have to be trained which is a time and computationally expensive task. To counter that, a multi-stage transfer learning technique is proposed. To train the AUI network $g_k(\cdot)$ in (2.5) for $k \geq 2$ through this technique, start from the trained weights of $g_{k-1}(\cdot)$. More precisely, the weights of $g_1(\cdot)$ are initialized according to [33]. Then, $g_1(\cdot)$ is trained until the network converges, i.e., there is no significant change in the network weights. Instead of initializing the weights of $g_2(\cdot)$

randomly, they are initialized with the trained weights of $g_1(\cdot)$; this way, $g_2(\cdot)$ leverages the information learnt by $g_1(\cdot)$ and converges faster than its randomly initialized counterpart. In general, the weights of $g_k(\cdot)$ are hence initialized through the trained weights of $g_{k-1}(\cdot)$, for $k = 2, \ldots, K_{\max}$.

2.3.3 Computational Complexity

In this subsection, the computational complexity of the AUEI is presented in terms of floating point operations (FLOPs). The addition, subtraction, and multiplication computation are assumed as a single FLOP whereas division and exponential computation are considered as 4 and 8 FLOPs, as in [34]. The FLOPs of the convolutional layers are given by

$$C_{\text{conv}_2} = 2 \cdot N_{\text{conv}_2} \cdot F_{\text{conv}_2} \cdot N_d \cdot out_{\text{conv}_2}$$
$$C_{\text{conv}_4} = 2 \cdot N_{\text{conv}_4} \cdot F_{\text{conv}_4} \cdot N_d \cdot out_{\text{conv}_4}$$

where N_{conv_*}, F_{conv_*} and out_{conv_*} represent the number of convolution filters, size of the filter and output shape, respectively. The output of the convolutional layers is fed into a ReLU, having computational complexity

$$C_{\text{ReLU}_2} = N_{\text{conv}_2} \cdot out_{\text{conv}_2}$$
$$C_{\text{ReLU}_4} = N_{\text{conv}_4} \cdot out_{\text{conv}_4}\,.$$

The number of FLOPs in a fully-connected layer (1.3) is dictated by the input (in) and output (out) size

$$C_{\text{FC}} = in \cdot out + (in - 1) \cdot out + out\,.$$

The number of multiplication and addition operations in $\boldsymbol{W}\boldsymbol{a}$ is given by the term ($in \cdot out$) and ($in \cdot out - out$), respectively. The last term (out) is the number of addition operations due to the bias $\boldsymbol{b}$. The computational complexity of the fully-connected layer simplifies to

$$C_{\text{FC}} = 2 \cdot in \cdot out\,.$$

Consequently, the FLOPs of the input fully-connected layer can be defined as

$$C_{\mathrm{FC}_{in}} = 2\alpha \cdot [N_{\mathrm{conv}_2} \cdot out_{\mathrm{conv}_2} + N_{\mathrm{conv}_4} \cdot out_{\mathrm{conv}_4}] \,.$$

The batch normalization (2.6) involves four operations, therefore, the complexity of the input batch normalization layer can be expressed as

$$C_{\mathrm{BN}_{in}} = 4\alpha \,.$$

The hidden layer is composed of two fully-connected layers, two batch normalization layers, two activation functions, one dropout layer and one residual connection. The dropout layer and residual connection are elementwise multiplication and addition operations; therefore, each will contribute α complexity to the algorithm. The overall complexity of L hidden layers is given by

$$\begin{aligned} C_{\mathrm{hidden}} &= (2\alpha^2 + 2\alpha^2 + 4\alpha + 4\alpha + 2\alpha + \alpha + \alpha)L \\ &= 4\alpha^2 L + 12\alpha L \,. \end{aligned}$$

The computational cost incurred at the output fully-connected layer of AUE and AUI is

$$C^{\mathrm{AUE}}_{\mathrm{FC}_{out}} = 2\alpha K_{\max}$$

and

$$C^{\mathrm{AUI}}_{\mathrm{FC}_{out}} = 2\alpha N$$

respectively. The softmax layer (2.8) in AUE invokes $K_{\max}$ exponential, $K_{\max}$ divisions and $K_{\max} - 1$ additions operations

$$C_{\mathrm{softmax}} = 13K_{\max} - 1 \,.$$

Similarly, the number of FLOPs in a sigmoid layer (2.10) is:

$$C_{\text{sigmoid}} = 13N\,.$$

According to [35], finding the largest probabilities in (2.9) and (2.11) yields the following complexity

$$C_{\max}^{\text{AUE}} = K_{\max} - 1$$

and

$$C_{\max}^{\text{AUI}} = KN - \frac{K(K+1)}{2}$$

respectively. The overall computational complexity of the AUE is described below

$$\begin{aligned} C_{\text{AUE}} &= C_{\text{conv}_2} + C_{\text{conv}_4} + C_{\text{ReLU}_2} + C_{\text{ReLU}_4} + C_{\text{FC}_{in}} \\ &\quad + C_{\text{BN}_{in}} + C_{\text{hidden}} + C_{\text{FC}_{out}}^{\text{AUE}} + C_{\text{softmax}} + C_{\max}^{\text{AUE}} \\ &= 2 \cdot (N_{\text{conv}_2} \cdot out_{\text{conv}_2})(F_{\text{conv}_2} \cdot N_d + \alpha) \\ &\quad + 2 \cdot (N_{\text{conv}_4} \cdot out_{\text{conv}_4})(F_{\text{conv}_4} \cdot N_d + \alpha) \\ &\quad + 4\alpha + 4\alpha^2 L + 12\alpha L + 2\alpha K_{\max} + 14K_{\max} - 2\,. \end{aligned} \tag{2.14}$$

Likewise, the computational complexity of AUI is given as

$$\begin{aligned} C_{\text{AUI}} &= C_{\text{conv}_2} + C_{\text{conv}_4} + C_{\text{ReLU}_2} + C_{\text{ReLU}_4} + C_{\text{FC}_{in}} \\ &\quad + C_{\text{BN}_{in}} + C_{\text{hidden}} + C_{\text{FC}_{out}}^{\text{AUI}} + C_{\text{sigmoid}} + C_{\max}^{\text{AUI}} \\ &= 2 \cdot (N_{\text{conv}_2} \cdot out_{\text{conv}_2})(F_{\text{conv}_2} \cdot N_d + \alpha) \\ &\quad + 2 \cdot (N_{\text{conv}_4} \cdot out_{\text{conv}_4})(F_{\text{conv}_4} \cdot N_d + \alpha) \\ &\quad + 4\alpha + 4\alpha^2 L + 12\alpha L + 2\alpha N + 13N + KN \\ &\quad - \frac{K(K+1)}{2}\,. \end{aligned} \tag{2.15}$$

Finally, the complexity of the AUEI is

$$C_{\text{AUEI}} = C_{\text{AUE}} + C_{\text{AUI}} \,. \tag{2.16}$$

In the next section, this complexity is compared with that of the algorithm presented in [18].

2.4 Implementation and Results

2.4.1 Simulation Setup

The samples are generated according to the system model described by (2.2) for training and testing the DNNs networks. To compare the proposed approach with other algorithms from the literature, the same simulation parameters are chosen as in [18], namely a total number of users $N = 100$, a maximum number of active users $K_{\max} = 8$, spreading codewords with sparsity $n_S = 2$ and length $S = 10$, and $N_d = 7$ successive measurements. The case of zero active users can be handled with less computationally expensive spectrum sensing techniques or machine learning algorithms, as described, e.g., in [36, 37, 38]. The non-zero values of the LDS codewords are generated from the distribution $\mathcal{CN}(0, \sigma_w^2)$ with $\sigma_w^2 = 1$. A Rayleigh fading channel model with perfect power control is employed, so that $h_{i,j} \sim \mathcal{CN}(0, 1)$ are i.i.d. complex Gaussian. Note that owing to perfect power control, the distance of the devices from the BS does not contribute towards the received vector. The data symbols s_i are unit energy QPSK, so that the SNR is defined as $\text{SNR} = 1/\sigma_n^2$.

For the AUE network dataset, the number of active users in each sample varies from 1 to $K_{\max}$. For the training, $13.5 \cdot 10^6$ samples are generated. The dataset generation for the kth AUI network $g_k(\cdot)$ involves randomly activating k users from a total of N. For training and testing, $9 \cdot 10^6$ and 10^6 samples are generated per AUI network.

The architecture of both the AUE and AUI DNNs consists of $L = 2$ hidden layers. The convolutional layers consist of 64 filters. Except the

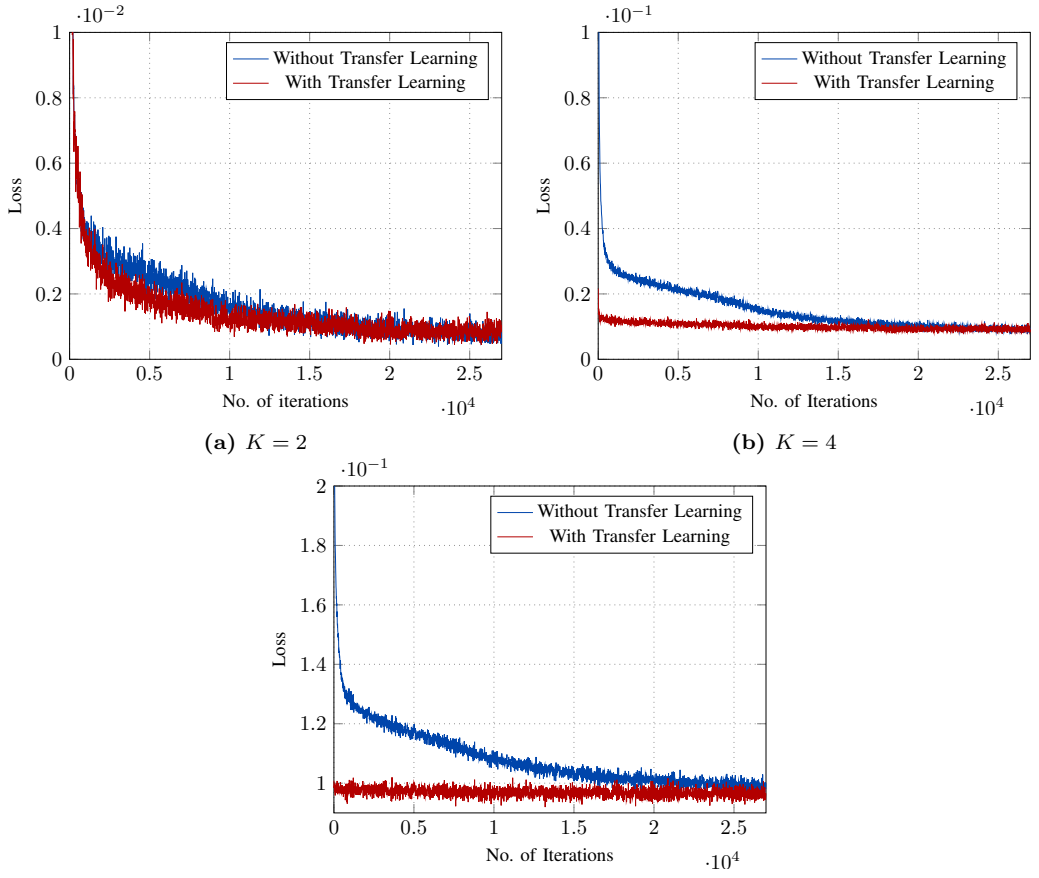
last fully-connected layer, each fully-connected layers consists of $\alpha = 1000$ neurons. In case of AUE and AUI, the last fully connected layer contains $K_{\max} = 8$ and $N = 100$ neurons, respectively.

The sparsity estimation DNN is trained for 10 epochs. Regarding the AUI networks, to minimize the training time, the multi-stage transfer learning approach is adopted. Hence, the first AUI network, $g_1(\cdot)$, is trained for 10 epochs with He initialization [33], while for the $g_k(\cdot)$ network the weights are initialized from the trained weights of $g_{k-1}(\cdot)$. The Adam optimizer is adopted for learning the weights in both DNN networks. For the optimizer, the following configuration is considered: learning rate = 0.001, $\beta_1 = 0.9$, and $\beta_2 = 0.999$. In the training phase, a mini-batch of size $|\boldsymbol{B}| = 1000$ is considered. The drop out rate is set to 0.1.

For the implementation of the deep learning algorithms, Keras deep learning framework with Tensorflow as backend is employed [28],[39]. The DNN algorithms are trained on a GPU server consisting of two Nvidia Quadro RTX 5000 cards, two Intel Xeon Gold 5222 Processors and 128 GB RAM.

2.4.2 Results

As for performance metrics, recall defined as $R = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN})$ and the false alarm rate $F = \mathrm{FP}/(\mathrm{FP} + \mathrm{TN})$ is used, where TP, TN, FP, and FN stand for true positive, true negative, false positive, and false negative, respectively. True positives (TP) and true negatives (TN) indicate the number of occurrences when the active/inactive users are correctly identified as active/inactive, respectively. Similarly, false positives (FP) and false negatives (FN) represent the number of occurrences when the inactive/active users are misclassified as active/active, respectively. In the following one iteration means updating the weights over a mini-batch. The rate of convergence of the weights in the training phase is investigated for the AUI networks. In this regard, in Fig. 2.3 the loss versus the number of iterations is reported. Comparing the curves with and without transfer learning, where $g_k(\cdot)$ for $k = 2$, 4 and 8 is trained for 3 epochs using the transfer learning approach, a considerable improvement in the speed of convergence of the training can



(a) $K = 2$ **(b)** $K = 4$

(c) $K = 8$

Figure 2.3: Training Loss with Transfer Learning and without Transfer Learning, SNR = 10 dB.

Table 2.1: Recall, $K = 8$, SNR $= 10\,\text{dB}$.

Epochs	Without TL	With TL
1	0.706	0.736
2	0.728	0.740
3	0.735	0.741
4	0.738	0.742
5	0.739	0.743
6	0.740	0.743
7	0.741	0.744
8	0.741	0.744
9	0.742	0.744
10	0.742	0.744

be observed. The improvement is substantial for all sparsity levels K, and it is particularly important for the networks designed for large K (see, e.g., the case $K = 8$). In the case $K = 2$ the advantage due to transfer learning is less pronounced. The improvement also in terms of recall is tabulated in Table 2.1, where the results with and without transfer learning is reported for $K = 8$ and SNR $= 10\,\text{dB}$. For obtaining the recall values through the multi-stage transfer learning, $g_1(\cdot)$ is trained for 10 epochs while $g_k(\cdot)$ for $2 \leq k \leq 8$ are trained for epochs as in the first column of the Table 2.1. The networks which are trained without the transfer learning approach are initialized through [33].

The recall for the proposed architecture is compared with the points taken from the literature proposing other algorithms, under the same simulation parameters, namely the Deep AUD (D-AUD) [18], and the compressed-sensing Approximate Message Passing (AMP) [18]. The curves for the proposed AUEI are obtained through the multi-stage transfer learning approach. The $g_1(\cdot)$ is trained for 10 epochs while $g_k(\cdot)$ for $2 \leq k \leq 8$ is trained for 3 epochs. The proposed approach shows improved recall values with respect to the other algorithms, as can be seen in Fig. 2.4 and Fig. 2.5 for SNR $= 10\,\text{dB}$ and SNR $= 20\,\text{dB}$, respectively. In contrast to the proposed approach, the other algorithms suffer from substantial performance degradation for high sparsity levels.

Table 2.2: False Alarm rate, Transfer Learning, epochs= 3.

Sparsity Level (K)	SNR = 10 dB	SNR = 20 dB
1	3.87×10^{-5}	1.11×10^{-7}
2	2.50×10^{-4}	2.05×10^{-6}
3	9.46×10^{-4}	4.50×10^{-5}
4	2.32×10^{-3}	2.54×10^{-4}
5	4.88×10^{-3}	8.10×10^{-4}
6	9.00×10^{-3}	1.96×10^{-3}
7	1.49×10^{-2}	4.59×10^{-3}
8	1.94×10^{-2}	6.24×10^{-3}

The false alarm rate for the proposed architecture with multi-stage transfer learning is presented in Table 2.2. It can be observed that the proposed approach, besides the previously discussed high recall, yields a negligible false alarm rate. In Fig. 2.6, the performance of the proposed algorithm is compared with D-AUD and AMP in terms of recall for the SNR range $0-20$ dB, $N_d = 7$ and $K = 4$. It can be observed that the proposed approach outperforms the other approaches, especially in the low SNR regime. To check the robustness of the proposed algorithm, the performance for overloading factors 125% and 250% is illustrated in Fig. 2.7 and 2.8, respectively. The overloading factor is defined as $N/(N_d S)$. For different overloading factors, a fixed length of the spreading sequence, S, and a number of users, N is assumed, while varying the number of measurements, N_d. A significant performance improvement can be observed for $N_d = 8$ in comparison to $N_d = 4$ for all the algorithms. In other words, increasing the number of measurements N_d or reducing the overloading factor yields better performance. It can be observed that the proposed algorithm outperforms the D-AUD and AMP in both scenarios, confirming the reliability of AUEI. Finally, the numerical comparison of computational complexity between AUEI (see Section 2.3.3) and D-AUD is presented, whose complexity for a given sparsity K is

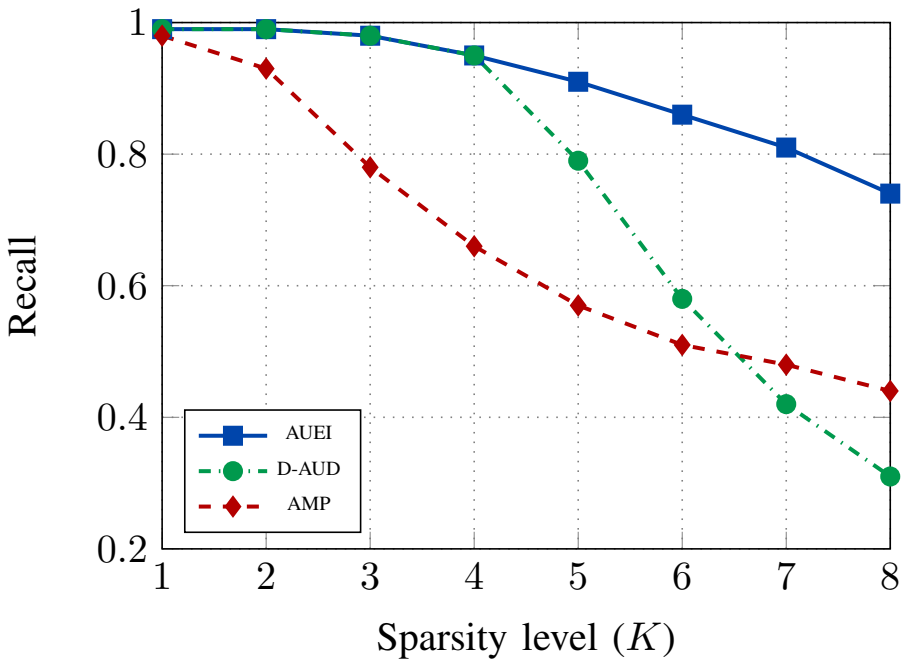


Figure 2.4: Recall vs. sparsity level, SNR = 10 dB.

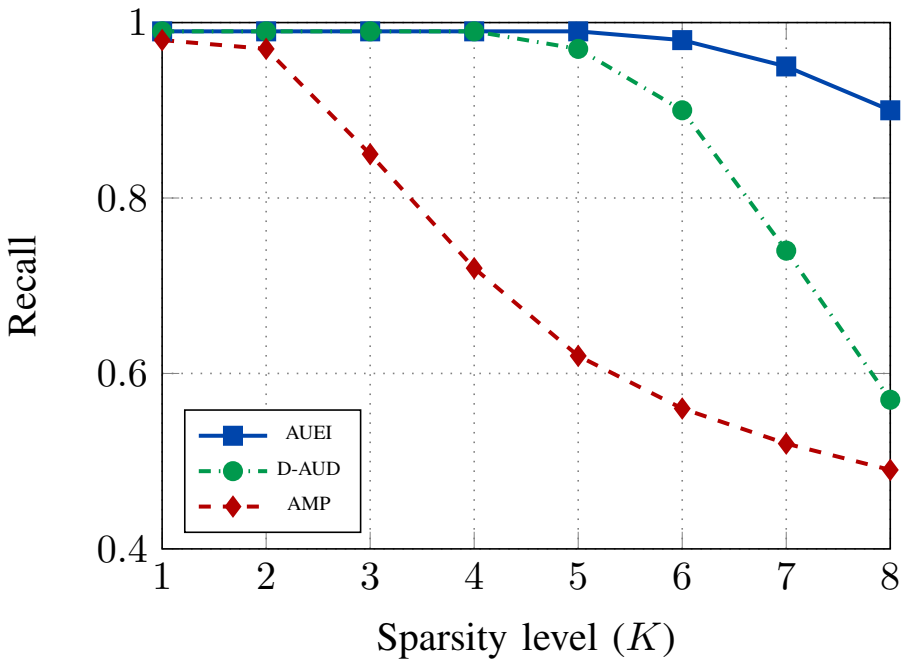


Figure 2.5: Recall vs. sparsity level, SNR = 20 dB.

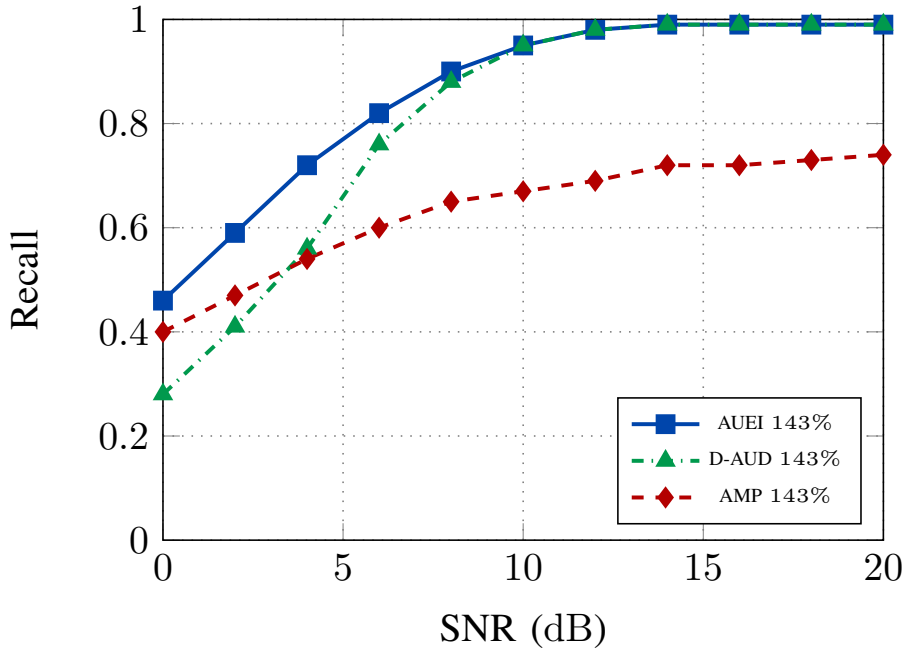


Figure 2.6: Recall vs. SNR, $N_d = 7, K = 4$.

