Radio Resource Management for RIS-aided Mobile Networks using Graph Neural Networks

J.J. van Breukelen

Radio Resource Management for RIS-aided Mobile Networks using Graph Neural Networks

by

J.J. van Breukelen

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday March 26, 2025 at 11:00.

Student number: 4704967
Project duration: January 8, 2024 – March 26, 2025
Thesis committee: Dr. R. Litjens MSc, TU Delft, TNO
Dr. J. L. Cremer, TU Delft
S. Agarwal MSc, TNO

TU Delft TNO innovation for life

Preface

After a long and beautiful study time, I am pleased to present this thesis, which marks the end of my academic time in Delft. However, finalizing the thesis project was not easy. The thesis period was a long and often very stressful period. At some point in time, it felt like it would never be over. But through all the challenges, I have learned a lot about the topic, doing research, and also about myself. In the end, the result is finally there, and it is something I can be truly proud of. This is made possible by the guidance and support of many people, whom I would like to thank in this preface.

First of all, I would like to thank my supervisors, *Dr. Remco Litjens MSc* and *Sakshi Agarwal MSc*, for the huge amount of time and effort they have put into this project. They were very involved in the project, week after week, which has been invaluable. Our meetings led to many new ideas, discussions, and insights that helped to improve this work. I would also like to thank *Dr. Jochen Cremer MSc* for assessing my work as the third committee member.

I am also grateful to *TNO* for providing the opportunity to work on this interesting project in the innovative field of 6G. Their support, both in terms of facilities and financially, enabled me to do this research. Additionally, I am also thankful for the new friends I have made at TNO, which made this experience even more rewarding.

Finally, I would like to thank *my family* and *friends* for the support they gave me throughout the year. They have always been there for me, to encourage me and to provide a listening ear whenever I needed to get something off my chest, which happened many times.

Besides obtaining the title of Master of Science, I hope that this thesis can have meaning beyond just being a requirement for my degree and can contribute to future research in this field.

J.J. van Breukelen
Delft, March 19, 2025

Abstract

Reconfigurable Intelligent Surfaces (RIS) have emerged as a promising technology to enhance the performance of next-generation wireless networks, particularly in terms of increasing coverage and enhancing user throughput. However, effective radio resource management for RIS-aided mobile networks introduces significant complexity due to the high dimensionality and the interdependency of these tasks.

This thesis proposes a solution that leverages the machine learning-based framework of Graph Neural Networks (GNNs) to jointly optimize radio resource management tasks in a multi-user scenario. The proposed model is capable of performing user scheduling, beamforming, and RIS configuration based on full channel information that maximizes proportional fairness. Additionally, the model also supports implicit power allocation.

A comprehensive simulation environment emulating a dense urban mobile network is developed to train and test the model. The performance across various mobile network deployments is evaluated and compared to an accurate existing method. Results demonstrate that the proposed GNN-based solution manages to achieve a large fraction of the user throughput gain achieved using the existing method in significantly less computation time, showcasing its potential for real-time radio resource management in future 6G networks.

Keywords: Reconfigurable Intelligent Surfaces (RIS), Radio Resource Management, Graph Neural Networks (GNN), Machine Learning (ML), Beamforming, RIS Optimization, User Scheduling, MIMO Systems, 6G Mobile Networks.

List of Acronyms

- **3GPP**: Third-Generation Partnership Project
- **AE**: Antenna Element
- **AI**: Artificial Intelligence
- **AO**: Alternating Optimization
- **AP**: Access Point
- **AR**: Augmented Reality
- **B5G**: Beyond 5G
- **BCD**: Block Coordinate Descent
- **BL**: Blocker Loss
- **BS**: Base Station
- **BW**: Bandwidth
- **CF**: Correction Factor
- **CSI**: Channel State Information
- **DL**: Downlink
- **DNN**: Deep Neural Network
- **DRL**: Deep Reinforcement Learning
- **eMBB**: Enhanced Mobile Broadband
- **GNN**: Graph Neural Network
- **IoT**: Internet of Things
- **IRS**: Intelligent Reflective Surface
- **JCAS**: Joint Communication and Sensing
- **KPI**: Key Performance Indicator
- **LMMSE**: Linear Minimum Mean Square Error
- **MIMO**: Multiple-Input Multiple-Output
- **MISO**: Multiple-Input Single-Output
- **MLP**: Multi-Layer Perceptron

- **ML**: Machine Learning
- **MMTC**: Massive Machine-Type Communications
- **MRC**: Maximum-Ratio Combining
- **MRT**: Maximum-Ratio Transmission
- **NOMA**: Non-Orthogonal Multiple Access
- **OFDM**: Orthogonal Frequency-Division Multiplexing
- **PF**: Proportional Fairness
- **PRB**: Physical Resource Block
- **QuaDRiGa**: Quasi-Deterministic Radio Channel Generator
- **ReLU**: Rectified Linear Unit
- **RIS**: Reconfigurable Intelligent Surface
- **SINR**: Signal-to-Interference-plus-Noise Ratio
- **SLA**: Sidelobe Attenuation
- **SAC**: Soft-Actor Critic
- **SNR**: Signal-to-Noise Ratio
- **SSB**: Synchronization Signal Block
- **STAR-RIS**: Simultaneous Transmitting and Reflecting RIS
- **SUS**: Semi-orthogonal User Selection
- **SVD**: Singular Value Decomposition
- **SWIPT**: Simultaneous Wireless Information and Power Transfer
- **TDD**: Time-Division Duplex
- **TTI**: Transmission Time Interval
- **UE**: User Equipment
- **UL**: Uplink
- **UMa**: Urban Macro
- **URLLC**: Ultra-Reliable Low-Latency Communications
- **VR**: Virtual Reality
- **WSR**: Weighted Sum-Rate
- **ZF**: Zero-Forcing

List of Figures

List of Tables

Contents

1 Introduction

Over the last decades, mobile communication has evolved rapidly and has become a worldwide network that almost everyone can access, connecting billions of devices, from voice calls to high-speed internet and beyond, as shown in Figure 1.1. Data rates have increased from kbit/s to Gbit/s, latency has dropped to the millisecond range, and cell capacity has expanded while costs and energy consumption per transferred bit have decreased.

The first generation of mobile communication was introduced in the 1980s. It was followed by GSM (global system for mobile communications) ten years later, which was considered to be the second generation of mobile communication. GPRS (general packet radio service) added the possibility of sending packet data. In the early 2000s, multiple organizations started to collaborate and formed the third-generation partnership project (3GPP), which, from that time onwards, was the organization responsible for developing and maintaining mobile communication standards. This led to the introduction of 3G with UMTS (universal mobile telecommunications system). The next generation of cellular network technology standards, LTE (long-term evolution), delivered even higher performances in terms of throughput and latency and was later upgraded to LTE-A (LTE-advanced).

The roll-out of 5G NR (5G new radio) started at the beginning of this decade. There are three service categories defined by 3GPP that characterize 5G, which are:

- Enhanced mobile broadband (eMBB)
- Ultra-reliable low-latency communication (URLLC)
- Massive machine-type communication (MMTC)

Enhanced mobile broadband aims to provide higher data rates and capacity to allow the growing demand for high-definition (HD) streaming, augmented reality (AR), and virtual reality (VR). The requirements for ultra-reliable low-latency communication are achieving a reliability of 99.999% and a latency of 1 ms. The particular combination of high reliability and low latency is hard to achieve, and in classical mobile networks, one is often compromised to achieve the other. URLLC is required for applications such as vehicular communication. Massive machine-type communication means that a vast number of devices simultaneously can establish a connection, up to a million connected devices per square kilometre. Potential use cases for MMTC are large automated factories, smart agriculture, or the rise of the internet of

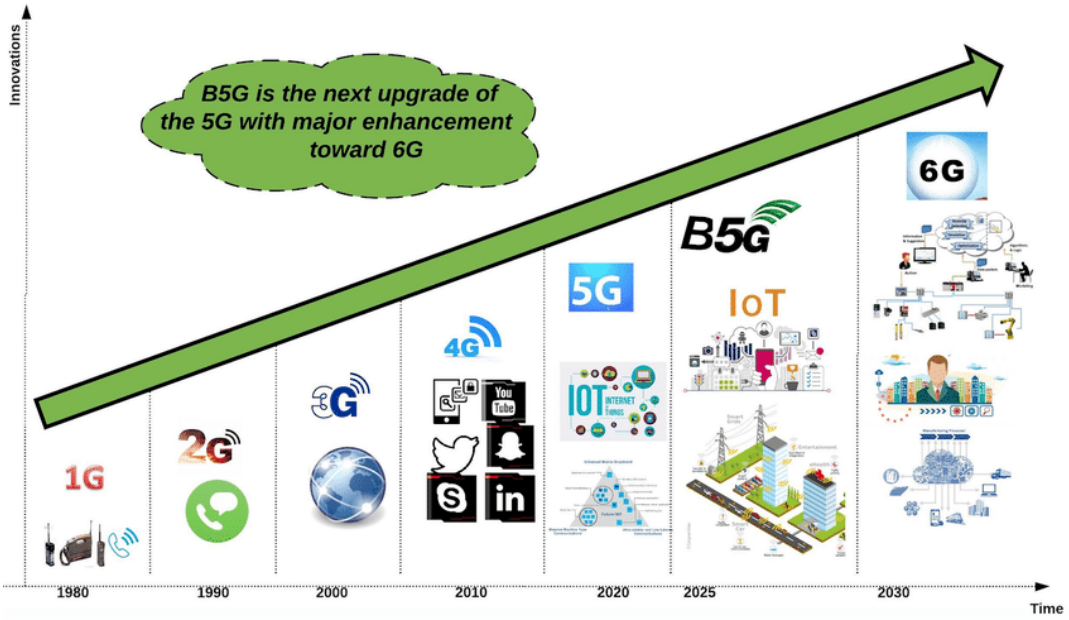


Figure 1.1: The evolution of mobile networks over the last decades.[1]

things (IoT) in urban areas.

As the world looks toward the future, early discussions on 6G are already underway, promising even greater advancements. Besides the still-growing traffic and growing demand for higher data rates by regular consumers, there are also new use cases that require improvements to the current standards.

Potential ways to meet the requirements of beyond-5G (B5G) and 6G are utilizing higher spectrum, AI/ML-driven radio resource management, network ultra-densification, cell-free networking, and joint communication and sensing (JCAS). Currently, there is a lot of ongoing research on including the terahertz (THz) frequency band in 6G to increase the available bandwidth, which can be utilised to provide higher data rates and traffic handling capacity. However, this will also lead to new challenges since higher frequencies have weaker propagation characteristics. Radio resource management tasks are often computationally complex and must be executed within short timescales for real-time applications. AI/ML can be very useful for optimizing these radio resource management-related tasks. Increasing the number of small cells significantly, a process called network ultra-densification should lead to higher capacity and better coverage, especially in densely populated urban areas. Cell-free networking means that a user is simultaneously served by multiple cooperative access points (APs), so the user is no longer bound to one cell. This can be exploited to enhance channel capacity, spatial throughput fairness, and link reliability. JCAS is also a new technology that combines communication and sensing networks into a single infrastructure. There is a growing demand for sensing and localization for various applications such as autonomous vehicles, industrial automation, and smart cities. Combining communication and sensing into one system is envisioned to increase the overall spectrum efficiency.

Another emerging technology that is receiving considerable attention is reconfigurable intelligent surfaces (RIS), which will be studied in this thesis. It has the potential to contribute to meeting the requirements of the sixth generation of mobile communication.

1.1. Reconfigurable intelligent surfaces (RIS)

Reconfigurable intelligent surface (RIS), sometimes also referred to as intelligent reflecting surface (IRS), is a promising technique to increase coverage, capacity, and throughput. A RIS is a surface consisting of many elements that can reflect a wireless signal and shift the phase of the reflected signals, as shown in Figure 1.2. Each reflecting element can be individually configured with a particular phase shift, making it 'reconfigurable'. Similar to beamforming from the base station to the users, significant user-specific directional gains from the RIS can be achieved through constructive and destructive interference of the reflected electromagnetic waves. Configuring the RIS and designing the right precoders should be done cooperatively, and therefore, a signalling connection between the RIS and the base station is necessary. Since the considered RIS will only reflect and does not add extra power to the reflected signal, it can be regarded as passive beamforming. There are also RISs that amplify the signal, also referred to as active RISs, and simultaneous transmitting and reflecting RIS (STAR-RIS), but this thesis will focus solely on passive reflecting RISs.

A couple of approaches are proposed in the literature to control the reflection of the RIS elements, such as mechanical actuation via mechanical rotation or functional materials, such as liquid crystal and graphene. However, the most feasible solution seems to be using electronic devices, like diodes, which are embedded in the circuitry that is connected to the metasurface of the elements. The controllable diodes can be switched on or off, which means that, in practice, the phase shift is not a continuous variable, but is selected from a discrete set of phase shift values. When each element contains three diodes, there are $2^3 = 8$ possible configurations for each element.

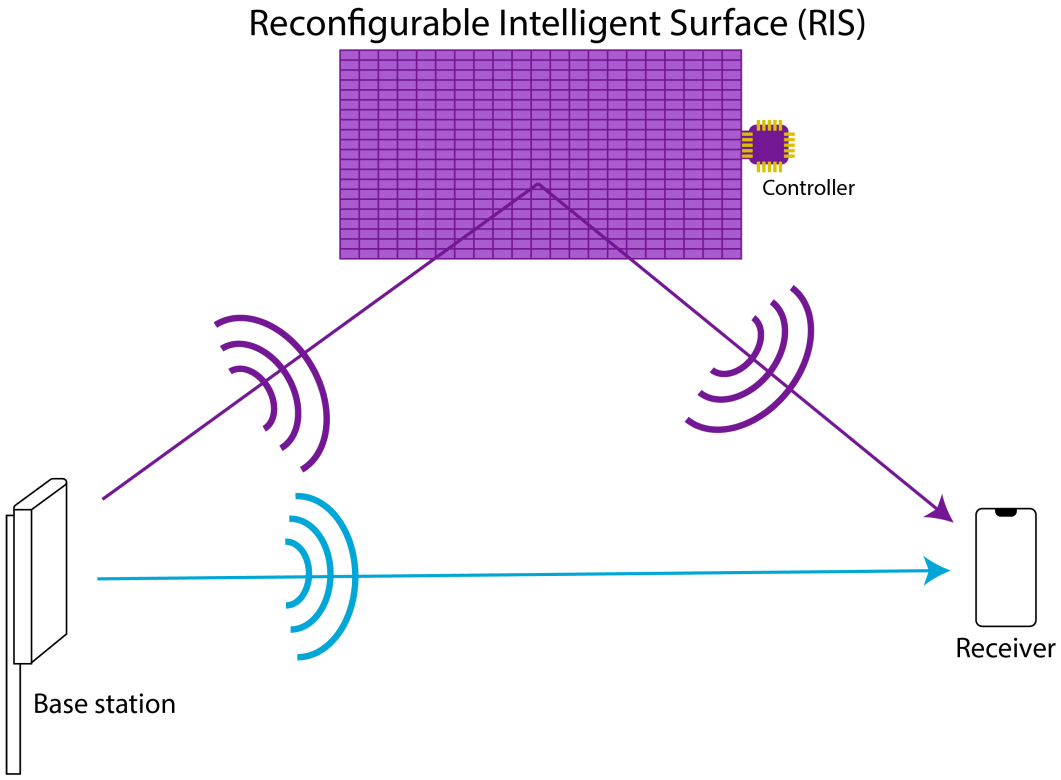


Figure 1.2: Illustration of the RIS with the two propagation paths from the base station to the receiver.

The RIS creates an extra path from the base station to the users via the RIS, which does not follow the direct path between the base station and the users. This is why RIS can help to improve a cellular network's performance. It can be used to extend coverage for users in dead zones. If the direct path is heavily blocked, a well-configured RIS can, in some occasions, allow the user to still have coverage via the indirect path. The RIS can also lead to an enhancement of the user throughput because the extra path could add to the channel to improve the overall channel quality and thus increase the SINR of the received signal by a user.

Although RIS has the potential to improve coverage and throughput, some challenges need to be addressed to exploit that potential. First, the channel state information (CSI) needs to be acquired. Estimating the CSI of the three links, the cell-user, cell-RIS, and RIS-user link, individually is a complex task. To accomplish this, the base station needs to send many pilot signals while varying the RIS configuration in a systematic way. Another challenge is the design and deployment of RIS. This includes design choices for the physical implementation, such as the number of reflecting elements and the materials used for the RIS elements, but also the deployment strategies, such as selecting the optimal location of the RIS, deciding between indoor or outdoor deployment, or whether to use a centralized RIS or multiple distributed RISs. Lastly, there are radio resource management-related tasks that need to be addressed, such as selecting precoders, finding a proper RIS configuration, user scheduling, RIS-user association, cell-RIS association, and cell-user association. These radio resource management tasks are rather difficult to solve and, when done well, may involve complex and hence time-consuming solutions. For real-time implementation, these tasks should ideally be executed in a short time. In this thesis, we will focus on solving resource management-related problems.

1.2. Literature review

This section presents the literature review that we have conducted on radio resource management for RIS-aided networks. We start with briefly discussing literature on heuristic-based radio resource management. In the following subsection, we present the findings on the literature of machine learning (ML)-based methods for radio resource management optimization We identify three main classes of ML-based approaches for radio resource management optimization, of which the first two are discussed in this subsection. The last subsection will elaborate on the third class, graph neural networks (GNNs).

1.2.1. Literature on heuristic-based radio resource management for RIS

This work is a follow-up study on earlier research on radio resource management for reconfigurable intelligent surfaces [2], in which two heuristic solutions were proposed to solve radio resource management-related problems. The addressed resource management tasks included beamforming, deriving the RIS configuration, cell-user association, and scheduling in a multi-cell environment. For cell-RIS association, the RIS was always associated with a given cell.

In [3], the authors focus on optimizing associations among the cells, RISs, and users, thus cell-RIS-user association. This was also done in [4], jointly with beamforming and finding the proper RIS configuration, but these two papers do not perform any scheduling of the users. On top of that, these methods are rather complex and, hence, time-consuming solutions.

1.2.2. Literature on ML-based radio resource management for RIS

In this thesis, we want to explore the possibilities of using Machine Learning (ML) for the optimization of radio resource management tasks to overcome the limitation of existing techniques in terms of computation time. In the literature, we have found many methods which can be categorized into the following three classes of machine learning methods, as listed below.

- Supervised learning
- Deep reinforcement learning (DRL)
- Graph neural networks (GNNs)

In this section, we first examine supervised learning and deep reinforcement learning before shifting our focus to GNNs, which are discussed separately in the next section.

An example of performing radio resource management tasks using supervised learning is shown in [5], in which they try to optimize the RIS-user association. First, the problem is split into RIS-user association, beamforming and configuring the RIS, which are then solved by convex optimization methods. Since the RIS-user association step is considered to be complex and time-consuming, a forward neural network (FNN) is trained on pre-calculated samples to decrease the calculation time of the entire process to make it feasible for real-time application. The RIS-user association matrix is a binary matrix, with ones indicating that users are associated with the RIS and zeros otherwise. This binary matrix is then converted to integer values. This way, this problem is transformed into a regression problem.

Another commonly used machine learning technique is deep reinforcement learning (DRL), which has been applied across various research topics. For instance, [6] explores the potential of DRL for cell-RIS-user association in multi-cell systems, [7] and [8] include scheduling, [9] and [10] focus on the energy efficiency of the network, [8] exploits DRL for RIS-aided systems for vehicular communication, and [10] also addresses the hardware limitations of RIS.

A DRL approach is presented in [6] using the soft actor-critic (SAC) DRL algorithm to jointly optimize beamforming and the association between cells, RISs, and users in mmWave networks when there are blockages. The SAC-DRL learns the best policy by interacting with the environment using prior information. The goal is to maximize the system sum rate by dynamically associating RISs to cells and users.

In [7], a DRL-based user scheduling, phase shift control, and beamforming optimization algorithm is proposed to solve the joint problem. The neural combinatorial optimization (NCO) technique was used for user scheduling. In NCO, a stochastic policy for solving the combinatorial optimization problem is learned by training deep neural networks (DNNs) based on reinforcement learning. For the joint phase shift control and beamforming optimization, a curriculum learning-deep deterministic policy gradient (CL-DDPG) was used. It is compared to two alternating optimization (AO)-based algorithms and a greedy algorithm. It outperforms them in terms of the sum of the proportional fairness ratios. Experiments were performed for a multi-cell scenario.

[8] investigates user scheduling and beamforming with RIS for vehicular communication. One of the main challenges is that it is a very dynamic environment where vehicle arrival, departure, and speeds are unknown. A DRL model is used for the user scheduling, and block coordinate descent (BCD) is used to determine the RIS configuration to maximize the minimum average

bit rates of all vehicles selected to be served at a particular time slot. A limitation of this work is that the cells only have one transmit antenna, and the users only have one receive antenna, making this a single-input single-output (SISO) scenario. The proposed algorithm is compared to three benchmark algorithms: greedy scheduling with BCD, random scheduling with BCD, and DRL with random RIS configurations.

In [9], the objective is to maximize the energy efficiency of the system instead of the sum rate, allowing RIS elements to be turned on or off as part of the optimization process. They split their optimization problem into two sub-problems. The first problem, the joint optimization of user-RIS association, the on/off states of the RIS elements, and the RIS phase shift matrix, is solved using deep reinforcement learning. The second sub-problem, the power control of the RIS elements, is a convex optimization problem. This is solved numerically with the help of the Python convex optimization solver CVXPY.

[10] addresses RIS hardware limitations and channel uncertainties by proposing an actor-critic-barrier (ACB) DRL algorithm. The model dynamically adjusts the base station transmit power and the RIS configuration, aiming to maximize energy efficiency, taking the hardware limits of the RIS into account. This approach is particularly valuable where RISs do not have a continuous range of RIS configurations but a discrete set, which is the case in current real-life implementations of RIS.

