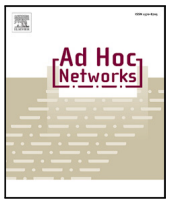
Ad Hoc Networks 169 (2025) 103738

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc



Data-driven cell-free scheduler

Yara Huleihel [a,*], Gil Maman [a], Zion Hadad [b], Eli Shasha [b], Haim H. Permuter [a]

[a] School of Electrical and Computer Engineering, Ben-Gurion University of the Negev, Be'er-Sheva, Israel
[b] RunEl NGMT Ltd, Rishon LeTsiyon, Israel

ARTICLE INFO

Keywords:
Cell-free
Deep-learning
Perfect match
Scheduler
Wireless communication
Power allocation

ABSTRACT

Efficient scheduling is essential in cell-free (CF) networks, where user equipments (UEs) communicate with multiple distributed transceivers (radio units (RUs)) linked to a centralized base station (BS) that coordinates and processes the received or transmitted signals. Unlike traditional cellular networks, CF networks operate without cell boundaries, allowing UEs to seamlessly connect to multiple RUs, and thus eliminating the conventional necessity for handoffs between transceivers. In this paper, we introduce a novel CF scheduler designed to enhance data quality of service (QoS) parameters, including throughput, and latency. The scheduler employs a neural network (NN) algorithm to autonomously manage interactions with users across a distributed network of transceivers. This approach utilizes both model and data driven methods to optimize user communication. To mitigate the high computational complexity of traditional model-driven algorithms, we propose a supervised NN that learns from the model-driven approach. We assess its performance using simulated data from orthogonal frequency division multiple access (OFDMA) waveforms in frequency, time, space, and polarization (e.g., resource blocks, OFDM symbols, beam ID), within multi-transceiver RU environments. Our results indicate that the model-driven algorithms exhibit competitive performance compared to the exhaustive search method, while the supervised NN demonstrates comparable efficiency after offline learning. Consequently, our NN-based scheduler emerges as a viable, efficient solution for optimizing CF network scheduling.

1. Introduction

Cell-free (CF) [1,2] is a key configuration of 5G distributed network infrastructure [3]. In the open radio access network (ORAN) concept, the centralized components of the RAN are moved to an edge cloud infrastructure, in accordance with 3GPP standards [4]. These include the centralized unit (CU) and distributed unit (DU), both of which housed in the cloud. Following the fronthaul split, the distributed radio units (RUs) are positioned at the cell sites. This configuration is expected to become increasingly vital in the forthcoming beyond 5G/6G era, where the integration of AI engines into the radio intelligence controller (RIC) is anticipated. These AI engines will manage the medium access control (MAC) scheduler engine in the DU, thereby facilitating applications that are low in complexity but high in throughput, ultra-reliable, and offer low-latency quality of service (QoS). The use of multiple RUs in the transmission significantly enhances system efficiency [5], thereby improving the information QoS key performance indicators (KPI) [6]. In the CF architecture, the available bandwidth, which scales polynomially with the number of RUs and user equipments (UEs), is limited. This scaling can lead to a mutual interference among the RUs and UEs. As a result of this, in the uplink, power control and RUs scheduling are essential in order to minimize co-channel interference and improve spectral efficiency. However, the scheduling problem presented by CF schedulers is NP-hard [7], requiring a simultaneous determination of frequency and transmitting power from all RUs to users and vice-versa, as a part of the downlink/uplink power control combined with the modulation coding scheme (MCS). Various sub-optimal, yet feasible, scheduling algorithms have been proposed, with varying levels of complexity and accuracy [8]. Panda [9] explored power control and allocation strategies to enhance the performance of CF massive MIMO systems by optimizing energy use and addressing pilot contamination through strategic access point selection and grouping, highlighting the importance of power management in improving system efficiency. With the advancements in machine learning (ML) techniques, particularly deep learning (DL), and the availability of large amounts of training data, there has been a growing interest in using DL to design and optimize wireless networks without sacrificing complexity or accuracy, particularly in the context of scheduler problems.

* Corresponding author.
E-mail addresses: halihal@post.bgu.ac.il (Y. Huleihel), gilma3222@gmail.com (G. Maman), zionh@runel.net (Z. Hadad), eli.shasha@runel.net (E. Shasha), haimp@bgu.ac.il (H.H. Permuter).

https://doi.org/10.1016/j.adhoc.2024.103738
Received 17 April 2024; Received in revised form 5 November 2024; Accepted 9 December 2024
Available online 16 December 2024

1.1. Network scheduler

In a packet-switching communication network, a network scheduler acts as an arbiter on a node, serving as a control unit that is responsible for managing the data transfer between multiple RUs and multiple UEs. The scheduler manages the sequence of network packets in the transmit and receive queues of the network interface controller. The scheduler in a network makes decisions, based on a variety of metrics, to determine: (1) which packet to forward next, (2) the RU from which it will be transmitted, (3) the allocation (resource blocks - RBs), and (4) at what transmitting power. These metrics include the signal-to-interference-plus-noise-ratio (SINR), the received signal strength indicator (RSSI), and the required packet QoS. The SINR is a crucial metric that quantifies the quality of the signal. It is given by the ratio of the desired signal power to the combined power of interference and noise. On the other hand, the RSSI, or in the context of 3GPP orthogonal frequency division multiple access (OFDMA), the reference signal received power (RSRP), measures the power present in a received radio signal. This measurement is used to determine the strength of the signal.

The corresponding RSSI between a RU and a UE should be larger than the interference from other transmitting distributed RUs to ensure successful transmission. Another factor that is considered part of the network scheduling process is latency. Reducing latency is important for real-time applications, such as, virtual reality (VR), online gaming, and voice over IP (VoIP), where even slight delays or jitters in communication can disrupt the synchronicity of the interaction, and lead to a poor user quality of experience (QoE). To minimize latency, packet scheduling algorithms prioritize packets based on their importance and urgency, aiming to minimize the time they spend in queues within the network. In addition, the need for high reliability, power saving, and low radiation requires careful management of the transmission power. To wit, the network scheduler plays a vital role in managing the communication resources in a packet-switching communication network. By utilizing the mentioned metrics, the scheduler can optimize the communication scheduling to improve the overall network performance and ensure efficient use of the network resources. In this work, we demonstrate our proposed scheduler for downlink data transmissions from the RUs to the UEs. However, similar algorithms and scheduler designs can be adapted for uplink data transmissions from the UEs to the RUs by incorporating the relevant constraints.

1.2. Cell-free principle

Fig. 1 illustrates the difference between conventional cellular networks, where each UE is connected to a single RU, and CF networks, where UEs can be connected to multiple RUs. CF communication, where many distributed RUs jointly process signals, has been identified as a potential infrastructure for beyond-5G networks [10]. The comparison between conventional and CF communication is non-trivial and highlights the trade-offs between these two approaches. Conventional cellular networks benefit from channel hardening and spatial interference suppression, but may have poor channel conditions for cell-edge UEs. In contrast, CF networks have strong macro diversity, but the capability for interference suppression is highly dependent on the specific operating conditions. The CF innovation allows for simultaneous scheduling between various RUs and UEs to be set up in advance. It determines who transmits and when, and also anticipates different metrics, such as the channel estimator using channel state information (CSI). This data is then used to collaboratively design the precoding matrices and relative power, leading to the required SNR per user and the MCSs for all the users on the same allocated RBs, OFDM symbols, beams, etc. These are all prepared before transmission. With diversity combining, the received signal from nearby RUs undergoes additional channel hardening. This approach contrasts with the current generation, where each cell operates independently, with only one base station (BS) available per cell, and scheduling is performed individually per cell, where it is assumed that some transmissions will collide and automatic repeat request/hybrid automatic repeat request (ARQ/HARQ) will recover the packet, albeit with a delay penalty.