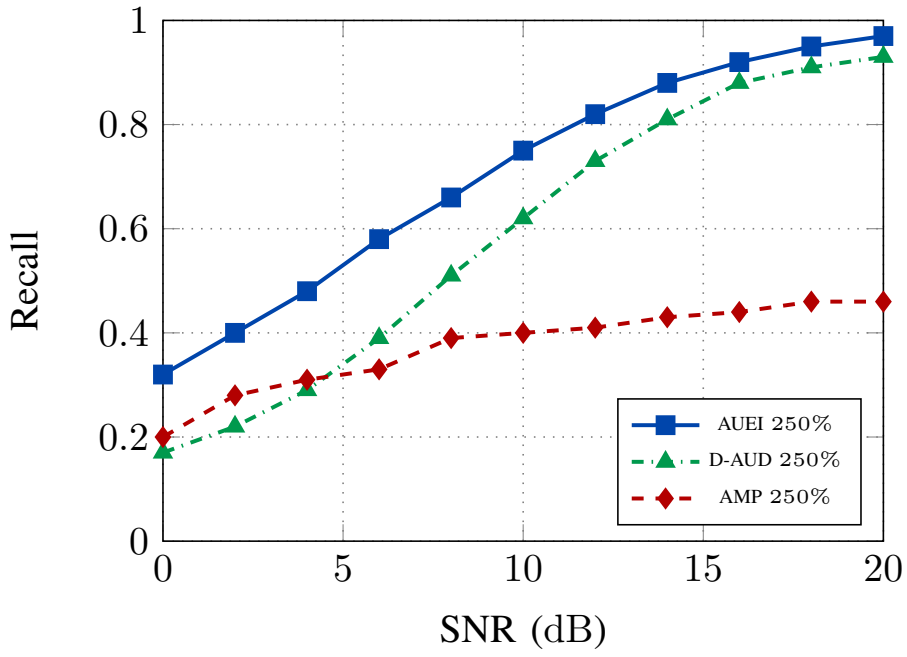


Figure 2.7: Recall vs. SNR, $N_d = 4, K = 4$.

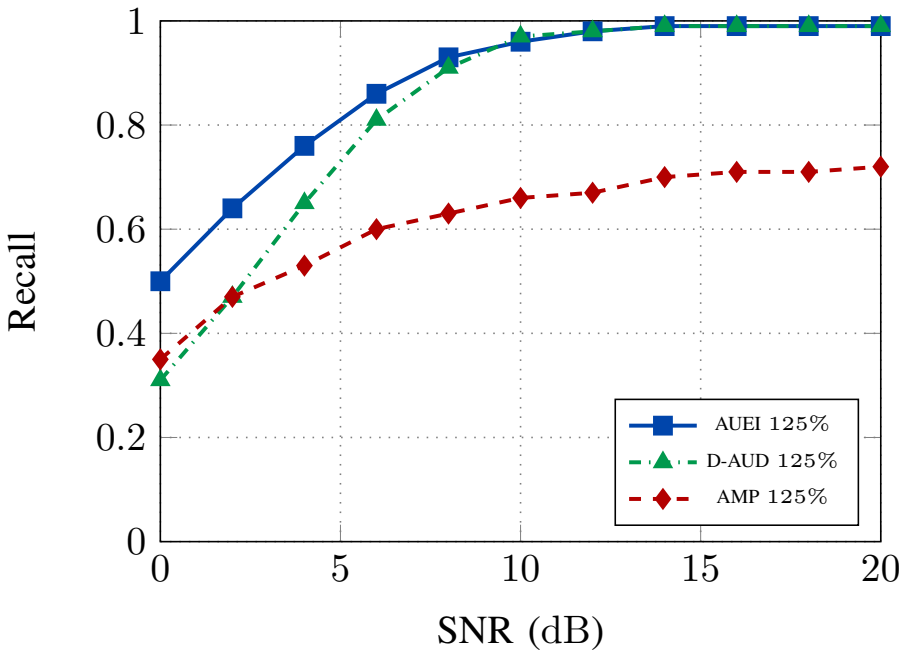


Figure 2.8: Recall vs. SNR, $N_d = 8, K = 4$.

stated in [18] as

$$C'_{\mathrm{D-AUD}} = 2L\alpha^2 + (4N_dS + 7L + 2N + 4)\alpha + (K+3)N - \frac{K(K+1)}{2} - 1\,.$$

For calculating the overall D-AUD complexity, the algorithm proposed in [18] for sparsity estimation is also taken into account . In this algorithm, the received vector is passed first through the D-AUD trained for sparsity level $K = 1$. If the output satisfies the threshold-based condition, this is considered as the sparsity level. Otherwise, the received vector is passed through the D-AUD network trained for $K = 2$, and so on. The procedure is repeated until the threshold-based condition is met or the maximum sparsity level is reached. Thus, for a given sparsity K, the received vector is passed through K D-AUDs. For this reason, the complexity of the D-AUD algorithm grows linearly with the sparsity level. Considering that, the overall computational

Table 2.3: Computational Complexity in FLOPs, $N_d = 7$.

	$K = 1$	$K = 2$	$K = 4$	$K = 8$
AUEI	2.02×10^7	2.02×10^7	2.02×10^7	2.02×10^7
D-AUD	1.25×10^7	2.51×10^7	5.01×10^7	1.00×10^8

complexity expression for D-AUD is

$$C_{\mathrm{D-AUD}} = KC'_{\mathrm{D-AUD}}\,. \tag{2.17}$$

Table 2.3 shows the computational complexity of AUEI and D-AUD for $N_d = 7$ and $K = 1, 2, 4$, and 8, calculated through (2.16) and (2.17). The number of hidden layers for AUEI and D-AUD is $L = 2$ and $L = 6$, respectively. As observed, the computational complexity of D-AUD increases linearly with the sparsity level, while the complexity of AUEI remains practically constant. This is because the dependence on the sparsity level K in (2.15) has a negligible effect on the overall computational complexity. Specifically, for all cases with more than one active user, the AUEI shows a significant gain in terms of complexity. So, despite having two separate architectures instead of one as in D-AUD, the proposed approach yields a lower complexity and better performance.

2.5 Conclusion

In this chapter, an active users detection method is proposed, realized by one DNN for active users enumeration and one for active users identification. The deep neural network architectures extract relevant features from the multiple measurements for enumeration and identification. Besides the fully-connected layers, both DNNs consist of convolutional layers to reduce the computational complexity. To minimize the training time for the active users identification networks, the multi-stage transfer learning technique is adopted. The numerical results demonstrate that the proposed approach is more effective than previously known methods in identifying the active

users, especially for high sparsity levels and low SNR. The false alarm rates are also analyzed, which are negligible for the scenarios of interest, and the computational complexity, which results lower than other approaches.

Future work will include analysis of the scalability of the proposed algorithm for a different number of users and further reduction of the computational cost.

References

[1] C. Bockelmann, N. K. Pratas, G. Wunder, S. Saur, M. Navarro, D. Gregoratti, G. Vivier, E. De Carvalho, Y. Ji, C. Stefanović, P. Popovski, Q. Wang, M. Schellmann, E. Kosmatos, P. Demestichas, M. Raceala-Motoc, P. Jung, S. Stanczak, and A. Dekorsy, "Towards massive connectivity support for scalable mMTC communications in 5G networks," *IEEE Access*, vol. 6, pp. 28969–28992, May 2018.

[2] F. Li, D. Wang, Y. Wang, X. Yu, N. Wu, J. Yu, and H. Zhou, "Wireless communications and mobile computing blockchain-based trust management in distributed internet of things," *Wireless Commun. and Mobile Computing*, vol. 2020, Dec. 2020.

[3] Y. Ahn, W. Kim, and B. Shim, "Active user detection and channel estimation for massive machine-type communication: Deep learning approach," *IEEE Internet Things J.*, vol. 9, pp. 11904 – 11917, July 2021.

[4] 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, p. 764–779, 2020.

[5] C. Bockelmann, H. F. Schepker, and A. Dekorsy, "Compressive sensing based multi-user detection for machine-to-machine communication," *Trans. on Emerging Telecommun. Technol.*, vol. 24, pp. 389–400, Apr. 2013.

[6] J. W. Choi, B. Shim, Y. Ding, B. Rao, and D. I. Kim, "Compressed sensing for wireless communications: Useful tips and tricks," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1527–1550, 2017.

[7] S. Haghighatshoar, P. Jung, and G. Caire, "Improved scaling law for activity detection in massive MIMO systems," in *IEEE Int. Sym. on Inf. Theory*, pp. 381–385, June 2018.

[8] T. Hara, H. Iimori, and K. Ishibashi, "Hyperparameter-free receiver for grant-free NOMA systems with MIMO-OFDM," *IEEE Wireless Commun. Lett.*, vol. 10, pp. 810–814, Apr. 2021.

[9] Y. Bai, B. Ai, and W. Chen, "Deep learning based fast multiuser detection for massive machine-type communication," in *IEEE 90th Veh. Tech. Conf.*, Sept. 2019.

[10] R. Garg and R. Khandekar, "Block-sparse solutions using kernel block RIP and its application to group lasso," in *Proc. of the Fourteenth Int. Conf. on Artificial Intelligence and Statistics*, vol. 15, pp. 296–304, Apr. 2011.

[11] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, pp. 359–366, Mar. 1989.

[12] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, pp. 82–97, Nov. 2012.

[13] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," in *Proc. 31st AAAI Conf. on Artificial Intelligence*, (San Francisco, California, USA), p. 4278–4284, Feb. 2017.

[14] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst. (NeurIPS)*, vol. 2, (Montreal, Canada), pp. 3104–3112, Dec. 2014.

[15] Z. Qin, H. Ye, G. Y. Li, and B.-H. F. Juang, "Deep learning in physical layer communications," *IEEE Wireless Commun.*, vol. 26, pp. 93–99, Apr. 2019.

[16] T. Wang, C.-K. Wen, H. Wang, F. Gao, T. Jiang, and S. Jin, "Deep learning for wireless physical layer: Opportunities and challenges," *China Commun.*, vol. 14, pp. 92–111, Nov. 2017.

[17] L. Dai, R. Jiao, F. Adachi, H. V. Poor, and L. Hanzo, "Deep learning for wireless communications: An emerging interdisciplinary paradigm," *IEEE Wireless Commun.*, vol. 27, pp. 133–139, Aug. 2020.

[18] W. Kim, Y. Ahn, and B. Shim, "Deep neural network-based active user detection for grant-free NOMA systems," *IEEE Trans. Commun.*, vol. 68, pp. 2143–2155, Apr. 2020.

[19] J. Wu, W. Kim, and B. Shim, "Pilot-less one-shot sparse coding for short packet-based machine-type communications," *IEEE Trans. Veh. Technol.*, vol. 69, pp. 9117–9120, Aug. 2020.

[20] L. Wei, S. Lu, H. Kamabe, and J. Cheng, "User identification and channel estimation by DNN-based decoder on multiple-access channel," in *Proc. IEEE Global Commun. Conf.*, Dec. 2020.

[21] J. H. I. de Souza and T. Abrão, "Deep learning-based activity detection for grant-free random access," *IEEE Syst. J.*, vol. 17, pp. 940–951, Mar. 2023.

[22] R. Hoshyar, F. P. Wathan, and R. Tafazolli, "Novel low-density signature for synchronous CDMA systems over AWGN channel," *IEEE Trans. Signal Process.*, vol. 56, pp. 1616–1626, Apr. 2008.

[23] Y. Fu, H. Li, Q. Zhang, and J. Zou, "Block-sparse recovery via redundant block OMP," *Signal Process.*, vol. 97, pp. 162–171, Apr. 2014.

[24] A. Elzanaty, A. Giorgetti, and M. Chiani, "Weak RIC analysis of finite Gaussian matrices for joint sparse recovery," *IEEE Signal Process. Lett.*, vol. 24, pp. 1473–1477, Oct. 2017.

[25] A. Elzanaty, A. Giorgetti, and M. Chiani, "Limits on sparse data acquisition: RIC analysis of finite gaussian matrices," *IEEE Trans. Inf. Theory*, vol. 65, pp. 1578–1588, Mar. 2019.

[26] T. Sivalingam, S. Ali, N. Huda Mahmood, N. Rajatheva, and M. Latva-Aho, "Deep neural network-based blind multiple user detection for grant-free multi-user shared access," in *IEEE 32nd Annual Int. Sym. on Personal, Indoor and Mobile Radio Commun.*, 2021.

[27] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[28] F. Chollet, "Keras." `https://github.com/fchollet/keras`, 2015.

[29] K. Eckle and J. Schmidt-Hieber, "A comparison of deep networks with ReLU activation function and linear spline-type methods," *Neural Netw.*, vol. 110, pp. 232–242, Feb. 2019.

[30] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, pp. 157–166, Mar. 1994.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," June 2015.

[32] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proc. 3rd Int. Conf. Learning Representations (ICLR)*, (San Diego, CA, USA), May 2015.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, (Santiago, Chile), pp. 1026–1034, Dec. 2015.

[34] X. Wu, *Performance evaluation, prediction and visualization of parallel systems*, vol. 4. Springer Science & Business Media, 1999.

[35] J. Wang, S. Kwon, and B. Shim, "Generalized orthogonal matching pursuit," *IEEE Trans. on Signal Process.*, vol. 60, pp. 6202–6216, Dec. 2012.

[36] A. Mariani, S. Kandeepan, and A. Giorgetti, "Periodic spectrum sensing with non-continuous primary user transmissions," *IEEE Trans. Wireless Commun.*, vol. 14, pp. 1636–1649, Mar. 2015.

[37] E. Recayte, A. Munari, and F. Clazzer, "Grant-free access: Machine learning for detection of short packets," in *Proc. 10th Adv. Satellite Multimedia Syst. Conf. and 16th Signal Process. for Space Commun. Workshop*, (Graz, Austria), Oct. 2020.

[38] E. Testi and A. Giorgetti, "Blind wireless network topology inference," *IEEE Trans. Commun.*, vol. 69, pp. 1109–1120, Feb. 2021.

[39] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

Chapter 3

Preamble Detection in Asynchronous Random Access

3.1 Introduction

In recent years, the demand for wireless data transmission has grown tremendously, leading to the rise of new applications involving communication among machines [1, 2]. In a conventional cellular communication system, resources are allocated to the users in a coordinated manner. However, resource allocation would be highly inefficient in mMTC scenario due to control signalling overhead [3]. To address these challenges, GF random access-based approaches have been proposed. In such schemes, devices transmit packets without coordination with the BS over the shared time or frequency resources [4]. Over the years, random access protocols have evolved from ALOHA [5] to more sophisticated protocols involving repetitions of packets and successive interference cancellation (SIC), with the aim to reduce signalling related to grants management and packets retransmission [6, 7, 8].

The GF random access schemes could be either synchronous or asynchronous [9]. In asynchronous systems, there is no common time reference between the BS and the active devices, which are therefore allowed to access the channel according to a "transmit at will" policy: since time is "fluid" and not organized into slots and frames, packets can arrive at the BS at any

time and data can in principle be transmitted as soon as they are generated and available at a node [9, 10]. In contrast with synchronous systems, where any two packets from different devices have either a complete overlap in time or no overlap at all, in asynchronous ones partial overlapping is the typical situation [11, 12]. While in synchronous GF schemes a minimum amount of control signalling is necessary to synchronize the active devices to the BS slot and frame time, in asynchronous ones absence of coordination is taken to an extreme level [13, 14].

Asynchronous GF schemes are of interest from several viewpoints. In asynchronous, the signalling overhead is reduced to the minimum, since even a downlink beacon signal for synchronization of active devices to the frame or slot becomes unnecessary [15, 16]. On the one hand, this reduces the burden on the network control plane. On the other hand, after wake up, active devices need not turn their radio on awaiting for the synchronization beacon, with a positive effect on the battery lifetime [17]. As such, asynchronous communication becomes particularly interesting for ultra-low-cost IoT devices. Moreover, since an active device can in principle start its transmission as soon as the data is available, latency in asynchronous access protocols tends to be lower that in synchronous ones.

The asynchronous random access setting also comes with its drawbacks and poses several challenges. Asynchronous random access systems simplify the access protocol on the device side but increase considerably the computational burden on the BS side [15, 18]. The first problem is related to the fact that there are no medium access control (MAC) frames that can be individually processed. The strategy that is usually adopted to overcome this issue is to proceed in a sliding window fashion, where new received signal samples are stored in memory overwriting the most outdated ones, as in [19, 20].

Another fundamental problem is connected to the random times of arrival of the users' packets, which makes the packet detection problem much more challenging than in synchronous systems where all packets are aligned with the global slots. The problem is further complicated by the fact that packet detection, representing the first step of the whole processing, must be performed prior to (or at least jointly with) channel estimation.

In [21], a correlator-based approach is used to detect packets in a satellite-based scenario, where devices start private and asynchronous virtual frames (VFs) independently of each other and transmit multiple replicas of a packet within them. A random access scheme based on correlation using DFT is proposed in [22] to detect preambles in a satellite scenario. In [23], a DL-based solution is proposed for the detection of preambles in satellite communication. Both approaches assumed single antenna receiver and AWGN channel. A CNN architecture is presented in [24] to identify the active user preambles in a slotted synchronous GF random access scenario with a single antenna at the BS. In [25] a neural network and logistic regression was developed to detect orthogonal preambles, and their multiplicity, for random access in Long Term Evolution (LTE) systems. In [26], a closed-form expression for the probability of detection of tagged preamble sequences at Next Generation NodeB (gNB) is proposed.

In a mMTC scenario, consideration must be given to a distinct propagation model, which is characterized by fading, shadowing, and possibly multiple antennas at the receiver. The main contributions of this chapter are summarised as follows:

- preamble detection in an asynchronous GF random access uplink scenario exploiting multiple antennas at the BS is performed;

- a channel model with fading, path-loss, and shadowing, assuming no power control is considered. Due to uncoordinated transmissions, preamble detection is performed by the BS before channel estimation;

- a DL-based preamble detection method consisting of a CNN is proposed that strikes a good trade-off between performance and complexity, compared to a classical correlator-based approach.

The rest of the chapter is organized as follows. The system model is presented in Section 3.2. In Section 3.3, the CNN architecture and correlator-based approach are explained. Section 3.4 contains the computational complexity analysis. Numerical results along with simulation setup are given in

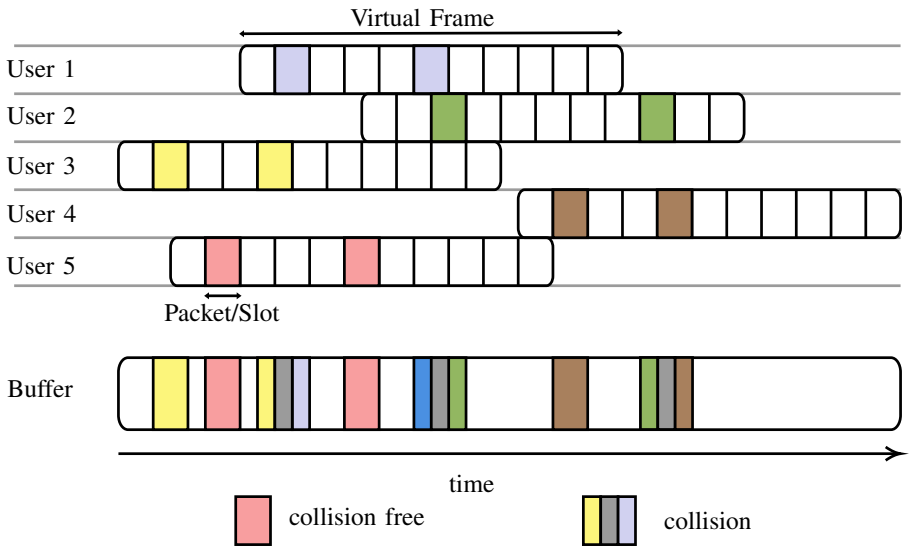


Figure 3.1: Pictorial representation of the users initiating virtual frame and transmitting replicas in an asynchronous scenario.

Section 3.5. Paylaod association using DNN is discussed in Section 3.6. The Conclusions are drawn in Section 3.7.

Throughout this chapter, matrices, vectors, and scalars are represented by boldface uppercase, boldface lowercase, and lowercase letters, respectively. The real and imaginary parts of a complex number are indicated as $\Re(\cdot)$ and $\Im(\cdot)$, respectively. The operations $(\cdot)^T$ and $(\cdot)^H$ denote the transpose and conjugate transpose, respectively. Notation $\mathcal{U}(a, b)$ indicates a uniform distribution between a and b. The normal and circularly-symmetric complex normal distributions with mean 0 and variance σ^2 are denoted by $\mathcal{N}(0, \sigma^2)$ and $\mathcal{CN}(0, \sigma^2)$, respectively.

3.2 System Model

Consider an asynchronous GF random access uplink scenario, where users are uniformly distributed within an annulus with inner and outer circles of radius $D_{\min}$ and $D_{\max}$, respectively, as shown in Fig. 3.2. The BS is positioned in the center of the annulus. Each device has a single antenna whereas the BS is equipped with M antennas. The number of users becoming active in

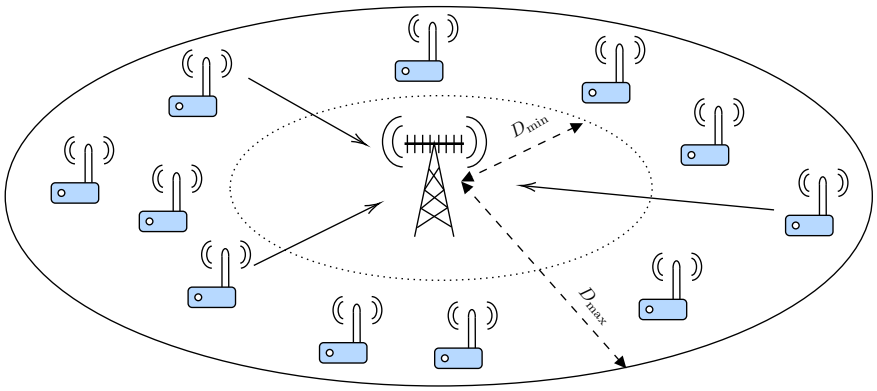


Figure 3.2: The depiction of uplink communication scenario.

an uplink symbol time follows a Poisson distribution with mean λ. When a user becomes active, it initiates a VF comprising N_S slots, with each slot duration equal to the packet length as shown in Fig. 3.1. The VF is local to the device: the BS is unaware of the starting time of VFs but it is aware of the number of slots in a VF. Each user transmits multiple packet replicas to boost performance as in [6, 8]. To transmit N_{rep} replicas of the packet, the user selects N_{rep} slots from the set $\{1, \cdots, N_S\}$ without replacement and with uniform probability. The packet transmission is considered symbol-wise synchronous. A packet consists of a preamble of N_P symbols, $\boldsymbol{s} = [s_1, \cdots, s_{N_P}]^T \in \mathbb{C}^{N_P \times 1}$, which is the same for all users, and a user-specific data payload of length N_D.

A Rayleigh block fading channel model is assumed with no power control and with a coherence time equal to the packet (and virtual slot) time. Accordingly, the channel gain between a device and one BS antenna is constant during the transmission of a packet, but independent from replica to replica from the same user. The channel gains between a single device and different BS antennas are considered independent. The N_{rep} replicas from the same user experience the same path-loss and large-scale fading, but independent Rayleigh-distributed small-scale fading. The vector of received samples at the M BS antennas at symbol time i, $\boldsymbol{y}(i) \in \mathbb{C}^{M \times 1}$, may be expressed as

$$\boldsymbol{y}(i) = \sum_{j \in \mathcal{A}_P} \boldsymbol{h}_j \, p_j(i) + \sum_{l \in \mathcal{A}_D} \boldsymbol{h}_l \, q_l(i) + \boldsymbol{n}(i) \tag{3.1}$$

where

- $\mathcal{A}_P$ and $\mathcal{A}_D$ are the set of users transmitting a preamble and data symbol at ith sample time, respectively;
- $p_j(i)$ is the symbol of preamble $\boldsymbol{s}$ transmitted by user $j \in \mathcal{A}_P$ and $q_l(i)$ represents the data symbol transmitted by user $l \in \mathcal{A}_D$ at the ith sample time;
- $\boldsymbol{h}_k = [h_{k,1}, \cdots, h_{k,M}]^T \in \mathbb{C}^{M\times 1}$ is the vector of channel gains between the kth user and the BS, where $h_{k,m} \sim \mathcal{CN}(0, \sigma^2_{h_k})$ for $m = 1, ..., M$. The variance $\sigma^2_{h_k}$ is given by $\gamma\,(D_{\max}/d_k)^{\beta}$, where γ is the log-normal shadowing coefficient in linear scale, i.e., $\gamma_{\text{dB}} \sim \mathcal{N}(0, \sigma^2_{\text{dB}})$, β is the path-loss exponent, and d_k is the distance between the kth device and the BS. The distance d_k is randomly distributed as

$$\sqrt{D^2_{\min} + (D^2_{\max} - D^2_{\min}) \cdot \mathcal{U}(0,1)};$$

- $\boldsymbol{n}(i) \in \mathbb{C}^{M\times 1}$ is the vector of independent and identically distributed noise samples, each distributed as $\mathcal{CN}(0, \sigma^2_{\text{n}})$.