Power allocation among users is also an important aspect in [11]. A DRL-based solution for joint optimization of beamforming, optimizing the RIS configuration, and power allocation is designed for a network with multiple RISs. The model integrates these variables into an action output with a weighted reward system to balance beamforming and power. The DRL significantly improves in the sum of rates, especially for high-density network environments with higher frequencies.

A summary of all the described studies utilizing supervised learning and deep reinforcement learning (DRL) is presented in Table 1.1. This table provides a comparative overview of the objectives and key methodologies of each study. All the studies that we have described so far assume a fixed number of scheduled users, which is not practical for real-world deployment. This is a limitation that can be overcome by using GNNs.

1.2.3. Graph neural networks (GNNs)

Graph neural networks (GNNs) is a machine learning technique that is oftentimes proposed to solve the joint problem of beamforming and RIS configuration because of their ability to adapt to a varying number of users. GNNs operate on graphs consisting of nodes that exchange information with each other via messages along edges between the nodes. Each node can represent various entities, such as a user, a RIS, an individual RIS element, or a transmitting antenna element, depending on the chosen GNN architecture. The nodes aggregate the received messages, which makes them scalable in terms of the number of users and RISs. Based on the aggregated messages they update their states, which can be read out in the end to obtain the desired outcome for the specific optimization objective the model was trained for.

In [12], the authors improved on their already developed unsupervised learning model of [13] by adopting graph neural networks to better model the interference among the users. The GNN can jointly perform beamforming and configure the RIS based on received pilot signals. The GNN architecture allows the design of the precoders that minimize the interference in the

Table 1.1: Summary of papers with ML techniques to solve RIS-related radio research management.

Paper	MIMO/ MISO	Single/ Multiple cells	Single/ Multiple RISs	Discrete/ Continuous phase	Method	Optimization objective	Benchmark algorithm
[3]	SISO	Multiple	Multiple	Continuous	Optimal via Branch-and-Bound (BB) method, Suboptimal via Successive refinement	Sum throughput	Fixed BS-user Association, Fixed BS-IRS Association, Random Scattering
[4]	MISO	Multiple	Single	Continuous	fractional programming-Majorization minimization-Alternating direction method of multipliers (FP-MM-ADMM)	Sum throughput,	Gain-based
[5]	MISO	Single	Multiple	Continuous	Alternating Optimization, Forward Neural Networks (FNN)	Sum throughput	Without RIS, Random, exhaustive search
[6]	MISO	Multiple	Single	Discrete	Soft Actor Critic (SAC)-DRL	Sum throughput	RA, w/o RIS, lower values of B
[7]	MISO	Single	Single	Continuous	Neural combinatorial optimization (NCO), Curriculum Learning (CL), Deep Deterministic Policy Gradient (CL-DDPG)	Sum throughput, distribution of rates	Greedy Scheduling (GS), Alternating Optimization (AO)
[8]	SISO	Single	Single	Discrete	DRL, Block Coordinate Descent (BCD)	min average bit rate, throughput fairness	Greedy Scheduling (GS), RS, DRL with random phase
[9]	MISO	Single	Multiple	Continuous	DRL	Sum throughput, energy efficiency, distribution of rates	S-RIS, M-RIS-NPS
[10]	MISO	Single	Single	Discrete	Actor-critic-barrier(ACB) Reinforcement learning	Average bit rate, energy effiency	DQN, DDPG
[11]	MISO	Single	Multiple	Continuous	DRL (DDPG)	Sum throughput	RP, CF

direction of other users, and is also scalable in terms of the number of users. The model was trained in an unsupervised learning procedure, maximizing the sum of the user throughputs. After [12] introduced GNNs to jointly perform beamforming and configuring the RIS, others further expanded on the research considering different objectives and scenarios.

Instead of received pilot signals, [14] assumes full channel knowledge, which is used as features for the nodes and edges of the GNN. The graph neural networks are bipartite graphs that separate RIS nodes and user nodes and allow multiple RISs.

[15] shifts the focus from total throughput optimization to maximizing the sum of proportional fairness ratios, which they call the weighted sum rate (WSR). They are also using a GNN based on perfect channel knowledge. Instead of optimizing all the RISs for all the users, they perform explicit RIS-user association by assigning only one RIS to each user, which leads to an increase of the WSR of around 30%.

[16] primarily looks at single-RIS deployments in a large-scale wireless network. It aims to optimize the GNN across a large area, ensuring the GNN model is adaptable to other regions without having to retrain the model. They are testing various system layouts and evaluating the performance impact of the placement of the RIS. On top of that, they also assess the system throughput improvement by adding a second RIS.

[17] also presents a GNN approach for the maximization of the sum rate for single- and double-RIS scenarios based on received pilot signals. Likewise, different RIS deployment strategies are analyzed for both the single-RIS scenario and the double-RIS scenario. They benchmark their results against BCD, assuming perfect CSI to derive the upper performance bound and BCD combined with linear minimum mean square error (LMMSE) as the lower bound.

[18] analyses the permutation equivariance of GNNs for beamforming. First, they create two separate graphs, one for channel estimation and one for beamforming, that take received pilot signals as input. The first GNN consists of RIS nodes, transmit antenna nodes, and user nodes, while the second GNN maintains the same structure but uses RIS element nodes instead of full RIS nodes. It is compared to [13], and it uses fewer training samples, which is beneficial since the training samples are usually expensive to gather from real-world wireless systems [18].

Most papers do not consider scheduling. However, this is an important aspect that should not be overlooked. Selecting all users simultaneously without a proper scheduling mechanism can lead to very poor results, because channels of the scheduled users can be correlated. [19] extends the work of [12] by also incorporating the scheduling aspect into the system. A two-stage GNN approach was taken: A first GNN is used to make a scheduling decision, and a second GNN is used for the final beamforming and RIS configuration. Instead of maximizing the total throughput of the system, the sum of proportional fairness ratios is maximized. After the first GNN, they select a fixed number of users with the highest proportional fairness ratios to be scheduled, who are then passed to the second GNN to derive the final precoders and RIS configuration for the scheduled users only. The number of scheduled users is set equal to the number of transmit antennas, to serve as many users as possible. This number is fixed irrespective of the channel information, which may not lead to the best scheduling decision. As in [12], received pilot signals were used as input for the GNNs. A short set of pilot signals is used for the first GNN, while for the final GNN a more extended set of pilot signals is used.

Table 1.2 shows the main differences between the papers on RIS optimization with GNNs with respect to the considered scenarios, optimization objectives and available input information.

Table 1.2: Table with the main differences between the GNN-related works.

Paper	Scheduling	RIS scenario	Optimization goal	Provided information
[12]	No	Single-RIS	Sum throughput	Pilot signals
[14]	No	Multi-RIS	Sum throughput	Perfect CSI
[15]	No	Multi-RIS	Sum PF ratios	Perfect CSI
[16]	No	Double-RIS	Sum throughput	Pilot signals
[17]	No	Double-RIS	Sum throughput	Pilot signals
[18]	No	Single-RIS	Sum throughput	Pilot signals
[19]	Yes	Single-RIS	Sum PF ratios	Pilot signals

We have presented a literature review on performing radio resource management-related tasks for RIS-aided wireless networks using machine learning-based methods, primarily because of their fast inference time compared to traditional optimization techniques. From this literature review, we identified various potential machine learning techniques that could possibly be applied in this context. The GNNs in particular stood out because of their scalability in terms of the number of users. However, we observed that most of the GNN-based approaches do not take scheduling into account. The only exception is [19], which makes scheduling decisions which are not based on channel information, which can lead to co-scheduling users whose channels are insufficiently uncorrelated, which may consequently lead to poor performance. This leads us to the objective of this thesis.

1.3. Objective and thesis overview

This section presents the objective of this thesis, how it is structured and the mathematical notation we use throughout.

1.3.1. Objective

First of all, this thesis aims to demonstrate the potential of reconfigurable intelligent surfaces to enhance user throughput performance in 6G mobile networks. Secondly, we will explore the possibilities of AI/ML for radio resource management in RIS-aided wireless networks.

The objective of this thesis is to optimize radio resource management-related tasks for RIS-aided mobile networks. More specifically, we want to develop a model that is able to **jointly design precoders for the scheduled users and configure the RIS** for a multi-cell multi-input, multiple-output (MIMO) scenario, in which the model further makes **scheduling** decisions based on the full channel information. Additionally, the model will also be able to provide a **power allocation** among users. The model that we develop will be a **GNN-based** solution, enabling these tasks to be executed in a short time.

The developed GNN will be extensively analyzed across various scenarios and RIS deployments to assess its capabilities. The solution will be compared to existing benchmark solutions, which will showcase the advantages and limitations of the proposed ML-based approach compared to alternative, non-ML-based, approaches. Different types of architectures of the GNN model will also be compared to each other to find the best possible model and to understand the reasoning behind the model's behaviour.

1.3.2. Thesis overview

In the next chapter, we introduce the simulation environment and all the parameters that characterize it. The chapter further covers the radio resource management concepts important for both the proposed solution and the benchmark algorithms to which it is compared. Chapter 3 will provide a complete overview of the proposed solution for our optimization problem. This solution comprises multiple steps, including both ML-based and non-ML-based steps. The chapter will further elaborate on GNNs, which is one of the key components of the proposed solution. We will explain the concept of GNNs and describe the architecture of the GNN model in our solution. In Chapter 4, we will dive deeper into the machine-learning aspect of our solution. We will discuss the training procedure, the specific ML choices made, and the process for generating the training data. In this chapter, the GNNs will also be tested for four cases: single-user without RIS, multi-user without RIS, single-user with RIS, and multi-user with RIS. Once the GNN is tested, the GNN is fine-tuned to obtain the optimal GNN model configuration. Chapter 5 presents and analyzes the results of the optimized GNN across different mobile network scenarios. The results are compared against cases where the RIS is either configured to a fixed default setting or configured using existing radio resource management solutions. Finally, in the conclusion, we will reflect on this work. We will summarize the findings of this thesis and discuss its limitations. We will also be giving recommendations for possible future research on this topic.

1.3.3. Notation

The notation used in this thesis is as follows. A scalar value a is denoted with regular typeface, while a vector $\mathbf{a}$ is always represented by a boldface lowercase letter and a matrix by a boldface capital letter $\mathbf{A}$. $\mathbf{A}$[i,j] denotes the entry in the i^{th} row and j^{th} column of matrix $\mathbf{A}$. When a matrix varies over time, it is expressed as $\mathbf{A}(t)$. Since we are dealing with complex numbers in this thesis, the real and imaginary components of scalars, vectors, and matrices are denoted by $Re(\cdot)$ and $Im(\cdot)$, respectively. The absolute value of a scalar a is denoted with $|a|$, the L2-norm of a vector $\mathbf{a}$ with $\|\mathbf{a}\|$, and the Frobenius norm of a matrix with $\|\mathbf{A}\|_F$. Moreover, for a matrix $\mathbf{A}$, the $\mathbf{A}^T$, $\mathbf{A}^{-1}$, $\mathbf{A}^+$, and $\mathbf{A}^*$ are the transpose, the inverse, the Moore-Penrose inverse, and the complex conjugate of that matrix. The notation of $diag(\mathbf{a})$ means a diagonal matrix with the entries of $\mathbf{a}$ on the diagonals. A concatenation of vectors and matrices along the vertical axis is denoted with $\oplus$.

2

Modelling

In this chapter, we will explain the modelling of our mobile network, the antennas and RIS, and the propagation channels. In the second part of this chapter, we will explain the radio resource management concepts that are important to understand for this study, which encompasses the coverage check, user scheduling, beamforming, a method for deriving the RIS configuration and how we calculate the throughput.

2.1. Network modelling

For our network, we consider a single-cell MIMO scenario with multiple users in a dense urban environment. The cell is part of a larger cellular network characterized by an inter-site distance (ISD) of 200 m. It has a hexagonal shape, as shown in Figure 2.1, with a cell range 133.3 m ($2/3 \cdot ISD$). We will also look at the effect of the cell size, so we will also consider cells that are characterized by ISDs of 100 m, 300 m, 400 m, and 500 m. The base station (BS) is placed at the left cell edge at a height of 25 meters and is tilted downwards by an angle dependent on the ISD, which is 12°for an ISD of 200 m. The users are uniformly distributed over the cell, and we consider two distinct scenarios where all users are either indoors or outdoors. The number of active users is denoted with N_{users}.

The network is operating at the 3.5 GHz frequency band with a 20 MHz wide carrier. The transmitted signals are modulated using orthogonal frequency-division multiplexing (OFDM) signals with numerology 1. In this numerology, the time transmission interval (TTI) is 0.5 ms, with a sub-carrier spacing (SCS) of 30 kHz. Since twelve sub-carriers make up one physical resource block (PRB), each PRB is 360 kHz wide. This leads to 50 PRBs in total, with a guard band of 805 kHz at either end of the carrier. The carrier makes use of time-division duplexing (TDD), where in each frame of five TTIs, four slots are dedicated to downlink communication, followed by one slot for uplink communication. In this thesis, we will focus only on the downlink communication.

The total transmit power, P_{tx}, of the base station is 120 W and is equally divided over the 50 PRBs. The noise present in the system is modelled as thermal noise using $N = k \cdot T \cdot 10^{NF/10} \cdot BW$, assuming a noise temperature of 290 K and a user equipment (UE) noise figure of 8 dB.

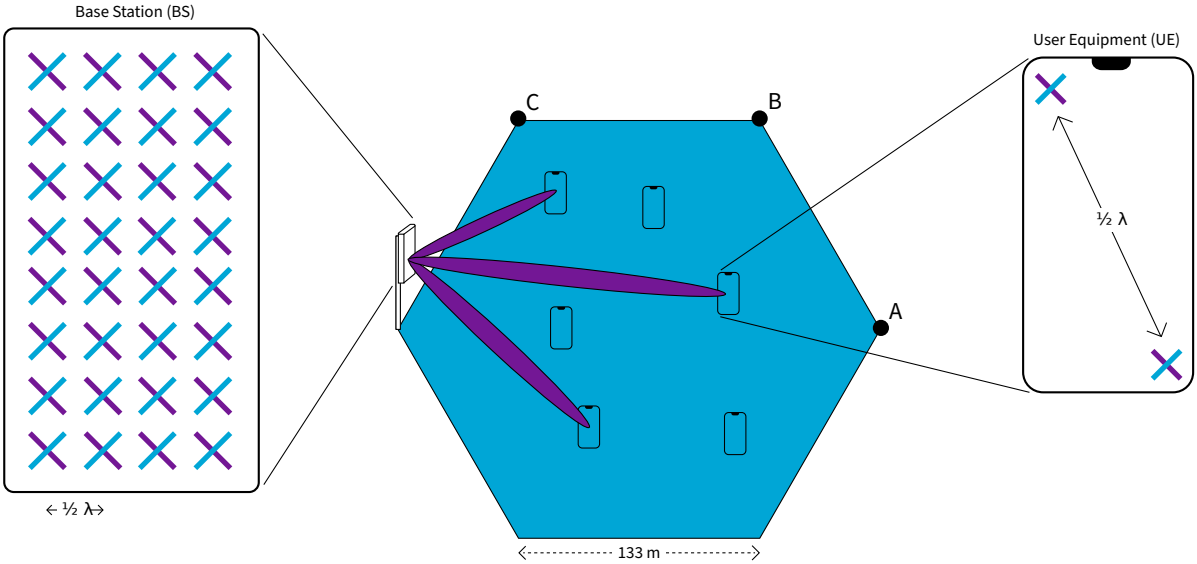


Figure 2.1: Illustration of the simulated hexagonal-shaped area with the base station, user equipment and the possible locations of the RIS, denoted with A, B, C.

2.2. Antenna and RIS modelling

The base station is equipped with a 64T64R antenna array, which means it comprises 64 antenna elements for transmitting and receiving, denoted with N_{tx}, enabling the base station to perform three-dimensional beamforming. The 64 antenna elements are arranged in eight rows and four columns of cross-polarized pairs of antenna elements, as shown in Figure 2.1. The spacing between distinct pairs is half a wavelength ($\lambda/2$).

Each antenna element has a maximum antenna gain $G_{ant,max}$ of 8 dBi. The antenna gain of the antenna elements is dependent on the specific direction. The antenna radiation power pattern of a single antenna element is modelled as specified in Table 7.3-1 of the 3GPP technical report TR38.901 [20]. The antenna radiation power pattern in any three-dimensional direction can be derived by summing the horizontal cut and vertical cut of the radiation power pattern, as shown in Equation 2.1a. The vertical cut of the radiation power pattern is given by Equation 2.1b, in which the vertical half-power beamwidth θ_{3dB} is specified to be 65°and the sidelobe attenuation (SLA)) to be 30 dB. The horizontal cut of the radiation power pattern is similar to the vertical cut (Equation 2.1c) with a horizontal half-power beamwidth ϕ_{3dB} of 65°and a maximum radiation power pattern A_{max} of 30 dB.

$$G_{ant}(\theta,\phi) = G_{ant,max} - \min(-(A(\theta,\phi=0) + A(\theta=0,\phi)), A_{max}) \quad [dB] \tag{2.1a}$$

$$A(\theta,\phi=0) = -\min(12\left(\frac{\theta}{\theta_{3dB}}\right)^2, SLA) \quad [dB] \tag{2.1b}$$

$$A(\theta=0,\phi) = -\min(12\left(\frac{\theta}{\theta_{3dB}}\right)^2, A_{max}) \quad [dB] \tag{2.1c}$$

On the receiving side, each UE (user equipment) is provided with four receive antennas (N_{rx}=4), two cross-polarized antennas in the top left corner and two cross-polarized antennas in the bottom right corner, separated by 157 mm. However, to reduce complexity, our proposed solution will utilize only a single receive antenna. The model's architecture is capable

of extending to multiple receive antennas, but this has not been tested due to time constraints. All UEs are assumed to be at 1.5 m above the ground with a random orientation.

In this thesis, we will consider three different RIS sizes; the RIS could contain 100, 400, 900 or 1600 reflective elements, denoted with $N_{elements}$, arranged in square-shaped grids of 10x10, 20x20, 30x30, or 40x40 elements, respectively. The spacing between adjacent RIS elements is $\lambda/2$.

The phase shift of each RIS element can be tuned individually. The RIS configuration is represented by the RIS configuration vector, $\boldsymbol{\phi}(t) \in \mathbb{C}^{N_{elements}}$, in which each element is constrained by $|\boldsymbol{\phi}_i(t)| = 1$, which means that signals are fully reflected without loss or amplification. The RIS configuration matrix $\boldsymbol{\Phi}(t)$ is defined as $diag(\boldsymbol{\phi}(t))$, representing a diagonal matrix constructed from the configuration vector. The RIS is configured for every TTI of 0.5 ms to adapt to the dynamic wireless environment. The default RIS configuration is represented by the identity matrix, $\mathbf{I}_{N_{elements}}$. In this configuration, none of the elements shift the phase of the signals, and they reflect the signals as a mirror.

A RIS is most effective when it is deployed at the cell edge, because it helps to improve the signal power of cell-edge users. In this study, we, therefore, consider three different RIS locations at the edges of the cell. The locations are denoted by the letters A, B and C, as shown in Figure 2.1 and the RIS is positioned at the height of 20 m, with its physical orientation aligned toward the centre of the cell at all three locations. Since only users within the cell are considered, the RIS beam range is a 120° sector, which allows for optimal signal reflection to the intended users.

Up to this point, we have assumed that each reflective element can achieve a continuous range of reflection values. However, in most practical implementations, these reflective elements are controlled by diodes that operate in discrete states, such as on or off. As a result, the reflection of each element is restricted to a finite, discrete set of values. In this study, we will examine the effect of the discretization of the phase shift by looking at different values for the bit resolution B. The bit resolution B represents the number of bits per RIS element used to control this phase shift, leading to 2^B possible phase shifts. Besides the case of no discretization, we will have 1, 2, and 3 bits per reflective element, which lead to 2, 4, and 8 possible phase shifts, respectively.