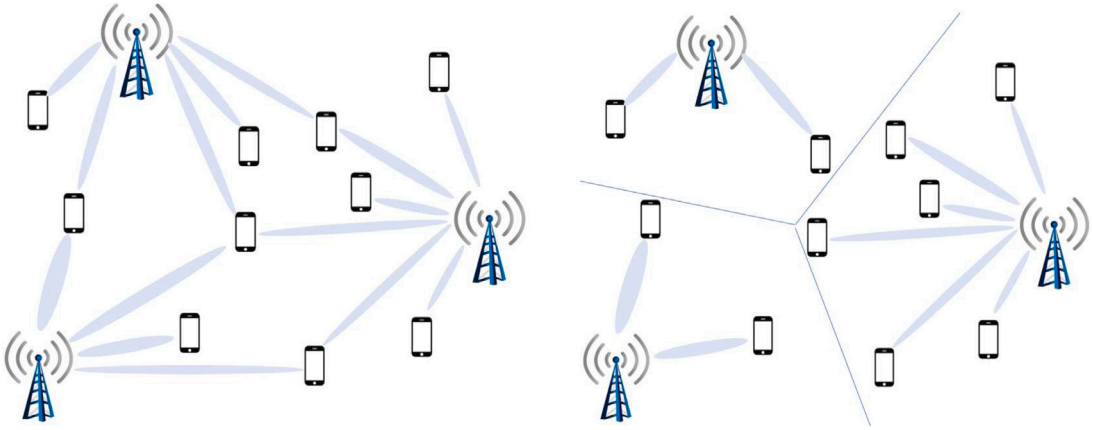
In traditional cellular networks, handovers are essential for maintaining connectivity as UEs move between cell boundaries, transferring sessions from one BS to another. This process, although crucial, often results in delays, increased control plane load, and potential packet loss, especially during the transition between cells. Such interruptions can be challenging in high-mobility or edge-of-cell scenarios where signal quality may degrade, affecting the overall QoS. Various algorithms have been proposed to improve handover success rates by considering factors such as speed, position, and direction [11]. However, most handovers remain "hard" handovers, dropping the existing connection before a new link is established, often causing delays and straining the control plane.

In contrast, CF networks eliminate the need for handovers by connecting each UE to multiple distributed RUs simultaneously, enabling seamless mobility across the network. Operating above the MAC layer, the CF scheduler dynamically coordinates packet transmissions across RUs within a single, unified geographic area. This approach avoids the disruptions associated with traditional handovers by maintaining continuous communication with nearby RUs. By integrating all RUs under a common MAC, the CF network ensures a consistent QoS, enhanced coverage, and reduced latency and packet loss. Centralized scheduling thus avoids the bottlenecks and complexities of handover management, making CF networks particularly advantageous in scenarios requiring uninterrupted connectivity.

1.3. Machine learning approaches

Various classical model-driven strategies have been developed to address scheduling tasks in wireless communication networks. These strategies typically involve optimization techniques, exhaustive search methods, or heuristic algorithms that, while effective, can be computationally intensive and may not scale efficiently with increasing network complexity. To overcome these limitations, DL techniques have been increasingly explored for their potential to enhance performance while reducing computational demands. The motivation behind DL-based schedulers lies in their ability to learn optimized mappings from resources and packets to RUs for transmission, directly from training data. This capability enables these models to outperform classical scheduling algorithms in terms of computational efficiency. Our study presents a supervised neural network (NN) approach designed to emulate the behavior of a high-performing model-driven algorithm, with the goal of achieving similar results in reduced time and with lower computational complexity. The NN is trained using the outputs of deterministic algorithms as labels, ensuring high accuracy in predicting packet scheduling and resource allocation.

Several studies have explored ML methodologies for network scheduling. For example, Mennes et al. [12] tackle the problem of shared spectrum management in wireless communication. This becomes particularly crucial when multiple nodes try to access the same spectrum in traditional cellular networks. Accordingly, they introduce two NN-based algorithms designed to accurately predict free slots in a multiple frequencies time division multiple access network. By forecasting the spectrum behavior, these algorithms are capable of reducing collision instances. Using prior observations, they employ a supervised NN to estimate the probability of slot utilization in subsequent frames. Another ML approach utilized in network scheduling is reinforcement learning (RL). Unlike supervised learning, where both input and output are available for decision-making, RL involves sequential decision-making, with the next input depending on previous decisions. An agent interacts with the environment, transitioning from one state to another, and receives rewards for successful actions. Over time, the agent learns from its experiences and improves its decision-making. Chinchal et al. [13] introduced an RL-based scheduler capable of



(a) Cell-free Network (b) Conventional cellular network

Fig. 1. Illustration of cell-free and conventional cellular networks.

dynamically adapting to traffic variations and reward functions set by network operators, optimizing the scheduling of IoT traffic in cellular networks. Further advancements in ML-based network scheduling have been made by Guenach et al. [14,15], who proposed a low-complexity solution for iterative joint optimization of power allocation and access point scheduling in CF massive MIMO networks. Their approach utilizes a U-Net NN model to address the association problem as an image segmentation task, combined with linear programming (LP) for power allocation. This method is designed to maximize the worst-case SNR for uplink scheduling.

A recent contribution [16] exploits NNs for resource allocation in CF massive MIMO networks to reduce grid energy consumption while maintaining user spectral efficiency. This work highlights the potential of AI-driven solutions in achieving efficient energy resource management in wireless networks. Additionally, Fan et al. [17] introduced a novel approach for full-duplex cell-free mmWave network that incorporates hybrid MIMO processing and multi-agent deep RL for power allocation. This method effectively reduces cross-link interference and enhances spectral efficiency, demonstrating the potential of deep RL in the management of complex network environments. Furthermore, Arshad et al. [18] demonstrated the effectiveness of unsupervised ML in user clustering within CF massive MIMO systems using non-orthogonal multiple access (NOMA). Their approach significantly improves spectral efficiency, sum-rate, and bit error rates by employing user-centric clustering for resource allocation.

Additional works have addressed related challenges with different approaches. For instance, Zaher et al. [19] explore a DL approach to approximate power allocation in downlink scenarios, focusing on spectral efficiency maximization while balancing computational complexity. The use of deep NNs in this context enables a distributed solution, reducing the need for centralized control and allowing real-time adaptability in power allocation. Lacava et al. [20] present an RL-based approach for traffic steering in 5G networks, optimizing UE throughput through handover management. The problem is formulated as a Markov decision process and solved using a conservative Q-learning approach. Although this method achieves significant performance improvements, RL-based solutions often require substantial data collection and computational resources and can struggle with stability and convergence in highly dynamic environments [21–23]. In contrast, our proposed NN-based scheduler offers a more deterministic approach, providing stable and predictable performance with lower computational demands, which is particularly advantageous in dynamic and resource-constrained environments like CF networks.