3.3 Preamble Detection

This section presents the proposed DL-based approach, which consists of a CNN that performs preamble detection starting from raw received samples at the BS. A correlator-based methodology is introduced as a benchmark, showing how it can be derived from the generalized likelihood ratio test (GLRT) design method.

3.3.1 CNN Architecture

Assume we want to check if N_P consecutive samples at an initial offset i_0 correspond to a preamble or not. For this purpose, the observation matrix $\boldsymbol{R} = \{r_{i,j}\} = [\boldsymbol{y}(i_0), \boldsymbol{y}(i_0+1), \cdots, \boldsymbol{y}(i_0+N_P-1)]$ is considered. As the samples are complex, the received samples are split into real and imaginary

parts, and then the reference preamble sequence is added, obtaining the matrix

$$\boldsymbol{Y} = \begin{bmatrix} \Re(\boldsymbol{R}) & \Im(\boldsymbol{R}) \\ \Re(\boldsymbol{s}^T) & \Im(\boldsymbol{s}^T) \end{bmatrix} \tag{3.2}$$

$$= \begin{bmatrix} \Re(r_{1,i_0}) & \cdots & \Re(r_{1,i_0+N_P-1}) & \Im(r_{1,i_0}) & \cdots & \Im(r_{1,i_0+N_P-1}) \\ \Re(r_{2,i_0}) & \cdots & \Re(r_{2,i_0+N_P-1}) & \Im(r_{2,i_0}) & \cdots & \Im(r_{2,i_0+N_P-1}) \\ \vdots & & \vdots & \vdots & & \vdots \\ \Re(r_{M,i_0}) & \cdots & \Re(r_{M,i_0+N_P-1}) & \Im(r_{M,i_0}) & \cdots & \Im(r_{M,i_0+N_P-1}) \\ \Re(s_1) & \cdots & \Re(s_{N_P}) & \Im(s_1) & \cdots & \Im(s_{N_P}) \end{bmatrix}. \tag{3.3}$$

Matrix $\boldsymbol{Y} \in \mathbb{R}^{(M+1)\times 2N_P}$ is a feature map obtained from the raw received samples at the BS and is the input to the DL model. Extensive investigation revealed that concatenating the reference preamble with the received symbols and feeding the resulting matrix into the DL model yields better performance.

Various architectures with different numbers, types, and sizes of layers were explored, to find a good balance between performance and complexity. Finally, considering the 2-dimensional nature of the input feature map, the CNN architecture depicted in Fig. 3.3 is selected. At the BS, each antenna receives the same transmitted symbols with different channel gains, as depicted through (3.2). The purpose of the convolutional filter is to extract the common features shared among the multiple antennas, i.e. identifying the symbols transmitted by the device. The filter also tries to learn the mapping between the received symbol and the reference preamble, which helps classify the received symbols as a preamble or non-preamble. In particular, two convolutional layers with 8 and 4 filters of the same size is employed, respectively, without any padding. At the BS, each antenna receives the same transmitted symbols with different channel gains. The purpose of the convolutional filter is to extract the common features shared among the multiple antennas, i.e. identifying the symbols transmitted by the device. The filter also tries to learn the mapping between the received symbol and the reference preamble, which helps classify the received symbols as preamble or

non-preamble.

The convolutional layers are followed by fully-connected (1.3) and dropout layers. To mitigate overfitting and enhance the model's generalization capacity, the CNN incorporates a dropout layer, which randomly drops connections between the fully-connected layers during the training phase. This process helps to minimize the dependencies between neurons.

The preamble detection is essentially a binary classification problem, i.e., classifying the received symbols as preamble or non-preamble. Consequently, the last fully-connected layer consists of only one neuron employing sigmoid as an activation function. It is defined as

$$\hat{q} = \frac{1}{1 + e^{-z}}\,, \tag{3.4}$$

where $\hat{q}$ estimates the likelihood of the input being a preamble. A threshold of 0.5 is applied to this value to perform classification.

The neural network architecture is linked to a cost function, which equals zero for ideal classification and increases when the inputs are misclassified. From this standpoint, a binary cross-entropy loss is utilized, which is formulated as

$$\mathcal{J}(q, \hat{q}) = -q \log \hat{q} - (1 - q) \log(1 - \hat{q}) \tag{3.5}$$

where q is the true label, equal to 1 if the input samples correspond to a (possibly interfered) preamble and to 0 otherwise. The objective of the training is to determine the suitable weights of the DL model that minimizes the cost function. In this approach, Adam optimizer is utilized, which is an extended version of the gradient descent algorithm [27], adopting mini-batches to enhance training efficiency.

3.3.2 Correlator-based Approach

The proposed DL-based solution is compared with a classical approach based on hypothesis testing. To simplify the analytical derivation and make the problem tractable, consider a simpler scenario in which the BS antenna can

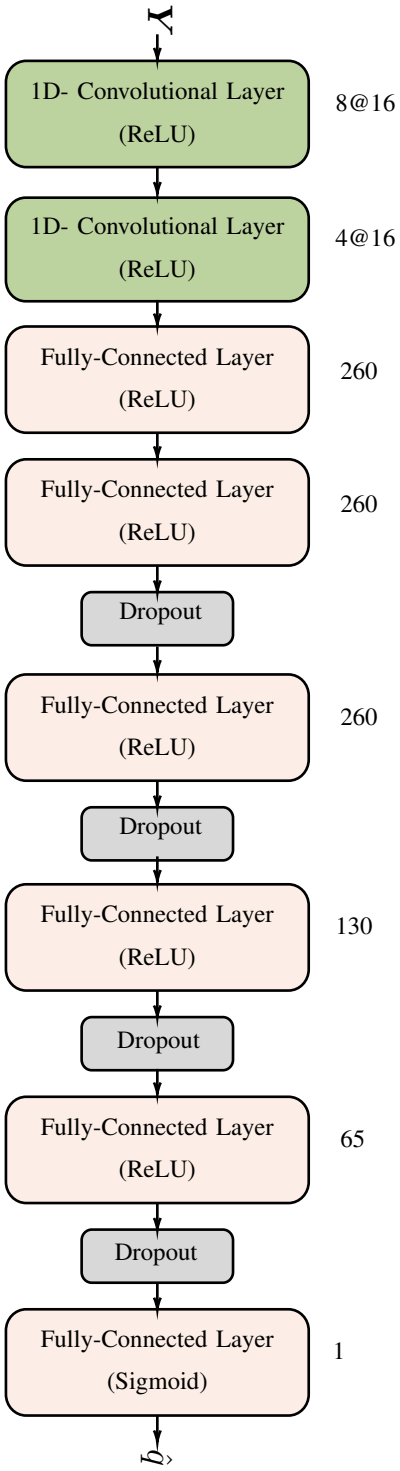


Figure 3.3: A schematic representation of the architecture of the proposed CNN for preamble detection, where the size of each layer is specified. For instance, the first convolutional layer has 8 filters of dimensions 16, and the first fully-connected layer contains 260 neurons.

receive either the symbols corresponding to a pilot sequence and noise, or only noise. After observing N_{P} subsequent complex samples, $r_{m,i}$, where i and m are the sample and BS antenna indexes, respectively, the pilot detector must choose between two possible hypotheses

$$\mathcal{H}_0 : r_{m,i} = n_{m,i}, \qquad i = 1, \ldots, N_{\mathrm{P}}, \quad m = 1, \ldots, M \tag{3.6}$$

$$\mathcal{H}_1 : r_{m,i} = h_{k,m} s_i + n_{m,i}, \qquad i = 1, \ldots, N_{\mathrm{P}}, \quad m = 1, \ldots, M \tag{3.7}$$

where k is the user transmitting the preamble. Hypothesis $\mathcal{H}_0$ represents the case where there is no pilot, while $\mathcal{H}_1$ corresponds to the case in which the pilot is present.

One widely adopted approach to the noncoherent hypothesis test problem in the case of $M = 1$ is to use the noncoherent correlation as a metric. Here, a better test is derived which is valid also for any finite M. Consider the optimum likelihood ratio test (LRT) which in general, can be written as

$$\Lambda(\boldsymbol{R}) = \frac{f_{\boldsymbol{R}|\mathcal{H}_1}(\boldsymbol{R}|\mathcal{H}_1)}{f_{\boldsymbol{R}|\mathcal{H}_0}(\boldsymbol{R}|\mathcal{H}_0)} \tag{3.8}$$

where $f_{\boldsymbol{R}|\mathcal{H}_k}(\boldsymbol{R}|\mathcal{H}_k)$ is the probability density function (PDF) of the matrix of random vectors $\boldsymbol{R}$ in the hypothesis $\mathcal{H}_k$ and $\boldsymbol{r}_m$ is the random vector describing the received samples at the mth antenna. The general expression (3.8) is now tailored to the preamble detection problem. For the $\mathcal{H}_0$ hypothesis, the conditional PDF is given as

$$f_{\boldsymbol{R}|\mathcal{H}_0}(\boldsymbol{R}|\mathcal{H}_0) = \prod_{m=1}^{M} \prod_{i=1}^{N_{\mathrm{P}}} \frac{1}{2\pi\sigma_{\mathrm{n}}^2} e^{-\frac{|r_{m,i}|^2}{2\sigma_{\mathrm{n}}^2}}. \tag{3.9}$$

On the other side, in the $\mathcal{H}_1$ hypothesis the observed samples are generated by the deterministic and known pilot symbols, multiplied by the Rayleigh fading coefficients. Hence, the conditional PDF for the $\mathcal{H}_1$ hypothesis is given as

$$f_{\boldsymbol{R}|\mathcal{H}_1,\boldsymbol{h}}(\boldsymbol{R}|\mathcal{H}_1, \boldsymbol{h}) = \prod_{m=1}^{M} \prod_{i=1}^{N_{\mathrm{P}}} \frac{1}{2\pi\sigma_{\mathrm{n}}^2} e^{-\frac{|r_{m,i} - h_{k,m} s_i|^2}{2\sigma_{\mathrm{n}}^2}}. \tag{3.10}$$

Substituting the unknown channel coefficient $h_{k,m}$ in (3.10) with its maximum likelihood (ML) estimation, we obtain

$$f_{\boldsymbol{R}|\mathcal{H}_1}(\boldsymbol{R}|\mathcal{H}_1) = \prod_{m=1}^{M}\prod_{i=1}^{N_\mathrm{P}} \frac{1}{2\pi\sigma_\mathrm{n}^2} e^{-\frac{|r_{m,i}-\hat{h}_{k,m}^{\mathrm{ML}} s_i|^2}{2\sigma_\mathrm{n}^2}} \tag{3.11}$$

where $\hat{h}_{k,m}^{\mathrm{ML}} = \frac{\sum_{i=1}^{N_\mathrm{P}} r_{m,i} s_i^H}{\|\boldsymbol{s}\|^2}$. Now, substituting (3.9) and (3.11) in (3.8) we get the GLRT

$$\Lambda(\boldsymbol{R}) = \frac{\prod_{m=1}^{M}\prod_{i=1}^{N_\mathrm{P}} \frac{1}{2\pi\sigma_\mathrm{n}^2} e^{-\frac{|r_{m,i}-\hat{h}_{k,m}^{\mathrm{ML}} s_i|^2}{2\sigma_\mathrm{n}^2}}}{\prod_{m=1}^{M}\prod_{i=1}^{N_\mathrm{P}} \frac{1}{2\pi\sigma_\mathrm{n}^2} e^{-\frac{|r_{m,i}|^2}{2\sigma_\mathrm{n}^2}}} \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \eta \tag{3.12}$$

which, in logaritmic form becomes $\Lambda^{(1)}(\boldsymbol{R}) \underset{\mathcal{H}_1}{\overset{\mathcal{H}_0}{\lessgtr}} \eta$ with metric

$$\Lambda^{(1)}(\boldsymbol{R}) = \sum_{m=1}^{M}\sum_{i=1}^{N_\mathrm{P}} |r_{m,i} - \hat{h}_{k,m}^{\mathrm{ML}} s_i|^2 - |r_{m,i}|^2 \tag{3.13}$$

where η is the test threshold and (3.13) is obtained by removing all the constant terms. Incorporating the ML estimation of the channel coefficients in (3.13) we obtain

$$\begin{aligned}\Lambda^{(1)}(\boldsymbol{R}) &= \sum_{m=1}^{M} -2\Re\left\{ \frac{\sum_{j=1}^{N_\mathrm{P}} r_{m,j} s_j^H}{\|\boldsymbol{s}\|^2} \sum_{i=1}^{N_\mathrm{P}} r_{m,i}^H s_i \right\} \\ &\quad + \left| \frac{\sum_{j=1}^{N_\mathrm{P}} r_{m,j} s_j^H}{\|\boldsymbol{s}\|^2} \right|^2 \left| \sum_{i=1}^{N_\mathrm{P}} s_i \right|^2 \\ &= \sum_{m=1}^{M} \frac{|\sum_{i=1}^{N_\mathrm{P}} r_{m,i} s_i^H|^2}{\|\boldsymbol{s}\|^2}.\end{aligned}$$

Since all the constant terms can be incorporated in the threshold, the term $\|\boldsymbol{s}\|^2$ can be removed to obtain the final expression of the metric

$$\Lambda(\boldsymbol{R}) = \sum_{m=1}^{M} \left| \sum_{i=1}^{N_\mathrm{P}} r_{m,i} s_i^H \right|^2 \underset{\mathcal{H}_1}{\overset{\mathcal{H}_0}{\lessgtr}} \eta\,. \tag{3.14}$$

3.4 Computational Complexity

The computational complexity is evaluated in terms of FLOPs. The real addition, subtraction, and multiplication, are taken as a single FLOP while division and exponential operations as 4 and 8 FLOPs, respectively. The complex addition and subtraction operations are considered as two FLOPs and complex multiplication as six FLOPs. [28, 29, 30]

3.4.1 CNN Complexity

The number of FLOPs of a convolutional layer is given by

$$C_{\text{cv}} = 2N_{\text{cv}}F_{\text{cv}}G_{\text{cv}}D_{\text{cv}} \tag{3.15}$$

where N_{cv}, F_{cv}, G_{cv}, and D_{cv} represent the number of convolution filters, size of the filter, number of channels, and output shape, respectively. The output shape D_{cv} is expressed as $(I - F + 2 \cdot P)/S + 1$, where I, F, P, and S specify the input size, filter size, padding, and stride. The ReLU is applied to the output of the convolutional layers, resulting in

$$C_{\text{ReLU}} = N_{\text{cv}}D_{\text{cv}}\,. \tag{3.16}$$

The number of FLOPs in a fully-connected layer (1.3) can be expressed as

$$C_{\text{FC}} = 2 \cdot \text{in} \cdot \text{out} + \text{out}\,. \tag{3.17}$$

The dropout layer involves elementwise multiplication operations; for a single operation, the complexity is 1. The sigmoid function and thresholding in the last layer yield 14 FLOPs. The total complexity of the CNN, with α representing the number of neurons in the first fully-connected layer, is given

by

$$\begin{aligned} C_{\text{CNN}} =& 2N_{\text{cv}_1}F_{\text{cv}_1}\,M\,D_{\text{cv}_1} + 2N_{\text{cv}_2}F_{\text{cv}_2}N_{\text{cv}_1}D_{\text{cv}_2} \\ &+ N_{\text{cv}_1}D_{\text{cv}_1} + N_{\text{cv}_2}D_{\text{cv}_2} + 2N_{\text{cv}_2}D_{\text{cv}_2}\alpha \\ &+ \frac{13\alpha^2}{4} + 5\alpha + 14\,. \end{aligned} \tag{3.18}$$

3.4.2 Correlator Complexity

The inner sum $\sum_{i=1}^{N_{\text{P}}} r_{m,i}s_i^H$ for the mth antenna requires N_P and $N_P - 1$ complex multiplication and addition operations, respectively. The $|\cdot|^2$ operation results in two real multiplication operations and one real addition operation for each antenna. The total computational cost is then

$$C_{\text{corr}} = 8M\,N_P + 2M - 1\,. \tag{3.19}$$

3.5 Implementation and Results

3.5.1 Simulation Setup

The performance analysis, for both the correlator-based approach (3.14) and the CNN, is conducted assuming $M = 32$ and $M = 64$ antennas at the BS, with SNR per antenna ranging from $-10\,\text{dB}$ to $20\,\text{dB}$. The SNR is defined as $\text{SNR} = 1/\sigma_n^2$, and represents the median SNR per antenna element for a user on the edge of the cell. Clearly, the average SNR inside the cell is higher than that on the boundary. The minimum and maximum distances of a user from the BS are $D_{\min} = 5\,\text{m}$ and $D_{\max} = 100\,\text{m}$, respectively. The path-loss exponent is set to $\beta = 2$ and the standard deviation of the log-normal shadowing is taken as $\sigma_{\text{dB}} = 3$.

A preamble and payload of length $N_P = 63$ and $N_D = 150$ are considered, respectively. The preamble sequence is generated by a linear feedback shift register of length 6 with primitive polynomial $p(x) = x^6 + x + 1$ over the Galois field GF(2). The sequence is designed to have good (aperiodic) auto- and cross-correlation properties and allow accurate channel estimation. The

pilot sequence bits are then converted to $N_P = 63$ binary phase shift keying symbols with unitary energy using $x_i = e^{j(\pi/4+\phi_i\pi)}$, where $\phi_i \in \{0, 1\}$ is the ith bit of the pilot sequence and x_i is the corresponding complex symbol. The payload of each user is populated randomly with quadrature phase-shift keying symbols having an equal probability of occurrence.

For generating a dataset, a buffer of $M \times 213{,}000$ complex symbols is considered, i.e., one sub-buffer for each antenna. The number of active users in a symbol time in the buffer is randomly generated by Poisson distribution with λ equal to $[0.05, 0.25, 0.5, 0.75, 1, 1.2, 1.45] \times 10^{-2}$, such that the average number of packet collisions per slot ranges from 1 to 7 [1] When a user becomes active, it initiates a virtual frame consisting of $N_S = 100$ slots, where each slot equals the packet size. The user sends $N_{\text{rep}} = 2$ replicas in slots chosen randomly without replacement. The user packet may get partially or fully interfered by packets from other users; at time i the received sample as in an asynchronous scheme is mathematically expressed by (3.1).

The samples for training and test sets are extracted from the buffer after the placement of packets, as described above. For the preamble case, N_P consecutive samples are obtained from the buffer that contains the entire preamble sequence. For the non-preamble case, N_P consecutive samples are randomly selected from the buffer that do not satisfy the preamble case condition. Ensuring a balanced dataset involves acquiring an equal number of instances for both preamble and non-preamble scenarios, while also considering an equal number of examples for each λ value. For instance, $8 \cdot 10^3$ examples are generated per λ per class (preamble or non-preamble). For each SNR value, a separate CNN is trained but with the same architecture as depicted in Fig. 3.3. Each dataset comprises $1.12 \cdot 10^5$ samples, which are split into 70% training set and 30% test set.

For each hyperparameter (learning rate, epochs, mini-batch size, dropout rate, number of neurons, etc.) of the model, the performance is evaluated on a range of values by fixing other hyperparameters and selecting the one that results in the best performance [27]. The final result of this search yielded learning rate, epochs, and mini-batch size 0.001, 20, and 512, respectively,

[1] The full derivation can be found in the appendix A.

Table 3.1: Comparison between CNN and CNNx, $\eta = 0.5$

	CNN		**CNNx**		
SNR (dB)	R	F	R	F	
20	0.9984	0.0011	0.9553	0.0192	$M = 32$
10	0.9973	0.0012	0.9578	0.0126	
0	0.9899	0.0012	0.9529	0.0175	
−10	0.9639	0.0157	0.9292	0.1185	
20	0.9976	0.0014	0.9546	0.0078	$M = 64$
10	0.9989	0.0018	0.9563	0.0081	
0	0.9919	0.0008	0.9442	0.0145	
−10	0.9794	0.0028	0.9298	0.0472	

with the architecture depicted in Fig. 3.3. A drop-out rate of 0.2 and 0.3 for $M = 32$ and $M = 64$ is employed, respectively.

3.5.2 Numerical Results

As for performance metrics, the detection rate (or recall) which is defined as

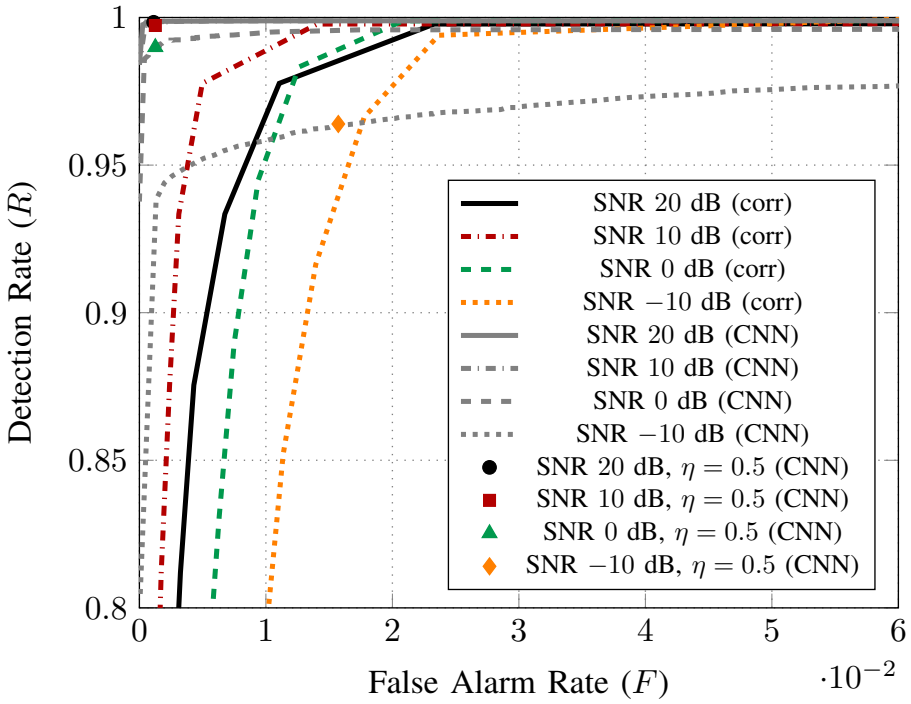
$$R = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

and the false alarm rate

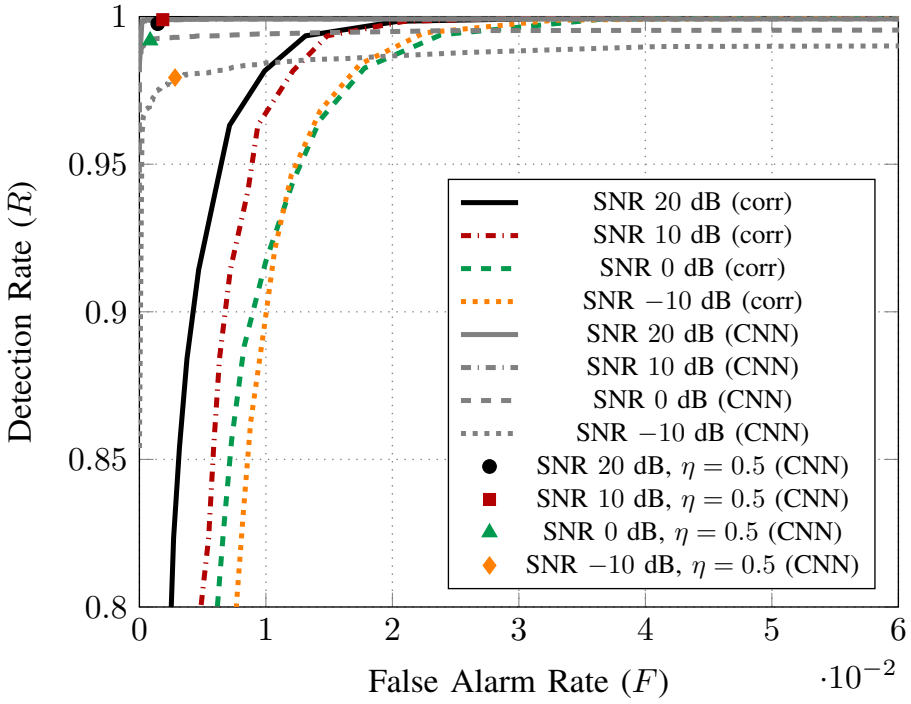
$$F = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$

is employed, where the true positives, true negatives, false positives, and false negatives are denoted by TP, TN, FP, and FN, respectively. TP and TN correspond to the instances when the preamble and non-preamble cases are correctly identified, respectively. Likewise, FP and FN indicate the number of instances when the non-preamble/preamble is misclassified as preamble/non-preamble, respectively.