2.3. Channel modelling

The propagation behaviour of the signals through different channels is represented by the channel response matrices $\mathbf{H}_{ab}$, which capture the gain and the phase response of the signals through the specific channels. Given the TDD duplexing mode of the assumed carrier, all channels are reciprocal, but we will only focus on the downlink communication. As shown in Figure 2.2, there are three channels in our simulation. These three channels, along with their corresponding channel response matrices, are as follows:

- The cell-user link: $\mathbf{H}_{cu}(t, user) \in \mathbb{C}^{N_{tx} \times N_{rx}}$
- The cell-RIS link: $\mathbf{H}_{cr} \in \mathbb{C}^{N_{tx} \times N_{elements}}$
- The RIS-user link: $\mathbf{H}_{ru}(t, user) \in \mathbb{C}^{N_{elements} \times N_{rx}}$

The cell-user channel is the channel between the base station and a given user and has a non-line-of-sight (nLoS) character. It is the only channel in which the RIS is not involved. We will

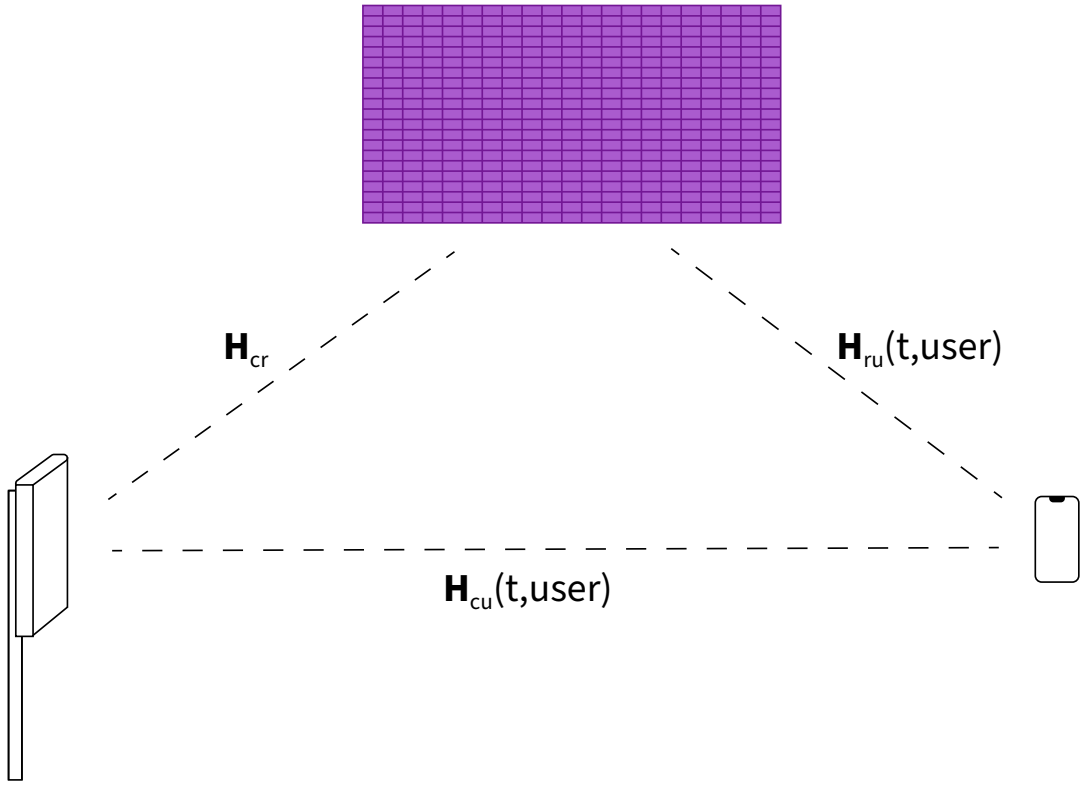


Figure 2.2: The channels between the base station, the RIS and a user.

also refer to this channel as the *direct* channel. The cell-RIS channel is the channel between the base station and the RIS. Since both are placed relatively high above the ground, the channel is assumed to be line-of-sight (LoS), and so there is no multipath fading or shadowing on this channel. However, small-scale fading still occurs on this single-path channel. The RIS-user channel contains the paths between RIS elements and a given user, and it is also a nLoS channel. The cell-RIS channel and the RIS-user channel together form an alternative path from the base station to each user as the signals are reflected by the RIS, we therefore call this path the *indirect* channel. Since the indirect channel is the product of two channels, it is usually weaker than the direct channel without a proper RIS configuration.
The full channel between the base station and each user is the combination of the direct channel and the indirect channel. Mathematically, the full channel response can be described as

$$\mathbf{H}(t, user) = \mathbf{H}_{cu}(t, user) + \mathbf{H}_{cr}\mathbf{\Phi}(t)\mathbf{H}_{ru}(t, user) \tag{2.2}$$

The cell-user channel and the RIS-user channel are non-line-of-sight (nLoS) channels and are changing over time because of the assumed local mobility of the users and multipath fading. The channels consist of two components, the average channel gain $\sqrt{G_{av}}$ and the time- and frequency-dependent small-scale channel component $\mathbf{H}_{small-scale}(t, f)$. The average gain is derived differently for each channel, as explained in the next section. The method for calculating the small-scale component differs between the LoS and nLoS channels. This will be discussed in detail in Section 2.3.2.

$$\mathbf{H}(t, f, user) = \sqrt{G_{av}(user)} \cdot \mathbf{H}_{small-scale}(t, f, user) \tag{2.3}$$

2.3.1. Average channel gain

The average gain of the channel responses is influenced by a couple of factors. It is characterized by the antenna gain of the antenna elements of the base station, as described in Section 2.2, and the propagation losses that arise in the channel, in which the location of the users and the position at which the RIS is deployed play a key role. We assume this channel gain to be static over time, hence the name average channel gain. The average channel gains for the three different channel responses are given in Equations 2.4a, 2.4b, 2.4c. The specific losses are the path loss, shadowing, human body loss, building penetration loss, and an additional blocking loss, which will described below.

$$G_{av,cu}(user) = G_{ant}(\theta_{cu}(user), \phi_{cu}(user)) - max(MCL, L_{path,cu}(d_{cu})) \\ -L_{shadowing,cell}(x,y) - L_{body} - L_{building} - L_{blocker} \quad [dB] \tag{2.4a}$$

$$G_{av,cr} = G_{ant}(\theta_{cr}, \phi_{cr}) - max(MCL, L_{path,cr}(d_{cr})) \quad [dB] \tag{2.4b}$$

$$G_{av,ru} = -max(MCL, L_{path,ru}(d_{ru})) - L_{shadowing,RIS}(x,y) - L_{body} \\ -L_{building} - L_{blocker} \quad [dB] \tag{2.4c}$$

Path loss

The loss due to the propagation over distance is the path loss, represented by L_{path}. Because of the different channel characteristics of the links, we will use different models to derive the path loss for the LoS channel and the nLoS channels. The path loss of the LoS channel between the base station and the RIS, $\mathbf{H}_{cr}$. Given the LoS character of the cell-RIS channel, we adopt the free-space path loss model. The free space path loss can be calculated using Equation 2.5, in which d_{cr} is the distance between the base station and the RIS, and the path loss exponent γ is 2. The path loss PL_0 at the reference distance d_0 of 1 m is calculated by the Friis free space loss equation (2.6).

$$L_{path,cr}(d_{cr}) = PL_0 + 10\gamma \log(d_{cr}/d_0) \quad [dB] \tag{2.5}$$

$$PL_0 = 20 \cdot log\left(\frac{\lambda}{4\pi d_0}\right) \tag{2.6}$$

The path loss of the other two channel responses, $\mathbf{H}_{cu}(t, user)$ and $\mathbf{H}_{ru}(t, user)$, are calculated by using the UMa non-line-of-sight model derived from Table 7.4.1 of [20], This path loss model is shown in Equation 2.7, which is dependent distance between the transmitter and receiver d, the center frequency f_{center} of 3.5 GHz, and the height of the UE h_{UE}.

$$L_{path,UMa-nLoS}(d) = 13.54 + 39.08 \cdot \log(d) + 20 \cdot \log(f_{center}) - 0.6 \cdot (h_{UE} - 1.5) \quad [dB] \tag{2.7}$$

To prevent scenarios with unrealistically small losses, a minimum coupling loss (MCL) of 70 dB is imposed on all three channels [21].

Shadowing
Shadowing, also known as shadow fading or slow fading, refers to the fluctuations in gain due to obstacles and other environmental factors beyond the path loss. The shadowing effect experienced on the link between the cell or RIS to a given UE can be described by a map in which each pixel is sampled from a zero-mean Gaussian distribution. However, the shadowing model in this study is a little bit more complicated. We need to create two cross-correlated maps, one for the cell and one for the RIS. On top of that, the shadowing for users close to each other will be correlated, so it is necessary to generate cross-correlated and spatially correlated shadowing maps. To achieve this, we will apply the procedure as follows, derived from [22].

First, we create three independent maps, a common map G_0 and two other maps, one for the base station and the RIS, G_{cell} and G_{RIS}. These three shadowing maps are two-dimensional maps with a resolution of 5 m in which each pixel value is sampled from a zero-mean Gaussian distribution with a standard deviation σ_{sh} of 6 dB for the scenario with indoor users. When considering the scenario with only outdoor users, a standard deviation σ_{sh} of 7 dB is used [20]. To create the cross-correlation, we combine the RIS and the cell map with the common map, using Equations 2.8a and 2.8b, in which the cross-correlation coefficient ρ is assumed to be 0.5 [22].

$$L_{shadowing,cell}(x,y) = \sqrt{\rho} \cdot G_0(x,y) + \sqrt{1-\rho} \cdot G_{cell}(x,y) \tag{2.8a}$$

$$L_{shadowing,RIS}(x,y) = \sqrt{\rho} \cdot G_0(x,y) + \sqrt{1-\rho} \cdot G_{RIS}(x,y) \tag{2.8b}$$

We will model the spatial correlation of the slow fading effect with a decorrelation distance of 50 m [20]. To create spatial correlation to the cross-correlated maps, we will apply a filter based on the autocorrelation function, as described in [22], by convolving the maps with the filters' impulse response.

At the end of this process, we obtain two sets of two cross-correlated and spatially correlated shadowing maps, one for the indoor and one for the outdoor scenario. Each set comprises one map for the shadowing of the cell-user channel and another for the RIS-user channel shadowing. For all the simulations, these two sets of shadowing maps are reused, which means that we will look at the same propagation environment for all the simulations.

Human body loss
The UEs are assumed to be carried close to the users' bodies. The human body forms an extra blockage that attenuates the wireless signal. The loss due to this effect, the human body loss, is considered to be 6 dB in all cases in our simulations for the cell-user link and RIS-user link [23].

Building penetration loss
For indoor users, the propagation of wireless signals is also significantly attenuated as they pass through buildings. This attenuation of the signals is called the building propagation loss. The building penetration loss is given in Equation 2.9 and is derived from [20]. Indoor users experience a building penetration loss, which is dependent on the distance of the user within a building, d_{inside}. This value is simulated as the minimum of two independent uniformly

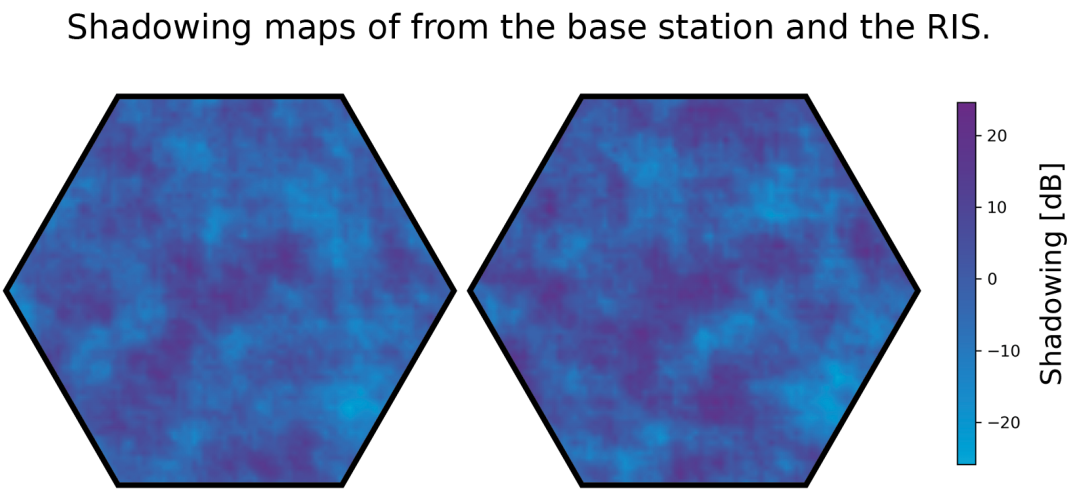


Figure 2.3: The cross-correlated shadowing maps for the base station and the RIS as radio sources, for indoor users.

distributed variables between 0 and 25 m [20]. When considering outdoor users, the building penetration loss will obviously not be present.

$$L_{building} = \begin{cases} 20 + 0.5 \cdot d_{inside} & \text{for indoor users} \\ 0 & \text{for outdoor users} \end{cases} \tag{2.9}$$

Additional blocker

On top of the other losses, both the cell-user channel and the RIS-user channel can be blocked by large temporary physical obstacles. In the case of a strong blocker between the base station and a user, a RIS could be useful to overcome this obstruction since the RIS can provide an alternative path that might not be blocked. We assume that this additional blocker is present in 20% of the cases in the default scenario, but we will also consider scenarios with different blockage probabilities p ranging from 0% to 100%. In case the additional blocker is present, the blocking strength is Gaussian distributed with a mean $\mu_{blocker}$ of 30 dB and a standard deviation $\sigma_{blocker}$ of 10 dB.

$$L_{blocker} \sim \mathcal{N}(\mu_{blocker}, \sigma_{blocker}) \quad [dB] \tag{2.10}$$

2.3.2. Small-scale fading

There are different ways to derive the small-scale component of the channel response, as the channel characteristics are different. For the LoS channel between the cell and the RIS, we will assume a simple distance-based phase, while the nLoS (non-line-of-sight) channels to the users require more complex modelling techniques. We will begin by explaining how the small-scale component of the cell-RIS channel is derived, followed by the cell-user and RIS-user channels.

Cell-RIS channel

As mentioned before, the cell-RIS channel $\mathbf{H}_{cr}$ is a pure line-of-sight (LoS), and so there is no multipath fading on this link. The distance between the base station and the RIS is fixed and user-independent. Consequently, it is considered static across all simulations of the same RIS location. The small-scale component of the channel response is entirely determined by the

lengths of the individual paths within the channel and can be calculated from 2.11, in which $d_{i,j}$ is the distance between base station antenna element i and RIS element j.

$$\mathbf{H}_{small-scale}[i,j] = e^{-j2\pi d_{i,j}/\lambda} \quad \forall \quad \begin{matrix} i = \{0,1,...,N_{tx}-1\}, \\ j = \{0,1,...,N_{elements}-1\} \end{matrix} \tag{2.11}$$

Cell-user channel and RIS-user channel
For the two channels with the non-line-of-sight character, the small-scale component is represented by a more complicated time- and frequency-dependent multipath fading model, which is generated using QuaDRiGa [24].
In QuaDRiGa, the multipath fading is modelled according to the UMa nLoS model [20]. The users are assumed to be dynamic over time. To model their local movement, the users are constantly moving along a circle with a radius of 0.5 m at a speed of 0.8 m/s. The outcome of QuaDRiGa is a set of traces. Ideally, we would like to generate unique traces for all the simulations. However, due to computational constraints, we generate ten different traces at arbitrary positions to be a good representation of all the possible situations. For each channel in the simulation, we randomly select one of these traces and a starting point in this selected trace. Since the traces are generated for specific user locations in QuaDRiGa, we first need to remove the distance-based phases, which is shown in Equation 2.12, in which $d_{i,j}$ is the distance of the path between antenna element i and RIS element j for the cell-RIS link, or between RIS element i and antenna element j for the RIS-user link. Every trace consists of 1000 TTIs for all 50 PRBs. To save on memory, we will only store the sample of every fourth TTI. The samples in between can be recovered through interpolation.

$$\mathbf{H}^{small-scale}_{trace,phaseless}[i,j] = e^{j2\pi d_{i,j}/\lambda} \cdot \mathbf{H}^{small-scale}_{trace,original}[i,j] \tag{2.12}$$

We will reintroduce the correct phase corresponding to the user location from our simulation to the phaseless trace to obtain the time-varying small-scale fading component of the channel. This component will be multiplied with the average gain of each channel to form the complete channel responses.

$$\mathbf{H}^{small-scale}_{trace,phases}[m,n](t,f) = e^{-j2\pi d_{m,n}/\lambda} \cdot \mathbf{H}^{small-scale}_{trace,phaseless}[i,j] \tag{2.13}$$

2.3.3. Wideband channels

Although the channels are comprised of 50 PRBs, the GNN model we will propose in Chapter 3 makes use of the wideband equivalent channel response to reduce the input dimension for the developed model. The wideband equivalent channel response is calculated using Equation 2.14, in which the normalized sum of the channel over all PRBs is multiplied with the average of the absolute channel gains over all PRBs.

$$\mathbf{H}_{wideband}(t) = \frac{\sum_{f=1}^{N_{PRB}} \mathbf{H}(t,f)}{|\sum_{f=1}^{N_{PRB}} \mathbf{H}(t,f)|} \cdot \frac{\sum_{f=1}^{N_{PRB}} |\mathbf{H}(t,f)|}{N_{PRBs}} \tag{2.14}$$

2.4. Radio resource management

Efficient operation of our simulated network requires effective radio resource management strategies. In this section, we will present the existing radio resource management techniques that will either be part of our proposed solution or serve as a benchmark for the performance evaluation of our solution. The existing radio resource management techniques include the coverage check, scheduling, beamforming, a state-of-the-art method for finding the RIS configuration. We will also describe how the throughput is calculated.

2.4.1. Coverage check

At first, there is the coverage check. In 5G, the base station frequently transmits a synchronization signal block (SSB) signal to all users. When a UE-received signal strength, $P^{SSB}(user)$, in the single-cell scenario is higher than the coverage threshold of -150 dBm, the user will be associated with the base station. When the received signal strength is lower than the threshold, the user is not able to make a connection to the base station and will not be regarded in our simulation. The received signal strength by the UE is calculated using Equation 2.15. It is based on the average channel gains and it assumes the full potential of the RIS, which results in a gain of $N^2_{elements}$, and the full potential beamforming gain of N_{tx}. There are 12 SSB signals at a time, each being transmitted with the power per PRB divided by 12.

$$P^{SSB}(user) = (G_{av,cu}(user) + G_{av,cr} \cdot N^2_{elements} \cdot G_{av,ru}(user)) \cdot \frac{P_{tx} \cdot N_{tx}}{12 \cdot N_{PRBs}} \quad [dB] \quad (2.15)$$

2.4.2. Scheduling

To efficiently utilize network resources, it is often not practical to serve all users simultaneously. Instead, a scheduling mechanism is necessary that determines which users will be served at each TTI. The set of all potential users is denoted with $\mathcal{U}$, and the set of scheduled users will be denoted by $\mathcal{S}$. We are wideband scheduling in the time and spatial domain. Through beamforming (spatial multiplexing), multiple users can be served simultaneously in each downlink TTI, in which all the PRBs are assigned to all scheduled users.

There are multiple ways of scheduling, each having its trade-offs. The simplest single-user scheduling method is to distribute the resources equally among the users, which is called Round-Robin scheduling. All the resources are assigned to another user every TTI in a circular order. Although the resources are fairly distributed, it is not very efficient. In maximum rate scheduling the user with the highest attainable throughput will be scheduled. This leads to a better overall network performance, but the resources are not fairly distributed because users with poor channel conditions may receive little to no resources.

Proportional fairness scheduling

A more fair approach would be using the proportional fairness principle. The scheduling decision is based on the proportional fairness ratio, which is the achievable throughput in the upcoming TTI, as if the user is scheduled in isolation with an MRT precoder, divided by the average throughput that the user already has experienced, as shown in Equation 2.16 in which t denotes the TTI.

$$PF(t, user) = \frac{R(t, user)}{R_{av}(t, user)} \quad (2.16)$$

Users with poor channel quality are more likely to have a low average throughput and get more priority in scheduling, which improves fairness among users. In the case of single-user scheduling, all the PRBs are assigned to the user with the highest proportional fairness ratio. To extend the proportional fairness principle to multi-user scheduling, we employ the semi-orthogonal user selection (SUS) algorithm.

Semi-orthogonal user selection (SUS) algorithm

The semi-orthogonal user selection (SUS) algorithm [25] is a multi-user scheduling algorithm based on the proportional fairness principle and the orthogonality between channels. Multi-user scheduling is possible because of multiplexing in the spatial domain (MU-MIMO). Ideally,

the channels of the scheduled users should be sufficiently orthogonal to each other to achieve higher beamforming gains. The orthogonality between two channels can be computed using Equation 2.17, where both channels are assumed to only have a single receive antenna. The resulting orthogonality is a value ranging from 0 to 1, with a value closer to 1 indicating higher orthogonality between the channels. In this study, the threshold for orthogonality γ_0 is set to 0.5 [25].

$$\Gamma(\mathbf{h}_1, \mathbf{h}_2) = \frac{\mathbf{h}_1^T \mathbf{h}_2}{|\mathbf{h}_1||\mathbf{h}_2|} \tag{2.17}$$

The SUS algorithm for single receive antenna users is presented in Algorithm 1. In the SUS algorithm, the user with the highest proportional fairness ratio is always scheduled. Based on their proportional fairness ratios in descending order, more users are considered to be co-scheduled. However, for a user to be scheduled alongside the others, its channel must be sufficiently orthogonal to those of the already scheduled users. This ensures that all scheduled users can be served simultaneously with a high beamforming gain.

Algorithm 1 SUS algorithm

Input: $PF = \{PF_1, PF_2, ..., PF_{N_{users}}\}$, $\{\mathbf{h}(t,1), \mathbf{h}(t,2), ..., \mathbf{h}(t,N_{users})\}$, γ_0
Output: $\mathcal{S}$ (list of scheduled users)

```
sort (PF)
add user_index of PF[1] to S
S ← { }
i ← 2
while i ≤ N_users and len(S) < N_tx do
    for s in S do
        if Γ(h(t,i), h(t,s)) < γ_0 then
            add user_index of PF[i] to S
    i ← i + 1
```

2.4.3. Beamforming

Beamforming is the technique of transmitting signals from an antenna array toward specific spatial directions through the principles of constructive and destructive interference. The process of configuring the antenna elements is called precoding. For each scheduled user, a precoder needs to be determined, which is represented by the precoding vector $\mathbf{w}(t, user)$. The precoder contains the phase and transmit power of each antenna element of the base station. The summed power of all the elements of the array is normalized to match the transmit power of the base station. There are two precoding techniques we will use, MRT and ZF, for single-user beamforming and multi-user beamforming, respectively.

Maximum ratio transmission (MRT)

Maximum ratio transmission (MRT) is a precoding technique used for single-user communication. The principle of MRT is to transmit from each antenna element with a phase that is opposite to the phase response of the corresponding path in the channel and with a power proportional to the gain of that specific path in the channel response. The MRT precoder can be calculated using Equation 2.18.

$$\mathbf{w}_{MRT}(t, user, rx) = \frac{\boldsymbol{h}^*(t, user, rx)}{\|\boldsymbol{h}(t, user, rx)\|} \tag{2.18}$$

If multiple users are scheduled, MRT is not the optimal precoding method. The precoder is optimized for a single user specifically and does not take other users into account, which can lead to a lot of interference to other users.