The contributions of these studies collectively mark a significant step toward integrating ML in network scheduling, offering valuable insights into the potential for AI-driven advancements in wireless communication networks. Using the strengths of various ML techniques, our work bridges the gap between classical deterministic methods and modern machine learning approaches. This hybrid approach offers a novel, scalable, and practical solution for addressing the unique challenges posed by network scheduling in CF networks.

1.4. Main contributions

In this paper, we tackle the scheduling challenge within CF networks through a dual-approach framework. Initially, we adapt conventional model-driven algorithms to the CF environment, evaluating their efficacy in offline settings for this specific scenario. More significantly, our primary contribution lies in developing a supervised NN model. This model not only learns from the outcomes of these established algorithms, but also substantially reduces the computational complexity. This NN-based approach offers a novel path for efficient scheduling by leveraging the strengths of traditional algorithms while overcoming their limitations in terms of computational complexity. Our solution employs a frequency reuse factor, which refers to the rate of reuse of the same frequency-time resource within the network [24]. We evaluate the performance of the proposed algorithm using multi-allocations multi-RUs simulated data, and compare it to different deterministic scheduling algorithms. Our results demonstrate that our algorithm offers competitive performance in terms of packet loss, throughput, and computational and time complexity compared to the other algorithms. We suggest using an iterative “row-column softmax" output layer to enhance the performance in generating a permutation matrix that adheres to the scheduler assignment matrix constraints. The “row-column softmax" refers to the process of sequentially applying the softmax function across the rows and then the columns of the matrix. To reduce latency, the proposed scheduling algorithms prioritize packets by their time-to-live (TTL), thereby decreasing packet expiration. The TTL in our context is a system parameter derived from the QoS requirements, specifically related to the maximum allowable delay for packet transmission. The initial value of the TTL is set based on the desired latency performance of the network and typically varies depending on the application requirements. For instance, a low-latency application like real-time video streaming, where packet delay requirements are stringent, typically requires a short TTL (e.g., in the range of 5–10 ms) to ensure timely delivery and reduce the risk of packet expiration [23,25]. Conversely, for less latency-sensitive applications such as file downloads, a higher TTL (e.g., 50 ms or more) is acceptable, allowing for more transmission flexibility. In our simulations, the TTL is represented as a normalized discrete time unit, ensuring timely delivery within the cell-free network. The network scheduler decrements the TTL at each timeslot, which serves as the scheduling cycle in our simulation, to reflect the remaining time available for successful packet delivery.

Once the TTL reaches zero, the packet is dropped to avoid excessive delays, ensuring compliance with QoS constraints. Additionally, when the distance between source and destination is considerable, scheduling based on RSSI becomes relevant to latency, as RSSI is influenced by this distance.

The rest of this paper is organized as follows. In Section 2, we define and formulate the problem for each distinct block of the scheduling task. Section 3 delves into the model and data driven strategies we propose to tackle the scheduling challenge, elucidates the DL-based system architecture for scheduling optimization and power management tasks, and outlines the training procedure. The learning-based approach's interference aspect and the post-processing protocol are comprehensively explained in Section 4. Sections 5 and 6 present an analysis of the results of the numerical simulation and provide pertinent information. Lastly, our final thoughts and conclusions appear in Section 7.

2. Notation and problem definition

In this section, we introduce the notation and the problem definition.

2.1. Notation

Throughout this paper, we use the following notation. The set of real numbers is denoted by $\mathbb{R}$. Random variables will be denoted by capital letters, and their realizations will be denoted by lower-case letters, e.g., X and x, respectively. We use the notation X^n to denote the random vector $(X_1, X_2, \ldots, X_n)$ and x^n to denote its realization. The indicator function, denoted by $\mathbb{1}_A$, equals to unity if event A occurs and zero, otherwise.

2.2. Problem definition

In this section, we mathematically describe the objective and constraints of the scheduling and power management problems. We provide the condition for successful transmission, and describe the scheduling and power matrices. First, we briefly introduce the complete setup used in our system. The 5G New Radio (NR) utilizes waveforms with OFDM and OFDMA arrangements for data transmission from the BS to the handset. In the frequency domain, each OFDMA symbol is divided into a pre-defined number of subcarriers, which are then grouped in sets of size 12 to form a single RB, as specified by the third generation partnership project (3GPP) specifications [4]. In the time domain, a radio frame with a duration of 10 ms is structured. Each frame is divided into subframes, each comprising one or multiple adjacent slots, depending on the subcarrier spacing. Each slot contains 14 OFDMA symbols. Within this framework, the scheduler is responsible for allocating blocks of several RBs in frequency domain and multiple OFDMA symbols in time domain for each user. This allocation process considers several factors, including transmitted power, user-specific modulation, coding, and MIMO waveforming techniques.

In this study, we consider the transmission characteristics of a CF network, comprising of K RUs and N UEs. The communication between the RUs and the UEs is facilitated by F frequency-time allocations, hereafter referred to as "frequency-time resource" or just "frequency allocations". These resources are the multidimensional units through which the scheduler operates, encompassing both frequency bands and time intervals to optimize network performance. The RSSI is employed to measure the power of the received signal at each UE. Consequently, each UE is associated with K RSSI measurements, one from each RU/beam. Furthermore, for each considered RU serving a dedicated UE, we evaluate the interference. This interference originates from the other surrounding RUs that are transmitting and affects the packet transmission from the ith RU to the corresponding UE. Each UE has a unique identifier, denoted by, $\mathcal{U}_n$ (e.g., different DMRS ports), for $1 \le n \le N$. To determine the combined effect or resultant interference from multiple RUs, we employ vector addition, taking into account only non-coherent sources. By summing the squares of the interference powers from each RU and incorporating the thermal noise, represented by N_0^2, we account for the combined effect of the interference sources. The square root of this sum yields the magnitude of the resultant interference, representing the non-coherent power summation. Accordingly, we mathematically quantify the interference in a linear scale ('lin') as follows,

$$\mathsf{IN}_{i,n,lin} \triangleq \sqrt{\left(\sum_{\substack{1\le k\le K\\ i\neq k}} \left[\mathsf{RSSI}_{k,lin}(\mathcal{U}_n)\right]^2\right) + N_0^2}, \tag{1}$$

where $\mathsf{IN}_{i,n,lin}$ is the interference encountered by the nth UE from all RU's excluding the designated ith RU, and RSSI_k is defined as

$$\mathsf{RSSI}_k(\mathcal{U}_n) \triangleq \mathsf{P}_k - L_{k,\mathcal{U}_n}(D_{k,\mathcal{U}_n}) + G_{k,\mathcal{U}_n} - F_{k,\mathcal{U}_n}, \tag{2}$$

where, P_k is the transmit power of the kth RU, $L_{k,\mathcal{U}_n}(D_{k,\mathcal{U}_n})$ represents the path loss between the kth RU and $\mathcal{U}_n$, which is a function of distance $D_{k,\mathcal{U}_n}$ between them and possibly other environmental factors. $G_{k,\mathcal{U}_n}$ denotes the gains, such as from beamforming or antenna design, and $F_{k,\mathcal{U}_n}$ accounts for fading effects. To simplify calculations, we can disregard interference from sources that are at least an order of magnitude weaker than the dominant ones, i.e., when the reference signal received quality (RSRQ) is greater than a predefined threshold. If the RSSI value of one interfering RU is significantly higher than those of the other RUs, the interference from the other RUs can be neglected, and only the RU with the largest RSSI value will be considered for interference analysis.