The receiver operating characteristics (ROC) curves are reported in Fig. 3.4a and Fig. 3.4b for $M = 32$ and $M = 64$, respectively. The curves are obtained for the correlator-based approach by varying the threshold η,



(a) Base station with $M = 32$ antennas.



(b) Base station with $M = 64$ antennas.

Figure 3.4: Comparison between the CNN and the correlator.

from 0 to $\eta_{\max}$ with a step size of 10000, where $\eta_{\max}$, depending on the number of antennas at the BS, is the maximum correlation value over all the examples. In the same figures, the CNN ROC curves are obtained by varying the threshold η, from 0 to 1, with a step size of 0.01. The points represent the performance of the CNN classifier at $\eta = 0.5$.

The CNN-based classifier exhibits a substantial improvement over the correlation-based detector. Indeed, it can be observed that the same detection rate provided by the CNN can be achieved with the correlator-based approach but at a higher false alarm rate, for all SNRs. For example, with SNR $= 20\,\text{dB}$, $M = 32$, and assuming a target detection rate $R = 0.998$, the correlator gives a false alarm rate $F = 0.023$, while the CNN achieves $F = 0.001$. As the number of antennas increases from $M = 32$ to $M = 64$, the improvement given by the CNN is even more pronounced. In Fig. 3.4a, the correlation-based approach for SNR 10 dB outperforms the 20 dB one because the hypothesis testing-based method does not consider interference.

To address the motivation behind using convolutional layers in the proposed architecture, the performance of the CNN is compared with that of a vanilla network consisting of fully-connected layers with ReLU activation functions, specifically (FC, ReLU, FC, ReLU, FC, and sigmoid). The first, second, and last fully-connected layer contains 130, 65, and 1 neurons, respectively. The computational complexity of the vanilla network is equivalent to the proposed CNN. The results of both the vanilla network and CNN for $M = 64$ antennas are presented in Table 3.2. The table clearly demonstrates that despite sharing the same computational complexity, the CNN outperforms the vanilla network. This observation serves as a motivation for adopting the CNN-based approach.

To assess the robustness of the proposed CNN architecture in scenarios where the BS is also unaware of the median SNR at the edge of the cell, a single CNN model is trained, referred to as CNNx, on examples obtained with all the considered SNR values. The performance of the CNNx is compared with CNN models trained specifically for each SNR value, simply regarded as CNN. The results of this comparison are presented in Table 3.1 for $M = 32$ and $M = 64$. In CNNx, a dropout rate of 0.1 and 0.2 for $M = 32$ and $M = 64$

Table 3.2: Vanilla Network versus CNN. $M = 64$

	CNN		**Vanilla Network**	
SNR (dB)	R	F	R	F
20	0.9976	0.0014	0.9893	0.0276
10	0.9989	0.0018	0.9838	0.0235
0	0.9919	0.0008	0.9669	0.0320
-10	0.9794	0.0028	0.9324	0.0537

Table 3.3: Computational cost

	$M = 32$	$M = 64$
Correlator	1.62×10^4	3.24×10^4
CNN	1.23×10^6	2.14×10^6

is utilized, respectively. As expected, the numerical results show that training a single model on multiple SNRs leads to performance degradation. However, the performance degradation is only about $3.5 - 4.3\%$ and $4.3 - 5\%$ for the detection rate, in the case of $M = 32$ and $M = 64$, respectively.

The computational cost of the algorithms is reported in Table 3.3. It can be observed that the correlator is computationally less expensive than the CNN. However, the latter outperforms the former as discussed earlier. Furthermore, as the number of antennas at the BS increases from $M = 32$ to $M = 64$, the computational complexity of the correlator doubles, while for the CNN it increases by a factor 1.74. This is due to the fact that convolutional layers are employed which reduce the computational complexity, as only the first layer has a linear relationship with the number of antennas M, while the rest of the architecture is independent of M. Besides this, the convolutional layer allows the extraction of the relevant features shared among the multiple antennas.

The execution time of code can be influenced by various factors, including the hardware platform and programming language used. For instance, the same code written in C++ and Python may offer different performance characteristics due to Python being an interpreted language while C++ is

Table 3.4: Execution Time per sample in seconds

	$M = 32$	$M = 64$
Correlator	2.45×10^{-6}	2.69×10^{-6}
CNN	1.39×10^{-4}	3.08×10^{-4}

compiled. Additionally, deep learning libraries like Keras, TensorFlow, and PyTorch can exhibit varying performance in terms of training and execution time for a given neural network. Furthermore, the choice between running code on a CPU or GPU can significantly impact execution time.

To have a fair comparison of the execution time, the code for both algorithms was written in Keras with Tensorflow as backend and executed on a GPU server consisting of an Nvidia Quadro RTX 5000 card. Based on the above-mentioned configurations, the execution time per sample in seconds is obtained as depicted in Table 3.4. It is evident from Table 3.4 that the CNN offers higher execution time than the correlator-based approach. For the weighted metrics, consider the following formula

$$W_M = (1 - A)L \tag{3.20}$$

where A is the accuracy defined as

$$A = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

and L is the normalized order of magnitude of the number of FLOPs. Calculating the normalized order of magnitude of FLOPs involves determining the order of magnitude of the computational complexity for both the CNN and correlator-based approach (i.e., applying a base-10 logarithm), followed by normalizing them through division by the maximum value between the two. The order of magnitude of the number of FLOPs is taken into account to ensure a fair comparison since computational complexities present an exponential growth. For example, for $M = 32$ antennas, the complexity of the CNN approach is 1.62×10^4 (with a weight of 4.21), while the correlator-based approach has a complexity of 1.23×10^6 (with a weight of 6.09). The

Table 3.5: Performance evaluation using Weighted Metrics, $M = 64$

(a) False alarm rate = 0.001

SNR (dB)	Correlator	CNN	CNNx
20	0.1837	0.0020	0.0610
−10	0.3383	0.0155	0.0911

(b) False alarm rate = 0.005

SNR (dB)	Correlator	CNN	CNNx
20	0.0324	0.0031	0.0339
−10	0.1670	0.0127	0.0710

results for the weighted complexity for the false alarm rate of 0.001 and 0.005 are shown in Table 3.5a and 3.5b, respectively. The lower the value of the weighted metrics the better the performance of the algorithm. Although the CNN offers higher computational complexity than the correlator-based approach, the CNN outperforms the latter due to higher performance.

3.6 Payload Association

In the preceding sections, preamble detection is addressed within an asynchronous random access scenario. Building upon this groundwork, this section takes the next step towards associating packet replicas using the DL-approach. To enhance packet decoding and reduce the probability of packet loss, merging replicas of the same packet presents a viable solution, however, identifying these replicas poses a significant challenge. We can employ DL-based approach for identification of replicas. The key idea is to input the two payloads into the deep learning algorithm and its task is to classify the two payloads as replicas or not. After associating the replicas using DL algorithm. Besides designing a DL architecture for payload association task, considerate time has been spent on feature extraction.

- As a traditional DNN does not directly operate on complex numbers, therefore, both payloads are split into their real and imaginary parts,

which then serve as inputs to the DNN. The simplest approach doesn't supersede the performance of the traditional correlator-based approach.

- To achieve better results, a new feature is introduced, i.e., the correlation between the payloads. Now, the input of the DNN is composed of real and imaginary parts of the payloads, and the correlation between the payloads. The DL approach shows a slight improvement gain in comparison to the first approach, however, it doesn't exceed the performance of the correlator-based approach.

- In previous approaches, all the possible combinations of payloads are considered for payload association. However, replicas transmitted by the same user must be separated by an integer multiple of the packet size and must lie within a VF. Given this fact, only those packet pairs are considered which are positioned N_S slots before and after the transmitted packet. A DNN model is trained and evaluated on dataset, revealing that the DL algorithm consistently falls short of surpassing the performance achieved by the correlator-based approach.

- To assess the implementation of the algorithm, a simple scenario is considered where payloads encountered no interference with other packets. Under these ideal conditions, the DNN demonstrated superior performance compared to cases where packets are transmitted fully or partially by other packets. Despite this improvement, it did not exceed the performance achieved by the correlator-based approach.

- In all the previously discussed scenarios, the DNN consistently falls short of outperforming the correlation-based approach. In an attempt to explore alternative avenues, sequence modeling is considered. Long short-term memory (LSTM) model is considered for this purpose, where the real and imaginary parts of both payloads are input symbol by symbol, treating the sequence of symbols as the temporal dimension. However, even with this sequence modeling approach, the LSTM fails to surpass the performance achieved by the traditional correlation-based method.

In conclusion, the performance of DL algorithms is compromised due to the interference caused by packets transmitted by other users. For the packet association problem, the correlator-based approach should be preferred as it is computationally less expensive than the DL models.

3.7 Conclusion

In this chapter, a CNN architecture is proposed to detect the preamble in an asynchronous GF random access uplink scenario with no power control. The proposed deep learning model employs convolutional layers, which not only reduce the computational complexity but also extract the features shared between the antennas. The results, obtained for several values of the SNR and number of antennas, show that the CNN achieves better performance when compared to a classical solution based on the correlation, at the price of an increase in complexity. Furthermore, a DL-based approach is investigated for payload association. While the proposed preamble detection using DL demonstrates extremely good results, the additional task of payload association falls short of expectations.

References

[1] M. Z. Chowdhury, M. Shahjalal, S. Ahmed, and Y. M. Jang, "6G wireless communication systems: Applications, requirements, technologies, challenges, and research directions," *IEEE Open Journal of the Communications Society*, vol. 1, pp. 957–975, July 2020.

[2] Z. Qadir, K. N. Le, N. Saeed, and H. S. Munawar, "Towards 6G internet of things: Recent advances, use cases, and open challenges," *ICT Express*, vol. 9, pp. 296–312, June 2023.

[3] L. Miuccio, D. Panno, and S. Riolo, "A DNN-based estimate of the PRACH traffic load for massive IoT scenarios in 5G networks and beyond," *Computer Networks*, vol. 201, p. 108608, Dec. 2021.

[4] C. Bockelmann, N. K. Pratas, G. Wunder, S. Saur, M. Navarro, D. Gregoratti, G. Vivier, E. De Carvalho, Y. Ji, C. Stefanović, P. Popovski, Q. Wang, M. Schellmann, E. Kosmatos, P. Demestichas, M. Raceala-Motoc, P. Jung, S. Stanczak, and A. Dekorsy, "Towards massive connectivity support for scalable mMTC communications in 5G networks," *IEEE Access*, vol. 6, pp. 28969–28992, May 2018.

[5] N. Abramson, "The ALOHA system: Another alternative for computer communications," in *Proc. of the Fall Joint Computer Conf.*, p. 281–285, Nov. 1970.

[6] E. Casini, R. De Gaudenzi, and O. Del Rio Herrero, "Contention resolution diversity slotted aloha (CRDSA): An enhanced random access scheme for satellite access packet networks," *IEEE Trans. Wireless Commun.*, vol. 6, pp. 1408–1419, Apr. 2007.

[7] G. Liva, "Graph-based analysis and optimization of contention resolution diversity slotted ALOHA," *IEEE Trans. Commun.*, vol. 59, pp. 477–487, Feb. 2011.

[8] E. Paolini, G. Liva, and M. Chiani, "Coded slotted ALOHA: A graph-based method for uncoordinated multiple access," *IEEE Trans. Inf. Theory*, vol. 61, pp. 6815–6832, Dec. 2015.

[9] M. Doudou, D. Djenouri, and N. Badache, "Survey on latency issues of asynchronous MAC protocols in delay-sensitive wireless sensor networks," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 2, pp. 528–550, 2013.

[10] W. Zhang, J. Li, X. Zhang, and S. Zhou, "A joint user activity detection and channel estimation scheme for packet-asynchronous grant-free access," *IEEE Wireless Commun. Lett.*, vol. 11, pp. 338–342, Feb. 2022.

[11] M. Hasan, E. Hossain, and D. Niyato, "Random access for machine-to-machine communication in LTE-advanced networks: issues and approaches," *IEEE Commun. Mag.*, vol. 51, pp. 86–93, June 2013.

[12] J. Choi, J. Ding, N.-P. Le, and Z. Ding, "Grant-free random access in machine-type communication: Approaches and challenges," *IEEE Wirel. Commun.*, vol. 29, pp. 151–158, Feb. 2022.

[13] S. Kim, J. Kim, and D. Hong, "A new non-orthogonal transceiver for asynchronous grant-free transmission systems," *IEEE Trans. Wireless Commun.*, vol. 20, pp. 1889–1902, Mar. 2021.

[14] L. Liu and W. Yu, "Massive connectivity with massive MIMO—part i: Device activity detection and channel estimation," *IEEE Trans. Signal Process.*, vol. 66, pp. 2933–2946, June 2018.

[15] Z. Zhang, Y. Chi, Q. Guo, Y. Li, G. Song, and C. Huang, "Asynchronous grant-free random access: Receiver design with partially uni-directional message passing and interference suppression analysis," 2023.

[16] R. B. D. Renna and R. C. de Lamare, "Dynamic message scheduling based on activity-aware residual belief propagation for asynchronous mMTC," *IEEE Wireless Commun. Lett.*, vol. 10, pp. 1290–1294, June 2021.

[17] J. Kim, J. On, S. Kim, and J. Lee, "Performance evaluation of synchronous and asynchronous MAC protocols for wireless sensor networks," in *Proc. 2nd Int. Conf. Sens. Device Technol. Appl*, (Cap Esterel, France), pp. 500–506, IEEE, Aug. 2008.

[18] F. Clazzer and A. Munari, "IoT via satellite: Asynchronous random access for the maritime channel," in *Proc. IEEE 91st Veh. Technol. Conf.*, (Antwerp, Belgium), May 2020.

[19] R. De Gaudenzi, O. del Río Herrero, G. Acar, and E. Garrido Barrabés, "Asynchronous contention resolution diversity ALOHA: Making CRDSA truly asynchronous," *IEEE Trans. Wireless Commun.*, vol. 13, pp. 6193–6206, Nov. 2014.

[20] P. Li, Y. He, G. Cui, J. He, and W. Wang, "Asynchronous cooperative Aloha for multi-receiver satellite communication networks," *IEEE Commun. Lett.*, vol. 21, pp. 1321–1324, June 2017.

[21] F. Clazzer, F. Lazaro, G. Liva, and M. Marchese, "Detection and combining techniques for asynchronous random access with time diversity," in *Proc. 11th Int. ITG Conf. on Syst., Commun. and Coding*, (Hamburg, Germany), Feb. 2017.

[22] J. Yang, A. Wang, N. Ye, Y. Liu, and H. Xu, "Simplified random access design for satellite internet of things with NOMA," in *Proc. Int. Wireless Commun. and Mobile Comput.*, pp. 128–132, June 2021.

[23] E. Recayte, A. Munari, and F. Clazzer, "Grant-free access: Machine learning for detection of short packets," in *Proc. 10th Adv. Satellite Multimedia Syst. Conf. and 16th Signal Process. for Space Commun. Workshop*, (Graz, Austria), Oct. 2020.

[24] J. H. I. de Souza and T. Abrão, "Deep learning-based activity detection for grant-free random access," *IEEE Syst. J.*, vol. 17, pp. 940–951, Mar. 2023.

[25] D. Magrin, C. Pielli, Č. Stefanović, and M. Zorzi, "Enabling LTE RACH collision multiplicity detection via machine learning," in *Proc. Int. Symp. Model. Optim. Mobile Ad Hoc Wireless Netw.*, (Avignon, France), June 2019.

[26] S. Riolo, D. Panno, and L. Miuccio, "Modeling and analysis of tagged preamble transmissions in random access procedure for mMTC scenarios," *IEEE Trans. Wireless Commun.*, vol. 20, pp. 4296–4312, July 2021.

[27] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[28] X. Wu, *Performance evaluation, prediction and visualization of parallel systems*, vol. 4. Springer Science & Business Media, 1999.

[29] W. Kim, Y. Ahn, and B. Shim, "Deep neural network-based active user detection for grant-free NOMA systems," *IEEE Trans. Commun.*, vol. 68, pp. 2143–2155, Apr. 2020.

[30] M. U. Khan, E. Paolini, and M. Chiani, "Enumeration and identification of active users for grant-free NOMA using deep neural networks," *IEEE Access*, vol. 10, pp. 125616–125625, Nov. 2022.

Chapter 4

Joint Power Control and Pilot Assignment in Cell-Free Massive MIMO using Deep Learning

4.1 Introduction

As the demands for device connectivity and mobile data traffic escalate, strategic BS densification within the network emerges as a pivotal solution [1]. Network densification can be achieved in two ways: adding more BSs for local spectrum reuse or implementing massive multiple-input multiple-output (mMIMO) to reduce interference [2, 3]. The first architecture that combines the advantages of both approaches was proposed in [4], known as CF-mMIMO. The CF-mMIMO can achieve the objectives of 6G by improving the (usually poor) quality of service for the users at the edge of the cell and reducing inter-cell interference [5]. In such a distributed mMIMO-based network, a large number of service antennas, called APs, serve a group of users distributed in a wide area. Unlike conventional cellular networks, this approach discards the concept of cells and cell boundaries entirely. The CF-mMIMO approach relies on seamless cooperation among numerous APs,

all operating within the same time-frequency resource using time-division duplexing (TDD). At the center of the network is the central processing unit (CPU) through which the cooperation between the APs takes place. The APs are connected to the CPU through a fronthaul connection.

With reference to mMTC services, a typical MMA problem arises in the uplink, in which a myriad of devices physically located in the same area contend to transmit their packets, consisting of a pilot and data payload, over the radio access network. Due to limited coherence time intervals and a very large number of devices, assigning orthogonal pilots to every user becomes impractical [3]. Consequently, we are compelled to reuse pilots, inadvertently giving rise to the pilot contamination phenomenon, which deteriorates the quality of channel estimation [6]. Additionally, the challenge of inter-user interference emerges, demanding innovative solutions, e.g., using advanced power control mechanisms. However, the mMIMO system benefits from channel hardening, i.e., the effect of small-scale fading becomes negligible at the receiver due to the presence of multiple antennas. This allows optimization of the power coefficients based on the large-scale fading coefficients instead of small-scale fading which requires frequent updates [7].

Strategic power control and careful pilot assignment are pivotal in mitigating inter-user interference and enhancing network performance. The most diffused power control strategies focus on maximizing the minimum user rate to ensure uniform service quality regardless of the spatial user distribution [4, 8, 9]. The max-min problem for power/pilot assignment can be solved through optimization [10, 11, 12]. However, the high computational burden inherent in optimization algorithms poses substantial challenges in terms of meeting stringent time constraints, making classical optimization techniques impractical. Leveraging the universal function approximation capability of artificial neural networks (ANNs), DL-based methodologies emerge as an innovative solution yielding high performance while simultaneously reducing the computational complexity, compared to traditional optimization algorithms [13, 14]. The only drawback of DNNs is that they need extensive training to achieve operational efficiency; however, the training is usually performed offline [15].

4.1.1 Related Works

Power Control

The advantages offered by DL algorithms have catalyzed a significant body of work in various domains [16, 17, 18, 19**?**], and power control in the CF-mMIMO scenario is no exception. The approaches for power control through DL in the literature can be divided into two: supervised and unsupervised. In supervised learning, a crucial requirement is the availability of output labels (specifically, the power coefficients of each user) for both training and testing the network. The output labels are generated using optimization algorithms, requiring significant execution time. A supervised LSTM network and a DNN for power control is proposed in [5] and [20], respectively, feeding the position of the users as input to the learning model. In [13], a DNN that takes the large-scale fading coefficients as input and produces the optimized power coefficients and the total power budget as output is proposed.

Conversely, in unsupervised learning, no prior knowledge of output labels is required. This reduces the time to generate the datasets but necessitates the development of a problem-tailored loss function. In [21, 22], a specific loss function that maximizes the minimum user rate is proposed. In [23], the model complexity is reduced by feeding aggregated large-scale fading coefficients to the DNN, rather than individual ones. Notably, the DNN with the proposed loss function achieves better performance than the optimization algorithm in [4]. In [24], a soft max-min problem is proposed. Most of the above-mentioned strategies presume that mutually orthogonal pilots are assigned to the users. Yet, in massive access scenarios, this approach becomes impractical due to the limited coherence intervals.

Pilot Assignment

A substantial body of research has explored pilot assignment strategies, with random pilot assignment emerging as the most widely recognized approach [25]. A repulsive clustering-based method for the pilot assignment in CF-mMIMO is proposed in [26]. Graph coloring-based pilot assignment is

presented in [27]. The study in [6] is focused on forming groups of the users and APs to reduce pilot contamination. A supervised learning-based approach that maps the users' location to a pilot sequence is presented in [28]. A multi-agent reinforcement learning-based approach for pilot assignment is proposed in [3].

Power Control and Pilot Assignment

Only few works in the literature address both power control and pilot assignment simultaneously. In [4], a greedy iterative algorithm is proposed that assigns a different pilot to the user having the minimum user rate while solving the power control problem via bisection. Mai et al. designed pilots and formulated optimization problems for joint pilot power and data power control [29]. In [12], a pilot assignment strategy is proposed focusing on AUD, and then they developed a power control scheme for coexisted human-type communication (HTC) and machine-type communication (MTC) traffic. In [30], a deep reinforcement learning (DRL)-based approach for joint power and pilot assignment is presented. The authors perform clustering of the users and then assign pilots and allocate power using DRL, to increase AUD performance.

4.1.2 Main Contributions

The literature predominantly emphasizes either power control or pilot assignment tasks, with limited attention given to both optimizations. Even in cases where both optimizations are addressed, most studies rely on time-inefficient optimization algorithms. Although there are works exploring DRL for joint power control and pilot assignmnet, they tend to concentrate on AUD rather than enhancing the SE, leaving room for more comprehensive and efficient methodologies in this domain. Thus, the main contributions of the chapter are summarised.

- A DNN for joint pilot and data power control and pilot assignment (JPDCPA) is designed that maximizes the minimum user rate in a

CF-mMIMO network. To the best of my knowledge, the problem of jointly controlling pilot and data transmit powers and assigning pilot to each user in the network user has not been tackled yet.

- A massive access scenario is considered consisting of a large number of users to which a much smaller number of orthogonal pilots must be assigned. The DNN-based approach is designed to be scalable and to deal with large cell-free networks.

- The proposed model is validated via extensive simulation, providing a comparison with state-of-the-art methods. Moreover, an in-depth analysis of the average transmit power per device is performed in the CF-mMIMO network after optimization.

- The versatility and adaptability of the proposed approach is demonstrated by assessing its performance in a UMa and an indoor industrial scenarios [31].

The rest of the chapter is organized as follows. In Section 4.2, the system model is presented and in Section 4.3, signal-to-interference-plus-noise ratio (SINR) analysis is presented. The problem for maximizing the minimum SINR is formulated in Section 4.4. The DL-based approach for joint pilot and data power control and pilot assignment in CF-mMIMO is described in Section 4.5. The computational complexity analysis of the proposed approach is analyzed in Section 4.6. Simulation setup along with the numerical results is provided in Section 4.7. Conclusions are drawn in Section 4.8.