Zero-forcing (ZF)
For the case of multiple users, there are various ways of deriving the precoders for the base station. One commonly used technique is zero-forcing (ZF), in which the objective is to create a precoder for each user so that this precoder has zero gain in the direction of the co-scheduled users while still aiming to maximize the gain in the direction of the intended user. The designed precoders will not have the highest achievable received signal powers, but the overall signal-to-interference-plus-noise ratio (SINR) will be high because of zero intra-cell interference. Since the SINR is lower compared to a scenario where each user is scheduled individually, ZF does not lead to the highest achievable throughput per user. However, the system benefits from multi-user gain, meaning that the total throughput of the system will be higher. Although zero-forcing is very efficient, It will not necessarily lead to the highest obtainable system throughput. The equation for zero-forcing is shown in 2.19.

$$\mathbf{w}_{ZF}(t, user) = \frac{(\mathbf{H}(t))^{+}(user)}{\|\boldsymbol{h}(t, user, rx)\|} \tag{2.19}$$

2.4.4. RIS configuration

Finding the right RIS configuration, $\boldsymbol{\phi}(t)$, is a challenging task that needs to be done jointly with designing the precoders. Currently, there is no low-complexity algorithm for deriving a RIS configuration. This is, however, important since the RIS will be configured within very short time intervals, specifically for each TTI. Since the indirect path reflects via the RIS, the total channel is dependent on the RIS configuration. Altering the RIS configuration will change the total channel response. That is why configuring the RIS needs to be done in combination with designing the precoders. One heuristic method is the fixed-point iteration algorithm.

Fixed-point iteration algorithm
The fixed-point iteration algorithm [26] is an algorithm to derive a RIS configuration optimized for a single user. That means that the resulting RIS configuration is optimized for combination with the MRT precoder, which should be calculated on the full channel after obtaining the RIS configuration with this process. The fixed-point iteration algorithm operates on a matrix $\mathbf{R}$, which is first constructed as in Equation 2.20 from the three channel responses.

$$\mathbf{R} = \begin{bmatrix} diag(\mathbf{H}_{ru}^{H})\mathbf{H}_{cr}^{T}\mathbf{H}_{cr}^{*}diag(\mathbf{H}_{ru}) & diag(\mathbf{H}_{ru}^{H})\mathbf{H}_{cr}^{T}\mathbf{H}_{cu} \\ \mathbf{H}_{cu}^{H}\mathbf{H}_{cr}^{*}diag(\mathbf{H}_{ru}) & \mathbf{0} \end{bmatrix} \tag{2.20}$$

The fixed-point iteration algorithm is presented in Algorithm 2. It has been mathematically proven in [26] that for the single-user scenario the most optimal RIS configuration $\mathbf{v}$ is obtained by maximizing $\mathbf{v}^{H}\mathbf{R}\mathbf{v}$. However, this is a non-convex optimization problem. When multiplying $\mathbf{v}$ by the $\mathbf{R}$ matrix and then normalizing each element to maintain the unit modulus constraint $|\mathbf{v}_i| = 1$, it slowly converges to a local optimum of $\mathbf{v}^{H}\mathbf{R}\mathbf{v}$. In this algorithm, we initialize $\mathbf{v}$ as the default RIS configuration. After the initialization, we perform these iterations until the change in $\mathbf{v}$ falls below a small threshold ϵ_0 or if we have reached a maximum number of iterations i_{max} that we impose to ensure that it does not take too long. In the end, the resulting RIS configuration $\mathbf{v}$ is obtained, which is close to a locally optimal solution for the single-user scenario.

Due to this iterative process that involves multiplications on large matrices, it can be a computationally expensive method. The complexity increases as the size of the channel responses

Algorithm 2 Fixed point iteration algorithm

Input: $\mathbf{R}$, i_{max}, ϵ_0
Output: $\boldsymbol{\phi}$

$i \leftarrow 0$
$\epsilon \leftarrow 1$
$\mathbf{v} \leftarrow \mathbf{1}_{N_{elements}}$
while $i < i_{max}$ **and** $\epsilon > \epsilon_0$ **do**
    $\mathbf{v_1} \leftarrow \frac{\mathbf{R}\cdot\mathbf{v}}{|\mathbf{R}\cdot\mathbf{v}|}$
    $\epsilon \leftarrow |\mathbf{R}\cdot\mathbf{v_1}| - |\mathbf{R}\cdot\mathbf{v}|$
    $i \leftarrow i + 1$
    $\mathbf{v} \leftarrow \mathbf{v_1}$
$\boldsymbol{\phi} \leftarrow \mathbf{v}^*[: N_{elements}]$

grows, making it a time-consuming approach, especially for networks with large RISs. Since the algorithm will provide a configuration for a single user, it requires assigning the RIS to a specific user. As a result, the fixed-point iteration method enhances the performance of the selected user, but it does not necessarily maximize the overall system performance in multi-user settings. Additionally, it is not intended to work with Zero-Forcing, which is used for multi-user scenarios. We will use this fixed-point algorithm as a benchmark solution to compare the outcome of our solution, in which we set the maximum number of iterations i_{max} is set to 100.

2.4.5. Throughput calculation

The throughput of a given user is calculated with the truncated Shannon formula, as shown in Expression 2.21. In this expression, $SINR(t, user)$ is the signal-to-interfence-plus-noise ratio for the given user, BW is the bandwidth of the entire carrier, which is 18 MHz ($N_{PRB} \cdot 360$ kHz) and CF is the correction factor, which we assumed to be 0.75 in our simulation based on typical values for the correction factor [27]. We also impose a rate limit R_{max}, which we set 114.752 Mbit/s based on 3GPP TS38.214 [28].

$$R(t, user) = \min(BW \cdot CF \cdot {}^2\log(1 + SINR(t, user)), R_{max}) \tag{2.21}$$

3

Graph Neural Networks (GNN)

In this chapter, we will present the proposed radio resource management solution, which combines the machine learning technique of Graph Neural Networks (GNN) with scheduling. We will elaborate on Graph Neural Networks, and describe the architecture of the GNN model that is used in our solution. While the overall structure of the GNN model is defined, we also define a couple of hyperparameters and design choices of the model that still require fine-tuning to achieve optimal performance. This optimization will be conducted in the next chapter.

3.1. Full Radio Resource Management Solution

The objective of the full radio resource management solution is to determine the optimal precoders, RIS configuration, and scheduling decisions in order to maximize the sum of the proportional fairness ratios for all users. All these processes are assumed to be performed at the base station by a dedicated controller. The available input information is the full set of channel responses and the average throughputs experienced so far $R_{av}(t, user)$ by the users who have coverage, as determined by the coverage check described in Section 2.4.1. This coverage check is performed prior to the radio resource management model. We also assume that all channel responses have already been correctly estimated before this radio resource management process.

The full resource management solution consists of three stages, as illustrated in Figure 3.1. These stages include two stages with GNN blocks and a scheduling step in between. A GNN block produces precoders and a RIS configuration and can be applied to a variable number of users. It is a machine learning model that is trained for a given number of users but may be

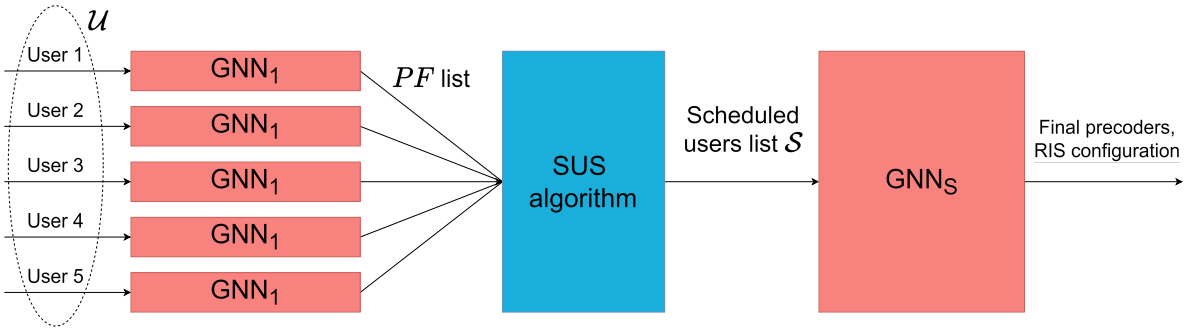


Figure 3.1: The full radio resource management solution consisting of three stages, with Graph Neural Networks (red) and the scheduling stage (blue), illustrated for five connected users.

applied for a different number of users, although the best performance is expected when the model is used for the same number of users as it was trained for. In the first stage, we consider all active users, denoted as $\mathcal{U}$. A GNN trained for the single-user case is applied to each of these users individually in parallel. The resulting precoder and RIS configuration from these GNNs is used to estimate an attainable bit rate for each user if it were scheduled in isolation, which represents how well the user can receive the signal with a RIS deployed and optimized for the user's link. Together with the average per-user throughput $R_{av}(t, user)$, we will create a list of proportional fairness ratios of all the users. In the second stage, we will use the SUS algorithm, as described in Section 2.4.2.1 of the previous chapter, to make a scheduling decision. The SUS algorithm selects users based on the list of proportional fairness ratios and the full channel information of all the users in $\mathcal{U}$. The result is a set of scheduled users $\mathcal{S}$ of size S. The selected users are deemed to have sufficiently orthogonal channels with respect to each other and will have a high expected combined sum of proportional fairness ratios. In the third and last stage, a single GNN is applied to obtain the final set of precoders and the RIS configuration jointly for all the scheduled users in $\mathcal{S}$.

3.2. Graph Neural Networks

The main ML-based component of this proposed solution is the Graph Neural Network. For the derivation of the precoders and the RIS configuration, the GNN takes as input the cell-user and the RIS-user channel responses and the users' average throughput $R_{av}(t, user)$ to optimize the sum of proportional fairness ratios. We define two GNN variants with respect to the power allocation among the users. In one variant, the GNN will divide the power equally among all the scheduled users, and in the other variant, the GNN will provide an implicit power allocation based on the optimization of the sum of proportional fairness ratios. This implicit power allocation is captured in the relative amplitudes of the precoders. This will be further described in Section 3.3.

Graph Neural Networks (GNNs) are based on the principle of graphs, which are collections of nodes connected by edges. When the graph contains multiple types of nodes, it is referred to as a heterogeneous GNN. In these cases, the edges also have different types for the relations between the different node types. Each node in the graph is characterized by a state vector and multiple feature vectors. The state vector is dynamically changing and will be updated throughout an iterative process in the GNNs, in which the nodes will exchange information with each other via messages for a couple of rounds. Features do not change after the initialization, reflecting the inherent properties of the nodes. Edges do not have a state vector but can include static features. After completing the message passing, the states will be read out to obtain the desired result.

Our proposed GNN operates on heterogeneous graphs with user nodes and one RIS node. The user nodes are connected to each other via user-user edges, and the user nodes are also connected to the RIS node via RIS-user edges. The user nodes contain a state, denoted as $\mathbf{s}_{i,k}$ for user node i in round k, which will be used to derive the precoders for the users in the final round. The state of the RIS node in round k is denoted as $\mathbf{v}_k$, and will be used to derive the RIS configuration. The sizes of both state vectors are hyperparameters of the GNN. Although this graph representation closely resembles the physical model, it is a mathematical abstraction to compute the precoders and RIS configuration. Importantly, not all the edges in the graph have a direct physical interpretation, as shown in Figure 3.2, but are introduced purely for computational purposes, e.g. the user-user edges that are used to exchange information to minimize the inter-user interference. The GNN framework allows for multiple RIS

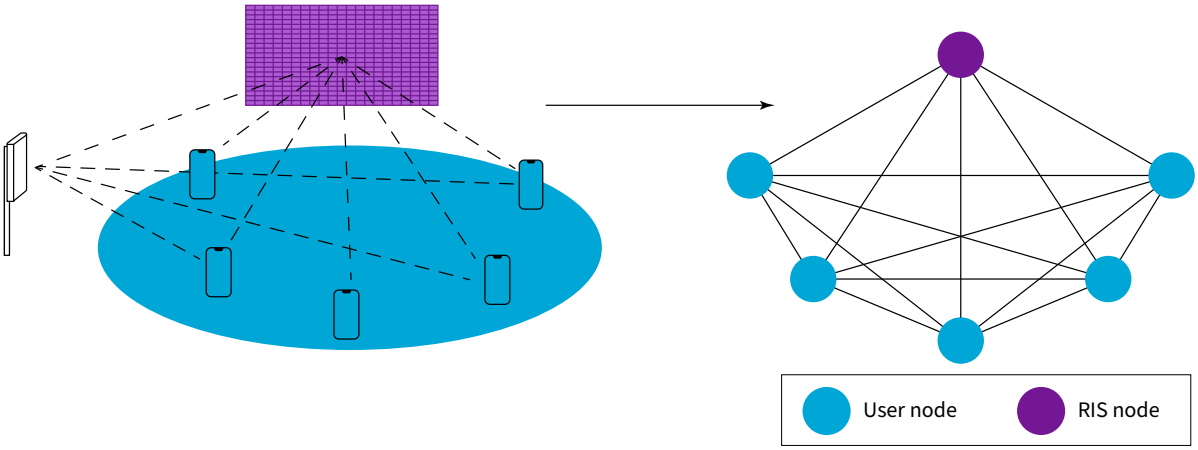


Figure 3.2: The mobile network is transformed into a mathematical graph structure.

nodes, so multiple RISs can be added to the system. However, for the scope of this thesis, we will only focus on the single-RIS scenario, and thus, the GNN has only one RIS node.

There are multiple reasons to adopt the GNN structure. One of the main reasons is the scalability of GNNs, as the GNN can be trained for a specific number of users and later be inferred on a different number of users without retraining. This makes it adaptable to a varying number of served users. Another advantage is the ability of GNNs to take the interference between the users into account when designing the precoders and the RIS configuration because of the message passing rounds. In addition to scalability and interference, GNNs also offer low computation time, which is a key requirement for implementation. While training the model may be time-consuming, once the model is trained, the inference process can be executed in a very short time frame.

Our GNN implementation is developed in Python using TensorFlow (version 2.15.0) [29] and TensorFlow-GNN (version 1.0.3) [30].

3.3. Graph Neural Networks Architecture

The proposed Graph Neural Network comprises three steps: the initialization of the nodes and edges, the message passing rounds, and the final read-out of the nodes' states to obtain the precoders and the RIS configuration. A schematic overview of this entire process is shown in Figure 3.3. We will explain the architecture of the GNN in these three steps.

Initialization

First, we need to initialize the graphs' nodes and edges and their associated states and features. A set of user nodes is created based on the number of users, and furthermore, one RIS node is created. All the user nodes are connected to each other, and all the user nodes are connected to the RIS node. For initializing the states and features, the methods are slightly different. The user nodes and the RIS-user edges will be initialized based on channel responses and the average throughput of the users. The channel responses, however, need to be preprocessed first.

Preprocessing of the input data is required because the difference in values across different samples can be significant, while within a sample, the variation is relatively small. The subtle differences are crucial, as they capture the phase information and relative gain differences

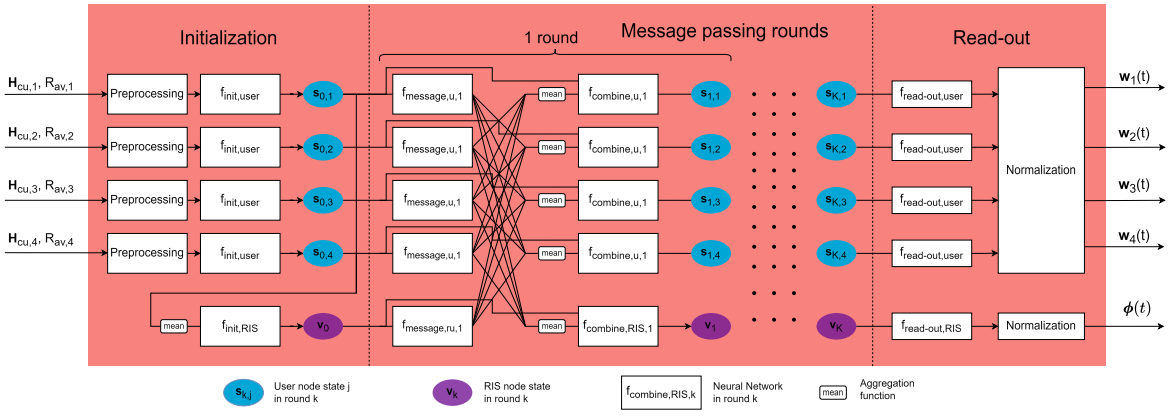


Figure 3.3: The graph neural networks block with all internal neural networks interacting with each other.

between the different propagation paths within the channel. To prevent channel responses with larger magnitudes from disproportionately influencing the learning process, we scale all features to a similar order of magnitude. Specifically, we divide the entire channel response matrix by the highest gain in the matrix. This preserves the phases and the relative gain differences within the matrix while scaling all the samples to a similar range. Since TensorFlow does not support complex numbers, we split the real and imaginary parts of the resulting channel response and work with their concatenated representation. Additionally, the highest gain of the channel response matrix, as used for rescaling, is converted to decibels and appended to the preprocessed input vector to retain the channel gain information. The preprocessing method for the cell-user channel response is formally defined in Equation 3.1, and the same preprocessing approach is applied to the RIS-user channel response.

$$\hat{\mathbf{H}}_{cu}(t,user) = \frac{Re\{\mathbf{H}_{cu}(t,user)\} \oplus Im\{\mathbf{H}_{cu}(t,user)\}}{\max(|\mathbf{H}_{cu}(t,user)|)} \oplus \max(|\mathbf{H}_{cu,dB}(t,user)|) \tag{3.1}$$

The average user throughputs are preprocessed by subtracting the mean and dividing by the standard deviation, which was retrieved from a different independent set, that was not used for training or testing.

The user nodes' states are initialized with the preprocessed channel responses of the direct cell-user link $\mathbf{H}_{cu}(t,user)$ and the preprocessed average throughput $\hat{R}_{av}(t,user)$ of the corresponding user, using an initialization neural network $f_{init,user}$, as shown in Equation 3.2. This neural network is reused for all the user nodes.

$$\mathbf{s}_{0,j} = f_{init,user}(\hat{\mathbf{H}}_{cu}(t,user), \hat{R}_{av}(t,user)) \qquad \forall\, j = \{1,2,...,N_{users}\} \tag{3.2}$$

The edges between the user nodes and the RIS node contain a feature vector, denoted with $\mathbf{e}_j$, in which j represents the corresponding edge. This vector is initialized with the preprocessed channel response of the RIS-user link, as shown in 3.3. The feature vector carries a real physical meaning, representing the RIS-user channel. The RIS-user edges are bi-directional, which means that these edges are used to generate messages for the information exchange in both directions in message-passing rounds.

$$\mathbf{e}_j = f_{init,RIS-user}(\hat{\mathbf{H}}_{ru}(t,user)) \qquad \forall\, j = \{1,2,...,N_{users}\} \tag{3.3}$$

The channel response of the RIS-user link is quite a high-dimensional vector, due to the large number of RIS elements. Since we need the separation of real and imaginary parts of the

channel, this vector would span $2N_{elements}$ numbers. For larger-sized RISs, this channel can dominate the input space and increase the risk of overfitting. A possible solution to this problem is to apply a technique that uses Singular Vector Decomposition (SVD) to reduce the dimensionality of this channel response. The decision to apply this technique is considered a hyperparameter in our model.

For this technique, we first rearrange the channel response matrix $\mathbf{H}_{ru}(t, user)$ to a square matrix of size $\sqrt{N_{elements}} \times \sqrt{N_{elements}}$. For example, a RIS with 900 elements would be transformed into a 30×30-sized channel response matrix. A Singular Vector Decomposition (SVD) will be applied to each input sample individually, decomposing each transformed channel response matrix into three matrices: the left singular matrix $\mathbf{U}$, a diagonal matrix containing singular values $\mathbf{\Sigma}$, and the right singular matrix $\mathbf{V}$. The singular values in the $\mathbf{\Sigma}$ matrix indicate the importance of each principal component in the decomposition. We can reduce the data by taking the x most relevant rows of the right singular matrix $\mathbf{V}$ as in Equation 3.4b. From a separate independent large set, we observed that 10 singular values already capture more than 80% of the total channel response information for RISs with 900 elements. This would reduce the size of the input sample to one-third of the original size of the channel.

$$\mathbf{U\Sigma V} = \text{SVD}(\mathbf{H}_{ru,30\times30}(t, user)) \tag{3.4a}$$

$$\hat{\mathbf{H}}_{ru}(t, user) = \mathbf{V}[: x, :] \tag{3.4b}$$

Although SVD is used in the dimensionality reduction technique Principal Component Analysis (PCA), this technique is different from normal PCA, in which features are reduced based on the variance of features across samples. In our SVD-based approach, we reduce the input feature based on the variance within a sample.