Fig. 2 illustrates the proposed general process, as outlined below. The scheduler's input is determined by a fixed-size buffer that is ordered based on the packets' TTL priority. The buffer size should be determined as the maximum number of packets that can be efficiently handled without causing significant delays or overflows. For example, in applications like real-time voice transmission, where minimizing delay is critical, the buffer is managed to prioritize quick processing of packets with low TTL, even if some packets are occasionally dropped to maintain low latency. In contrast, applications such as file transfer (FTP) prioritize data integrity over speed, often using larger buffers to ensure that packets are not lost due to buffer overflow. Each cell in the buffer contains a tuple ($\mathcal{U}_n$, TTL) of the packet's $\mathcal{U}_n$ and its priority. Based on each cell's $\mathcal{U}_n$, QoS flow, and timeslot, the input for the scheduler is generated in accordance to the procedures outlined in Section 5.1. This input includes the K relevant UE's RSSI values, one from each RU/beam. At each timeslot, the $K \times F$ packets in a queue with the lowest TTL are selected to be scheduled and accordingly transmitted.

The middle component of Fig. 2 showcases the scheduling mechanism, which determines the allocation of packets to be transmitted from specific RBs and RUs, based on specified constraints. For each allocation, the SINR for each RU and UE (determined according to the chosen packets) is calculated, and the scheduling algorithm is applied accordingly. Under the suggested setup, the maximum number of packets that can be transmitted per RB allocation is directly proportional to the number of RUs available, assuming there is no overlap in the coverage areas of the RUs.

To guarantee the successful transmission of a packet for a given allocation, the RSSI from the assigned RU to the UE must exceed the cumulative interfering signal strength from other RUs by a predefined threshold, known as the fade margin. These relationships are expressed on a logarithmic scale (dB) for clarity. This constraint can be formulated as follows:

$$\mathsf{SINR}_{i,n} + \mathsf{P}_i > \kappa, \tag{3}$$

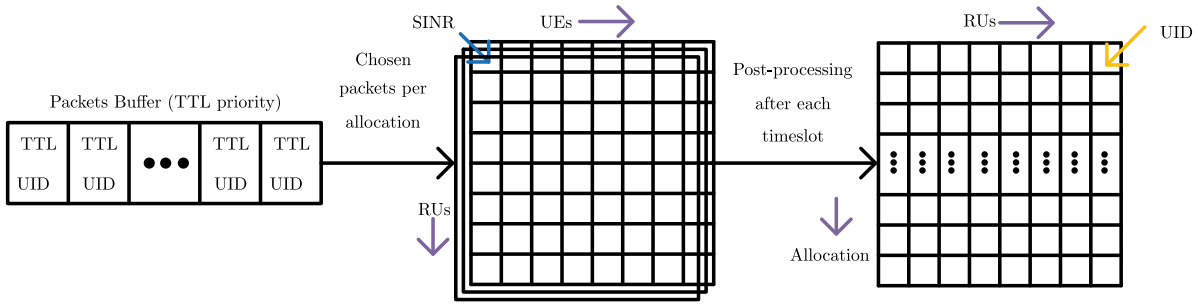


Fig. 2. Full suggested scheduling process diagram.

where κ is a certain threshold, representing the minimum SINR required to achieve acceptable error performance while aiming to maximize the code rate, determined through empirical measurements and system simulations. $\mathsf{SINR}_{i,n}$ is defined as,

$$\mathsf{SINR}_{i,n} \triangleq \mathsf{RSSI}_i(\mathcal{U}_n) - \mathsf{IN}_{i,n}, \tag{4}$$

and P_i denotes the minimum transmission power necessary, in addition to the RSSI from the ith RU to the $\mathcal{U}_n$ UE, to meet the successful transmission condition in (3). By appropriately adjusting the transmission power, we aim to increase the throughput while minimizing unnecessary power consumption. The added/subtracted transmission powers correspond to the adjustments made to ensure a successful transmission according to the SINR threshold condition. Added transmission power ensures that the SINR requirement is met, while subtracted transmission power improves efficiency by reducing interference and conserving energy when an RU does not transmit. However, it is crucial to consider the impact of the added power on the transmission interference experienced by other packets sharing the same RBs. In the event that the conditions for successful transmission cannot be met, a packet switch will be carried out, as described in Section 4, and is depicted on the right component of Fig. 2. The output as shown in the right component of Fig. 2 is the assigned UE for each of the RUs per frequency-time allocation.

The algorithm's output is represented by two matrices. The first matrix captures the final suggested scheduling of transmission packets among all available RUs and UEs for every available frequency resource, as depicted in the UE-RU matrix. If an RU is not engaged in transmission at a particular frequency resource, its corresponding matrix cell will be assigned a value of "-1". For each frequency resource and participating RU, we assign an index indicating the scheduled UE. We denote this UE index by $\mathcal{I}_{f,k}$, for the kth RU at frequency f. For simplicity of notations, we gather all of these $F \times K$ possible indices using the following matrix,

$$\mathcal{I} \triangleq \begin{bmatrix} \mathcal{U}_{\mathcal{I}_{1,1}} & \dots & \mathcal{U}_{\mathcal{I}_{1,K}} \\ \dots & & \dots \\ \mathcal{U}_{\mathcal{I}_{F,1}} & \dots & \mathcal{U}_{\mathcal{I}_{F,K}} \end{bmatrix}. \tag{5}$$

A power matrix is also defined, with entries representing the added/subtracted transmission powers, $\mathsf{P}_{f,k}$, for all RUs, k, and frequency resources, f, i.e.,

$$\mathcal{P} \triangleq \begin{bmatrix} \mathsf{P}_{1,1} & \dots & \mathsf{P}_{1,K} \\ \dots & & \dots \\ \mathsf{P}_{F,1} & \dots & \mathsf{P}_{F,K} \end{bmatrix}. \tag{6}$$

The algorithm is designed to balance the following objectives:

1. *Minimizing packet loss*: Reduce the number of packets not transmitted due to the expiration of their TTL.
2. *Minimizing the total transmission power*: Optimize the algorithm to minimize the amount of additional transmission power required to ensure successful transmission.

It is important to note that these objectives are in conflict with each other, and the algorithm must balance them in order to achieve the best possible performance. By finding the optimal trade-off between these goals, the algorithm can provide reliable and efficient packet transmission while minimizing energy consumption.

3. Proposed scheduling approaches

In this section, we suggest the use of different methods for addressing the discussed scheduling task. These methods can be broadly classified into two categories: *model-driven* and *data-driven* techniques. The first category encompasses the standard approaches we have implemented to solve our scheduling problem, including exhaustive search, round robin, perfect match, Hungarian method, and stable match. The second category encompasses ML-based techniques. The following subsections provide an overview of the various methods we have considered for the described scheduling problem.

3.1. Proposed model-based methods

In the following, we will provide a detailed overview of the model-driven methods that we have explored for the scheduling task. These approaches are evaluated based on their performance and computational complexity. While in many cases high performance can be achieved, it often comes with the price of high computational complexity and longer run times. This challenge becomes more pronounced in complex configurations, such as those with a large number of RUs. The exhaustive search algorithm is presented as a reference for the best possible performance. However, it is only suitable for small-scale scenarios due to its high computational complexity. The round robin algorithm [26] is a well-established network scheduling approach. The perfect match algorithm [27] and stable match [28] are both matching algorithms that are capable of handling large-scale configurations. The Hungarian method [29] is an optimization algorithm used to solve assignment problems, with the goal of finding the optimal assignment of resources to tasks with the lowest overall cost. Below we discuss the above methods in more detail.