Matrices, vectors, and scalars are represented by boldface uppercase, boldface lowercase, and lowercase letters, respectively. The fields of real and complex numbers are denoted by $\mathbb{R}$ and $\mathbb{C}$, respectively. The operations $(\cdot)^T$, $(\cdot)^*$, and $(\cdot)^H$ denote the transpose, conjugate, and conjugate transpose, respectively. The expectation and euclidean norm operators are defined as $\mathbb{E}[\cdot]$ and $\|\cdot\|^2$, respectively. The normal and circularly-symmetric complex normal distributions with mean 0 and variance σ^2 are denoted by $\mathcal{N}(0, \sigma^2)$ and $\mathcal{CN}(0, \sigma^2)$, respectively.

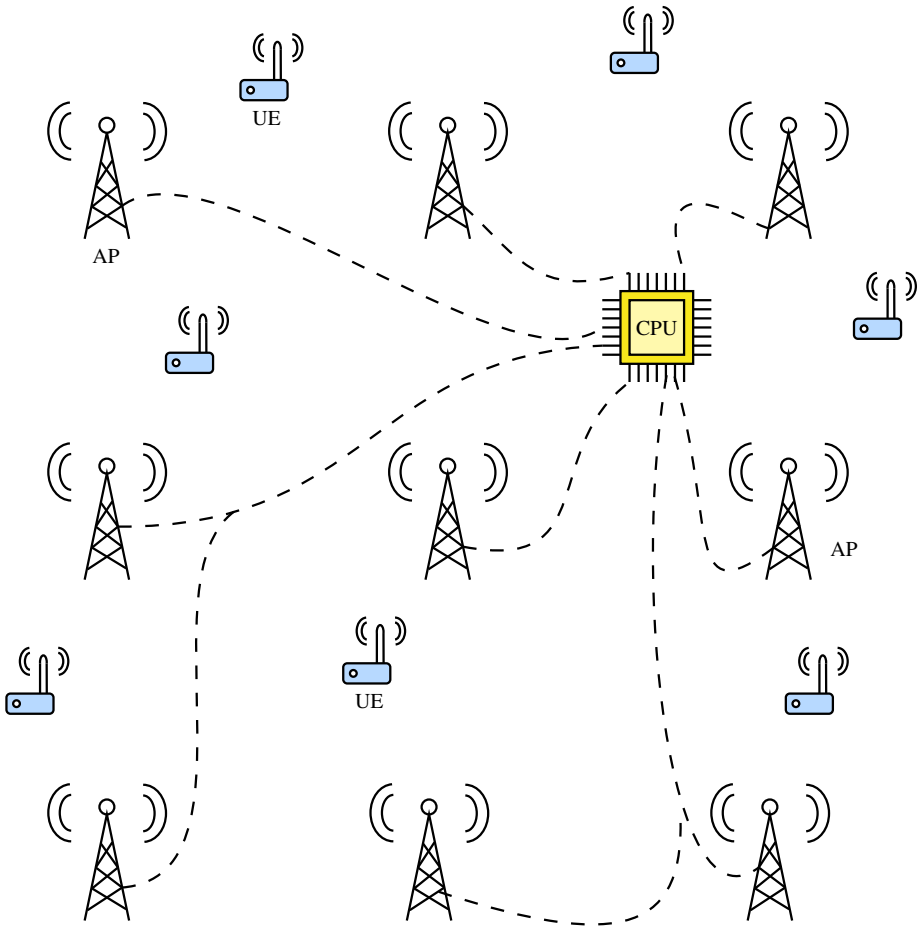


Figure 4.1: Cell-Free massive MIMO scenario.

4.2 System Model

A CF-mMIMO system is considered with M single-antenna APs arranged on a grid and K users randomly deployed in an area measuring $D \times D$ m^2, as illustrated in Fig. 4.1. The APs are connected to a CPU through fronthaul links. The number of mutually orthogonal pilot sequences P is far less than the number of users K, i.e., $P \ll K$. The set of available orthogonal pilot sequences is denoted by $\mathcal{P} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_P\}$. The large-scale fading coefficients between every user and AP are assumed to be known at the CPU whenever necessary, as in [13, 21, 23]. The APs serve all the users in the same time-frequency resources and each channel coherence interval is divided into

downlink and uplink phases, such that the system operates in TDD. During the downlink phase, the CPU communicates the optimized power coefficients and pilot assignments to the users of the network through the APs.

4.2.1 Uplink Transmission

The uplink transmission consists of two phases: pilot transmission and data transmission.

Pilot Transmission

In this phase, all K users synchronously transmit the pilot they have been assigned to via optimization. The vector of received symbols at AP m, $\boldsymbol{y}_m \in \mathbb{C}^{\tau \times 1}$, is

$$\boldsymbol{y}_m = \sqrt{\tau \rho_{\mathrm{p}}} \sum_{k=1}^{K} \sqrt{b_k} g_{mk} \boldsymbol{\phi}_k + \boldsymbol{w}_m \tag{4.1}$$

where ρ_{p} is the normalized pilot SNR, $\boldsymbol{\phi}_k \in \mathbb{C}^{\tau \times 1}$ are the pilot symbols transmitted by user k with $\|\boldsymbol{\phi}_k\|^2 = 1$, τ is the pilot sequence length, $b_k \in [0, 1]$ is the pilot power control coefficient, g_{mk} is the channel coefficient between the kth user and the mth AP, $\boldsymbol{w}_m \in \mathbb{C}^{\tau \times 1}$ is additive noise and its elements are i.i.d. $\mathcal{CN}(0, 1)$ random variables (r.v.s). The normalized pilot transmit SNR is defined as

$$\rho_{\mathrm{p}} = \frac{\bar{\rho}_{\mathrm{p}}}{\sigma_n^2} \tag{4.2}$$

where $\sigma_n^2 = B k_B T_0 \mathrm{NF}$, $\bar{\rho}_{\mathrm{p}}$ is the pilot transmit power, B is the bandwidth, k_B is the Boltzmann constant, T_0 is the equivalent noise temperature, and NF denotes the noise figure of the receiver. The channel coefficient between user k and AP m is modeled as

$$g_{mk} = \sqrt{\beta_{mk}} h_{mk} \tag{4.3}$$

where β_{mk} is the large-scale fading incorporating both path-loss and log-normal shadowing, and $h_{mk} \sim \mathcal{CN}(0,1)$ is the small-scale fading. The large-scale fading coefficient is

$$\beta_{mk} = \mathrm{PL}_{mk} 10^{\sigma_{\mathrm{sh}} s_{mk}/10} \tag{4.4}$$

where PL_{mk} is the path-loss from the kth user to the mth AP, σ_{sh} is the shadowing intensity, and $s_{mk} \sim \mathcal{N}(0,1)$. The mth AP estimates the channel associated with the kth user by projecting $\boldsymbol{y}_m$ along $\boldsymbol{\phi}_k$, as

$$\begin{aligned} \tilde{y}_{mk} &= \boldsymbol{\phi}_k^H \boldsymbol{y}_m \\ &= \sqrt{\tau \rho_p} \left(\sqrt{b_k} g_{mk} + \sum_{k' \neq k}^{K} \sqrt{b_{k'}} g_{mk'} \boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'} \right) + \boldsymbol{\phi}_k^H \boldsymbol{w}_{p,m} \end{aligned} \tag{4.5}$$

such that the minimum mean square error (MMSE) channel estimate is

$$\begin{aligned} \hat{g}_{mk} &= \frac{\mathbb{E}[\tilde{y}_{mk}^* \, g_{mk}]}{\mathbb{E}[\, |\tilde{y}_{mk}|^2]} \tilde{y}_{mk} \\ &= c_{mk} \tilde{y}_{mk} \end{aligned} \tag{4.6}$$

where c_{mk} is defined as

$$c_{mk} = \frac{\sqrt{\tau \rho_{\mathrm{p}} b_k} \beta_{mk}}{\tau \rho_{\mathrm{p}} \sum_{k'=1}^{K} b_{k'} \beta_{mk'} |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 + 1} \, . \tag{4.7}$$

Note that the quality of the channel estimate depends on the pilots assigned to all the users of the network.

Uplink Data Transmission

The uplink phase is then concluded with the simultaneous transmission of the data payload of all the users. The generic received signal sample at the

mth AP can be written as

$$z_m = \sqrt{\rho} \sum_{k=1}^{K} \sqrt{q_k} g_{mk} x_k + \nu_m \tag{4.8}$$

where ρ is the normalized data transmit SNR, $q_k \in [0, 1]$ is the power control coefficient of user k, x_k is the data payload symbol transmitted by user k with $\mathbb{E}[\,|x_k|^2] = 1$, $\nu_m \sim \mathcal{CN}(0, 1)$ is additive noise. The normalized data transmit SNR is defined as

$$\rho = \frac{\bar{\rho}}{\sigma_{\mathrm{n}}^2} \tag{4.9}$$

where $\bar{\rho}$ is the maximum data transmit power.

For the detection of the symbols transmitted by the kth user each APs processes the received signal by multiplying it with the complex conjugate of the locally derived channel estimate [4]. The resultant quantity is then forwarded to the CPU through a fronthaul link to perform joint detection. The aggregated received signal at the CPU is

$$\begin{aligned} r_k &= \sum_{m=1}^{M} \hat{g}_{mk}^* z_m \\ &= \sqrt{\rho} \sum_{k'=1}^{K} \sum_{m=1}^{M} \sqrt{q_{k'}}\, \hat{g}_{mk}^* g_{mk'} x_{k'} + \sum_{m=1}^{M} \hat{g}_{mk}^* \nu_m \,. \end{aligned} \tag{4.10}$$

4.3 SINR Analysis

In this section, a closed-form expression is obtained for the uplink achievable rate using the formulation introduced in [4, 11]. A key distinction between the proposed method and the approach outlined in [4, 11] pertains to pilot power allocation. The two approaches assume equal pilot power allocation for each user, while in this approach pilot power control is also performed. The derivation of the achievable rate expression assumes that each user possesses knowledge of channel statistics but not specific channel realizations. The

received signal r_k can be expressed as

$$r_k = \mathrm{DS}_k x_k + \mathrm{BU}_k x_k + \sum_{k' \neq k}^{K} \mathrm{IUI}_{kk'} x_k + \mathrm{TN}_k \tag{4.11}$$

where

$$\mathrm{DS}_k \triangleq \sqrt{\rho}\, \mathbb{E}\left\{ \sum_{m=1}^{M} \sqrt{q_k} g_{mk} \hat{g}_{mk}^{*} \right\}, \tag{4.12}$$

$$\mathrm{BU}_k \triangleq \sqrt{\rho} \left(\sum_{m=1}^{M} \sqrt{q}_k g_{mk} \hat{g}_{mk}^{*} - \mathbb{E}\left\{ \sum_{m=1}^{M} \sqrt{q}_k g_{mk} \hat{g}_{mk}^{*} \right\} \right) \tag{4.13}$$

$$\mathrm{IUI}_{kk'} \triangleq \sqrt{\rho} \sum_{m=1}^{M} \sqrt{q_{k'}} g_{mk'} \hat{g}_{mk}^{*} \tag{4.14}$$

$$\mathrm{TN}_k \triangleq \sum_{m=1}^{M} \hat{g}_{mk}^{*} \nu_m. \tag{4.15}$$

The terms DS_k, BU_k, $\mathrm{IUI}_{kk'}$, and TN_k denote the desired signal for the kth user, the uncertainty in beamforming gain for the kth user, the inter-user interference introduced by the k'th user, and the total noise, respectively. The first term of (4.11) demonstrates no correlation with the second, third, and fourth terms, i.e., the desired signal and effective noise terms are uncorrelated. By considering uncorrelated Gaussian noise as a worst-case scenario, the achievable SINR of the received signal of the kth user can be expressed as

$$\mathrm{SINR}_k = \frac{|\mathrm{DS}|_k^2}{\mathbb{E}\{|\mathrm{BU}_k|^2\} + \sum_{k' \neq k}^{K} \mathbb{E}\{|\mathrm{IUI}_{kk'}|^2\} + \mathbb{E}\{|\mathrm{TN}_k|^2\}}. \tag{4.16}$$

After simplifying the expression in (4.12), (4.13), (4.14), (4.15) and substituting in (4.16), the final expression of achievable uplink rate of the kth users is obtained in (4.19), shown on the top of the next page. The detailed calculations are outlined in the appendix B.

$$\mathrm{SINR}_k = \frac{q_k(\sum_{m=1}^{M}\gamma_{mk})^2}{\sum_{k'\neq k}^{K} q_{k'}(\sum_{m=1}^{M}\gamma_{mk}\frac{\sqrt{b_{k'}}\beta_{mk'}}{\sqrt{b_k}\beta_{mk}})^2|\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}|^2 + \sum_{k'=1}^{K} q_{k'}\sum_{m=1}^{M}\gamma_{mk}\beta_{mk'} + \frac{1}{\rho}\sum_{m=1}^{M}\gamma_{mk}}. \tag{4.19}$$

4.4 Problem Formulation

In [4], the authors have demonstrated that uniform and good-quality service can be ensured to all the users of a CF-mMIMO system via max-min power control. Let's define the kth user uplink throughput rate as

$$R_k = \log_2(1 + \mathrm{SINR}_k) \tag{4.17}$$

where the SINR of user k is given by (4.19) and $\gamma_{mk} = \sqrt{\tau\rho_{\mathrm{p}}b_k}\beta_{mk}c_{mk}$ [4]. The max-min power control aims to maximize the minimum user uplink throughput rate so that all the network users can experience good service quality. In this work, the number of orthogonal pilots is assumed to be considerably smaller than the total number of users, so orthogonal pilot reuse becomes necessary. Adding such a constraint, the max-min problem can be formulated as

$$\begin{aligned} &\max_{b_k,q_k,\boldsymbol{\phi}_k} \min_k R_k \\ \text{s.t. } &0 \leq b_k \leq 1, \quad k = 1,2,\ldots,K, \\ &0 \leq q_k \leq 1, \quad k = 1,2,\ldots,K\,. \end{aligned} \tag{4.18}$$

4.5 Deep Learning-based Approach

In this section, a DNN is introduced that is designed to solve the problem in (4.18), allocating pilot and data power coefficients b_k and q_k, respectively,

and simultaneously assigning pilot sequences $\boldsymbol{\phi}_k$. The proposed algorithm maximizes the minimum user rate based on the large-scale fading experienced by the users. Based on assumptions taken in other approaches in the literature [13, 21, 23], this approach also assumes that the large-scale fading coefficients between every user and AP are known at the CPU. Hereafter, the data pre-processing strategy, the architecture of the proposed DL model, and the loss function used for the unsupervised training are described.

4.5.1 Pre-processing

To make unsupervised training effective and decrease the input layer size, the data is pre-processed by aggregating the large-scale fading coefficients and applying proper normalization. The data pre-processing procedure is detailed in the following.

1. **Large-scale fading coefficients aggregation:** The large-scale fading coefficients are aggregated related to user k as

$$\beta_k^{(j)} = \sum_{i=1}^{M} \beta_{ik}^{(j)} \tag{4.20}$$

   where the superscript (j) refers to the jth sample of the dataset. This operation reduces the input layer size of the DL model, making the architecture scalable and suitable for large networks [23].

2. **Logarithmic scale conversion and scaling:** Then z-score normalization is performed of the aggregated fading coefficients in logarithmic scale, $\xi_k^{(j)} = \log_{10}(\beta_k^{(j)})$, such that the normalized data are zero mean and have standard deviation one, i.e.,

$$\eta_k^{(j)} = \frac{\xi_k^{(j)} - \bar{\mu}}{\bar{\sigma}} \tag{4.21}$$

   where $\bar{\mu}$ and $\bar{\sigma}$ are the sample mean and sample standard deviation of $\xi_k^{(j)}$, respectively, computed over all the network users and the training samples. Note that the test dataset is normalized using $\bar{\mu}$ and $\bar{\sigma}$

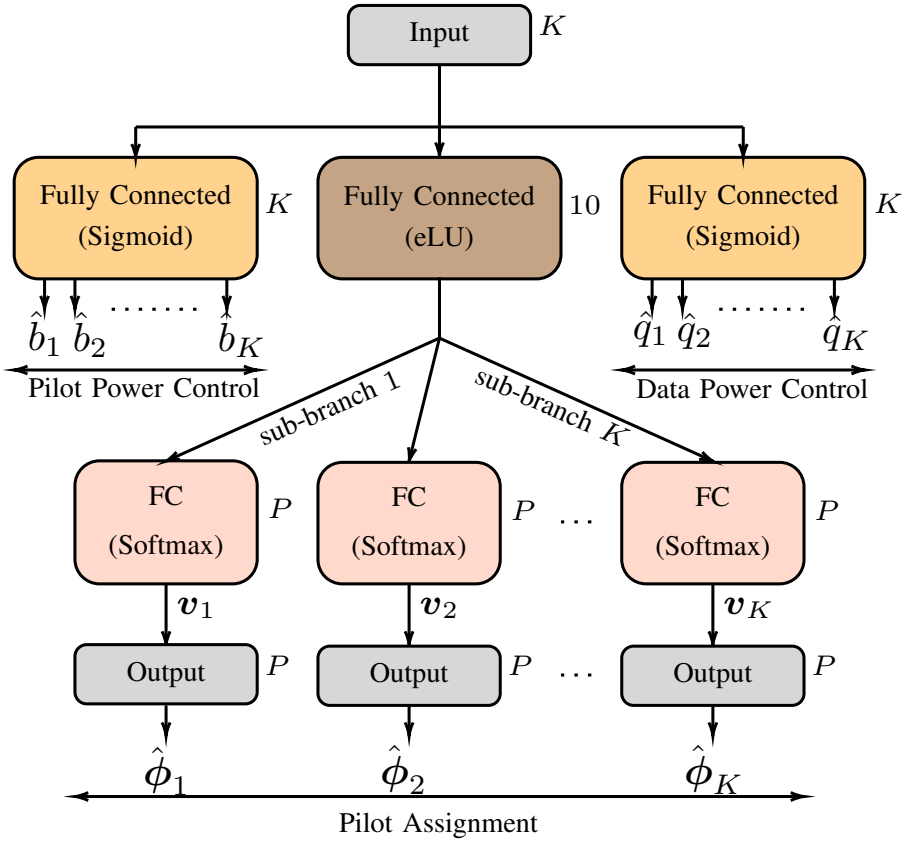


Figure 4.2: Model layout of JPDCPA for power control and pilot assignment.

calculated over the training dataset.

3. **Normalization:** L^2-normalization is applied to each sample of the dataset, such that the kth normalized feature of the jth sample is

$$\chi_k^{(j)} = \frac{\eta_k^{(j)}}{\sqrt{\sum_{i=1}^{K} (\eta_i^{(j)})^2}}. \tag{4.22}$$

4.5.2 Architecture

In this subsection, the proposed DNN architecture is presented for joint pilot and data power control, and pilot assignment, which is referred as JPDCPA. Various architectures were explored by varying the numbers, types, and sizes of layers, aiming to identify an architecture that strikes a good balance be-

tween complexity and performance. The input of the DNN is the vector of aggregated and pre-processed large-scale fading coefficients obtained from (4.22), while the outputs are the pilot and data power control coefficients, and the pilot indexes for each network user. The architecture comprises three branches, each of which performs one task as shown in Fig. 4.2. The pilot assignment branch is organized into K sub-branches, each responsible for the allocation of a pilot to a specific user. The DL model consists of fully connected layers with multiple neurons, defined in (1.3). The number of neurons employed in each fully connected layer is specified in Fig. 4.2. For instance, the number of neurons in the fully connected layer of the pilot power control branch is equal to K. Different non-linear activation functions have been used for the network layers depending on their purpose. The activation functions are applied element-wise to the layer input vectors, whose ith element is generically denoted by a_i. The sigmoid function is used in the pilot and data power control branches such that the optimized pilot (or data) power coefficient for user i is

$$\hat{b}_i(\text{or } \hat{q}_i) = \sigma(a_i) = \frac{1}{1 + e^{-a_i}}. \tag{4.23}$$

The eLU function is employed for the hidden layer of the pilot assignment branch, defined as

$$\text{eLU}(a_i) = \begin{cases} \Gamma \cdot (e^{a_i} - 1), & \text{if } a_i < 0, \\ a_i, & \text{otherwise} \end{cases} \tag{4.24}$$

where Γ determines the function saturation point for the negative input values. The softmax function is used in all the output layers of the pilot assignment branch. In particular, the kth sub-branch activation is calculated as

$$v_{ki} = \text{softmax}(a_{ki}) = \frac{e^{a_{ki}}}{\sum_{i=1}^{P} e^{a_{ki}}} \tag{4.25}$$

where v_{ki} represents the ith element of vector $\boldsymbol{v}_k \in \mathbb{R}^{P\times 1}$. The softmax values are then mapped to the pilots as

$$w = \underset{i\in\{1,2,\ldots,P\}}{\arg\max}\ v_{ki} \tag{4.26a}$$

$$\hat{\boldsymbol{\phi}}_k = \boldsymbol{p}_w \tag{4.26b}$$

where $\boldsymbol{p}_w$ is the wth pilot sequence.

4.5.3 Loss Function

Since optimal power coefficients and pilot assignment schemes are unknown, unsupervised training is performed for the DNN employing the loss function

$$\begin{aligned}\mathcal{L}(\hat{\boldsymbol{b}}, \hat{\boldsymbol{q}}, \boldsymbol{\Theta}) = \frac{\lambda_1}{K}\sum_{k=1}^{K}\sigma\left(\frac{0.3}{R_k}\right) - \lambda_2 R_{\min} + \frac{\lambda_3}{K}\sum_{k=1}^{K}\hat{q}_k \\ + \frac{\lambda_4}{K}\sum_{k=1}^{K}\hat{b}_k + \frac{2\lambda_5}{K^2-K}\sum_{i=1}^{K}\sum_{j=1}^{i-1}\frac{\Theta_{ij}}{e^{\Omega_{ij}}}\end{aligned} \tag{4.27}$$

where λ_1, λ_2, λ_3, λ_4, and λ_5 are the weights associated with the loss terms. Furthermore, the vectors $\hat{\boldsymbol{b}}$ and $\hat{\boldsymbol{q}}$ represent the pilot and data power coefficients assigned to the users. The first two terms of the loss function lay the foundation of this approach, solving the max-min problem as outlined in [23]. However, unlike [23], the joint pilot and data power control and pilot assignment optimization performance is further enhanced by incorporating three additional terms into the loss function. The average of the assigned data power coefficients, $\frac{1}{K}\sum_{k=1}^{K}\hat{q}_k$, penalizes the allocation of high data power coefficients to the users during the training process, reducing the overall network transmit power and potentially increasing the nodes battery life. Similarly, the average of the assigned pilot power coefficients, $\frac{1}{K}\sum_{k=1}^{K}\hat{b}_k$ penalizes the network to reduce the pilot transmit power. The last term of (4.27) promotes the reuse of pilots among users that are far apart in the network area, limiting inter-user interference. The penalty becomes significant when nearby users utilize the same pilot, while it is minimized when

users, whether in close proximity or at a significant distance, are allocated distinct pilots. The matrix $\boldsymbol{\Theta} = \{\Theta_{ij}\}$ is defined as $\boldsymbol{\Theta} = \boldsymbol{V}\boldsymbol{V}^T$, where $\boldsymbol{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_K]^T \in \mathbb{R}^{K \times P}$. The normalized distance between the ith and the jth users is denoted by Ω_{ij} and is obtained by dividing the actual distance by $D\sqrt{2}$. For computing the average, the last term is divided by the number of entries below the main diagonal. Note that the distances between the nodes and the APs are exclusively used for neural network training. After the training phase, only the small-scale fading coefficients are necessary for performing power control and pilot assignment.