Unlike the user nodes and the RIS-user edges, which are initialized directly from the channel responses, the RIS node will be initialized based on the information from these already initialized nodes and edges. The RIS node contains a state $\mathbf{v}_k$ in round k that will be used to derive the RIS configuration at the final output, and it does not contain any additional features. The optimization of the RIS configuration is dependent on the RIS-user channels, the cell-RIS channel, and the precoders. Therefore, the initial state of the RIS node will be determined by combining messages from the user nodes, which are produced using the user node states and the channel information held by the user-RIS edges, as shown in Expression 3.5. This procedure is equivalent to how the states of the user and RIS nodes will be updated, as will be described in the next section.

$$\mathbf{v}_0 = f_{init,RIS}(\underset{i}{\text{mean}}(f_{message,ur}(\mathbf{s}_{0,i}, \mathbf{e}_i))) \qquad \forall\, i = \{1, 2, ..., N_{users}\} \tag{3.5}$$

At last, we have the user-user edges between every possible pair of users. While they have no direct physical interpretation, they are essential for modelling the interference between the users. The edges don't hold a state or a feature, but they do exist to exchange the state and feature information between the user nodes. Similar to how in zero-forcing the precoders are designed to create a null in the direction of the other users, the GNN model is expected to learn to minimize the interference towards other users.

It is important to note that the channel response $\mathbf{H}_{cr}$ of the cell-RIS link is not used as an input for the GNN model. This is because the channel response $\mathbf{H}_{cr}$ remains constant across all the samples and, due to the large size of the matrix ($N_{tx} \times N_{elements}$), it could become

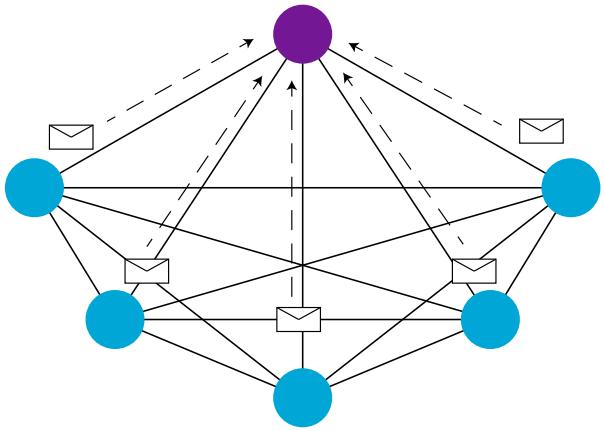


Figure 3.4: The user nodes send a message vector to the RIS node to update the RIS nodes' state.

dominant. However, the cell-RIS channel response is involved in the training process, which will be described in Chapter 4, so the model should learn this implicitly.

Message passing rounds

After the initialization of the nodes and edges, the iterative process of message passing starts. The nodes exchange information messages with each other for multiple rounds to update their states. The number of rounds is represented by K and is a hyperparameter of our model. During each round, all the user and RIS node states are updated concurrently, which means that we will only progress to the next round of message passing or read-out after all nodes have been updated.

The exchange of information between nodes is an essential part of the GNN. For the user nodes, this exchange is necessary because the user nodes need the information about the channels and precoders of the other user nodes to design precoders with little gain to these other users. The RIS configuration also needs to be taken into account for the design of the precoders because the full cell-user channel response is dependent on the RIS configuration (Equation 2.2), which is why the user nodes are updated based on the state of the other users and the RIS. Similarly, for the RIS node, the precoders and the RIS-user channel responses are necessary information to find the optimal RIS configuration.

To update the user nodes, all the user nodes first generate a message vector for all other user nodes, using a neural network $f_{message,uu}$ that is applied to the user nodes' own state. These messages are sent to the other users via the user-user edge. Simultaneously, the RIS node generates a message for each user node using another neural network $f_{message,ru}$, which takes as input its own state combined with the feature vector of the corresponding RIS-user edge. Each generated message is subsequently sent to its respective user node. The optimal length of the message vectors is a tunable hyperparameter, which will be determined in the next chapter.

For each user node, the messages from the other user nodes are aggregated using an aggregation function. The choice of the aggregation function is also a hyperparameter and can be either the mean function or the maximum function. The new state of each user node is obtained by combining its current state with the aggregated messages from the other user nodes and the message received from the RIS node. This process is mathematically described in Equation 3.6 and visually depicted in Figure 3.5.

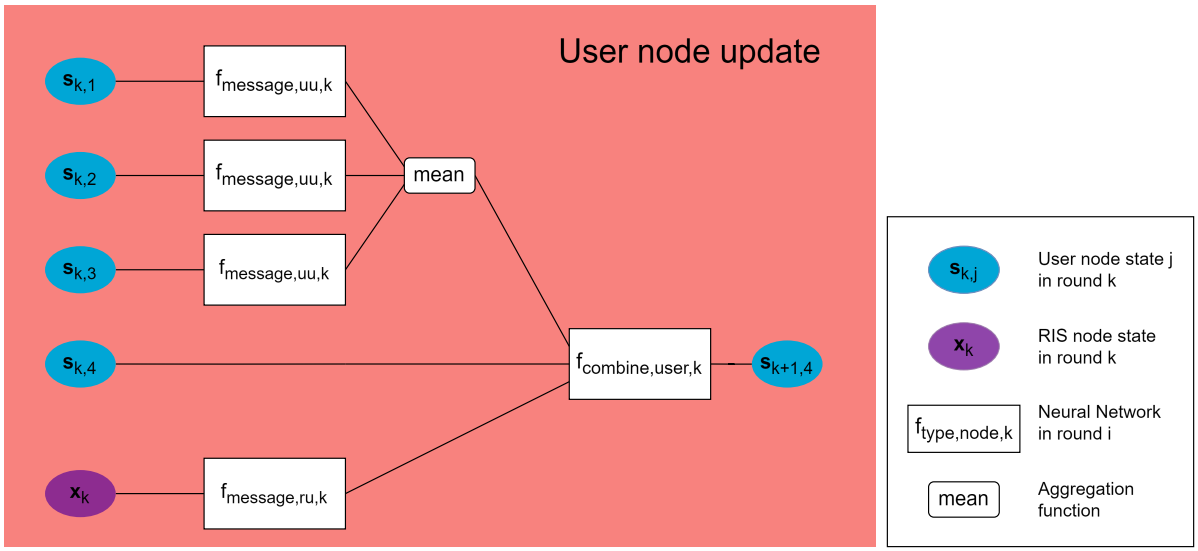


Figure 3.5: One round of message passing to update the state of a single user node.

$$\mathbf{s}_{k+1,j}(user) = f_{combine,user}\Big(\mathbf{s}_{k,j}, f_{message,ru}(\mathbf{v}_k), \underset{i;\ i\neq j}{\text{mean}}(f_{message,uu}(\mathbf{s}_i, \mathbf{e}_i))\Big),$$
$$\forall\ k = \{1, 2, \dots K\},\ j = \{1, 2, \dots N_{users}\} \tag{3.6}$$

Similarly, to update the RIS node, each user node generates a unique message using its own state and the corresponding RIS-user edge feature, and sends it to the RIS node. The RIS node aggregates the received messages from the user nodes and combines them with its current state using the neural network $f_{combine,RIS}$ to determine its new state. This is shown mathematically in Equation 3.7 and visually in Figure 3.6.

$$\mathbf{v}_{k+1} = f_{combine,RIS}\Big(\mathbf{v}_k, \underset{j}{\text{mean}}(f_{message}(\mathbf{s}_{k,j}, \mathbf{e}_j))\Big), \quad \forall\ k = \{1, 2, \dots, K\} \tag{3.7}$$

It is important to highlight that the neural networks for generating the same type of messages and combining the messages are reused for all user nodes and the RIS node within a single message-passing round. This reuse is what enables the GNN to adapt to a varying number of users. However, the neural networks differ across different rounds. While the neural network architecture remains the same, the weights of these networks are different. The number of trainable parameters, therefore, scales linearly with the number of message-passing rounds and, thus, training time.

Read-out

After the message-passing rounds, we will read out the states of the user nodes to obtain the precoders, and we will read out the state of the RIS node to obtain the RIS configuration.
Since we have as many user nodes as scheduled users, each node's state corresponds to the precoder of that user. A final neural network $f_{read-out,\ user}$ with a linear activation function is applied to the user nodes. This will result in a $2N_{tx}$-dimensional vector for each user, representing the real and imaginary components of that user's precoding vector, which are then combined to form a N_{tx}-dimensional complex vector. It needs to satisfy the transmit power constraint of the base station, so the result still needs to be normalized.

$$\tilde{\mathbf{w}}(t, user) = f_{read-out,\ user}(\mathbf{s}_{K,j}) \tag{3.8}$$

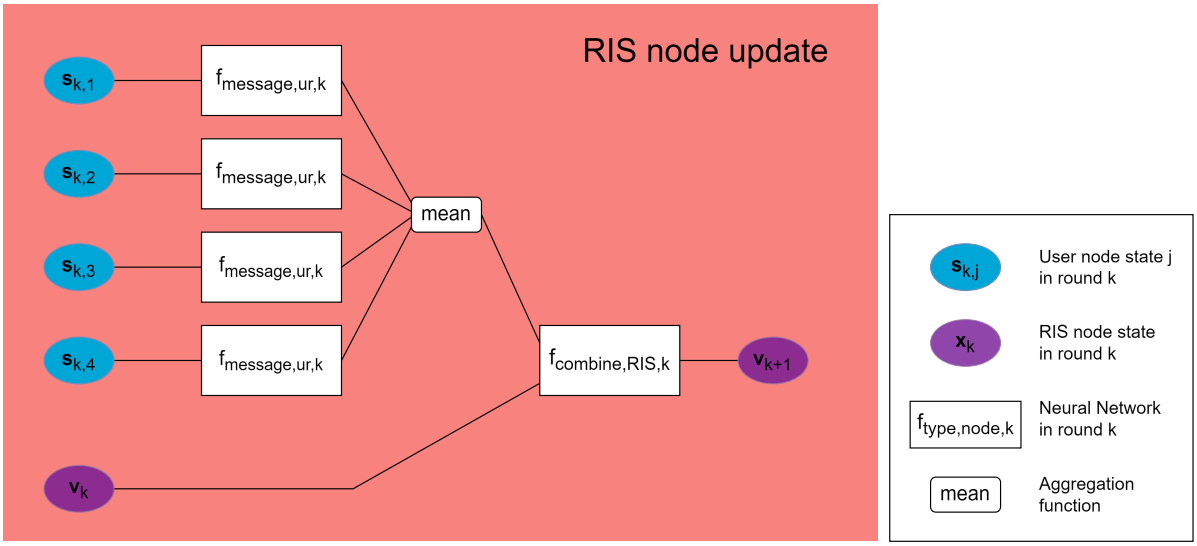


Figure 3.6: One round of message passing to update the state of the RIS node.

We define two variants of the GNN architecture by how the precoders are normalized. In the first variant, we assume an equal power distribution among the users. Each output vector is divided by the L2-norm of the corresponding vector individually, as shown in Equation 3.9. As a result, the power transmitted to every user is equal, though the power across the antenna elements still varies.

$$\mathbf{w}(t, user) = \sqrt{\frac{P}{N_{users}}} \frac{\tilde{\mathbf{w}}(t, user)}{\|\tilde{\mathbf{w}}(t, user)\|} \tag{3.9}$$

In the second variant, the GNN model performs an implicit power allocation. Instead of normalizing each precoding vector separately, normalization is applied collectively to all precoding vectors, by taking the Frobenius norm of the matrix $\tilde{\mathbf{W}}(t)$ containing all the unormalized precoding vectors. The result is then multiplied by the total power to ensure that the total transmit power of the base station of 120 W is not exceeded. This is shown in Equation 3.10.

$$\mathbf{w}(t, user) = \sqrt{P} \frac{\tilde{\mathbf{w}}(t, user)}{\|\tilde{\mathbf{W}}(t)\|_F} \tag{3.10}$$

A potential pitfall of this implicit power allocation is that, during the learning process, the GNN model might learn to assign a disproportionate amount of power to a single user or a small group of users to optimize the individual throughputs rather than designing the precoders intelligently such the interference is reduced by minimizing gain in each other's directions. Although this power concentration may initially increase throughput, it ultimately limits the overall combined throughput, as it sacrifices spatial multiplexing gains.

For the RIS configuration, the state of the RIS node is read out to obtain the RIS configuration. Similar to the read-out of the user nodes, a final output neural network is applied to the state of the RIS node. The result should be normalized such that the gain of each RIS element is equal to 1 for the full reflection of each RIS element. This can be achieved in two ways.
The first approach would be to output a real and imaginary part of a complex number per reflective element, which results in $2N_{elements}$ values. These are combined into complex numbers and then divided by their absolute values to ensure proper normalization.

$$\boldsymbol{\phi}(t) = \frac{\mathbf{v}_K[: N_{elements}] + \mathbf{v}_K[N_{elements} :] \cdot j}{\left|\mathbf{v}_K[: N_{elements}] + \mathbf{v}_K[N_{elements} :] \cdot j\right|} \tag{3.11}$$

Another approach would be to output the phase of each reflective element directly, which is subsequently converted into a complex vector using 3.12, which does not need to be normalized anymore since it results in an amplitude of 1. In this case, the final layer of the output network will have $N_{elements}$ values corresponding to the phases of the RIS elements.

$$\boldsymbol{\phi}(t) = e^{-j \cdot \mathbf{v}_K} \tag{3.12}$$

Although the two methods for the normalization of the RIS configuration are different, they are expected to produce equivalent results. The performance difference between the two methods will be evaluated in Chapter 4.

Concluding remarks

In this chapter, we presented the overall architecture of the Graph Neural Network. The GNN consists of multiple internal neural networks, which are listed below. All these neural networks don't need to be trained individually, but the GNN is trained as a whole. In the next chapter, we will dive deeper into the training process of the GNN. Specifically, we will explain the loss function for the optimization We will examine how different training parameters affect the performance of the training process, and we will look at ML techniques to improve the training process. The internal neural networks of the GNN include:

- Initialization networks: $f_{init,user}$, $f_{init,RIS}$
- Message-generation networks: $f_{message,uu}$, $f_{message,ur}$, $f_{message,ru}$
- Combining networks: $f_{combine,user}$, $f_{combine,RIS}$
- Read-out networks: $f_{read-out,user}$, $f_{read-out,RIS}$

We have defined a couple of hyperparameters of the GNN that need to be tuned for optimal performance. We will be looking at these hyperparameters to find the optimal GNN model configuration in the next chapter. The main hyperparameters are summarized in Table **??**.

Currently, the GNN produces both precoders and RIS configurations. However, further investigation is needed to determine whether the precoders should be retained or discarded to simplify the model without compromising performance. Additionally, we will evaluate the GNN's performance step-by-step to identify areas for improvement.

Parameter	Value
Number of message passing rounds	0,1,2,3
Power allocation	Equal, Implicit
RIS node output format	Complex number, Phase
SVD input reduction	Yes, no
User state size	64,128,256
RIS state size	64,128,256,512,1024,1800
Message size	64,128,256,512,1024
Neural network parameters	
Number of hidden layers	1,2,3
Number of neurons per hidden layer	64,128,256,512,1024
Activation function final layer	Linear, ReLU
Activation function hidden layer	Linear, ReLU

Table 3.1: Table with tunable hyperparameter for the GNN architecture.

4

Machine learning

Now, we know what the architecture of the GNN model looks like in terms of inference. In this chapter, we will delve into the machine learning aspects of the GNN model. First, we will explain the training process of the GNN with the corresponding loss function, how the training data is generated and how the training parameters are determined. We will step-by-step test the model starting from the scenario for a single user and no RIS, followed by the scenario for multiple users and no RIS, for a single user with RIS, and for multiple users with RIS. During the testing, we will make the necessary adjustments to ensure that the GNN works properly. When we have found the working model, we will do a final hyperparameter fine-tuning to obtain the optimal GNN model configuration. Finally, the inference time of this fine-tuned GNN model is measured and compared to the computation time required to derive the RIS configuration using an existing method.

4.1. Training

Our GNN is trained using unsupervised learning, which means that the model optimizes its performance without using labeled data. In contrast to supervised learning, where each training sample has a corresponding output label to which the model's output is compared, unsupervised learning does thus not require prior knowledge of the optimal solution. Instead, the model learns by optimizing a measurable performance metric based on the feedback from the loss function. We train our GNN using unsupervised learning because we don't have the optimal solution and we have a clear performance metric for which we want to optimize. For beamforming, we can apply Maximum Ratio Transmission (MRT) and Zero-Forcing (ZF). MRT leads to the best possible outcome for the single-user scenario. ZF does not necessarily lead to the most optimal precoders for the multi-user scenario in terms of total throughput but is still very effective. For the other objective of our model, finding the RIS configuration, we can use the fixed-point iteration algorithm. However, this method is specifically optimized for the single-user case in combination with MRT. Finding the optimal RIS configuration in the multi-user scenario, we do not know the optimal solution and the GNN has the potential to provide a more suitable RIS configuration by using unsupervised learning.

4.1.1. Loss function

We use the sum of the proportional fairness ratios to define the loss function for training our model, as given by Equation 4.1. Since the objective is to maximize the sum of proportional fairness ratios, we introduce a negative sign in front of the sum to ensure that minimizing the loss function leads to maximizing the sum of the proportional fairness ratios.

$$Loss(t) = -\sum_{user}^{N_{users}} \frac{R(t, user)}{R_{av}(t, user)} \tag{4.1}$$

Here, the throughput $R(t, user)$ for a given user is computed based on the precoder and RIS configuration obtained from the model. The throughput is then determined using Equation 2.21. For each user, it is divided by the average throughput the specific user has already experienced, $R_{av}(t, user)$, which is assumed to be an input variable to the GNN model. For the calculation of the throughput in the loss function, we remove the rate limit of 114.752 Mbit/s in Equation 2.21. It should help the model to improve on learning the underlying principle of optimizing the throughput, and it prevents discontinuity issues when calculating the gradient, which is based on the loss function. In the single-user case, given that the denominator is constant, the loss function boils down to just optimizing the $R(t, user)$ for the given single user.

Important to highlight is that the loss function is dependent on the outcome of the model, the input features of the GNN model and the channel response of the cell-RIS link. The GNN implicitly learns the impact of the cell-RIS channel response because the neural network weights are updated based on feedback from the loss function, which inherently includes the information about the cell-RIS link. As a result, the model adapts to the cell-RIS channel, even though this information is not provided as an input feature to the GNN. This cell-RIS channel response changes for different RIS locations and RIS sizes. A GNN model trained for one such scenario cannot be directly reused in another if one of these parameters has been changed. In real-life deployment, this would not be an issue because these parameters do not change dynamically. Thus, once a GNN has been trained for a specific scenario, it should be sufficient as long as there is no structural change.

However, in the testing phase, which will be described in Section 4.4, we have observed that for the single-user with RIS scenario to derive, the model was not able to learn the RIS configuration properly with this loss function. Instead, when training this particular model, an alternative loss function is used, considering the throughput $R(t, user)$ on the indirect link only, as will described in Section 4.4.

4.1.2. Datasets

We generate three different sets of samples: the training set, the validation set and the test set. The training set is used to train the model, which means optimizing the weights of the GNN internal networks. To prevent overfitting the training data, we use the validation set to monitor the training process and to determine the optimal model configuration. However, since the validation set influences the training process and the hyperparameter tuning, the GNN model may also become overfitted to this set. To ensure a truly unbiased evaluation of the GNN's performance, we use the test set, which remains completely unseen during training and validation. This final evaluation set provides an accurate measure of how well the model generalizes to new data.

Since we generate the datasets ourselves using our channel models, we can create as many samples as required. Choosing the right number of samples for each dataset is important for the performance of our GNN model. In general, increasing the number of samples in the training set improves the representativeness of the dataset. A small dataset can lead to a low training loss, but to a high validation loss, which is the loss calculated on the validation set, as the model may overfit to the small training set. This large gap between training loss and

validation loss indicates a high variance, meaning the model performs well on the training data but poorly on unseen data. Increasing the training set size helps to reduce the variance and to improve the GNN's ability to generalize. However, more training samples also lead to longer training times per epoch. For the training set, increasing the number of samples can thus improve the model performance, while for the validation and test sets, the number of samples should be large enough to provide statistically representative results but does not necessarily need to be as large as the training set. For the training set, we generate 10,000 samples. For both the validation and test set, we generate 3000 samples per set.

4.1.3. Optimizer

To minimize the loss function, an optimizer calculates gradients and updates the weights of the GNN's internal networks accordingly. Our GNN model is trained using the *ADAM (Adaptive Moment) optimizer* [31]. The Adam optimizer combines an adaptive learning rate with momentum-based methods. In adaptive learning rate methods, the learning rate is adapted based on past gradients. The learning rate determines the step size for each iteration as the model updates its weights to minimize the loss function. Adapting the learning rate improves convergence efficiency and speed. Momentum-based optimizing means that in the gradient descent, the loss moves more smoothly towards the minimum by using a moving average of past gradients, which will reduce oscillations in the gradients. Since the Adam optimizer results in fast and reliable training, it was elected as the optimizer for training our model.

Although the Adam optimizer adapts the learning rate during training, an *initial learning rate* must still be chosen. A lower learning rate may increase the accuracy but may also increase the risk of either taking too long to converge or getting stuck in a local minimum. On the other hand, a higher learning rate can cause the training to overshoot and miss the optimal solution. We have tried the following initial learning rates: $5 \cdot 10^{-3}$, $1 \cdot 10^{-3}$, $5 \cdot 10^{-4}$, $1 \cdot 10^{-4}$, $1 \cdot 10^{-5}$. Among these, the initial learning rate of $1 \cdot 10^{-3}$ provided a good balance between accuracy and convergence time.

In the training process, the weights of the GNN's internal neural networks are updated in batches of a fixed number of samples, the *batch size*. A smaller batch size generally leads to more accurate results, but this comes at a cost of longer training times [32]. For computational efficiency, it is recommended to choose a batch size that is a power of two [29]. We considered batch sizes of 16, 32, 64, and 128. Since we are aiming for an accurate result, we have chosen a relatively small batch size of 16.