3.1.1. Exhaustive-search

The exhaustive search algorithm is a method for resolving problems, including scheduling tasks, by systematically examining all possible solutions to determine the optimal one. While this method guarantees the best outcome, its computational complexity is extremely high, and the run-time blows up as the dimensions of the problem increase. This makes exhaustive search infeasible for large-scale scheduling problems, as it requires significant computational resources and can take an unreasonable amount of time to complete [30]. However, for small-scale scheduling problems, the exhaustive search can provide a valuable benchmark for evaluating other scheduling algorithms and determining the best possible scheduling solution.

In this paper, we adopt a conventional assumption, motivated by complexity constraints, that treats the scheduling and packet transmission as independent processes across different frequency-time resources. The transmission outcome of each frequency-time resource relies on the availability of resources, allowing for the transmission of all or a subset of the designated packets based on the following condition,

$$\mathsf{A}_f \triangleq \sum_{i=1}^{K} \left[\mathsf{RSSI}_i(\mathcal{U}_{\mathcal{I}_{f,i}}) - \mathsf{IN}_{i,n} \right] > K \cdot \kappa. \tag{7}$$

To wit, if the value of A_f is greater than $K \cdot \kappa$, all assigned packets can be transmitted simultaneously at the same frequency-time resource. However, if the condition is not met, the unsent packet will be returned to the queue and replaced by a new packet from the buffer, provided the condition in (7) is satisfied. Otherwise, if this condition is not met, then the slot remains unoccupied. To increase the possibility for a higher number of packets to be transmitted under the same resources, a transmission power can be added as follows,

$$\mathsf{P}_{f,i} = \max\left\{0, \kappa + \mathsf{IN}_{i,n} - \mathsf{RSSI}_i(\mathcal{U}_{\mathcal{I}_{f,i}})\right\}. \tag{8}$$

To find the best schedule, we use a reward metric that considers both the total packets successfully transmitted (TP) per timeslot, and the power used. The optimal schedule maximizes TP and, in the event of a tie, prefers the schedule using the least power. The reward formula is,

$$\mathsf{Reward} \triangleq \mathsf{TP} - \frac{\sum_{f,i} \mathsf{P}_{f,i}}{\mathsf{TP} \cdot P_{\mathsf{norm}}}, \tag{9}$$

with P_{norm} defined as follows,

$$P_{\mathsf{norm}} \triangleq F \cdot K \cdot \max_{f,i} \mathsf{P}_{f,i}, \tag{10}$$

where we note that $\max_i \mathsf{P}_{f,i}$ is some predetermined reasonable value of the maximum transmit power of any RU at any frequency-time allocation.

3.1.2. Perfect-match

The perfect match algorithm, often implemented via linear programming (LP), is a method to establish one-to-one relationships between elements of two equally sized groups. In the context of our scheduling problem, there are K RUs, each transmitting a packet to one of the K UEs at each frequency slot. The objective is to transmit exactly one packet per UE in a way that maximizes the total SINR, denoted here as the reward of the transmission. Every connection between two elements has a score, which indicates the quality of the connection with respect to the purpose of the matching. The term "cost/quality of a match" refers to this score, which can be interpreted as either cost or quality depending on the context. Essentially, it represents the metric used to evaluate the effectiveness of each match under the given constraints. In this context, we aim to choose the match with the best score, which can either minimize the cost or maximize the quality of the connection. The perfect match is defined as the match with the maximum weight sum. Mathematically, the problem above can be formulated as a LP problem [31], as follows,

$$\begin{aligned} \max_{x} \quad & \sum_{i,j} w_{ij} \cdot x_{ij} \\ \text{s.t.} \quad & \sum_{i} x_{ij} = 1 \ \forall j, \\ & \sum_{j} x_{ij} = 1 \ \forall i, \ x_{ij} \in \{0,1\}, \ \forall i,j, \end{aligned} \tag{11}$$

where i and j denote elements from the first and second sets, respectively, and w_{ij} is the weight that represents the cost/quality of a match. Consequently, the constraints on x_{ij} ensure that the connection matrix cells are unique, namely, no two cells belong to the same row or column. A value of $x_{ij} = 1$ signifies a possible connection between elements i and j. It is assumed that the LP problem above is feasible, i.e., that at least one perfect match exists. The LP formulation in this work is a bipartite relaxed form, as the constraint $x \in [0,1]$ is imposed, which aligns with the nature of the scheduling problem. The relaxed form is utilized to allocate the UE-index into the appropriate slot in the scheduling matrix, with the scheduling of $F \cdot K$ packets with the lowest TTL will being prioritized.

The perfect match algorithm can be applied to our scheduling problem by considering two sets of elements. One set, comprises the indices of K UEs, and the other set comprises the indices of the available RUs. Every pairing between a RU and a UE is assigned a weight, calculated as,

$$w_{ij} = \mathsf{RSSI}_i(\mathcal{U}_j) - \mathsf{IN}_{i,j}. \tag{12}$$

where we recall that $\mathsf{RSSI}_i(\mathcal{U}_j)$ is the RSSI from RU i to UE j, and $\mathsf{IN}_{i,j}$ is the interference from all other available RUs.

3.1.3. Stable match

The stable match algorithm [32], tailored for our scenario, forms stable pairings between equal-sized groups, relying on preferences based on RSSI and interference. Initially, UEs propose to RUs according to these preferences. RUs then evaluate the proposals, accepting preferred ones and rejecting the rest. The procedure iterates until stable, mutually preferred pairings are established, ensuring each UE is appropriately matched to an RU.

3.1.4. Hungarian method

The Hungarian method, also known as the Kuhn-Munkres algorithm, is a widely used optimization technique for solving one-to-one assignment problems. Similar to the perfect match algorithm, the Hungarian method for our problem aims to assign each RU exactly one UE at each frequency slot such that the total SINR is maximized. The key difference is in how the two algorithms approach this problem. The Hungarian method creates a cost matrix, where each element denotes the "cost" or quality measure associated with assigning a packet from a RU to a UE. It then modifies this matrix through a series of row and column reductions, creating a set of zero entries. The algorithm iteratively applies augmenting paths and updates the matching until an optimal solution is reached. This way, instead of aiming for a single perfect match, the Hungarian method considers all possible matches and finds the assignment that maximizes the total weight of the matches. While both Hungarian and perfect match algorithms aim for the same objectives, they differ in their computational requirements and implementation details. The Hungarian method is particularly useful for larger problem sizes due to its computational efficiency.