4.6 Computational Complexity

This section performs the complexity analysis of the proposed approach JPDCPA, alongside joint power control and pilot assignment (JPCPA) and deep learning power control (DLPC). In [23], authors introduced a DNN for the assignment of data power coefficients in the CF-mMIMO, which we called DLPC. The JPCPA follows the same architecture as in Fig. 4.2 excluding the pilot power control branch. The next section delves into more details about the DLPC and JPCPA algorithms and evaluates the performance of these algorithms.

The computational complexity is evaluated in terms of FLOPs. The real addition, subtraction, and multiplication, are taken as a single FLOP while division and exponential operations as 4 and 8 FLOPs, respectively. The number of FLOPs in a fully-connected layer, without considering the activation function, can be denoted as

$$C_{\text{FC}} = 2 \cdot \text{in} \cdot \text{out}\,. \tag{4.28}$$

4.6.1 JPDCPA

Pilot Power Control

The computational complexity of the fully-connected layer in terms of the FLOPs can be given as

$$C_{\mathrm{FC}}^{\mathrm{PP}} = 2K^2 \,.$$

The sigmoid function in (4.23) results in 13 FLOP. However, considering the need to compute the sigmoid function K times, the total FLOPs becomes $13K$. The total computational complexity of the pilot power control branch is given as

$$C_{\mathrm{tot}}^{\mathrm{PP}} = 2K^2 + 13K \,.$$

Data Power Control

The computational complexity of the fully-connected layer in terms of the FLOPs can be written as

$$C_{\mathrm{FC}}^{\mathrm{DP}} = 2K^2 \,.$$

The sigmoid function in (4.23) requires 13 FLOPs. However, due to the necessity of evaluating this function K times, the total FLOPs amount to $13K$. The overall computational complexity of the data power control branch is expressed as

$$C_{\mathrm{tot}}^{\mathrm{DP}} = 2K^2 + 13K \,.$$

Pilot Assignment

The computational complexity of the fully-connected layer in terms of the FLOPs can be expressed as

$$C_{\mathrm{FC}_1}^{\mathrm{PA}} = 20K \,.$$

The first fully-connected layer of the pilot assignment branch utilizes the eLU activation function, specified in (4.24). For $a_i < 0$, the computational cost is 10 FLOPs, whereas for $a_i \geq 0$, it involves 1 FLOP. Considering the worst-case scenario, 10 FLOPs are considered for a single eLU operation. With the eLU operation iterated 10 times due to the layer's size, the overall complexity of the activation layer is expressed as follows

$$C_{\text{eLU}}^{\text{PA}} = 10 \cdot 10 = 100\,.$$

The pilot assignment branch is divided into K sub-branches. The cumulative computational complexity of all fully-connected layers is expressed as

$$C_{\text{FC}_2}^{\text{PA}} = 20PK\,.$$

The softmax layer consists of P exponential operation, $P-1$ summations, and P divisions. Following the softmax operation, we need to determine the maximum value in $\boldsymbol{v}_k$, which involves $P-1$ FLOPs. These two operations are iterated K times, corresponding to the number of sub-branches, contributing to the overall computational complexity

$$\begin{aligned} C_{\text{softmax}}^{\text{PA}} &= [(8P + P - 1 + 4P) + (P - 1)]K \\ &= [14P - 2]K \\ &= 14PK - 2K\,. \end{aligned}$$

The total computational complexity of the pilot assignment branch is given as

$$\begin{aligned} C_{\text{tot}}^{\text{PA}} &= C_{\text{FC}_1}^{\text{PA}} + C_{\text{eLU}}^{\text{PA}} + C_{\text{FC}_2}^{\text{PA}} + C_{\text{softmax}}^{\text{PA}} \\ &= 20K + 100 + 20PK + 14PK - 2K \\ &= 34PK + 18K + 100\,. \end{aligned}$$

Total Computational Complexity

The total computational complexity of the JPDCPA is given as

$$\begin{aligned} C_{\text{tot}} &= C_{\text{tot}}^{\text{PP}} + C_{\text{tot}}^{\text{PD}} + C_{\text{tot}}^{\text{PA}} \\ &= 2K^2 + 13K + 2K^2 + 13K + 34PK + 18K + 100 \\ &= 4K^2 + 44K + 34PK + 100\,. \end{aligned}$$

4.6.2 JPCPA

The JPCPA follows the same architecture as the JPDCPA, with the omission of the pilot power control branch. Thus, the computational complexity is given as

$$\begin{aligned} C_{\text{tot}} &= C_{\text{tot}}^{\text{PD}} + C_{\text{tot}}^{\text{PA}} \\ &= 2K^2 + 13K + 34PK + 18K + 100 \\ &= 2K^2 + 31K + 34PK + 100\,. \end{aligned}$$

4.6.3 DLPC

In [23], DLPC is presented, which consists of one input layer, two hidden layers, and one output layer. The ReLU activation function is employed for the input and hidden layers, and sigmoid for the output layer. The ReLU activation contributes 1 FLOP. The computational complexity of the input layer is detailed below

$$\begin{aligned} C_{\text{FC}_{\text{in}}} &= 256K \\ C_{\text{ReLU}_{\text{in}}} &= 128\,. \end{aligned}$$

The computational complexity in terms of FLOPs for the first hidden layer is expressed as

$$\begin{aligned} C_{\text{FC}_{\text{H1}}} &= 2 \cdot 128 \cdot 256 = 65536 \\ C_{\text{ReLU}_{\text{H1}}} &= 256\,. \end{aligned}$$

For the second hidden, the computational complexity in terms of FLOPs can be expressed as

$$\begin{aligned} C_{\mathrm{FC_{H2}}} &= 2 \cdot 128 \cdot 256 = 65536 \\ C_{\mathrm{ReLU_{H2}}} &= 128\,. \end{aligned}$$

The output layer comprises a fully-connected layer utilizing sigmoid activation, with the computational complexity specified as

$$\begin{aligned} C_{\mathrm{FC_{out}}} &= 2 \cdot 128 \cdot K = 256K \\ C_{\mathrm{sigmoid}} &= 13K\,. \end{aligned}$$

The total computational complexity of DLPC is given as

$$\begin{aligned} C_{\mathrm{tot}} &= C_{\mathrm{FC_{in}}} + C_{\mathrm{FC_{H1}}} + C_{\mathrm{ReLU_{H1}}} + C_{\mathrm{FC_{H2}}} + C_{\mathrm{ReLU_{H2}}} + C_{\mathrm{FC_{out}}} + C_{\mathrm{sigmoid}} \\ &= 256K + 128 + 65536 + 256 + 65536 + 128 + 256K + 13K \\ &= 525K + 131584\,. \end{aligned}$$

4.7 Numerical Results

This section describes the simulation setup and numerical results for the DL-based approach.

4.7.1 Simulation Setup

Simulations are conducted for two distinct scenarios: a UMa scenario and an industrial one. Both scenarios have identical simulation area sizes and nodes and AP distributions. The network area is a square of side $D = 1000\,\mathrm{m}$. For each sample of the dataset, the positions of the users are generated randomly and a different realization of the large-scale fading coefficients is considered. For both scenarios, two simulation settings are considered with different numbers of APs, users, and orthogonal pilot sequences, namely: 1. $M = 64$, $K = 250$, $P = 24$, and 2. $M = 121$, $K = 500$, $P = 48$. The

channel models adopted in the two scenarios are detailed below.

UMa scenario

Hata-COST231 propagation model is adopted as in [4], with path-loss

$$\mathrm{PL}_{mk}[\mathrm{dB}] = \begin{cases} -L - 35\log_{10} d_{mk}^{\mathrm{UM}}, \text{ if } d_{mk}^{\mathrm{UM}} > d_1 \\ -L - 15\log_{10} d_1 - 20\log_{10} d_{mk}^{\mathrm{UM}}, \text{ if } d_0 < d_{mk}^{\mathrm{UM}} \leq d_1 \\ -L - 15\log_{10} d_1 - 20\log_{10} d_0, \text{if } d_{mk}^{\mathrm{UM}} < d_0 \end{cases} \tag{4.29}$$

where

$$\begin{aligned} L = {} & 46.3 + 33.9\log_{10} f^{\mathrm{UM}} - 13.82\log_{10} h_{AP} \\ & - (1.1\log_{10} f^{\mathrm{UM}} - 0.7)h_u + (1.56\log_{10} f^{\mathrm{UM}} - 0.8) \end{aligned}$$

and where f^{UM} is the carrier frequency in MHz, d_{mk}^{UM} is the distance between the kth device and mth AP in kilometers, d_0 and d_1 are the threshold distances associated with the path-loss model in kilometers, h_{AP} is the AP antenna height in meters, and h_u is the device antenna height in meters.

Industrial Scenario

An indoor industrial scenario is considered with the following path-loss model [31]

$$\mathrm{PL}_{mk}[\mathrm{dB}] = -32.40 - 23\log_{10} d_{mk}^{\mathrm{IN}} - 20\log_{10} f^{\mathrm{IN}}\,. \tag{4.30}$$

where f^{IN} is the carrier frequency in GHz and d_{mk}^{IN} is the distance between the mth AP and kth user in meters. The complete list of simulation parameters is reported in Table 4.1.

A set consisting of 5×10^4 and 10^3 different samples are used for training and testing the DNN model, respectively. The network is initialized with He initialization [32]. The network is trained using mini-batches of 100 samples

Table 4.1: Simulation parameters

Parameters	Value
Carrier frequency (f^{UM}, f^{IN})	1.9 GHz
Shadowing coefficient $\sigma_{\mathrm{sh}}^{\mathrm{UM}}, \sigma_{\mathrm{sh}}^{\mathrm{IN}}$	8, 5.9
Height of AP antenna (h_{AP})	15 m
Height of user antenna (h_u)	1.65 m
Path-loss model d_0, d_1	10 m, 50 m
Transmit power ($\bar{\rho}_{\mathrm{p}}$, $\bar{\rho}$)	100 mW
Bandwidth (B)	20 MHz
Noise figure (NF)	9 dB
Noise temperature (T_0)	290 K
Length of a pilot sequence (τ)	24, 48
Packet length (τ_c)	400

for 30 epochs adopting the ADAM optimizer [33]. The initial learning rate δ is set to 0.01, which is updated after each epoch using $\delta_i = \delta_{i-1}e^{-0.1}$, where i represents the ith epoch. The weights associated with the loss function are $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \lambda_5 = 0.2$. The eLU saturation parameter Γ is set to 0.2.

4.7.2 Performance Evaluation

The performance metrics considered in this section are per-user network uplink throughput rate and minimum user rate, defined as [4]

$$R_k^{\mathrm{net}} = \frac{\tau_d}{\tau_c} B R_k \tag{4.31}$$

and

$$R_k^{\mathrm{min}} = \frac{\tau_d}{\tau_c} \min_k R_k \tag{4.32}$$

respectively, where $\tau_{\mathrm{d}} = \tau_c - \tau$ is the data payload length. Three algorithms are compared to the proposed approach: random access (RA), JPCPA, and DLPC. In the RA algorithm, equal power is allocated to all users, ensuring that the total transmit power matches with the proposed DL approach and

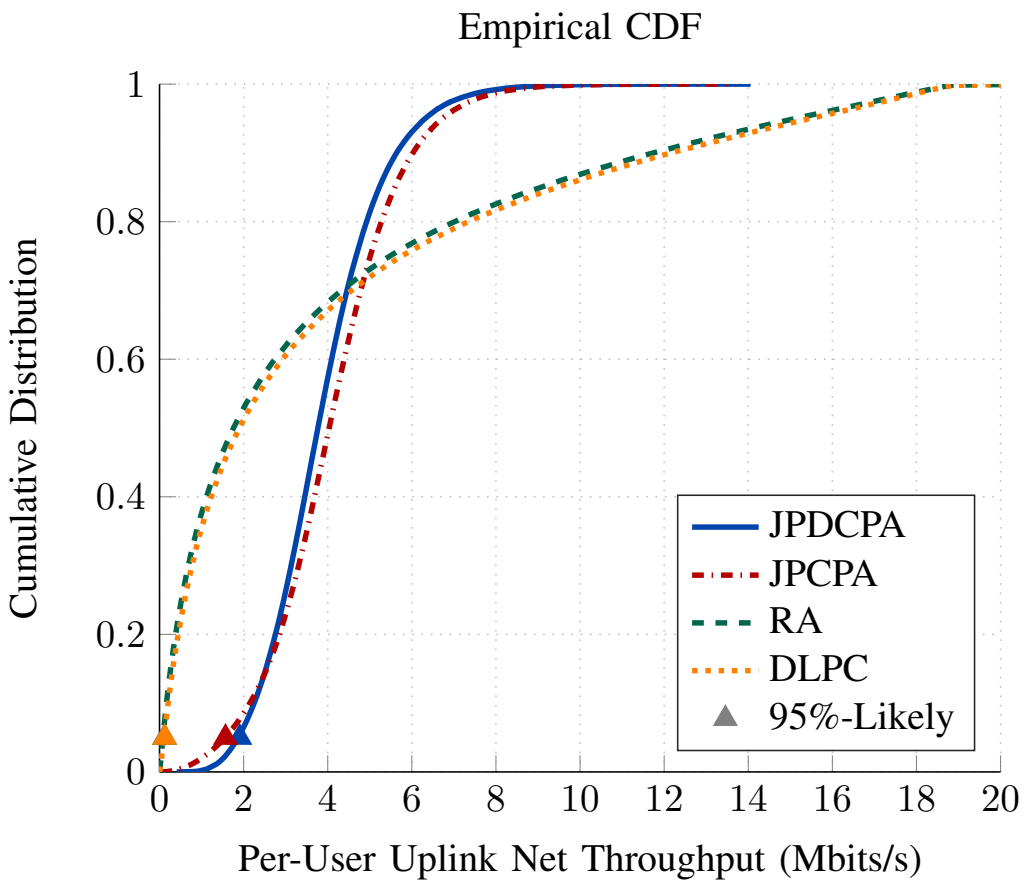


Figure 4.3: Cumulative distribution of per-user net throughput for $M = 64$, $K = 250$, $P = 24$ in an UMa scenario.

assigns pilots randomly. For instance, if JPDCPA distributes 10 mW among 10 users, RA algorithm assigns 1 mW to each user. In JPCPA, the focus is on optimizing data power allocation and pilot assignment. For JPCPA, the architecture depicted in Fig. 4.2 is employed with the omission of the pilot power allocation branch [**?**]. The network is trained for 30 epochs utilizing the loss function in (4.27) with $\lambda_1 = \lambda_2 = \lambda_4 = \lambda_5 = 0.25$ and $\lambda_3 = 0$. In DLPC, data power coefficients are obtained through the DNN proposed in [23] and the pilot sequences are randomly assigned to the users.

The empirical cumulative distribution functions (CDFs) for the per-user uplink throughput rate $M = 64$ and $M = 121$ is depicted in Fig. 4.3 and Fig. 4.4, respectively. It is evident that JPDCPA significantly outperforms

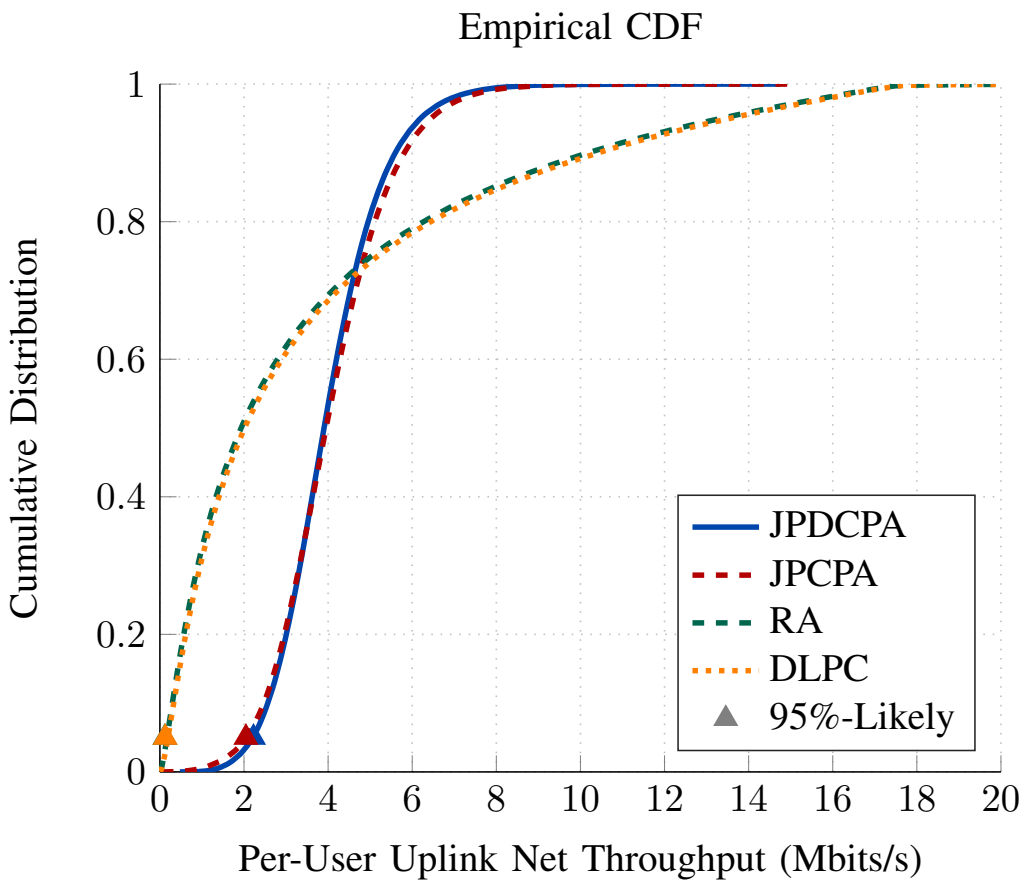


Figure 4.4: Cumulative distribution of per-user net throughput for $M = 121$, $K = 500$, $P = 48$ in an UMa scenario.

the RA and DLPC approaches. Specifically, DLPC lacks scalability, resulting in performance equivalent to RA. In the simulation setting with $M = 64$ as depicted in Fig. 4.3, the proposed approach has a significantly higher 95%-likely per-user net uplink throughput rate of 1.91 Mbits/s. In contrast, RA and DLPC attain only 0.07 Mbits/s and 0.13 Mbits/s, respectively. Furthermore, JPDCPA aligns with the trajectory of JPCPA, exhibiting a slight improvement in the lower region of the curve in both scenarios. For the simulation setting with $M = 64$ depicted in Fig. 4.3, JPDCPA observe an increase of 20% in the 95%-likely per-user net throughput with respect to JPCPA. Similarly, 8% increment in 95%-likely per-user net throughput with respect to JPCPA can be observed in Fig. 4.4. A performance increment

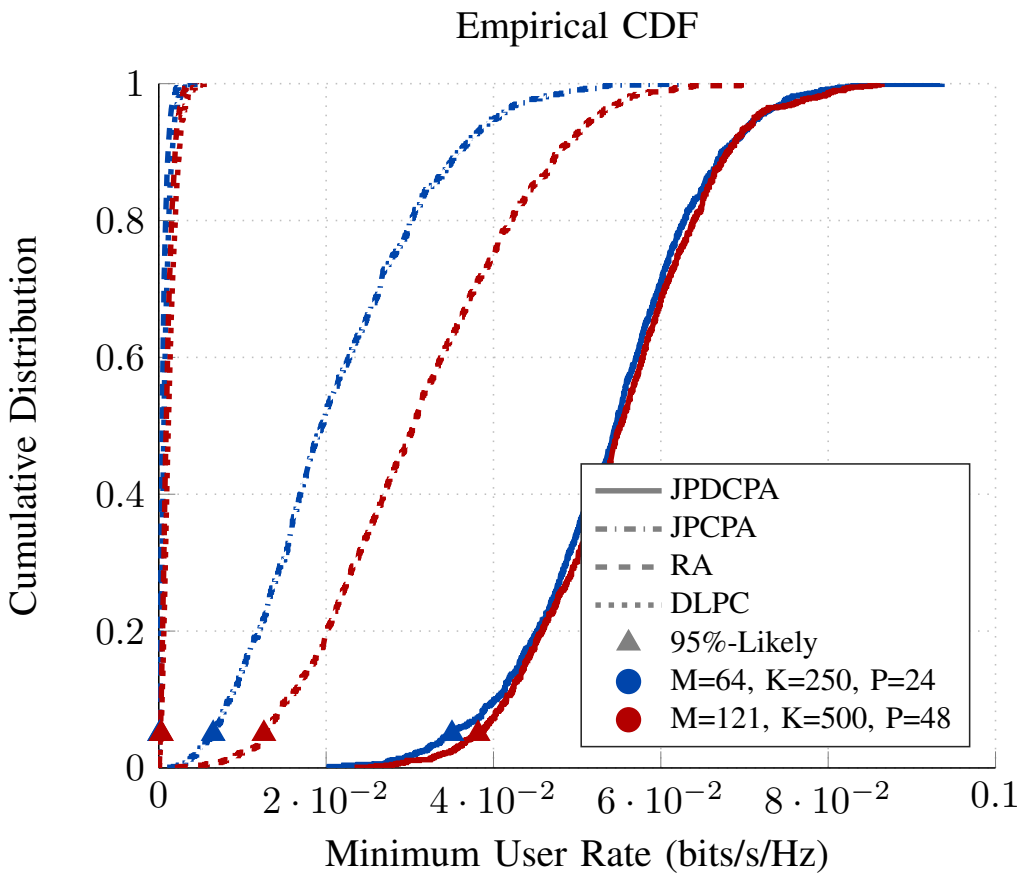


Figure 4.5: Cumulative distribution of minimum user rate in an UMa scenario.

for JPDCPA can also be observed in terms of 95%-likely and median from the scenario with $M = 64$ to $M = 121$, also highlighting the impact of the increased number of APs in the UMa scenario. Precisely, 95%-likely and median increases by 18% and 3.7%, respectively.

The minimum user rate depicted in Fig. 4.5 holds greater significance than the per-user throughput rate shown in Fig. 4.3 and Fig. 4.4 in terms of user fairness, i.e., quality service to all users. The plot in Fig. 4.5 clearly indicates a substantial increase in the minimum user rate in JPDCPA compared to other approaches. The benchmark approach JPCPA achieves a lower median value of 0.019 bits/s/Hz for the minimum user rate on the test set compared to 0.055 bits/s/Hz through JPDCPA with $M = 64$. The median of the empirical CDF of minimum user rate exhibited a substantial

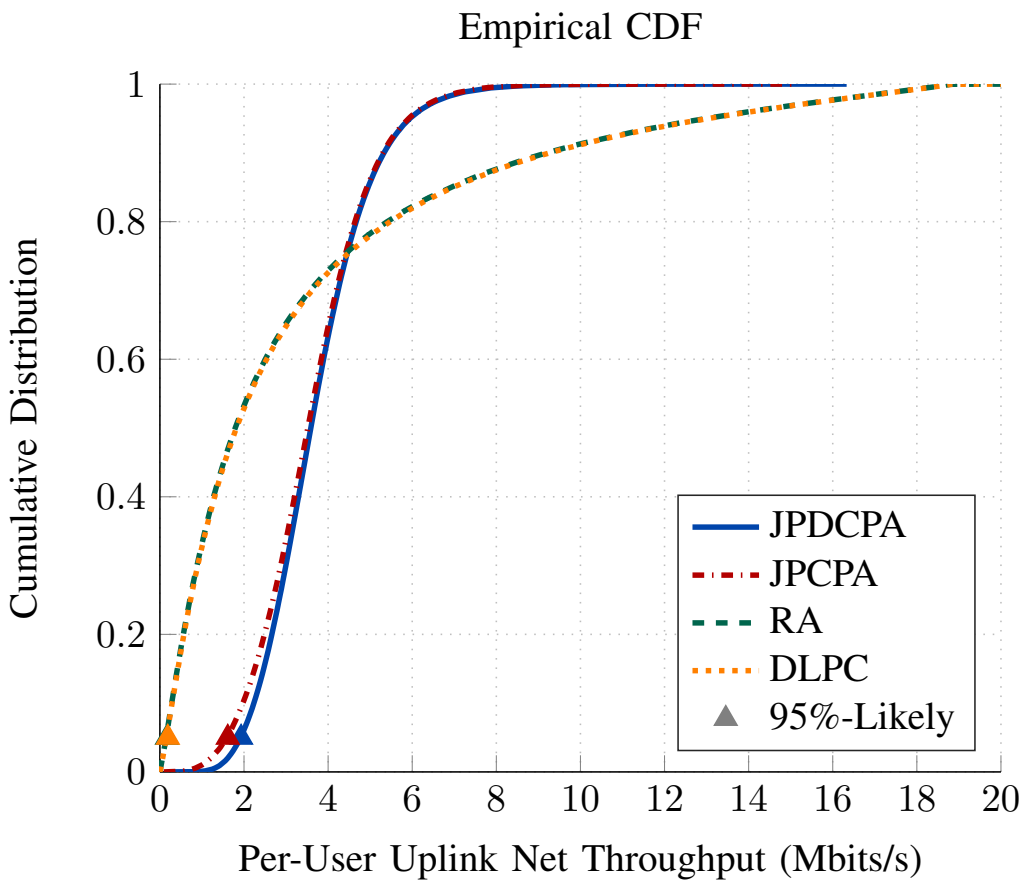


Figure 4.6: Cumulative distribution of per-user net throughput for $M = 64$, $K = 250$, $P = 24$ in an industrial scenario.

increase of approximately 56% from the simulation scenario with $M = 64$ to $M = 121$ with JPCPA. In contrast, with JPDCPA, the curves overlap, with the performance of $M = 121$ showing a marginal improvement over $M = 64$.