4.1.4. Convergence of training

The training of the GNN model is done in epochs, where each epoch represents one complete pass through the entire training set. To improve generalization and prevent the model from memorizing the order of the data, the training samples are shuffled at the start of each epoch. Determining the appropriate number of epochs is essential to ensure good training. Too few epochs will lead to an undertrained model, while too many epochs may lead to overfitting on the training set. The optimal number of epochs depends on many factors, such as the size of the training set and the model configuration.

To determine this optimal number of epochs, we want to terminate the training when the loss has converged to a stable value based on a convergence criterion. To prevent overfitting on the training data, the convergence criterion is based on the validation loss at the end of each epoch. The model is trained in windows comprising multiple epochs because we don't want

to interrupt the training to determine the validation loss after each epoch. After each window, we compute the average validation loss and compare it to the average loss from the previous window. If the relative improvement decrease in validation loss is below a defined threshold, we consider the validation loss to have converged, and we stop the training. This convergence threshold is set to 0.1%. Selecting an appropriate window size is important. Choosing a window size that is too small can lead to accidental early stopping due to short-term variations in the loss, while a window size that is too large may cause unnecessary delays in determining convergence.

Initial experiments showed that a window size of 20 was too small, leading to a validation loss that was not properly converged. When we increased the window size to 50, we achieved a more robust convergence criterion. Additionally, to prevent stopping too early, we enforce a minimum of 100 epochs before applying the convergence criterion.

4.2. Generation of training data

In our simulation, we can generate our own training data. Each training sample is a different constellation of users in different locations, where the locations are uniformly distributed over the defined propagation environment, as illustrated in Figure 2.1.

For each user in each sample, the cell-user and RIS-user channel responses are generated as described in Chapter 2. For each channel response, the path loss and shadowing vary due to the different user locations. The additional blocker and the building penetration loss also vary as their modelling depends on random variables. The small-scaling fading also differs for each channel. To simulate this, we randomly select a trace and a timestamp in this trace generated. Subsequently, we incorporate the distance-based phase into the small-scale fading.

A coverage check, as described in Section 2.4.1, is performed on the samples. If a user is found to be out of coverage according to the coverage check, a new user is generated with a different location until a user within coverage is found. This process ensures that each sample contains the same number of users and that they are all covered.

Since we apply the SUS algorithm for scheduling, as described in Section 2.4.2.1, all co-scheduled users will have channels that are sufficiently orthogonal to each other. To ensure this property also holds in our dataset, we apply an orthogonality check when generating users for each sample. Equivalent to the coverage check, if the channel of a newly generated user is not sufficiently orthogonal to those of the already generated users, we discard that user and generate a new one until the orthogonality criterion is satisfied.

4.2.1. Average rates

Since our GNN is trained based on the sum of proportional fairness ratios, we need an average experienced throughput for every user. In the generation of the training set, the training samples are independently generated at different time instances. As a result, there is no naturally occurring average experienced throughput $R_{av}(t, user)$ for any user. Therefore, these average experienced throughputs need to be artificially created.

The average experienced throughput $R_{av}(t, user)$ for a given user is assumed to be positively correlated with the quality of its channel and inversely affected by the total number of served users. A higher channel quality typically results in a higher average bit rate for that user. With an increasing number of served users, the resources need to be shared among more users.

Therefore, the average experienced throughput for all the users is assumed to decrease. To model this, we sample the average experienced throughput $R_{av}(t, user)$ from a uniform distribution, for each user in each sample, as shown in Expression 4.2. The upper bound of this distribution is the maximum achievable bit rate for this user if it were scheduled in isolation, calculated with the corresponding MRT precoder and the RIS at the default configuration. The lower bound is set as the maximum achievable bit rate divided by the total number of served users since we expect this value to be higher than the equal share of the maximum achievable bit rate because of the multi-user gain. Using the maximum achievable bit rate $R_{max}(t, user)$ as an upper bound is an optimistic choice. However, the exact choice for $R_{av}(t, user)$ does not play a crucial role, as the primary objective is to make a reasonable approximation of the average experienced throughput. In practical deployments, these $R_{av}(t, user)$ values would be operationally available anyway, eliminating the need for this assumption.

$$R_{av}(t, user) \sim U\left(\frac{R_{max}(t, user)}{N_{users}}, R_{max}(t, user)\right) \tag{4.2}$$

4.3. Regularization

Since we will observe a significant gap between the training and validation loss in Section 4.4, we can conclude that the model will overfit on the training set. To overcome this, regularization techniques can be used. We have tried two regularization techniques, L2 regression and dropout.

L2 regression, also known as Ridge regression [33], is a regularization technique that adds an extra penalty to the loss for the weights of the neural networks. This discourages large weights in the neural network, which helps to reduce overfitting. Our GNN model comprises several internal neural networks. When we tried to apply L2 regularization, it was applied to all these internal neural networks simultaneously. The extra penalty term is shown in Equation 4.3. In this equation, ω_i represents the optimizable weights of the GNN and α is the L2 coefficient, which should be determined based on the model. Because of the many optimizable weights of the neural networks and the high order of magnitude of the loss, it is hard to estimate the correct order magnitude for the L2 coefficient α. Therefore, we have tried a wide range of L2 coefficients: $1 \cdot 10^{-4}$, $1 \cdot 10^{-3}$, $5 \cdot 10^{-2}$, 0.1, 1, 10, 100. Unfortunately, in none of these cases, the L2 regularization seems to have reduced the overfitting issue.

$$Loss_{L2}(t) = -\sum_{user}^{N_{users}} \frac{R(t, user)}{R_{av}(t, user)} + \alpha \sum_{i} \omega_i^2 \tag{4.3}$$

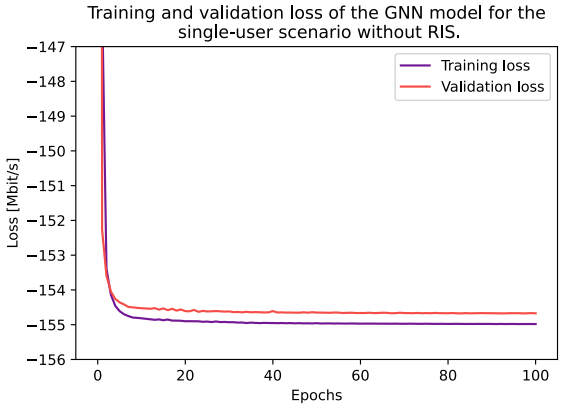
Another commonly used regularization technique is *drop-out regularization* [34]. In dropout regularization, neurons in the hidden layers of a neural network are randomly omitted with a certain probability, known as the dropout rate, during the training process. This helps to prevent the model from overfitting by ensuring that the network does not become overly dependent on specific neurons. As a result, the model will become more robust, and the model's ability to generalize should increase. In the context of GNNs, dropout regularization is often applied by randomly dropping nodes from the graph instead of neurons in the networks. However, this is not possible for our GNN model since the nodes correspond to the precoders at the final output. Instead, we have tried to apply dropout regularization to all the internal neural networks, similar to how for Ridge regression the penalty term was calculated with the weights of all the internal networks. We have tested dropout rates of 0.1, 0.2, and 0.5, but none of these values effectively reduced overfitting in the model.

4.4. Testing

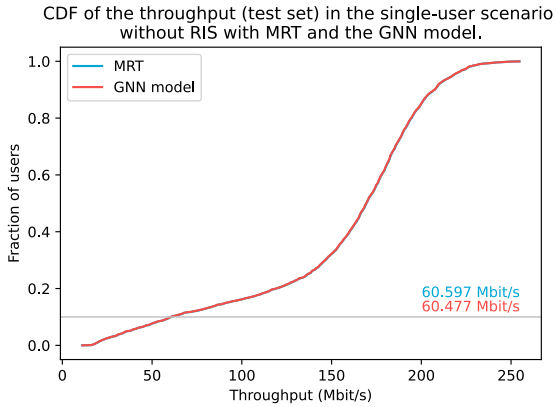
Our GNN is a large model with a lot of complexity, so we break down the testing process into different steps to ensure a thorough evaluation. We start with the single-user case without RIS, followed by the multi-user case without RIS. Next, we introduce the RIS in the single-user case, which leads to some adjustments in the loss function. Finally, we test whether the model can handle the most complex scenario: a multi-user mobile network with RIS.

4.4.1. Single user without RIS

At first, we start with the simplest scenario, the single-user case without a RIS deployed. In this case, the GNN's objective is to find a single precoder only. Since there is only one user and no RIS, there is only one node, which means there are no message passing rounds. The GNN in this scenario is reduced to a simplified form, in which we only have the preprocessing step, the initialization and the read-out of this single node. This testing step must, therefore, verify these steps and the loss function. We compare the throughput obtained from the GNN model to the throughput achieved using MRT precoding on the test set. Specifically, we evaluate both the 10^{th} percentile and the mean of the user throughput. Since MRT is the optimal precoder for the single-user scenario, we define an accuracy metric to quantify the performance of the GNN model relative to this benchmark. In Figure 4.1a, the training and validation losses over the training process are shown. In Figure 4.1b, the cumulative distribution function (CDF) of the user throughput is shown for the 3000 users in the test set, where each user is scheduled in isolation. The grey line indicates the 10^{th} percentile. The exact values of the 10^{th} user throughput using MRT and the GNN model are displayed on the right side above the grey line.



(a) Training and validation loss of the GNN in the single-user without RIS scenario.



(b) CDFs of the user throughput in the test set with the precoder derived via MRT and the GNN model.

Figure 4.1: The loss and resulting throughput distribution of the GNN in the single-user without RIS scenario.

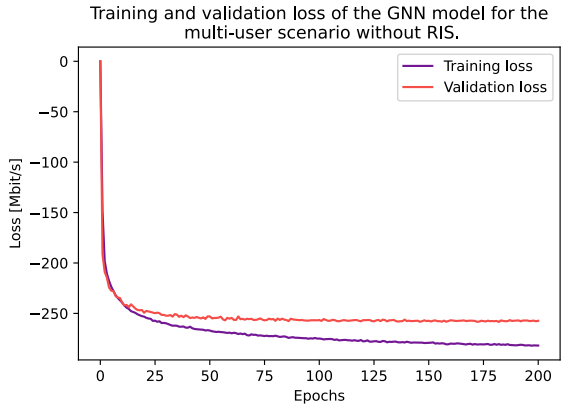
From the loss plots, we can conclude that the losses are fully converged. The small gap between the training and validation loss indicates that the model is not overfitting, which is a positive sign. However, the gap in the training and validation loss is larger than the difference between the GNN's throughput mean and the theoretical maximum of MRT, as can be seen in Figure 4.1b. This is caused by the fact that the training and validation loss are based on different datasets, whereas the two CDF curves in Figure 4.1b are both based on the same data set, the test set. The users in the training set are in slightly better positions than the users in the test set. As can be seen from the CDF plots, the distributions of the user throughput for the GNN model and MRT match almost exactly. The 10^{th} user throughput percentile and mean user throughput achieved using the GNN are extremely close to those obtained using MRT. The 10^{th} user throughput percentile achieved by the GNN model on the test set is 99.64% of

the 10th user throughput percentile when MRT is used. This test shows that the GNN model is capable of deriving the MRT precoder with very high accuracy with unsupervised learning.

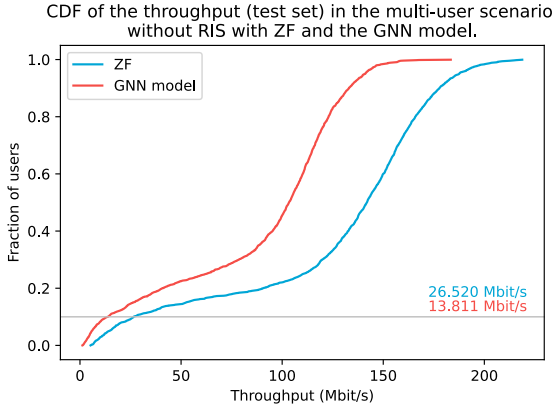
4.4.2. Multi-user without RIS

We will add another user node to the GNN. In this case, there will be message exchanges, but only between user nodes, which makes it a homogeneous GNN. This step must verify the message passing and the expected reduction of interference. We compare the results to the throughput obtained using ZF.

Figure 4.2a shows the training and validation loss of the multi-user GNN without RIS. The test set in this scenario contains 6000 users, which are scheduled in pairs that are sufficiently orthogonal to each other. The CDF of the individual throughputs of these users achieved using the GNN model and ZF is shown in Figure 4.2b. In the loss plot, we observe that after 200 epochs, the validation is fully converged, while the training loss is not. The accuracy of the 10th user throughput percentile of the GNN on the test set in this case is 52.08%, while the mean user throughput is 71.07%. This test shows that the GNN is capable of designing precoders such that the interference towards other users is reduced, but not as effectively as with using ZF.



(a) Training and validation loss of the GNN in the multi-user without RIS scenario.



(b) CDFs of the user throughputs in the test set with the precoder derived via ZF and the GNN model.

Figure 4.2: The loss and resulting throughput distribution of the GNN in the multi-user without RIS scenario.

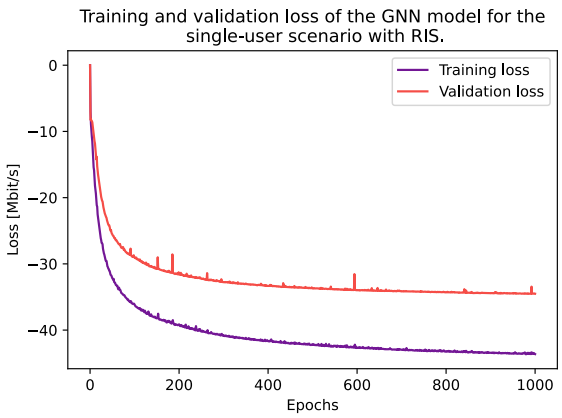
4.4.3. Single-user with RIS

Now we introduce the RIS node into our GNN while reducing the number of users again to a single user. The model will provide one precoder and a RIS configuration. There are no messages between users, but there are message exchanges between the user node and the RIS node. As observed in the multi-user case without RIS, the precoders generated by the GNN model cause significant inaccuracies compared to the benchmark. Since MRT is a low-complexity method for precoding, we discard the precoders from the GNN model and instead use the GNN solely to determine the RIS configuration. The precoder is calculated using MRT on the full channel response of Equation 2.2 with the RIS configuration of the GNN model. MRT is also incorporated into the training process, which means that in the loss function the throughput is calculated with the MRT precoder and the GNNs RIS configuration. The model still contains the user node to share necessary information with the RIS node.

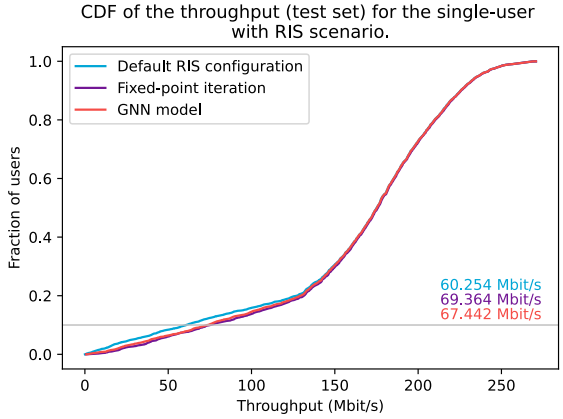
However, with this setup, the model is also not able to learn a proper RIS configuration. We

found that training the GNN model on only the indirect path significantly improved its performance. This means that in this approach, the throughput in the loss function is calculated as if there is only the indirect path ($\mathbf{H}_{cr}\mathbf{\Phi}(t)\mathbf{H}_{ru}(t, user)$). Training in this manner ensures that the model focuses entirely on optimizing the RIS configuration. For the inference of the GNN on the test set, the throughput is calculated using both the direct link and the indirect link as it would be in real-world deployment. To evaluate the RIS configuration from the GNN model, we compare it to the default RIS configuration and the fixed-point iteration, as described in Section 2.4.4.1.

The results of the loss in the training process and the resulting throughputs of the adjusted GNN are shown in Figures 4.3a and 4.3b, respectively. As can be observed, the loss is significantly lower than in Figure 4.1a because of the exclusion of the direct path. The training of this GNN model takes many more epochs compared to the cases without RIS. This is expected because the input and output space of this GNN model is much larger, due to the large number of configurable RIS elements. Both the training and validation loss seem to be converged. However, there is still a significant gap between the training and validation loss. The CDF plot shows that the GNNs RIS configuration achieves a gain in throughput compared to the default RIS configuration. This gain is mainly for the users with a weak channel. The GNNs throughput distribution almost matches the throughput distribution achieved using the fixed-point iteration algorithm, although the 10^{th} user throughput percentile is only 2 Mbit/s lower than when using the fixed-point iteration algorithm. Thus, the GNN model provides a result that is almost as good as the fixed-point iteration algorithm, which is a rather slow algorithm. In Section 4.6, we will evaluate the inference time of the GNN and determine how much faster it is compared to the fixed-point iteration algorithm.



(a) Training and validation loss of the multi-user without RIS GNN.



(b) Training and validation loss of the GNN in the single-user with RIS scenario.

Figure 4.3: The loss and resulting throughput distribution of the GNN in the single-user without RIS scenario.

4.4.4. Multi-user with RIS

For the multi-user with RIS scenario, we were unable to get a working GNN model due to time constraints. A good starting point for the multi-user with RIS scenario is to build upon the observations we found in the single-user with RIS scenario. This means that the GNN is only used to obtain the RIS configuration while deriving the precoders with zero-forcing. Training on the indirect link only might be an effective way for training strategy as it gives full attention to optimizing the RIS.

4.5. Hyperparameter tuning

In Chapter 3, we have already defined a list of tunable hyperparameters for the GNN. A couple of hyperparameters were tuned during the testing phase for the different cases. To refine the model configuration, we now perform a comprehensive hyperparameter tuning for the GNN in the single-user with RIS scenario, which is trained on the indirect path. Given the large number of hyperparameters, we focus on the parameters that have a significant impact on the performance of the GNN. The hyperparameters whose values were more straightforward to determine, are left out of this fine-tuning. We will perform a grid search, which means we will test every possible combination of the selected hyperparameters. The hyperparameters with their corresponding values that we will test in this grid search are shown in Table 4.1, which leads to 600 total model configurations we will test.

Parameter	Value
Phase output	Complex number, phase
Number of hidden layers (depth)	2, 3, 4
hidden layer sizes (width)	64, 128, 256, 512, 1024
hidden layer activation function	Linear, ReLU
Message size	64, 128, 256, 512, 1024
Output layer activation	Linear, ReLU

Table 4.1: Table with parameters for the grid search on the single-user GNN including the RIS to determine its optimal configuration.

Some of the hyperparameters in this grid search pertain to the internal neural networks of the GNN. Due to computational constraints, we were unable to tune each internal neural network configuration individually. Instead, we will test general hyperparameters that apply to all the internal neural networks. The 600 GNN models are evaluated on the 10^{th} user throughput percentile of the validation set. The three best GNN model configurations are presented in Table 4.2 with the corresponding results. The final tuned hyperparameters of the single-user with RIS GNN are shown in Table 4.3. The best performance of the GNN is achieved when the RIS output consists of complex numbers, with two hidden layers for every internal network with linear activation functions, and a ReLU activation function in the output layer. Additionally, configurations with smaller hidden layer sizes and message dimensions generally tend to yield better results.

4.6. Inference time

One of the main reasons to look into a machine learning-based model was because the existing methods to obtain the RIS configuration are too computationally complex. Ideally, the RIS configuration should be updated once every 20 ms (40 slots), necessitating an extremely low inference time for our GNN model. We test the inference time of the GNN model for the single-user with RIS scenario and compare it to the calculation time to derive the RIS configuration using the fixed-point iteration algorithm. This test is conducted on the test set that contains 3000 samples, with each sample processed sequentially in batches of size 1. The inference time experiments were conducted on a server equipped with an Intel Family 6, Model 55 processor with 48 cores operating at 2.6 GHz, along with 128 GB of RAM.

For the fixed-point iteration algorithm, it takes 387.49 s to compute the RIS configuration for the entire test set. This is a calculation time of *129.16 ms per sample* for the fixed-point iteration algorithm, which exceeds the desired calculation of 20 ms. The inference time on the entire test set using the GNN was 5.28 s, which means it took only *1.76 ms per sample*. This means

GNN configuration	10th user throughput percentile [Mbit/s] - training set	10th user throughput percentile [Mbit/s] - validation set	Training loss [Mbit/s]	Validation loss [Mbit/s]	Epochs
Complex, 2, 128, Linear, 128, ReLu	69.515	67.442	-33.878	-24.418	1050
Complex, 2, 256, Linear, 64, ReLu	67.978	65.897	-32.387	-23.186	1150
Complex, 2, 128, Linear, 64, ReLu	68.009	65.778	-32.782	-23.939	1000

Table 4.2: The three best GNN configurations for the single-user with RIS scenario from the grid search, based on 10th user throughput percentile on the validation set.

an approximate 73× speedup compared to the fixed-point iteration algorithm. Additionally, the GNN's inference time of 1.76 ms per sample is well below the target threshold of 20 ms, demonstrating its potential for real-world deployment.

Concluding remarks

We successfully developed a GNN model capable of deriving a RIS configuration for the single-user scenario that is much faster than the fixed-point iteration algorithm. This GNN is trained exclusively on the indirect cell-RIS link, and we have determined its optimal configuration through systematic evaluation. However, we were not able to achieve satisfactory results for the GNN in the multi-user scenario including the RIS, because of the limited time we had. In the next chapter, we will conduct an analysis of how our single-user GNN model performs across different mobile network scenarios.

Parameter	Value
Number of message passing rounds	0,1,2,3
Power allocation	Equal, Implicit
RIS node output format	Complex number, Phase
SVD input reduction	Yes, no
User state size	64,128,256
RIS state size	64,128,256,512,1024,1800
Message size	64,128,256,512,1024
Neural network parameters	
Number of hidden layers	1,2,3
Number of neurons per hidden layer	64,128,256,512,1024
Activation function final layer	Linear, ReLU
Activation function hidden layer	Linear, ReLU

Table 4.3: Table with tuned hyperparameter of the single-user GNN including the RIS.