3.2. Data driven - proposed NN model

Our target is to design a NN-based scheduler that optimizes scheduling allocation for maximum QoS user experience performance, while taking into account various factors such as TTL, capacity, coverage area, channel behavior, expected interference to others when one is transmitted, and the required SNR to deliver a packet. We aim to optimize and allocate the appropriate AMC and transmit power while accounting for mutual expected interferences. In this section, we propose a data-driven approach using a NN to efficiently perform the packet scheduling task with significantly reduced computational complexity and time compared to traditional model-driven algorithms. The proposed NN scheduler model is trained in a supervised manner using the perfect match algorithm scheduling results as labels to achieve similar scheduling performance. The architecture of the proposed NN scheduler model is presented in Figs. 3 and 4, and in the following we describe the training procedure in detail. In our implementation, the NN processes one frequency-time resource at a time through serial computation, allowing us to scale up the problem to hundreds of frequency bands in a reasonable timeframe. Additionally, we introduce the 'reuse factor' in our proposed scheduler implementation, which is defined as the rate at which the same frequency-time resource can be

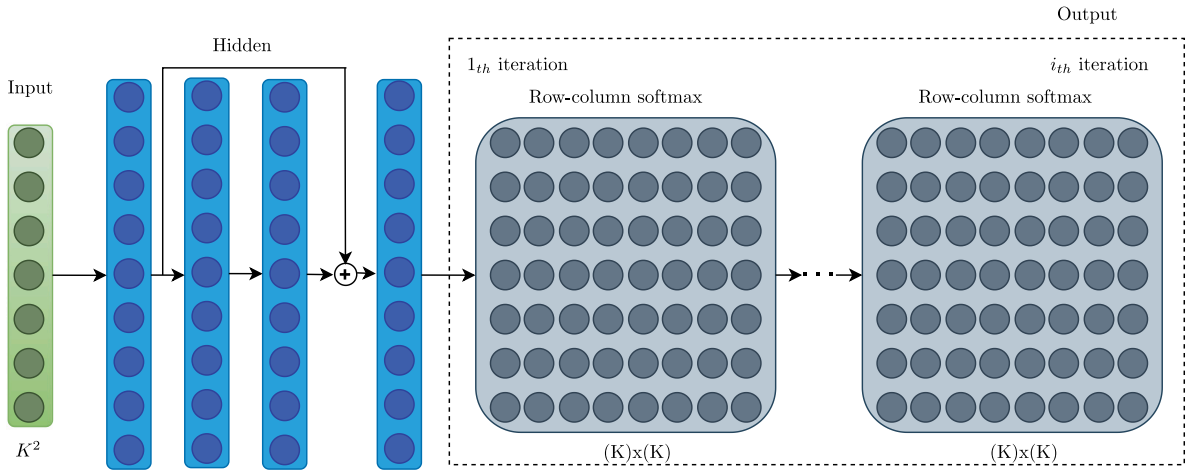


Fig. 3. Structure of the proposed FC NN-based scheduler.

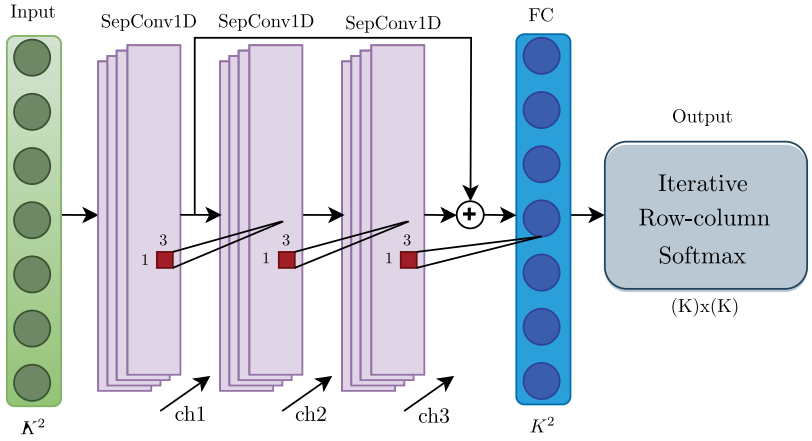


Fig. 4. Structure of the proposed convolutional NN-based scheduler.

reused in the network. By multiplying the number of handled packets under the same frequency-time resource by the reuse factor value, the NN can increase the total capacity of the system without increasing its allocated bandwidth.

3.2.1. Proposed architecture and training

The following details the NN architecture designed to address the packet scheduling task, with a default reuse factor 8. The scheduling is performed serially, one frequency-time resource at a time, to maintain low network complexity as the number of resources increases. This allows for scheduling a large number of frequency bands with low NN complexity and increased performance, with each frequency band scheduling learning based on earlier frequency band learning. The input to the NN consists of a concatenated vector of the SINRs for each of the K RUs and K selected packets denoted by each assigned UE. Each scheduling candidate packet contains K neurons, one from each RU, resulting in a total of K^2 neurons in the input layer. Hidden layers of varying sizes are used between the input and output layers, with each layer being fully connected (FC) and followed by a non-linear activation function. To enhance performance, batch normalization (BN) was added before the activation function. Additionally, a residual connection was added to the network architecture, which sums the input to the second layer and the output of the third layer element-wise. This modification improved the overall performance of the proposed scheme by providing another path for data to reach the latter parts of the NN, making it easier to optimize the mapping [33]. The network architecture can be described by the following:

$$f(\boldsymbol{x}) = \rho_{L_f}\left(\left|\boldsymbol{W}_{L_f}\left(\rho_{L_f-1}\left(\cdots\left(\rho_1\left(\left|\boldsymbol{W}_1\boldsymbol{x}+\boldsymbol{b}_1\right|_{bnorm}\right)\right)\right)\cdots\right)+\boldsymbol{b}_{L_f}\right|_{bnorm} + \left(\left|\boldsymbol{W}_1\boldsymbol{x}+\boldsymbol{b}_1\right|_{bnorm}\right)\right), \tag{13}$$

where L_f is the number of the network FC layers, $\boldsymbol{W}_i$, and $\boldsymbol{b}_i$ are the NN's weight matrix and bias vector, respectively, for the i'th layer, with size determined as a part of the network design. $\rho_i(\cdot)$ is the activation function of the i'th layer, and *bnorm* means the layer passes through a batch normalization.

The output layer is a $K \times K$ matrix, with rows representing the RUs and columns the UEs. Each entry in the output matrix signifies the probability of scheduling the packet to the corresponding RU. A softmax layer was used as the output layer to generate these probabilistic outputs. However, in the examined case, where classification is required across both dimensions, employing a single softmax function on one dimension can lead to slow convergence and low accuracy, as it only normalizes the values along one axis, generating a probability distribution for that specific axis, which can result in suboptimal outcomes. To overcome this limitation and enhance the model's performance, we implemented an iterative 'row-column softmax' layer. Each iteration consists of two consecutive softmax operations. The first softmax layer is applied to the rows, ensuring that multiple UEs/packets are not assigned to the same RU. The output from this layer is then fed into the second softmax layer, which is applied to the columns of the first softmax output. This prevents a single packet from being allocated

to multiple RUs. This iterative process is performed for a predetermined number of iterations, aiming to converge to a permutation matrix. This approach is similar to the softassign method [34], which enforces a two-way assignment constraint. At the end of this process, we set the maximum element in each column to 1 and all the other elements to 0. By applying two softmax layers in this manner, we generate the proposed scheduling matrix probabilities, aiming that each packet (represented by a UE) is uniquely assigned to an RU under the allocated RBs, as required. Each output of the proposed 'row-column softmax' function $\hat{y}:\mathbb{R}^K \to [0,1]^K$ is defined by,

$$\hat{y}_i = \sigma\left(\sigma(Z)_j\right)_i ; \text{for } i,j = 1,\dots,K \tag{14}$$

where $z = (z_1,\dots,z_K) \in \mathbb{R}^K$ represents the input to the first softmax layer, and $\sigma:\mathbb{R}^K \to [0,1]^K$ indicates the standard softmax function.