To demonstrate the adaptability of theJPDCPA, the proposed DNN is tested in the industrial scenario with $M = 64$ and $M = 121$. The per-user uplink throughput rates are depicted in Fig. 4.6 and 4.7, respectively. Similar to the UMa scenario, DLPC exhibits comparable performance to RA. Notably, JPDCPA exhibits 10 times better performance than DLPC and RA in terms of 95%-likely. Moreover, the median of the empirical CDF of the per-user uplink throughput rate in JPDCPA doubles that of the DLPC in

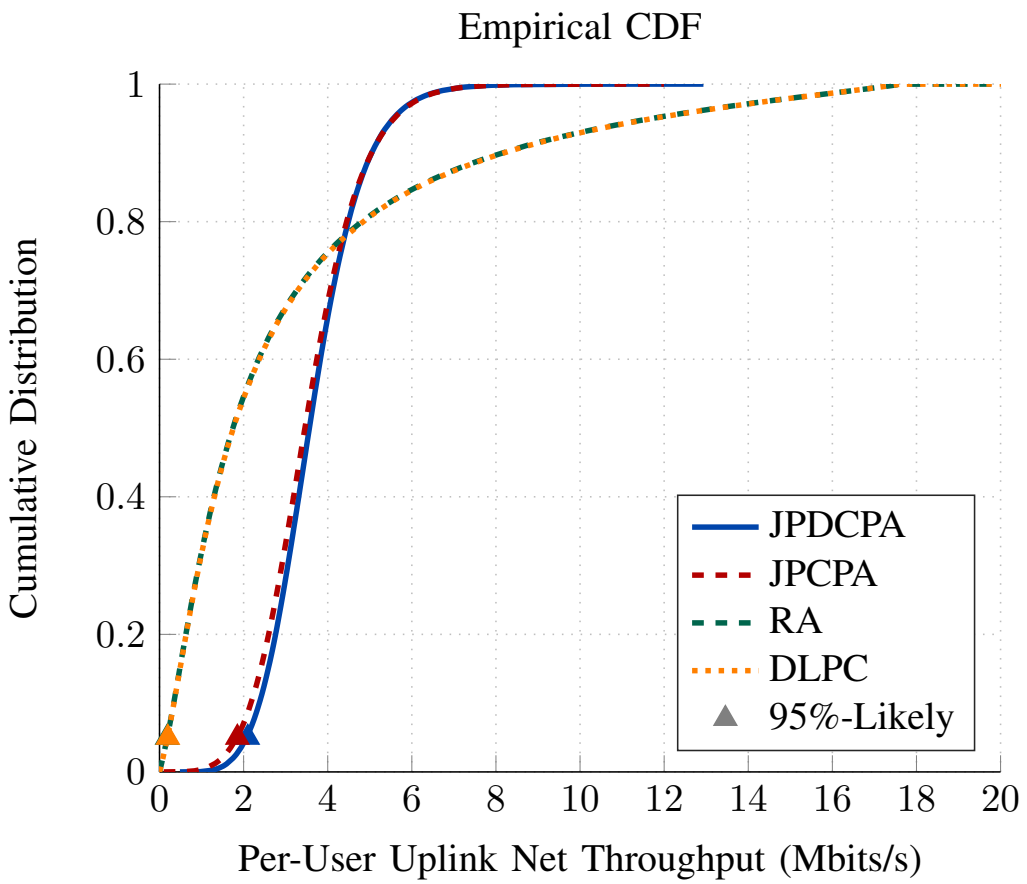


Figure 4.7: Cumulative distribution of per-user net throughput for $M = 121$, $K = 500$, $P = 48$ in an industrial scenario.

both simulation settings. The JPDCPA demonstrates a slight improvement over JPCPA in terms of both 95%-likely and median of the empirical CDF. For instance, with $M = 64$, JPCPA achieves 1.614 Mbits/s whereas JPDCPA obtains 1.920 Mbits/s 95%-likely per-user uplink throughput rate.

The minimum user rate in the industrial scenario is illustrated in Fig. 4.8. It is evident from the figure that the JPDCPA outperforms the baseline approaches considerably. Specifically, in a simulation setting with $M = 64$ and $M = 121$, JPDCPA exhibits an increase of 157% and 89% in the of 95%-likely throughput rate from the JPCPA, respectively. The performance of JPDCPA with $M = 121$ is slightly lower than that with the $M = 64$, showing an opposite behavior compared to the UMa scenario. This discrepancy arises

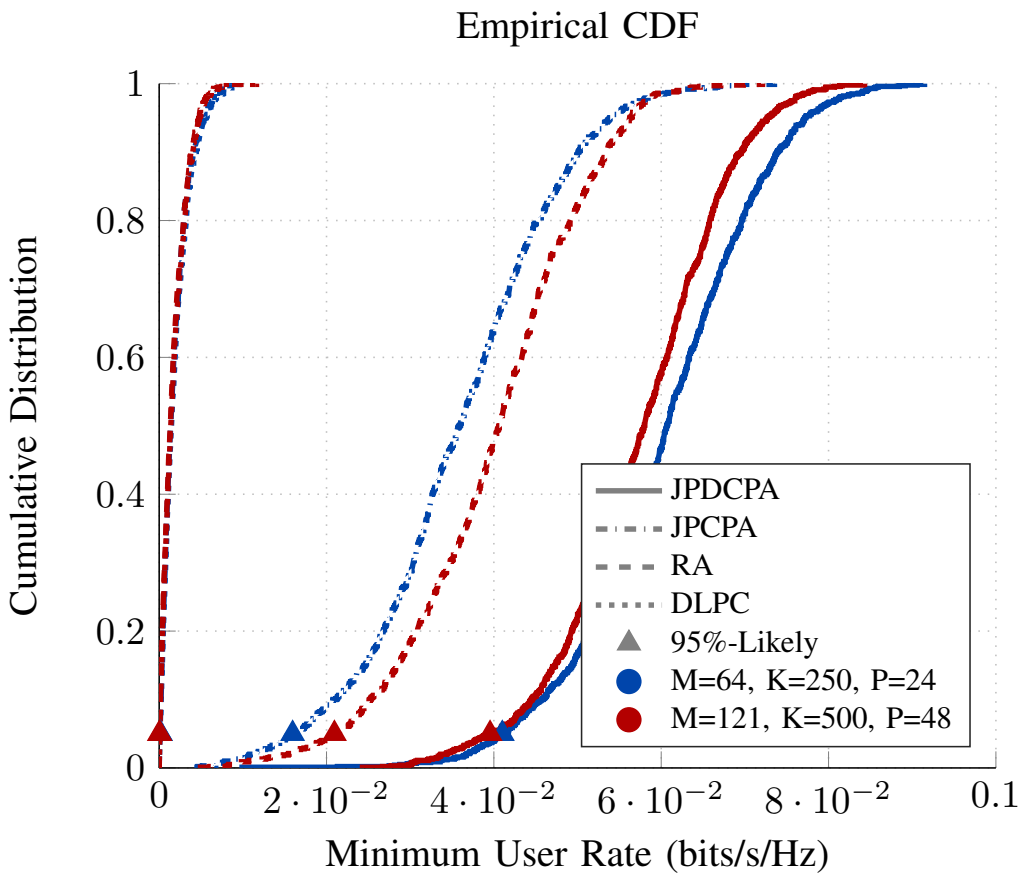


Figure 4.8: Cumulative distribution of minimum user rate in an industrial scenario.

due to the difference in the path-loss models and large-scale fading intensities between the two scenarios. In UMa scenario, the channel is more selective in space, which decreases interference and slightly boosts performance in $M = 121$ compared to $M = 64$.

4.7.3 Computational Complexity

The computational complexity of the JPDCPA, JPCPA, and DLPC is presented in Table 4.2. As the number of users increases from $K = 250$ to $K = 500$, there is a significant increase in computational cost for all the algorithms. It is evident from the table that the DLPC offers the lowest computational complexity. Nevertheless, it has been previously demonstrated

Table 4.2: Computational cost

	DLPC [23]	JPCPA	JPDCPA
$M = 64$, $K = 250$, $P = 24$	2.63×10^5	3.37×10^5	4.65×10^5
$M = 121$, $K = 500$, $P = 48$	3.94×10^5	1.33×10^6	1.84×10^6

that it results in poor performance in comparison to the other algorithms under consideration. The introduction of the pilot power control branch in JPDCPA contributes to a higher computational cost compared to JPCPA.

4.7.4 Per-User Power Usage

To demonstrate the advantage of the proposed method in terms of energy efficiency, the average per-user pilot and data transmit powers are computed as

$$P_{\mathrm{C}}^{\mathrm{p}} = \frac{\bar{\rho}_{\mathrm{p}}}{KS} \sum_{i=1}^{S} \sum_{k=1}^{K} \hat{b}_{k,i} \tag{4.33}$$

and

$$P_{\mathrm{C}} = \frac{\bar{\rho}}{KS} \sum_{i=1}^{S} \sum_{k=1}^{K} \hat{q}_{k,i} \tag{4.34}$$

respectively. Here, S represents the number of test samples. The result of this analysis is shown in Table 4.3. In both UMa and industrial scenarios, DLPC and JPCPA transmit pilots with maximum power. In comparison, the proposed loss function lowers the average pilot and data transmit power per user. Notably, DLPC shows a higher average power for transmitting data per user compared to JPCPA and JPDCPA. The inclusion of a term penalizing high power assignment in JPCPA prompts the network to allocate lower power to users. Although the average transmit power per user of JPDCPA is slightly higher than JPCPA, the overall transmit power, i.e., accounting for average pilot transmit power per user, is significantly lower than that of JPCPA.

Table 4.3: Transmit Power per User in dBm

	Pilot Transmit Power ($P_{\mathrm{C}}^{\mathrm{p}}$)			
	DLPC	JPCPA	JPDCPA	
$M = 64$, $K = 250$, $P = 24$	20.00	20.00	2.5208	Urban macro
$M = 121$, $K = 500$, $P = 48$	20.00	20.00	0.1460	
$M = 64$, $K = 250$, $P = 24$	20.00	20.00	−6.7448	Industrial
$M = 121$, $K = 500$, $P = 48$	20.00	20.00	−8.5263	
	Data Transmit Power (P_{C})			
	DLPC	JPCPA	JPDCPA	
$M = 64$, $K = 250$, $P = 24$	16.98	3.5388	3.8439	Urban macro
$M = 121$, $K = 500$, $P = 48$	16.99	1.5824	1.6411	
$M = 64$, $K = 250$, $P = 24$	16.93	−6.2342	−6.4897	Industrial
$M = 121$, $K = 500$, $P = 48$	16.96	−8.2740	−8.0632	

4.8 Conclusion

In this chapter, a scalable DNN-based solution is proposed called JPDCPA for joint pilot and data power allocation and pilot assignment in a CF-mMIMO network. A massive access scenario is considered where the number of users exceeds the available orthogonal pilots. The adaptability of JPDCPA is demonstrated by assessing its performance in a UMa and indoor industrial scenarios. Numerical results show that the JPDCPA shows an increase of 20% and 8% in the 95%-likely per-user uplink throughput rate with respect to the state-of-the-art considering $M = 64$ and $M = 121$ APs, respectively, in the UMa scenario. Furthermore, the unsupervised training of the JPDCPA using the proposed loss function not only outperforms alternative methods in enhancing users' spectral efficiency but also leads to a significant reduction in average transmit power per user compared to the considered state-of-the-art solution.

References

[1] J. Liu, M. Sheng, L. Liu, and J. Li, "Network densification in 5G: From the short-range communications perspective," *IEEE Commun. Mag.*, vol. 55, pp. 96–102, Dec. 2017.

[2] E. Björnson, J. Hoydis, and L. Sanguinetti, *Massive MIMO Networks: Spectral, Energy, and Hardware Efficiency*, vol. 11, pp. 157–655. Foundations and Trends in Signal Processing, 2017.

[3] M. Rahmani, M. J. Dehghani, P. Xiao, M. Bashar, and M. Debbah, "Multi-agent reinforcement learning-based pilot assignment for cell-free massive MIMO systems," *IEEE Access*, vol. 10, pp. 120492–120502, Nov. 2022.

[4] H. Q. Ngo, A. Ashikhmin, H. Yang, E. G. Larsson, and T. L. Marzetta, "Cell-free massive MIMO versus small cells," *IEEE Trans. Wireless Commun.*, vol. 16, pp. 1834–1850, Jan. 2017.

[5] A. Mazhari Saray and A. Ebrahimi, "Max-min power control of cell free massive MIMO system employing deep learning," in *Proc. 4th West Asian Symp. Optical Millimeter-wave Wireless Commun. (WASOWC)*, (Tabriz, Iran), May 2022.

[6] M. Sarker and A. O. Fapojuwo, "Granting massive access by adaptive pilot assignment scheme for scalable cell-free massive MIMO systems," in *IEEE 93rd Veh. Technol. Conf.*, Apr. 2021.

[7] A. Ghazanfari, H. V. Cheng, E. Björnson, and E. G. Larsson, "Enhanced fairness and scalability of power control schemes in multi-cell massive MIMO," *IEEE Trans. Commun.*, vol. 68, pp. 2878–2890, May 2020.

[8] T. Van Chien, E. Björnson, and E. G. Larsson, "Joint power allocation and user association optimization for massive MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 15, pp. 6384–6399, Sept. 2016.

[9] H. Yang and T. L. Marzetta, "Massive MIMO with max-min power control in line-of-sight propagation environment," *IEEE Trans. Commun.*, vol. 65, pp. 4685–4693, Nov. 2017.

[10] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, March 2004.

[11] M. Bashar, K. Cumanan, A. G. Burr, M. Debbah, and H. Q. Ngo, "On the uplink max–min SINR of cell-free massive MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 18, pp. 2021–2036, Apr. 2019.

[12] G. Liu, H. Deng, X. Qian, W. Zhang, and H. Dong, "Joint pilot and data power control for cell-free massive MIMO IoT systems," *IEEE Sensors J.*, vol. 22, pp. 24647–24657, Dec. 2022.

[13] M. Zaher, O. T. Demir, E. Björnson, and M. Petrova, "Learning-based downlink power allocation in cell-free massive MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 22, pp. 174–188, Jan. 2023.

[14] M. U. Khan, E. Paolini, and M. Chiani, "Enumeration and identification of active users for grant-free NOMA using deep neural networks," *IEEE Access*, vol. 10, pp. 125616–125625, Nov. 2022.

[15] Y. Zhao, I. G. Niemegeers, and S. H. De Groot, "Power allocation in cell-free massive MIMO: A deep learning method," *IEEE Access*, vol. 8, pp. 87185–87200, May 2020.

[16] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, pp. 82–97, Nov. 2012.

[17] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," in *Proc. 31st AAAI Conf. on Artificial Intelligence*, (San Francisco, California, USA), p. 4278–4284, Feb. 2017.

[18] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst. (NeurIPS)*, vol. 2, (Montreal, Canada), pp. 3104–3112, Dec. 2014.

[19] D. Mzurikwao, M. U. Khan, O. W. Samuel, J. Cinatl, M. Wass, M. Michaelis, G. Marcelli, and C. S. Ang, "Towards image-based cancer cell lines authentication using deep neural networks," *Scientific reports*, vol. 10, Nov. 2020.

[20] C. D'Andrea, A. Zappone, S. Buzzi, and M. Debbah, "Uplink power control in cell-free massive MIMO via deep learning," in *Proc. IEEE 8th Int. Workshop on Comput. Adv. in Multi-Sensor Adapt. Process. (CAMSAP)*, (Guadalupe, Mexico), pp. 554–558, Dec. 2019.

[21] N. Rajapaksha, K. B. Shashika Manosha, N. Rajatheva, and M. Latva-Aho, "Deep learning-based power control for cell-free massive MIMO networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, (Montreal, Canada), June 2021.

[22] N. Rajapaksha, K. B. S. Manosha, N. Rajatheva, and M. Latva-aho, "Unsupervised learning-based joint power control and fronthaul capacity allocation in cell-free massive mimo with hardware impairments," *IEEE Wireless Commun. Lett.*, vol. 12, pp. 1159–1163, July 2023.

[23] Y. Zhang, J. Zhang, Y. Jin, S. Buzzi, and B. Ai, "Deep learning-based power control for uplink cell-free massive MIMO systems," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, (Madrid, Spain), Dec. 2021.

[24] R. Nikbakht, A. Jonsson, and A. Lozano, "Unsupervised-learning power control for cell-free wireless systems," in *Proc. IEEE 30th Annual Int. Sym. on Personal, Indoor and Mobile Radio Commun. (PIMRC)*, (Istanbul, Turkey), Sept. 2019.

[25] H. Ahmadi, A. Farhang, N. Marchetti, and A. MacKenzie, "A game theoretic approach for pilot contamination avoidance in massive MIMO," *IEEE Wireless Commun. Lett.*, vol. 5, pp. 12–15, Feb. 2016.

[26] S. Mohebi, A. Zanella, and M. Zorzi, "Repulsive clustering based pilot assignment for cell-free massive MIMO systems," in *Proc. 30th Eur. Signal Process. Conf. (EUSIPCO)*, (Belgrade, Serbia), pp. 717–721, Aug. 2022.

[27] R. Chen, H. Wang, and R. Song, "Pilot assignment based on graph coloring with sum-rate maximization in cell-free massive MIMO," in *Proc. 7th Int. Conf. Comput. and Commun. (ICCC)*, (Chengdu, China), pp. 1890–1894, Dec. 2021.

[28] J. Li, Z. Wu, P. Zhu, D. Wang, and X. You, "Scalable pilot assignment scheme for cell-free large-scale distributed MIMO with massive access," *IEEE Access*, vol. 9, pp. 122107–122112, Sept. 2021.

[29] T. C. Mai, H. Quoc Ngo, and L.-N. Tran, "Design of pilots and power control in the cell-free massive MIMO uplink," in *Proc. 54th Asilomar Conf. on Signals, Syst., and Comput.*, (Pacific Grove, CA, USA), pp. 831–835, Nov. 2020.

[30] L. Diao, J. Li, P. Zhu, D. Wang, and X. You, "Adaptive federated learning-based joint pilot design and active user detection in scalable cell-free massive MIMO systems," in *Proc. 4th Inf. Commun. Tech. Conf. (ICTC)*, (Nanjing, China), pp. 232–236, May 2023.

[31] 3GPP, "Study on channel model for frequencies from 0.5 to 100 GHz," Technical Specification (TS) 38.901, 3rd Generation Partnership Project (3GPP), Mar. 2022. Version 17.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, (Santiago, Chile), pp. 1026–1034, Dec. 2015.

[33] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proc. 3rd Int. Conf. Learning Representations (ICLR)*, (San Diego, CA, USA), May 2015.

Chapter 5

Conclusions

The exponential growth of mIoT devices has led to the emergence of the term mMTC. These devices wake up intermittently to transmit short data packets and are usually powered by batteries. In this thesis, numerous challenges associated with integrating these devices into cellular networks through DL algorithms were addressed. It is established that relying solely on grant-based methods is inefficient due to control signaling overhead, resulting in prolonged delays. To address this, GF schemes are proposed in the literature. In GF schemes, the users transmit without prior resource allocation from the BS. This simplifies the device side by transferring the computational burden to the BS. Some of the major contributions of this thesis are presented below.

Chapter 2 delved into the implementation of a GF NOMA scheme to provide services to a large number of devices and to reduce the communication overhead in mMTC scenarios. In GF random access, BS is not aware of the devices that are trying to communicate. Consequently, AUD must be performed at the BS. In that respect, this chapter proposed the design of two DNN architectures for the AUD task, namely AUE and AUI. The former identifies the number of active devices, while the latter indicates which devices are active. The chapter analyzed the performance of the proposed algorithm, revealing that the proposed approach outperforms state-of-the-art methods and offers lower computational complexity.

Chapter 3 proposed a DL-based technique for preamble detection in an

asynchronous GF random access scenario. In the considered scenario, the active user initiates a VF consisting of many slots, where each slot is equal to the size of the packet. The user transmits two replicas in the randomly chosen slots. As a single packet contends with interference from numerous packets transmitted by other users, packet detection poses a formidable challenge in an asynchronous scenario. The study demonstrated that the proposed DL-based method provides a high detection rate for a negligible false-alarm rate and surpassed the performance of the conventional correlator-based approach. Furthermore, Chapter 3 delved into DL methodologies for payload association. As each user transmits multiple copies, the replicas can be combined to enhance the decoding process. The objective of the investigated DL methodologies was to determine the positions of the two replicas within the frame. Despite numerous attempts, achieving successful payload association has proven to be elusive.

The CF-mMIMO networks can improve the quality of service for the users at the edge of the cell and reduce inter-cell interference. Building on this premise, Chapter 4 has focused on strategies for enhancing the SE of the mIoT devices in a CF-mMIMO network, aiming to maximize the minimum user rate. Critical to this objective is the allocation of suitable power for transmitting the pilot and payload, and the assignment of pilots to each user in the network. A DNN algorithm called JPDCPA was proposed to achieve this objective. The algorithm was trained using an unsupervised learning approach and benchmarked against existing approaches in the literature. To the best of my knowledge, this is the first approach in the literature that deals with joint power and pilot allocation. The computational complexity analysis of the proposed approach is performed. The results demonstrated that the proposed algorithm outperformed the existing algorithms in terms of per-user uplink throughput rate and minimum user rate. Furthermore, it has been also shown that the proposed approach lowers the average pilot and data transmit power per user, in comparison to the other approaches.

In future generations of cellular networks, AI is poised to play a transformative role, revolutionizing various aspects of communication and connectivity. As detailed in this thesis, AI surpasses traditional algorithms in

performance, offering significant improvements in the reliability, throughput, and scalability of networks. Similarly, in the future, AI-based grant-free access protocols will catalyze innovation in emerging technologies, such as IoT, autonomous vehicles, and smart cities, thus laying the groundwork for a more interconnected and intelligent future.

Future Research Directions: Cell-Free networks offers higher SE for all the users in comparison to the Cell-based networks. This performance leap necessitates extending the activity detection algorithm to the Cell-Free networks. Furthermore, optimizing the architectures and feature selection for AUE and AUI can yield better performance and lower computational complexity.

A potential future extension of the preamble detection work could explore channel estimation through DL. While traditional methods, such as MMSE, can be effective, DL-based approaches may perform better in the considered scenario. Additionally, the proposed payload association approach can be improved by integrating transformer blocks into the architecture.