5 Results

In this chapter, we present the performance achieved using the GNN model that is found in the previous chapter, and we analyze how it is affected by different scenarios. Before doing so, we introduce the different scenarios that we will analyze, the benchmark methods we compare our GNN to and the key performance indicators that are used to assess the performance.

5.1. Results across different mobile network scenarios

We will analyze the performance achieved using the single-user GNN model across different scenarios based on six parameters. The parameters are listed in Table 5.1 with the corresponding range of values they can take. We have defined a default scenario, in which the RIS has 900 elements, the RIS is located at position B, the cell is characterized by an ISD of 200 m, the blockage probability is 20%, the user is indoors, and the phase shifts of the RIS elements are not discretized. For all other scenarios, we vary a single parameter while keeping the other parameters fixed at the default value to observe how this parameter impacts the performance. Each GNN model is trained on the specific scenario. This means that new datasets are generated for each scenario with the corresponding parameters for training, validation and testing. As a result, the channel characteristics differ across datasets due to variations in user locations and the stochastic nature of the dataset generation process, as described in Section 4.1.2. Each dataset used in the evaluation in this chapter contains 3000 samples to ensure that it is statistically representative. For the analysis of the phase discretization, the GNNs are not specifically trained on the scenarios because we will apply the discretization after the inference, which is thus not part of the GNN model itself.

To assess the performance achieved using the GNN model, we are comparing it to the results

Parameter	Value	Units	Symbol
Number of scheduled users	1	-	S
Number of RIS elements	100, 400, 900, 1600	-	$N_{elements}$
Location of RIS	A, B, C	-	-
Inter-site distance	100, 200, 300, 400, 500	m	ISD
Blockage probability	0, 20, 40, 60, 80, 100	%	p
Type of users	indoor, outdoor	-	-
Phase discretization	1, 2, 3, cont.	bits	B

Table 5.1: Table with the simulation parameters, with the default values underlined.

achieved using two benchmark methods. In the first benchmark method, the RIS in the default configuration, which means that the RIS is not optimized and reflects as a mirror. The other benchmark method is the RIS configuration obtained using the fixed-point iteration algorithm [26], as described in Section 2.4.2.1. In both benchmark methods, the precoder is derived via MRT on the full channel with the corresponding RIS configuration. The three methods are all applied on the same dataset, which is the test set that contains 3000 samples for each scenario.

We define two key performance indicators (KPIs) we will use to assess the performance achieved using the GNN. These are the:

- 10^{th} user throughput percentile
- Mean user throughput

The first KPI is the 10^{th} user throughput percentile. This is a good metric to assess the performance of users with a weak received signal strength. We are particularly interested in this because we expect the RIS to be more effective for these users. The second KPI is the mean user throughput, which we also expect to increase, though to a lesser extent, as it includes users with strong channels who generally benefit less from deploying a RIS.

5.1.1. Impact of the RIS size

We begin by analyzing how the performance achieved by using our GNN is affected by different RIS sizes. Specifically, we will look at four different numbers of RIS elements: 100, 400, 900 and 1600. This comparison is interesting not only from a wireless communication perspective but also from a machine learning perspective. Since we have different numbers of RIS elements, the output dimensions of our machine learning models will also change. This analysis therefore also evaluates whether the model can effectively handle outputs of a higher dimensionality. It is important to note that the hyperparameter grid search in the previous chapter was conducted on the GNN with 900 RIS elements. The same hyperparameters will be applied to the GNN models with 100, 400, and 1600 elements. This means that the GNN may not be optimally tuned for their respective RIS sizes.

The 10^{th} user throughput percentiles for the different RIS sizes are presented in Figure 5.1. From a physical perspective, the hypothesis is that the performance gain from properly configuring the RIS compared to configuring the RIS to the default setting will increase as the number of RIS elements grows. This trend becomes evident when observing the 10^{th} user throughput percentile for both the fixed-point iteration algorithm and the default configuration. Where the 10^{th} user throughput percentile of the default RIS configuration does not show any significant improvement when the number of RIS elements exceeds 400, the performance of the fixed-point iteration algorithm continues to increase with the number of RIS elements.

The performance obtained using the GNN does not follow this trend. The GNN model outperforms the fixed-point iteration algorithm for 100 RIS elements, but its performance gain flattens as the RIS size increases. While the GNN still provides improvements over the default RIS configuration, it does not scale as effectively as the fixed-point iteration algorithm when the RIS size becomes larger. This suggests that the GNN does not fully exploit the potential of larger RIS deployments. From an ML perspective, the results indicate that the GNN model works better for smaller RIS sizes. A possible solution could be to increase the neural network size, such as expanding the number of neurons in the hidden layers and the state sizes

of the RIS. However, we have seen that the GNN for 900 RIS elements does not increase in performance with more complexity.

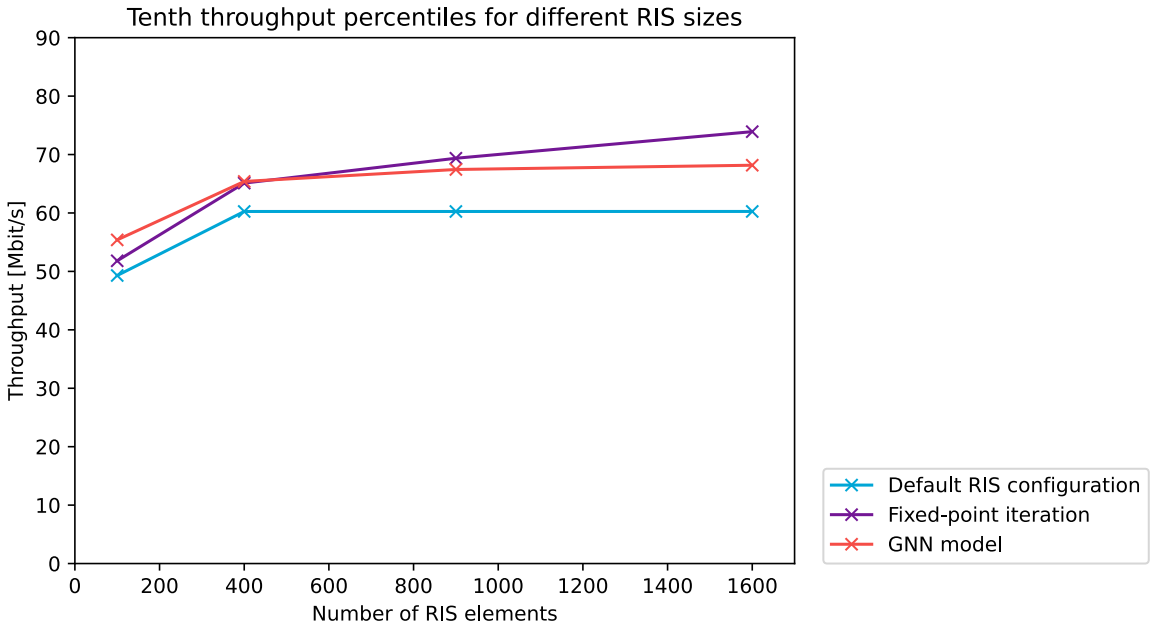


Figure 5.1: Performance for different RIS sizes.

5.1.2. Impact of the RIS location

Next, we want to examine the impact of the location of the RIS. We have defined three locations for deploying the RIS: positions A, B, and C. These locations are strategically chosen at the edges of our defined simulation area, as illustrated in Figure 2.1 in Chapter 2. The performances for the different RIS locations are shown in Figure 5.2.

For RIS location A, the gains of properly configuring the RIS using the fixed-point iteration algorithm and the GNN model are higher compared to gains in location B and even more so than in location C, both in the mean user throughput and the 10^{th} user throughput percentile. To explain why this is the case, we need to understand two things. First, users located close to the RIS benefit the most from the RIS, as there is less path loss on the RIS-user link. Secondly, the cell-RIS distance is larger for location A than for locations B and C. The users close to the RIS in RIS location A are cell-edge users that are also farther away from the base station and were already in the lower end of the users in terms of received signal strength. So, the RIS will improve the throughput of these users, and this contributes more in the 10^{th} user throughput percentile of the users. The RIS's contribution to improving these users at the cell edge outweighs the disadvantage of location A being farther away from the base station.

We can also observe another effect related to the cell-RIS distance. Although the gain from the RIS decreases as the RIS is closer to the base station, the gain in 10^{th} user throughput percentile of the fixed-point iteration algorithm relative to the GNN model decreases as the RIS goes from position A, to B, to C. This can be explained by the ML behind the GNN model. The GNN is trained on the indirect path only in our current configuration. Since locations B and C are closer to the base station, the indirect link is also stronger. Because of this stronger

link, the GNN is able to learn more effectively. Although the potential of RIS in locations B and C is lower, the GNN learns better how to configure the RIS.

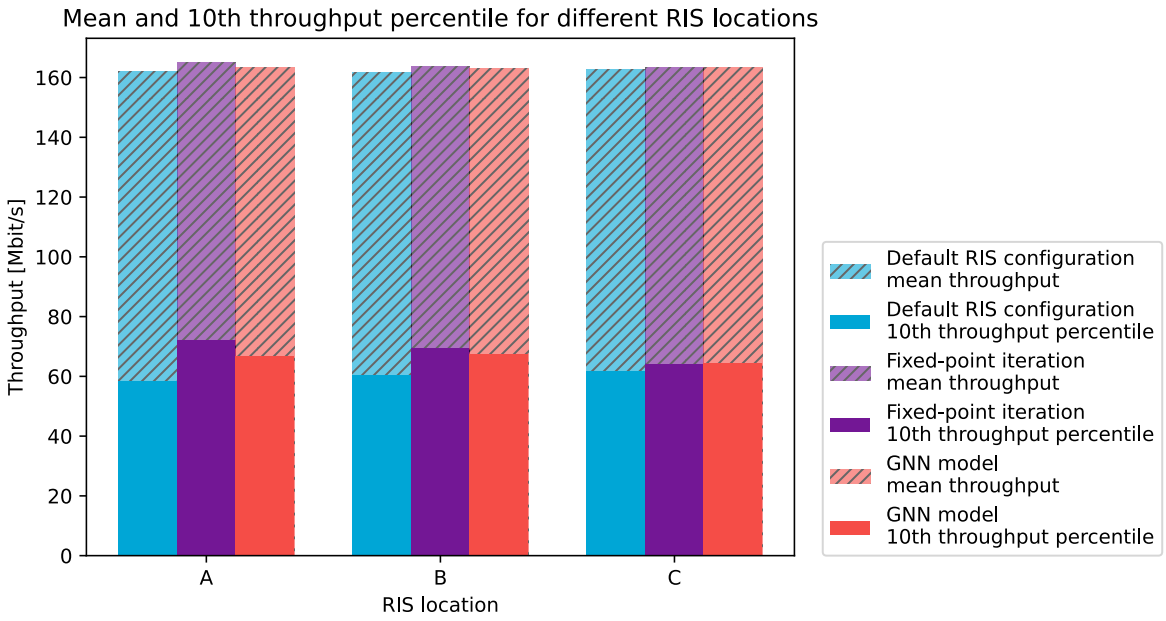


Figure 5.2: Performance for different RIS locations A, B, and C.

5.1.3. Impact of the cell size

In this section, we will investigate how the size of the cell affects the performance achieved by using the GNN, the fixed-point iteration algorithm, and the default RIS configuration. The mean user throughput and the 10^{th} user throughput percentile for different cell sizes can be seen in Figure 5.3. As the cell size increases, users are, on average, located farther from the base station, leading to greater path loss in the cell-user link. The distance between the RIS and the users also increases, on average, which will lead to more path loss on the indirect link. This will affect the effectiveness of the RIS.

We observe both effects in the plot. The overall performance decreases as the cell size increases, and the performance gain from properly configuring the RIS also declines with larger cells. The overall trend shows that the RIS is more effective in smaller cells.

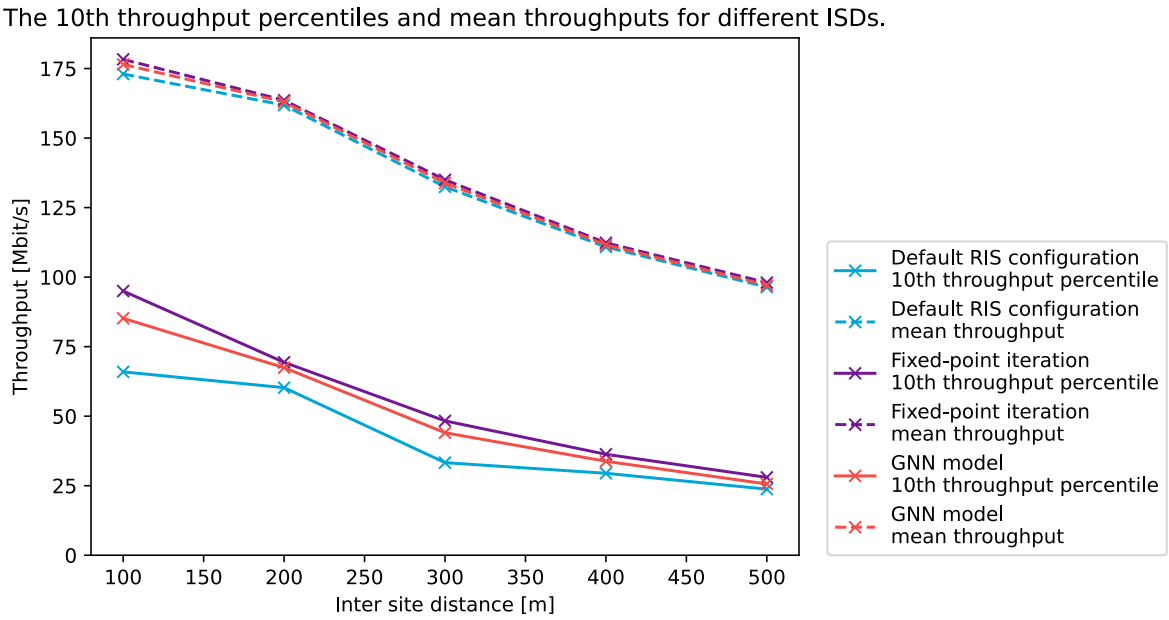


Figure 5.3: Performance for different cell sizes.

5.1.4. Impact of the blockage probability

The additional blocker as described in Section 2.3.1.5 has a significant influence on the potential gain from RIS. It is a loss imposed on both the cell-user link and the RIS-user link with a blockage probability p, which we had set to 20% for our default scenario. In this section, we will analyze the impact of the blockage probability on the gain from the RIS. For this analysis, we will be looking at different blockage probabilities for two different cell sizes, the default size with an ISD of 200 m and a cell that is characterized by an ISD of 500 m. In case of a low blockage probability, the direct link is already quite strong and the user will therefore not benefit much from the RIS. On the other hand, when the blockage probability is high, the direct link is more likely to be weak, but the alternative path via the RIS may also be blocked, and the RIS will therefore not be very effective as well. We expect the most gain from the RIS when the direct cell-user link is blocked, but the indirect link is not blocked. This happens with a probability of $p(1-p)$, which is maximal if the blockage probability is 50%. In our analysis, given the set of considered blockage probabilities, we expect the most gain from the RIS in the case of a blockage probability of 40% and 60%. In these scenarios, the probability that only the indirect path remains unblocked is 24%.

The results for varying the blockage probability for an ISD of 200 m are shown in Figure 5.4. As expected, the mean user throughput decreases as the blockage probability increases. The gain from the RIS is the highest at a blockage probability between 20% and 60% and is negligible for the 0% and 100% scenarios. This aligns with our expectations. For the 80% scenario, we observe that the GNN model is not performing well, while the fixed-point iteration does show a gain compared to the default RIS configuration. This can be explained by the fact that the indirect path is significantly weaker in this scenario, making it more difficult for the GNN model to learn how to effectively configure the RIS.

For the scenarios with the cell size that is characterized by an ISD of 500 m, we observe not much gain from the RIS for blockage probabilities close to either 0% or 100%, similar to in the

200 m case and in line with our expectations. The most gain from RIS is again in the 20% to 60% range of the blockage probability, with 60% yielding the highest gain.

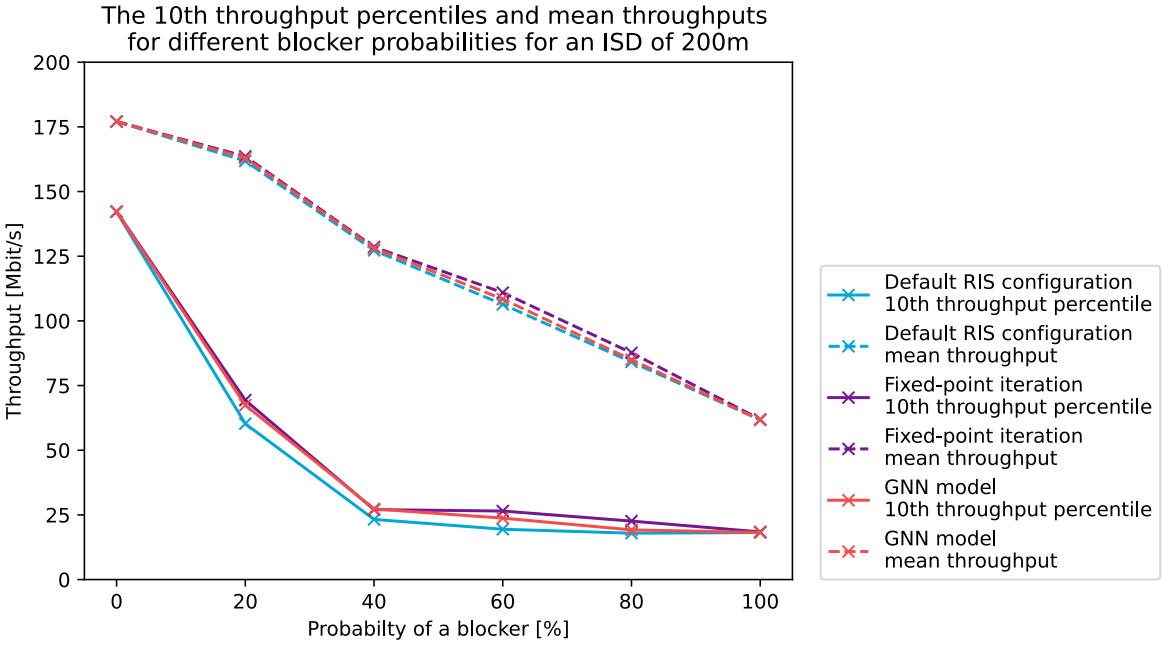


Figure 5.4: Performance for different blocker probabilities for an ISD of 200 m.

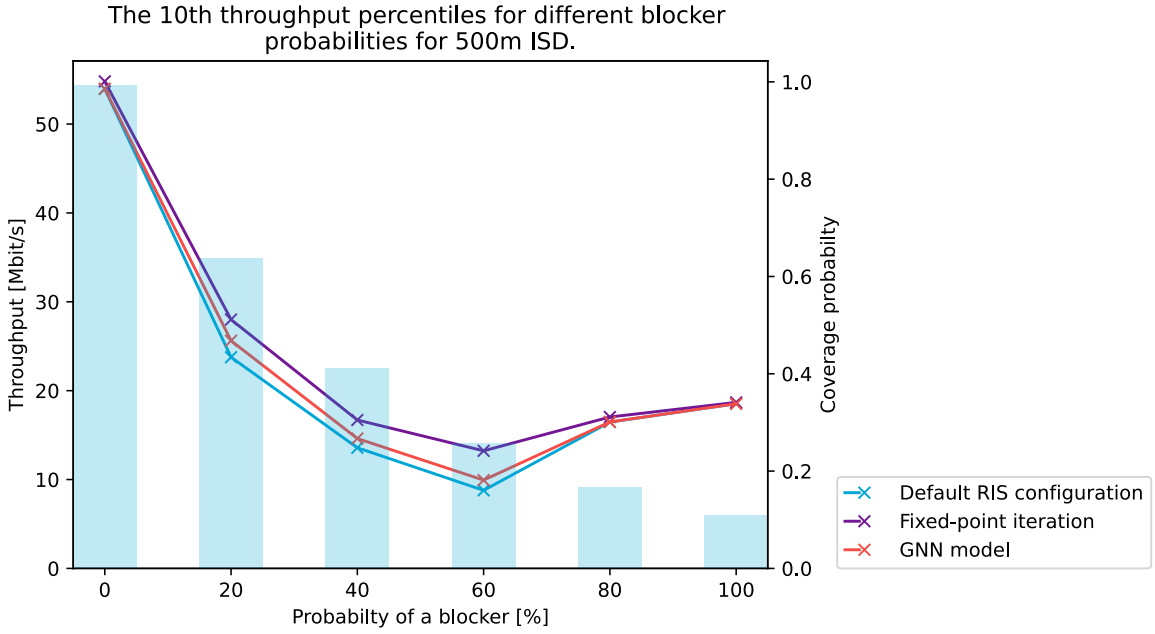


Figure 5.5: Performance for different blocker probabilities for an ISD of 500 m.