The negative log loss between the predicted output and the labels is the training loss function used to optimize the scheduler-based NN performance, with L_2 regularization to reduce over-fitting. The AdamW optimizer was used to run back-propagation and optimize the model during training. This optimizer is designed to improve gradients when L_2 regularization is used.

Denoting by y the targeted label, $\hat{y}$ as the predicted class, meaning RU scheduled for the packet, Θ as the model's weights, and λ_1 as a hyperparameter for tuning the L_2 regularization, the loss function is given by,

$$\mathcal{L}(\mathbf{y},\hat{\mathbf{y}}) = -\left[\sum_{p=1}^{K}\sum_{k=1}^{K} \mathbb{1}_{\{y_p=l_k\}} \log P_\theta\left(\hat{y}_p = l_k\right)\right] + \lambda_1 \|\Theta\|_2^2, \tag{15}$$

where the indicator function is used to indicate whether class label k is the correct classification. In this context, p represents the relevant UE.

Building upon this foundational structure, we introduce an alternative approach employing convolutional layers. This suggestion arises from the need to effectively handle the scalability of our scheduling problem, particularly as the size of the input expands. The convolutional neural network (CNN) replaces the FC layers with a series of 1D convolutional layers, concluding with a single FC layer before the iterative 'row-column softmax' approach at the output, facilitating the model's adherence to the constraints of the scheduling problem in CF networks. This adaptation leverages the efficiency of convolutional operations to process the input data to capitalize on the scalability advantages of convolutional layers for large-scale input data. In this convolutional architecture, we utilize separable convolutional layers to further enhance computational efficiency. These layers decompose the convolution operation into depthwise and pointwise convolutions, significantly reducing the number of parameters and computational load. Zero-padding ensures that the output dimensionality matches the input, preserving the integrity of the scheduling decision space, while dilation increases the receptive field of each convolutional layer without increasing the computational complexity. This convolutional approach suggests a scalable and computationally efficient alternative to the fully connected design, especially suited for large-scale scheduling problems where input size significantly influences model performance. By adopting this architecture, we aim to provide a robust solution for CF network scheduling, balancing computational efficiency with the capacity to handle extensive scheduling scenarios. Fig. 4 illustrates the modified convolutional architecture.

3.2.2. Reuse factor

The described resource allocation per user can be reused and reassigned to other users, who may employ different MCS based on the expected SNR and channel conditions. The same allocation can be replicated across all available RUs/beams under the same scheduling process, a practice known as the reuse factor. In modern wireless networks, especially in 5G, efficient spectrum utilization is paramount. One of the strategies to achieve this is through frequency reuse. To illustrate, consider a scenario in 5G where we split the bandwidth/time/space of a slot into 50 different allocations. If each of these allocations undergoes a frequency reuse of 8 times in a network comprising 4 RUs, with each RU associated with 2 beams, then a total of 400 users can be served within a single time slot. This approach highlights the potential of frequency reuse to boost network capacity without extra spectrum resources.

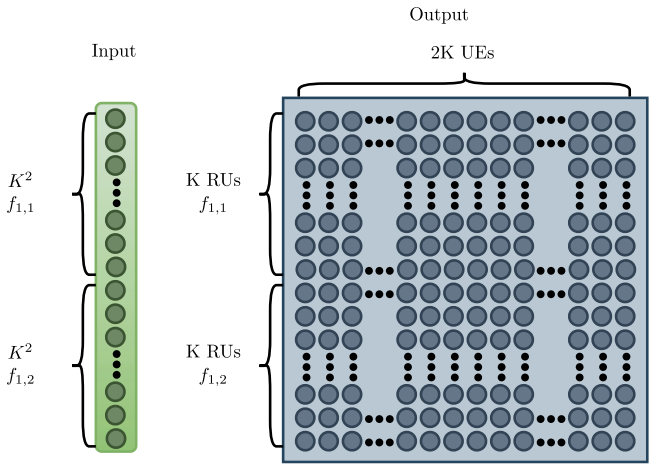


Fig. 5. Input and output layers under sub-allocations implementation.

In our implementation, the reuse factor specifies the maximum number of RUs/beams that can transmit packets at the highest MCS under the same frequency-time resource allocation. Essentially, a higher reuse factor enables multiple reuses of the same frequency resources to be used concurrently. This allows the scheduler to assign packet transmissions that share the same frequency and time resources. An increased reuse factor not only enables simultaneous packet scheduling but also facilitates the concurrent reception of reports from all associated UEs in a CF environment. While a higher reuse factor generally leads to more efficient spectrum utilization and enhanced network capacity, it also poses challenges, such as increased interference and potential degradation in signal quality. As previously stated, our default approach involves 8 RUs. That is, the maximal reuse factor for the examined topology is 8. This means that eight packets are queued for concurrently scheduling to eight RUs within a single frequency-time resource at the same coverage area.

In cases where the packet size and setup allow, the frequency-time allocation can be further divided into sub-allocations. This enables the same RUs to participate in each sub-allocation and use all the bandwidth, increasing the number of simultaneously scheduled packets to different UEs in the coverage area if conditions are met. By appropriately configuring the packet setup and increasing the number of fractions handled at each step, the overall capacity is improved. However, as the reuse factor increases, the NN layers' size and complexity of the network architecture may increase, reasonably degrading the NN performance. Fig. 5 illustrates the input and output NN tensors for dividing the frequency-time allocation into two sub-allocations to show how similar scenarios can be handled using the proposed learning method. The input tensor serially concatenates the data of the two divided frequency parts, $f_{1,1}$ and $f_{1,2}$, each containing all the SINRs between each RU and the appropriate UEs. The output tensor is a two-dimensional matrix where the rows represent the concatenation of the K RUs for each frequency allocation fraction, and the columns represent the $2K$ selected UEs. It is important to emphasize using the same frequency band across a given set of available RUs is indeed feasible. However, this is contingent upon ensuring that the K participating UEs per allocation are adequately spaced apart, so mutual cross-interference is kept within acceptable limits.

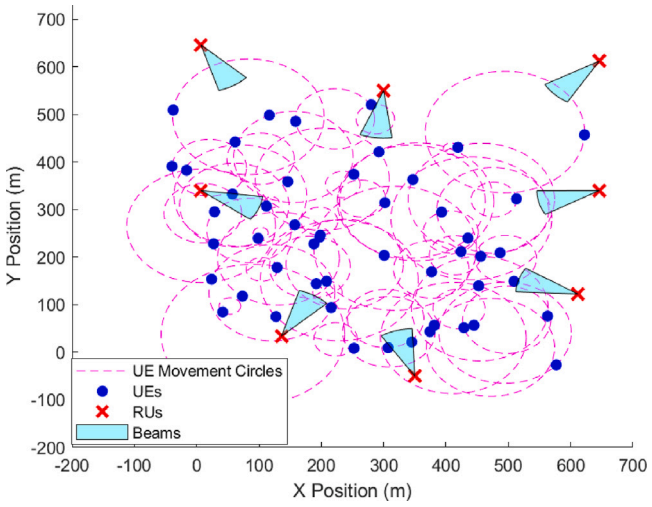


Fig. 6. Geographic map of 50 users and 8 RUs/beams.