For last work on pilot allocation and pilot assignment, the work could be extended to scenarios involving APs equipped with multiple antennas. Like the other approaches in the literature, this approach also assumes that the large-scale fading coefficient between every AP and the device is known at the CPU, however, this is often not the case. A future research direction may involve estimating the values of large-scale fading coefficients, and based on these estimates, power and pilots can be assigned.

Appendix A

Derivation of the Distribution of Collisions in a Random Access for Preamble Detection

Let us start by noting that the number of arrivals at the ith symbol time is $N = N_1 + N_2 + \cdots + N_S$ where N_a is the number of arrivals at the ith symbol time considering only the users that wake up at symbol $i - aM$, and M is the number of symbols in a slot. The number of users waking up during a symbol time, w, is distributed as a Poisson r.v. with density λ. The probability mass function of N_1, recalling that a virtual frame is composed by N_S slots, is given as

$$\begin{aligned} P(N_1 = n) &= \sum_{w=n}^{\infty} P(N_1 = n|w)P(w) \\ &= \sum_{w=n}^{\infty} \binom{w}{n} \left(\frac{2}{N_S}\right)^n \left(1 - \frac{2}{N_S}\right)^{w-n} \frac{\lambda^w}{w!} e^{-\lambda} \\ &= e^{-\lambda} \sum_{w=n}^{\infty} \frac{w!}{(w-n)!n!} \left(\frac{2}{N_S}\right)^n \left(1 - \frac{2}{N_S}\right)^{w-n} \frac{\lambda^w}{w!} \\ &= e^{-\lambda} \frac{1}{n!} \left(\frac{2}{N_S}\right)^n \sum_{w=n}^{\infty} \frac{\lambda^w}{(w-n)!} \left(1 - \frac{2}{N_S}\right)^{w-n}. \end{aligned} \tag{A.1}$$

Let $z = w - n$, then we have

$$\begin{aligned} P(N_1 = n) &= e^{-\lambda}\frac{1}{n!}\left(\frac{2}{N_S}\right)^n \sum_{z=0}^{\infty}\frac{\lambda^{(z+n)}}{z!}\left(1-\frac{2}{N_S}\right)^z \\ &= e^{-\lambda}\frac{1}{n!}\left(\frac{2\lambda}{N_S}\right)^n \sum_{z=0}^{\infty}\frac{\lambda^{z}}{z!}\left(1-\frac{2}{N_S}\right)^z \\ &= e^{-\lambda}\frac{1}{n!}\left(\frac{2\lambda}{N_S}\right)^n e^{\lambda\left(1-\frac{2}{N_S}\right)} \\ &= \frac{1}{n!}\left(\frac{2\lambda}{N_S}\right)^n e^{-\frac{2\lambda}{N_S}} \sim \text{Pois}\left(\frac{2\lambda}{N_S}\right). \end{aligned} \tag{A.2}$$

Let us recall that the probability generating function of the Poisson distribution Pois(μ) is

$$\begin{aligned} G_{N_1}(s) &= \sum_{k=0}^{\infty} s^k \frac{\mu^k}{k!} e^{-\mu} \\ &= e^{-\mu}\sum_{k=0}^{\infty}\frac{(s\mu)^k}{k!} \\ &= e^{\mu(s-1)} \end{aligned} \tag{A.3}$$

that for $\mu = \frac{2\lambda}{N_S}$ becomes $G_{N_1}(s) = e^{\frac{2\lambda}{N_S}(s-1)}$. Note that $G_{N_1}(s) = G_{N_2}(s) = \cdots = G_{N_S}(s)$. Then, the probability generating function of the number of arrivals in a symbol time is calculated as

$$\begin{aligned} G_N(s) &= \left(G_{N_1}(s)\right)^{N_S} \\ &= e^{\frac{2\lambda N_S}{N_S}(s-1)} \\ &= e^{2\lambda(s-1)} \end{aligned} \tag{A.4}$$

that is the probability generating function of Pois(2λ). Thus, the arrival rate in a symbol time follows a Poisson distribution with density 2λ. Thus, the average number of packet arrivals during a slot time is $2\lambda(N_\text{P} + N_\text{D})$, where N_P and N_D are the number of preamble and data symbols, respectively. This can be interpreted also as the average number of collisions during the packet transmission time. Since the aim is to test the performance of the proposed

Table A.1: Average number of collisions in a slot time varying λ.

λ	Average n° of collisions
0.0005	1
0.005	3
0.01	5
0.0145	7

schemes with different traffic loads, the dataset is generated considering different values of λ, i.e., scenarios with different average numbers of collisions. Table A.1 shows the average number of collisions (after ceiling operation) for some of the λ values used in the preamble detection work.As can be seen from the table, the average number of collisions in a slot time ranges from a low network load condition to a much higher one. The figures presented in Table A.1 are discussed in sub-section 3.5.1.

Appendix B

Derivation of closed-form expression for the achievable uplink rate

To derive the closed-form expression for the achievable rate given in (4.16), we need to compute DS_k, $\mathbb{E}\{|\mathrm{BU}_k|^2\}$, $\mathbb{E}\{|\mathrm{IUI}_{kk'}|^2\}$, and $\mathbb{E}\{|\mathrm{TN}_k|^2\}$

Computation of DS_k

Let $\varepsilon_{mk} \triangleq g_{mk} - \hat{g}_{mk}$ be the channel estimation error. Owing to the properties of MMSE estimation, ε_{mk} and $\hat{g}_{mk}$ are independent. Thus, we have

$$\begin{aligned}\mathrm{DS}_k =& \sqrt{\rho}\mathbb{E}\left\{\sum_{m=1}^{M}\sqrt{q_k}(\hat{g}_{mk}+\varepsilon_{mk})\hat{g}_{mk}^*\right\}\\ =& \sqrt{\rho}\mathbb{E}\left\{\sum_{m=1}^{M}\sqrt{q_k}(|\hat{g}_{mk}|^2+\varepsilon_{mk}\hat{g}_{mk}^*)\right\}\end{aligned}$$

Since $\mathbb{E}|\hat{g}_{mk}|^2 = \gamma_{mk}$, $\mathbb{E}\{\varepsilon_{mk}\} = 0$, and $\mathbb{E}\{\hat{g}_{mk}^*\} = 0$, it follows that

$$\mathrm{DS}_k = \sqrt{\rho q_k}\sum_{m=1}^{M}\gamma_{mk}. \tag{B.1}$$

Computation of $\mathbb{E}|\mathrm{BU}_k|^2$

The expression can be derived as follows

$$\begin{aligned}\mathbb{E}\{|\mathrm{BU}_k|^2\} &= \rho \sum_{m=1}^{M} q_k \mathbb{E}\left\{|g_{mk}\hat{g}_{mk}^* - \mathbb{E}\left\{g_{mk}\hat{g}_{mk}^*\right\}|^2\right\} \\ &= \rho \sum_{m=1}^{M} q_k \left(\mathbb{E}\left\{|g_{mk}\hat{g}_{mk}^*|^2\right\} - |\mathbb{E}\left\{g_{mk}\hat{g}_{mk}^*\right\}|^2\right)\end{aligned}$$

where the property $\mathbb{E}\{|X - \mathbb{E}\{X\}|^2\} = \mathbb{E}\{|X|^2\} - |\mathbb{E}\{X\}|^2$ has been used. Substituting $g_{mk} = \varepsilon_{mk} + \hat{g}_{mk}$, we get

$$\mathbb{E}\{|\mathrm{BU}_k|^2\} = \rho \sum_{m=1}^{M} q_k \Big(\mathbb{E}\left\{|(\varepsilon_{mk} + \hat{g}_{mk})\hat{g}_{mk}^*|^2\right\} - |\mathbb{E}\left\{(\varepsilon_{mk} + \hat{g}_{mk})\hat{g}_{mk}^*\right\}|^2\Big)$$

Using $\mathbb{E}\{\varepsilon_{mk}\} = 0$, $\mathbb{E}\{|\hat{g}_{mk}|^2\} = \gamma_{mk}$, we obtain

$$\mathbb{E}\{|\mathrm{BU}_k|^2\} = \rho \sum_{m=1}^{M} q_k \left(\mathbb{E}\left\{|\varepsilon_{mk}\hat{g}_{mk}^*|^2\right\} + \mathbb{E}\left\{|\hat{g}_{mk}|^4\right\} - \gamma_{mk}^2\right)$$

As $\mathbb{E}\{|\varepsilon_{mk}|^2\} = \beta_{mk} - \gamma_{mk}$ and $\mathbb{E}\{|\hat{g}_{mk}|^4\} = 2\gamma_{mk}^2$, we get

$$\mathbb{E}\{|\mathrm{BU}_k|^2\} = \rho \sum_{m=1}^{M} q_k \left(\gamma_{mk}(\beta_{mk} - \gamma_{mk}) + 2\gamma_{mk}^2 - \gamma_{mk}^2\right) = \rho \sum_{m=1}^{M} q_k \gamma_{mk}\beta_{mk}. \tag{B.2}$$

Computation of $\mathbb{E}|\mathrm{IUI}_{kk'}|^2$

The term is given as

$$\mathbb{E}\{|\mathrm{IUI}_{kk'}|^2\} = \rho \mathbb{E}\left\{\left|\sum_{m=1}^{M} \hat{g}_{mk}^* g_{mk'}\sqrt{q_{k'}}\right|^2\right\}$$

Substituting $\hat{g}^*_{mk} = c_{mk}\tilde{y}^*_{mk}$, where $\tilde{y}_{mk}$ is defined in (4.5).

$$\begin{aligned}\mathbb{E}\{|\mathrm{IUI}_{kk'}|^2\} =& \rho\mathbb{E}\Bigg\{\Bigg|\sum_{m=1}^{M} c_{mk}g_{mk'}\sqrt{q_{k'}} \\ &\times\left(\sqrt{\tau\rho_p}\sum_{i=1}^{K}\sqrt{b_i}g_{mi}\boldsymbol{\phi}_k^H\boldsymbol{\phi}_i + \boldsymbol{\phi}_k^H\boldsymbol{w}_{p,m}\right)^*\Bigg|^2\Bigg\} \\ =& \underbrace{\rho q_{k'}\mathbb{E}\left\{\left|\sum_{m=1}^{M} c_{mk}g_{mk'}(\boldsymbol{\phi}_k^H\tilde{\boldsymbol{w}}_{p,m})^*\right|^2\right\}}_{A} \\ &+ \underbrace{\tau\rho_p\rho\mathbb{E}\left\{\left|\sum_{m=1}^{M}\sqrt{q_{k'}}c_{mk}g_{mk'}\left(\sum_{i=1}^{K}\sqrt{b_i}g_{mi}\boldsymbol{\phi}_k^H\boldsymbol{\phi}_i\right)^*\right|^2\right\}}_{B},\end{aligned}$$

Since $\tilde{w}_{p,m} = \boldsymbol{\phi}_k^H\boldsymbol{w}_{p,m} \sim \mathcal{CN}(0,1)$ is independent of the term g_{mk}, the term A is given as

$$A = \rho q_{k'}\sum_{m=1}^{M} c_{mk}^2\beta_{mk'}.$$

Recalling that $\mathbb{E}\{|X+Y|^2\} = \mathbb{E}\{|X|^2\} + \mathbb{E}\{|Y|^2\}$ where X and Y are two independent random variables and $\mathbb{E}\{X\} = 0$, B can be expressed as

$$\begin{aligned}B =& \underbrace{\tau\rho_p\rho\mathbb{E}\left\{\left|\sum_{m=1}^{M}\sqrt{q_{k'}}c_{mk}g_{mk'}\left(\sqrt{b'_k}g_{mk'}\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}\right)^*\right|^2\right\}}_{B1} \\ &+ \underbrace{\tau\rho_p\rho\mathbb{E}\left\{\left|\sum_{m=1}^{M}\sqrt{q_{k'}}c_{mk}g_{mk'}\left(\sum_{i\neq k'}^{K}\sqrt{b_i}g_{mi}\boldsymbol{\phi}_k^H\boldsymbol{\phi}_i\right)^*\right|\right\}}_{B2}\end{aligned} \tag{B.3}$$

Manipulating B1 we obtain

$$
\begin{aligned}
B1 &= \tau\rho_p\rho\mathbb{E}\left\{\left|\sum_{m=1}^{M}\sqrt{q_{k'}}c_{mk}g_{mk'}\left(\sqrt{b_{k'}}g_{mk'}\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}\right)^*\right|^2\right\} \\
&= \tau\rho_p\rho q_{k'}b_{k'}|\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}|^2\mathbb{E}\left\{\left|\sum_{m=1}^{M}c_{mk}|g_{mk'}|^2\right|^2\right\} \\
&= \tau\rho_p\rho q_{k'}b_{k'}|\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}|^2\mathbb{E}\left\{\sum_{m=1}^{M}c_{mk}|g_{mk'}|^2\sum_{n=1}^{M}c_{nk}|g_{nk'}|^2\right\} \\
&= \tau\rho_p\rho q_{k'}b_{k'}|\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}|^2\mathbb{E}\left\{\sum_{m=1}^{M}c_{mk}^2|g_{mk'}|^4\right\} \\
&\qquad + \tau\rho_p\rho q_{k'}b_{k'}|\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}|^2\sum_{m=1}^{M}\sum_{n\neq m}^{M}c_{mk}c_{nk}\beta_{mk'}\beta_{nk'}
\end{aligned}
$$

As $\mathbb{E}\{|g_{mk'}|^4\} = 2\beta_{mk'}^2$, we get

$$
\begin{aligned}
B1 &= \tau\rho_p\rho q_{k'}b_{k'}|\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}|^2\sum_{m=1}^{M}c_{mk}^2 2\beta_{mk'}^2 \\
&\qquad + \tau\rho_p\rho q_{k'}b_{k'}|\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}|^2\sum_{m=1}^{M}\sum_{n\neq m}^{M}c_{mk}c_{nk}\beta_{mk'}\beta_{nk'} \\
&= \tau\rho_p\rho q_{k'}b_{k'}|\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}|^2\sum_{m=1}^{M}c_{mk}^2\beta_{mk'}^2 \\
&\qquad + \tau\rho_p\rho q_{k'}b_{k'}|\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}|^2\sum_{m=1}^{M}c_{mk}^2\beta_{mk'}^2 \\
&\qquad + \tau\rho_p\rho q_{k'}b_{k'}|\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}|^2\sum_{m=1}^{M}\sum_{n\neq m}^{M}c_{mk}c_{nk}\beta_{mk'}\beta_{nk'} \\
&= \tau\rho_p\rho q_{k'}b_{k'}|\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}|^2\sum_{m=1}^{M}c_{mk}^2\beta_{mk'}^2 + \tau\rho_p\rho q_{k'}b_{k'}|\boldsymbol{\phi}_k^H\boldsymbol{\phi}_{k'}|^2
\end{aligned}
$$

$$\times \sum_{m=1}^{M} c_{mk}\beta_{mk'} \left(\sum_{n\neq m}^{M} c_{nk}\beta_{nk'} + c_{mk}\beta_{mk'} \right)$$

$$B1 = \tau\rho_p\rho q_{k'} b_{k'} |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'}^2 + \tau\rho_p\rho q_{k'} b_{k'} |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \times \sum_{m=1}^{M} c_{mk}\beta_{mk'} \left(\sum_{n=1}^{M} c_{nk}\beta_{nk'} \right)$$

$$= \tau\rho_p\rho q_{k'} b_{k'} |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'}^2 + \tau\rho_p\rho q_{k'} b_{k'} |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'}^2$$

Multiplying and dividing the second term with $\sum_{m=1}^{M} b_k \beta_{mk}^2$

$$B1 = \tau\rho_p\rho q_{k'} b_{k'} |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'}^2 + \rho q_{k'} |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \left(\sum_{m=1}^{M} \frac{\sqrt{b_{k'}}\beta_{mk'}}{\sqrt{b_k}\beta_{mk}} \gamma_{mk} \right)^2$$

Manipulating B2, we obtain

$$B2 = \tau\rho_p\rho \mathbb{E} \left\{ \left| \sum_{m=1}^{M} \sqrt{q_{k'}} c_{mk} g_{mk'} \left(\sum_{i\neq k'}^{K} \sqrt{b_i} g_{mi} \boldsymbol{\phi}_k^H \boldsymbol{\phi}_i \right)^* \right|^2 \right\}$$

$$= \tau\rho_p\rho \mathbb{E} \left\{ \sum_{m=1}^{M} \sum_{i\neq k'}^{K} q_{k'} b_i c_{mk}^2 |g_{mk'}|^2 |g_{mi}|^2 |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_i|^2 \right\}$$

As $\mathbb{E}\{|g_{mk'}|^2\} = \beta_{mk'}$

$$
\begin{aligned}
B2 &= \tau\rho_p\rho \sum_{m=1}^{M} \sum_{i\neq k'}^{K} q_{k'} b_i c_{mk}^2 \beta_{mk'} \beta_{mi} |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_i|^2 \\
&= \tau\rho_p\rho \sum_{m=1}^{M} \sum_{i=1}^{K} q_{k'} b_i c_{mk}^2 \beta_{mk'} \beta_{mi} |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_i|^2 \\
&\quad - \tau\rho_p\rho \sum_{m=1}^{M} q_{k'} b_{k'} c_{mk}^2 \beta_{mk'}^2 |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2
\end{aligned}
$$

Adding and subtracting the term $\rho q_{k'} \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'}$

$$
\begin{aligned}
B2 &= \tau\rho_p\rho \sum_{m=1}^{M} \sum_{i=1}^{K} q_{k'} b_i c_{mk}^2 \beta_{mk'} \beta_{mi} |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_i|^2 \\
&\quad - \tau\rho_p\rho \sum_{m=1}^{M} q_{k'} b_{k'} c_{mk}^2 \beta_{mk'}^2 |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \\
&\quad + \rho q_{k'} \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'} - \rho q_{k'} \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'} \\
&= \rho q_{k'} \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'} \left(\frac{\sqrt{\tau\rho_p b_k} \beta_{mk}}{c_{mk}} \right) \\
&\quad - \tau\rho_p\rho \sum_{m=1}^{M} q_{k'} b_{k'} c_{mk}^2 \beta_{mk'}^2 |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \\
&\quad - \rho q_{k'} \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'} \\
&= \sqrt{\tau\rho_p b_k} \rho q_{k'} \sum_{m=1}^{M} c_{mk} \beta_{mk} \beta_{mk'} \\
&\quad - \tau\rho_p\rho \sum_{m=1}^{M} q_{k'} b_{k'} c_{mk}^2 \beta_{mk'}^2 |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \\
&\quad - \rho q_{k'} \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'}
\end{aligned}
$$

Using $\gamma_{mk} = \sqrt{\tau\rho_p b_k} c_{mk}\beta_{mk}$

$$
\begin{aligned}
B2 = \rho q_{k'} \sum_{m=1}^{M} \gamma_{mk}\beta_{mk'} \\
- \tau\rho_p\rho \sum_{m=1}^{M} q_{k'} b_{k'} c_{mk}^2 \beta_{mk'}^2 |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \\
- \rho q_{k'} \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'}
\end{aligned}
$$

Summing A, $B1$, and $B2$, we obtain

$$
\begin{aligned}
&\mathbb{E}\{|\mathrm{IUI}_{kk'}|^2\} = A + B1 + B2 \\
&= \rho q_{k'} \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'} \\
&+ \tau\rho_p\rho b_{k'} q_{k'} |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'}^2 \\
&+ \rho q_{k'} |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \left(\sum_{m=1}^{M} \gamma_{mk} \frac{\sqrt{b_{k'}}\beta_{mk'}}{\sqrt{b_k}\beta_{mk}} \right)^2 \\
&+ \rho q_{k'} \sum_{m=1}^{M} \gamma_{mk}\beta_{mk'} \\
&- \tau\rho_p\rho \sum_{m=1}^{M} q_{k'} b_{k'} c_{mk}^2 \beta_{mk'}^2 |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \\
&- \rho q_{k'} \sum_{m=1}^{M} c_{mk}^2 \beta_{mk'}
\end{aligned}
$$

After cancellation of terms, we get

$$
\begin{aligned}
\mathbb{E}\{|\mathrm{IUI}_{kk'}|^2\} = \rho q_{k'} |\boldsymbol{\phi}_k^H \boldsymbol{\phi}_{k'}|^2 \left(\sum_{m=1}^{M} \gamma_{mk} \frac{\sqrt{b_{k'}}\beta_{mk'}}{\sqrt{b_k}\beta_{mk}} \right)^2 \\
+ \rho q_{k'} \sum_{m=1}^{M} \gamma_{mk}\beta_{mk'}
\end{aligned}
\tag{B.4}
$$

Computation of $\mathbb{E}\{|\mathrm{TN}_k|^2\}$

$$\begin{aligned}\mathbb{E}\{|\mathrm{TN}_k|^2\} &= \mathbb{E}\left\{\left|\sum_{m=1}^{M} \hat{g}_{mk}^{*}\nu_m\right|^2\right\} \\ &= \sum_{m=1}^{M} \gamma_{mk} \end{aligned} \tag{B.5}$$

Substituting (B.1), (B.2), (B.4), (B.5) in (4.16) to obtain (4.19).

Acknowledgments

I express my gratitude to Allah Almighty, the Most Gracious, and the Most Merciful for the blessings bestowed upon me throughout my academic journey and in the successful completion of my PhD thesis. Furthermore, accolades are extended to Prophet Muhammad (SAW), whose exemplary way of life has been a constant source of guidance for me.

I would like to take this opportunity to express my heartfelt appreciation to all those who played a pivotal role in my Ph.D. journey. I extend my sincere gratitude to Prof. Marco Chiani, my esteemed supervisor, for graciously welcoming me into his research group. His guidance, support, and assistance in participating in Summer schools, conferences, and publishing in proceedings and journals have been invaluable. Prof. Chiani's profound passion for research and his adeptness in simplifying complex concepts have been a continual source of inspiration for me.

I also express my thanks to Prof. Andrea Giorgetti, my co-supervisor, for his invaluable encouragement, insightful comments, and dedicated time. His support has played a crucial role, providing guidance both academically and socially throughout my doctoral journey. I would also express my appreciation to Prof. Enrico Paolini and Enrico Testi for our fruitful collaboration on multiple projects, which resulted in several publications. Thanks and appreciation to Matteo Pizzotti for offering technical assistance whenever needed.

I would like to thank Prof. Marco Chiani and Lorenzo Valentini for ensuring my smooth arrival and managing all the necessary arrangements in Italy. A special acknowledgment goes to Alberto Faedi and Lorenzo Pucci for their assistance in navigating the intricacies of Italian bureaucracy. Their

support was indispensable, making my experience much more manageable.

I am deeply appreciative of my colleagues and dear friends for the incredible time: Giuseppe Capasso, Elisabetta Matricardi, Diego Forlivesi, Leonardo Franceschelli, Marina Lotti, Giulia Torcolacci, Oumaima Afif, Sana Fatima Syeda, Marcella Lucciardi, Nicola Lowenthal, Alessandro Mirri, Luca Arcangeloni, Tommaso Bacchielli, Elena Bernardi, Elena Macrelli, Vladislav Volosov, Jacopo Ferretti, Elia Favarelli, and Maurizio Millesimo.

I extend my profound appreciation to all my family members for their unwavering encouragement and support. I would like to thank my father Nisar Ahmad Khan, my mother Anjum Nisar, my sister Ayesha Sahar, my brothers Iqbal Ahmed Khan and Muhammad Omer Khan. I am profoundly grateful to my wife, Kashaf Ad Dooja, for her unwavering support, love, care, encouragement, understanding, patience, sacrifices, and perseverance over the years. Her steadfast commitment played a pivotal role in enabling me to complete my endeavors.

I would like to express my heartfelt gratitude to my teachers and supportive friends, whose encouragement has been instrumental in my academic journey. I extend my sincere thanks to my former supervisor and mentor, Dr. Omer Saleem Bhatti, whose guidance and belief in my capabilities played a pivotal role in my life.

Finally, many thanks to the Italian government for funding my Ph.D. scholarship for 39 months.