A remarkable trend is observed when the blockage probability reaches 80% and 100%. The 10^{th} user throughput percentile is unexpectedly higher than the 60% blockage scenario. Since this effect is also present with the default RIS configuration, it is not caused by the RIS con-

figuration techniques, but by the dataset itself. As described in Section 4.2, a coverage check is applied during dataset generation to ensure that only covered users are included in the dataset. In scenarios with a larger cell size and high blockage probability, this filtering process may result in a dataset where users are, on average, located in more favourable locations. This can inflate the 10^{th} user throughput percentile metric. To give a representative picture, we also need to show the coverage probability, which can be seen in Figure 5.5 in light blue. As expected, the coverage probability decreases as the blockage probability increases. For the blockage probabilities of 80% and 100%, the coverage probability is significantly lower compared to scenarios with lower blockage probabilities, but for the users that do have coverage in these high blockage probability scenarios, the 10^{th} user throughput percentile is higher.

5.1.5. User type

In a mobile network in which we want to deploy a RIS, there will likely be indoor and outdoor users. Up to this point, we have only considered indoor users in our study. To fully understand the effectiveness of the RIS with the configuration obtained from our GNN, we now evaluate its performance on outdoor users as well. The performance comparison between indoor and outdoor users is shown in Figure 5.6. Outdoor users are expected to receive a higher signal strength than indoor users because of the building penetration loss experienced by indoor users, as described in 2.3.1.4.

This is clear when looking at the chart. For both indoor and outdoor users, the GNN provides a performance gain compared to the default RIS configuration. Interestingly, the gain from properly configuring the RIS is higher for outdoor users than for indoor users, which is not immediately intuitive. Typically, in high-SINR conditions, achieving further gains is more challenging due to the logarithmic relationship between SINR and throughput. However, in this case, the higher gain for outdoor users can be explained by the fact that the RIS-user link is also stronger, as there is no building penetration loss on this link. Consequently, the RIS can provide a more effective enhancement in these scenarios.

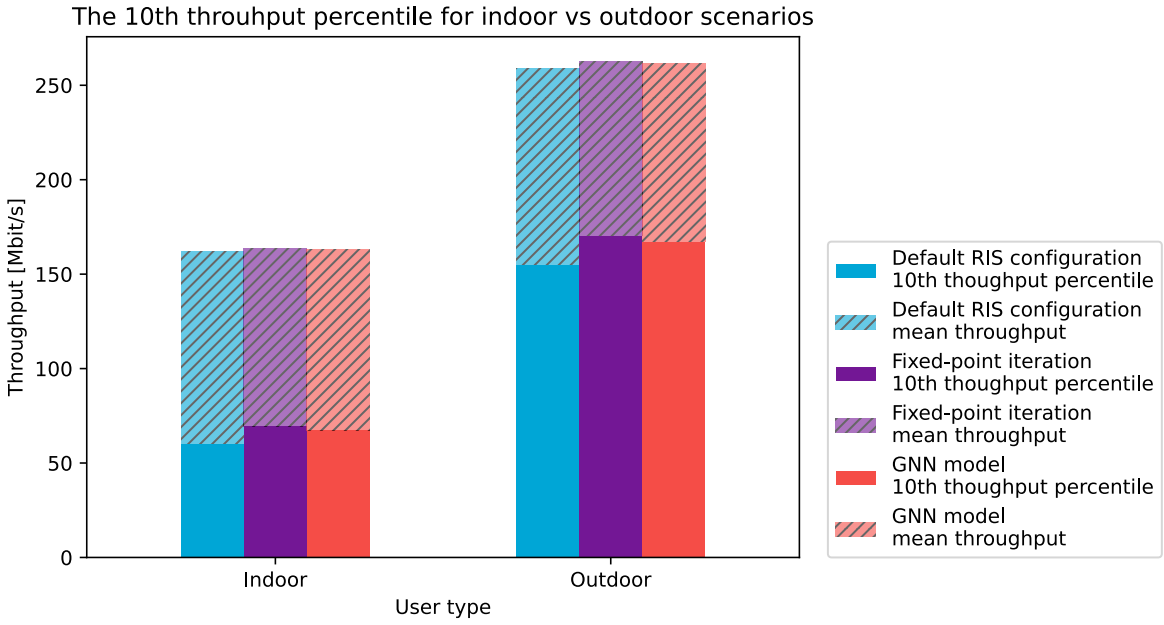


Figure 5.6: Performance of indoor users and outdoor users.

5.1.6. Phase shift discretization

In many practical implementations of RISs, the elements can only shift the phase of the reflected signal by a discrete set of phase shifts. That is why we analyze the performance achieved using the GNN with phase discretization in this section. The bit resolution B represents the number of bits per RIS element used to control this phase shift, leading to 2^B possible phase shifts. The ideal case in which the phase shift is continuous is equivalent to having an infinite bit resolution ($B = \infty$). In this analysis, the phase shifts are discretized after obtaining the RIS configurations from the GNN model and the fixed-point iteration algorithm. In the default RIS configuration, none of the RIS elements shift the phase of the reflected signal. The performance for the default configuration remains unchanged regardless of the bit resolution changes.

The performance in terms of the 10$^{\text{th}}$ user throughput percentile for different bit resolutions is shown in Figure 5.7. With a bit resolution B of 1, each element can either maintain the phase or shift it by π radians. Even at this low resolution, the RIS already provides a significant performance gain using both the GNN model and the fixed-point iteration algorithm. A higher bit resolution will reduce the discretization error and lead to a more accurate approximation of the actual of the GNN model and the fixed-point iteration algorithm, and is thus assumed to provide better performance. As expected, the 10$^{\text{th}}$ user throughput percentile improves with increasing bit resolution. The difference in 10$^{\text{th}}$ user throughput percentile between the 3-bits resolution and the ideal continuous phase shift is minimal for both the GNN model and the fixed-point iteration algorithm. This suggests that a bit resolution B of 3 is already sufficient for practical implementation in this scenario. Increasing the resolution further would require higher hardware complexity and cost, while providing only marginal performance gains.

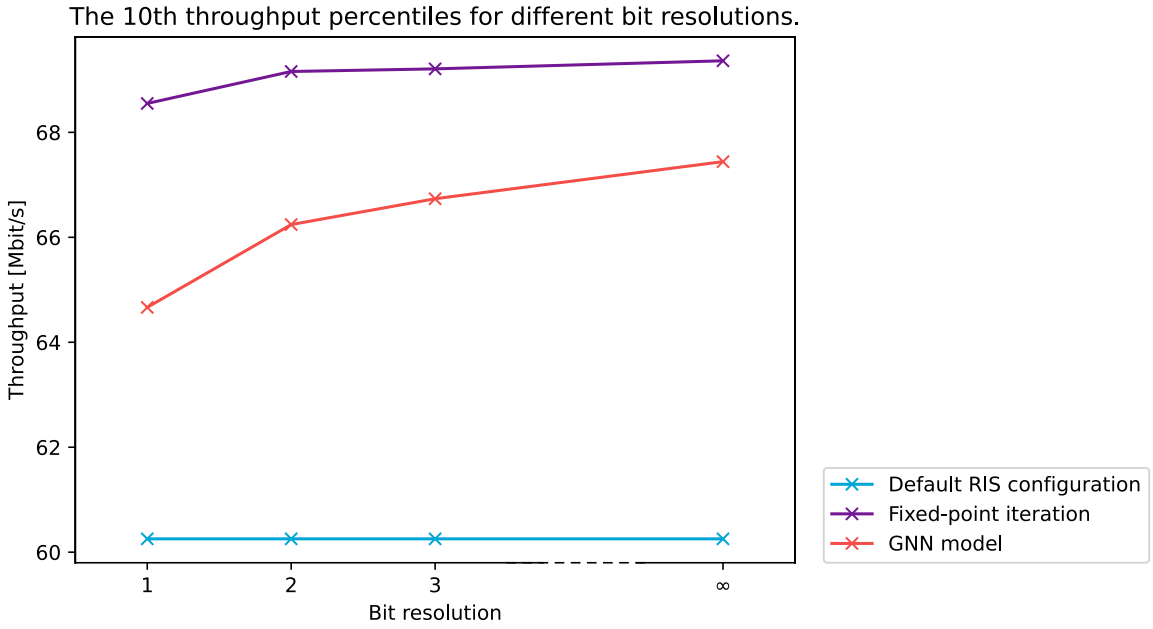


Figure 5.7: Performance for different phase shift discretization in number of bits per RIS element.

6

Conclusion

In this chapter, we will reflect back on this thesis. We start by having a discussion on the research conducted, we will draw some conclusions from the results of this work and finally, we will give some recommendations on possible future research.

6.1. Discussion

One of the objectives of this study has been to explore the possibilities of ML for RIS-related radio resource management tasks. From the literature review, we concluded that GNNs would be the most suitable approach because of their scalability in handling a varying number of users. The architecture of our GNN is not exactly the same as one of the GNNs from the literature, because of different optimization objectives and available input information. Instead, we have built upon a previous study on RIS-related radio resource management [2], and we developed our own GNN from scratch. However, this was to ensure that we considered a more realistic channel model for our study.

The GNN is a very complex model, both in terms of the number of hyperparameters and the number of trainable parameters, which makes it difficult to work with. The high number of hyperparameters and architecture choices makes it practically impossible to properly cross-validate all options, which means that there might be GNN configurations that achieve even higher performances. The combination of using GNNs, which inherently have a lot of trainable weights because of the internal neural networks, with the high dimensionality of both the input and the output space, resulted in a vast number of trainable parameters. We observe that the high number of weights led to overfitting in many test cases.

The methods employed to overcome this, such as the regularization techniques, were not able to achieve the desired performance in this thesis, which may be due to bad tuning or the fact that we applied these techniques on all the internal networks jointly.

6.2. Conclusion

In this work, we have successfully developed a GNN model to derive precoders and a RIS configuration for the scenario with a single user including a RIS. The GNN model, which is trained based on the indirect link only, is capable of deriving a RIS configuration, while we derive the precoder via MRT. The RIS configuration derived from our model manages to achieve a large fraction of the attainable performance gains within substantially less time than the fixed-point iteration algorithm [26], thus making it more operationally valuable in scenarios

with time-varying channels. It even outperforms the fixed-point iteration algorithm in terms of throughput for some scenarios, such as the scenario in which the RIS comprises 100 elements. We observed a large gap between training and validation loss. This may indicate overtraining, which means that there is still a lot of potential for even better results.

We have tested the performance that can be achieved by using our developed GNN across different scenarios and compared it to the fixed-point iteration algorithm. We observed that better results from the GNN were obtained relative to the benchmark when the number of RIS elements and, thus, the number of trainable parameters was lower. From this, we concluded that lower-complexity GNN models lead to better performance. However, smaller RISs also come at the cost of a lower gain in performance when properly configuring the RIS, which is a trade-off to be made. We also looked at different deployment locations for the RIS and different cell sizes. The gain from using our GNN was higher if the RIS was located farther away from the base station because it effectively improved the throughput for weaker users. However, when increasing the entire cell size, the gain obtained using the GNN decreases. Additionally, we observed that outdoor users benefited more from using the GNN model than indoor users in our study and that the most gain from the RIS is achieved when blockage probability is between 20% and 60%, in which the GNN manages to achieve a large fraction of that potential gain. Lastly, when analyzing the effect of the phase discretization, we concluded that the 3-bits phase resolution is already sufficient, and increasing this further will only provide marginal performance gains.

Even though we were unable to make it work for all scenarios due to time constraints, we have shown the potential of GNNs. This work serves as a valuable foundation for follow-up work.

6.3. Future Research

This brings us to the recommendations for future research. One of the main motivations for adopting the GNN framework was the scalability of the number of users that the GNN offers. For the continuation of this project, further research is needed to develop a GNN for the multi-user scenario including the RIS to fully leverage the scalability potential of GNNs in multi-user RIS-assisted networks.

The next logical step, after successfully developing a multi-user GNN, is the implementation of user scheduling. Although we have proposed a method for incorporating user scheduling, we have not yet implemented or tested it. Integrating user scheduling could further enhance the performance of a mobile network. This would be a novel contribution, as most of the literature we have encountered during our literature review does not consider scheduling within a GNN-based RIS-assisted mobile network.

Another interesting direction for future research is the addition of multiple receive antennas per user. The current implementation already supports handling multiple receive antennas. However, to reduce complexity, we limited our numerical study to a single receive antenna per user. Using the GNN trained with unsupervised learning is a promising approach to find a solution. For this possible research direction, it also holds that in the studied literature, we did not encounter studies that explore multiple receive antennas in the context of GNNs for RIS-related radio resource management, making this a novel and valuable research direction.

Introducing extra RISs could be another extension to this work. The GNN framework allows it, as extra RIS nodes can be integrated into the GNN model. This is something that also has

already been done by other studies, but can also enrich our model.

Further research can also focus on reducing the dimensionality of the input space to reduce the gap between the training and validation loss. Given that the GNN needs to handle high-dimensional input data in which the underlying pattern is important, exploring image recognition techniques could be a promising direction for future studies.

Our simulations have focused on the single-cell scenario. However, in real-world mobile networks, multiple cells coexist, where users and RISs operate in overlapping cell areas. Our GNN can be extended to the multi-cell scenario. This would likely introduce the need for a cell-RIS association mechanism, where RISs are dynamically assigned to different cells, which was beyond the scope of this thesis.

Bibliography

[1] Abdallah Aldosary et al. “Predictive Wireless Channel Modeling of MmWave Bands Using Machine Learning”. In: *Electronics* 10 (Dec. 2021), p. 3114. DOI: 10.3390/electronics10243114.

[2] Sakshi Agarwal, Kallol Das, and Remco Litjens. “Development and Assessment of Resource Management Solutions for Throughput Enhancement in a RIS-aided Mobile Network”. In: *2024 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*. 2024, pp. 1–6. DOI: 10.1109/EuCNC/6GSummit60053.2024.10597054.

[3] Weidong Mei and Rui Zhang. “Joint Base Station-IRS-User Association in Multi-IRS-Aided Wireless Network”. In: *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*. 2020, pp. 1–6. DOI: 10.1109/GLOBECOM42002.2020.9347975.

[4] Sifan Liu et al. “Joint BS-RIS-User Association and Beamforming Design for RIS-Assisted Cellular Networks”. In: *IEEE Transactions on Vehicular Technology* 72.5 (2023), pp. 6113–6128. DOI: 10.1109/TVT.2022.3231347.

[5] Hamid Amiriara et al. “IRS-User Association in IRS-Aided MISO Wireless Networks: Convex Optimization and Machine Learning Approaches”. In: *IEEE Transactions on Vehicular Technology* 72.11 (2023), pp. 14305–14316. DOI: 10.1109/TVT.2023.3282272.

[6] Yuqian Zhu et al. “DRL-based Joint Beamforming and BS-RIS-UE Association Design for RIS-Assisted mmWave Networks”. In: *2022 IEEE Wireless Communications and Networking Conference (WCNC)*. 2022, pp. 345–350. DOI: 10.1109/WCNC51071.2022.9771607.

[7] Rui Huang and Vincent W. S. Wong. “Joint User Scheduling, Phase Shift Control, and Beamforming Optimization in Intelligent Reflecting Surface-Aided Systems”. In: *IEEE Transactions on Wireless Communications* 21.9 (2022), pp. 7521–7535. DOI: 10.1109/TWC.2022.3159187.

[8] Ahmed Al-Hilo et al. “Reconfigurable Intelligent Surface Enabled Vehicular Communication: Joint User Scheduling and Passive Beamforming”. In: *IEEE Transactions on Vehicular Technology* 71.3 (2022), pp. 2333–2345. DOI: 10.1109/TVT.2022.3141935.

[9] Pyae Sone Aung et al. “Energy-Efficiency Maximization of Multiple RISs-Enabled Communication Networks by Deep Reinforcement Learning”. In: *ICC 2022 - IEEE International Conference on Communications*. 2022, pp. 2181–2186. DOI: 10.1109/ICC45855.2022.9838468.

[10] Yuzhu Zhang et al. “Reinforcement Learning based Optimal Dynamic Resource Allocation for RIS-aided MIMO Wireless Network with Hardware Limitations”. In: Institute of Electrical and Electronics Engineers Inc., 2023, pp. 148–152. ISBN: 9781665457194. DOI: 10.1109/ICNC57223.2023.10074116.

[11] Di Chen et al. “Integrated Beamforming and Resource Allocation in RIS-Assisted mmWave Networks based on Deep Reinforcement Learning”. In: *2023 21st IEEE Interregional NEWCAS Conference (NEWCAS)*. 2023, pp. 1–5. DOI: 10.1109/NEWCAS57931.2023.10198052.

[12] Tao Jiang, Hei Victor Cheng, and Wei Yu. “Learning to Beamform for Intelligent Reflecting Surface with Implicit Channel Estimate”. In: *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*. 2020, pp. 1–6. DOI: 10.1109/GLOBECOM42002.2020.9348156.

[13] Tao Jiang, Hei Victor Cheng, and Wei Yu. “Learning to Reflect and to Beamform for Intelligent Reflecting Surface With Implicit Channel Estimation”. In: *IEEE Journal on Selected Areas in Communications* 39.7 (2021), pp. 1931–1945. DOI: 10.1109/JSAC.2021.3078502.

[14] Byungju Lim and Mai Vu. “Graph Neural Network Based Beamforming and RIS Reflection Design in A Multi-RIS Assisted Wireless Network”. In: *2023 IEEE Statistical Signal Processing Workshop (SSP)*. 2023, pp. 120–124. DOI: 10.1109/SSP53291.2023.10207958.

[15] Mengbing Liu et al. “Cooperative Beamforming and RISs Association for Multi-RISs Aided Multi-Users MmWave MIMO Systems Through Graph Neural Networks”. In: *ICC 2023 - IEEE International Conference on Communications*. 2023, pp. 4286–4291. DOI: 10.1109/ICC45041.2023.10278986.

[16] Shuai Lyu, Limei Peng, and Shih Yu Chang. “Investigating Large-Scale RIS-Assisted Wireless Communications Using GNN”. In: *IEEE Transactions on Consumer Electronics* (2024), pp. 1–1. DOI: 10.1109/TCE.2023.3349153.

[17] Kaixin Li, Limei Peng, and Pin-Han Ho. “GNN-Based Data Rate Maximization in Double Intelligent Reflecting Surface (IRS)-Aided Communication System”. In: *2023 International Conference on Networking and Network Applications (NaNA)*. 2023, pp. 447–452. DOI: 10.1109/NaNA60121.2023.00080.

[18] Baichuan Zhao and Chenyang Yang. “Learning Beamforming for RIS-aided Systems with Permutation Equivariant Graph Neural Networks”. In: *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*. 2023, pp. 1–5. DOI: 10.1109/VTC2023-Spring57618.2023.10200544.

[19] Zhongze Zhang, Tao Jiang, and Wei Yu. “Learning Based User Scheduling in Reconfigurable Intelligent Surface Assisted Multiuser Downlink”. In: *IEEE Journal of Selected Topics in Signal Processing* 16.5 (2022), pp. 1026–1039. DOI: 10.1109/JSTSP.2022.3178213.

[20] 3GPP. “Study on channel model for frequencies from 0.5 to 100 GHz”. In: *3GPP Technical Specification Group Radio Access Network, Technical Report TR38.901* (2019).

[21] ETSI. *LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Frequency (RF) System Scenarios (3GPP TR 36.942 Version 8.2.0 Release 8)*. Tech. rep. European Telecommunications Standards Institute (ETSI), Sophia Antipolis Cedex, France, Jul 2009, Version 8.2.0, Release 8.

[22] Rubén Fraile et al. “Wireless Systems Mobile radio bi-dimensional large-scale fading modelling with site-to-site cross-correlation.” In: *European Transactions on Telecommunications* 19 (Jan. 2008), pp. 101–106. DOI: 10.1002/ett.1179.

[23] Zhihua Lai et al. “The Characterisation of Human Body Influence on Indoor 3.5 GHz Path Loss Measurement”. In: May 2010, pp. 1–6. DOI: 10.1109/WCNCW.2010.5487656.

[24] Stephan Jaeckel et al. “QuaDRiGa - Quasi Deterministic Radio Channel Generator, User Manual and Documentation”. In: *V1.2.3-307* (2014), pp. 1–133.

[25] Taesang Yoo and A. Goldsmith. “On the optimality of multiantenna broadcast scheduling using zero-forcing beamforming”. In: *IEEE Journal on Selected Areas in Communications* 24.3 (2006), pp. 528–541. DOI: 10.1109/JSAC.2005.862421.

[26] Xianghao Yu, Dongfang Xu, and Robert Schober. “MISO Wireless Communication Systems via Intelligent Reflecting Surfaces”. In: (Apr. 2019). URL: http://arxiv.org/abs/1904.12199.

[27] Preben Mogensen et al. “LTE Capacity Compared to the Shannon Bound”. In: *2007 IEEE 65th Vehicular Technology Conference - VTC2007-Spring*. 2007, pp. 1234–1238. DOI: 10.1109/VETECS.2007.260.

[28] 3GPP. “NR; Physical Layer Procedures for Data,” in: *3GPP Technical Specification Group Radio Access Network, Technical Report TR38.214* (2023).

[29] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[30] Oleksandr Ferludin et al. “TF-GNN: Graph Neural Networks in TensorFlow”. In: *CoRR* abs/2207.03522 (2023). URL: http://arxiv.org/abs/2207.03522.

[31] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015. URL: http://arxiv.org/abs/1412.6980.

[32] Pavlo M. Radiuk. “Impact of Training Set Batch Size on the Performance of Convolutional Neural Networks for Diverse Datasets”. In: *Information Technology and Management Science* 20 (1 Jan. 2018). DOI: 10.1515/itms-2017-0003.

[33] Andrew Ng. “Feature selection, L 1 vs. L 2 regularization, and rotational invariance”. In: *Proceedings of the Twenty-First International Conference on Machine Learning* (Sept. 2004). DOI: 10.1145/1015330.1015435.

[34] Nitish Srivastava et al. “Dropout: a simple way to prevent neural networks from overfitting”. In: *J. Mach. Learn. Res.* 15.1 (Jan. 2014), pp. 1929–1958. ISSN: 1532-4435.