4. Inference and post-processing

During inference, the trained NN model performs packet scheduling between available RUs and their corresponding UEs based on the provided data at each frequency-time resource. The scheduled RU-UE pairs are selected based on the highest probability obtained from the last softmax output of the NN. Post-processing is performed after each timeslot for both model-driven and data-driven approaches to account for additional power constraints.

Packet prioritization in the buffer is based on their TTL, where the most urgent packets are scheduled first. The output of the selected scheduling method suggests a scheduling scheme, which is then evaluated and adjusted to meet the constraints using a power matrix and packet switching before transmission.

4.1. Power matrix

To maximize throughput, the default transmission power for each $[RU, f]$ tuple is appropriately increased. This is accomplished by zero-initialized power matrix of size $F \times K$, then updating it via a greedy algorithm. Packets are processed in order of priority, and the minimum requisite transmission power to meet the successful transmission criterion (3) is incorporated into the power matrix, as shown in Eq. (8). This strategy aims to ensure successful transmission with minimal power expenditure. As mentioned in Section 3.2, adding transmission power to an RU escalates the interference for other concurrent transmissions on the same frequency-time resource. Such an increase must not compromise the integrity of pre-scheduled successful transmissions. If a power increment jeopardizes an existing transmission's success, the proposed match is considered untenable, and the $[RU, F]$ slot remains vacant in both the scheduler and power matrices during the packet-switching phase.

4.2. Packets switch

Following the power matrix update, the subsequent phase addresses the unresolved output slots. This phase seeks to identify packets that satisfy the scheduling conditions of available RUs for successful transmission. Packets are assessed based on priority, prioritizing those with the shortest TTL. Only packets with unique UIDs not previously scheduled in the same frequency slot qualify. For each eligible packet, the successful transmission condition is evaluated, ensuring no interference with other active transmissions on the same resource. If conditions are met, the packets are switched in the scheduling matrix, and the power matrix is updated according to (8). If no suitable packet is identified, the corresponding slot in the scheduling matrix is cleared, and the invalid packet returns to the buffer according to its TTL. Moreover, RUs unable to partake in this phase have their default transmission power reduced by a set amount to conserve energy and reduce interference. Once the scheduling and power matrices for the current timeslot are established, all buffer packet TTLs decrease by 1, and the timeslot is incremented by 1. If a buffer packet's TTL drops below zero, indicating its expiration, the packet is considered lost and is removed from the buffer. At the subsequent timeslot, empty buffer cells are refilled with new packets and reordered by TTL.

5. Design of experiments

We present the simulation environment and data generation used to train and evaluate the proposed schedulers. Subsequently, we provide a detailed description of our experimental setup.

5.1. Data generation and environment setup

The input data for all the scheduler methods evaluated in this study was produced using a MATLAB-based simulator designed specifically for this purpose, utilizing the MATLAB® 5G Toolbox™ [35]. This toolbox provides 5G radio-standard-compliant functions to generate accurate transmission data, adhering to the 3GPP standards.

We preset the number of UEs and RUs. Each RU was represented by a beam to reduce interference from other RUs and background noise, optimizing capacity allocation to the UEs while accommodating all of them. The packets were transmitted using different beams, and in the simulated environment, we required reports from each UE for each beam. The simulation incorporated dynamic UE locations, with each UE advancing a step along its circular trajectory every timeslot. Fig. 6 illustrates a snapshot of a simulated geographic map for a proposed CF environment with 50 UEs and 8 RUs. Each dashed circle denotes a UE's movement path, with each UE having its own speed and radius.

Packet information for the buffer, characterized by $[UEID, TTL]$, was generated using a random number generator. To obtain the input data for each frequency resource, we provided the simulator with the relevant UEIDs and the current timeslot, subsequently producing the updated RSSI data for the relevant packets. Training input data labels were dynamically produced by executing the perfect match for each batch. The labels can also be provided using any other deterministic approach, ensuring that the NN will be able to achieve comparable performance for any chosen reference.

In our simulation, we defined 2000 timeslots, with each timeslot representing a slot in the communication structure, typically 0.5 ms. Each timeslot is represented as a fixed normalized time interval in the simulation. During each timeslot, the positions of the UEs are updated. The UEs are modeled to move within a defined area of 500 m, such as an urban environment or a stadium. At each timeslot, the UEs change their positions based on a predefined movement model, with the radius of each UE's movement limited to 150 meters to reflect typical movement patterns within these environments.

To ensure the accuracy and relevance of our simulations, we employed the TDL-C type channel model as defined in the 3GPP 38.901 specification [4]. This model includes a delay spread of 30 ns, simulates cross-polarization, and considers medium-level antenna correlation. The MATLAB-based simulator calculates RSSI values for each UE-RU pair by incorporating path loss, shadowing, and antenna gain patterns. The transmit power for each RU is set to 30 dBm, and the shadowing standard deviation is 8 dB. The antenna gain pattern is given by $10^{((2\cos(\text{angleDifference}))/10)}$. By simulating realistic UE movements and employing a standardized channel model, we can generate reliable and reproducible data to evaluate our scheduling algorithm under realistic and challenging conditions where user positions and network conditions are constantly changing.

Table 1
CAE Proposed structure.

Parameter	Value	Kernel	Ch-in	Ch-out	Pad	Dilation
Input size	1×64	–	–	–	–	
Conv (GELU)	–	1×3	1	11	2	2
Conv (GELU)	–	1×3	11	7	2	2
Conv (GELU)	–	1×3	7	11	2	2
FC (Linear) output size	1×64	–	–	–	–	

5.2. Experimental setup

To evaluate the performance and capabilities of the proposed data-driven and model-driven scheduling methods, we selected configuration options for the environment setup and design, which adhered to a basic and didactic setup. The chosen parameters were carefully considered to ensure that our experiments can also be adapted to represent real-world scenarios. It should be noted that a trained model of the proposed CF scheduler is applicable to a range of scenarios, including time-varying numbers of UEs and allocations. However, we utilized a fixed number of RUs for training and testing. If the number of RUs is altered, the NN must be retrained. Nevertheless, modifications to the spatial distribution of the RUs can be made, and unseen configurations of RUs can be tested using the same trained model.

As part of the experimental analysis, we extensively explored different model structures and hyper-parameters, including the number of layers and neurons, regularization, training data size, dropout, batch-normalization, learning rate, and epoch number. We found the best performance versus complexity for the following design parameters. The proposed model was trained using AdamW optimizer and Gelu activation. The learning rate was initialized to 10^{-4} and weight decay of 10^{-5}. The model was trained for 200-300 epochs with input data of 5M samples and a batch size of 64. The input is shaped as 64 neurons, a concatenation of 8 SINRs for each packet, and accordingly, the output is 8×8 matrix. The proposed FC based NN model layers sizes are $64 \times 50 \times 100 \times 50 \times 64$, wheres the CNN based network is described in Table 1. The model was tested for 500K samples.

5.3. Algorithm comparison

We evaluated the performance between the suggested supervised NN based model for a reuse factor 8, exhaustive search, Hungarian method, perfect match, stable match, and round robin. The round robin algorithm is a straightforward and equitable scheduling technique that achieves a balanced allocation of resources among the available RUs. The algorithm sequentially cycles through the list of RUs, assigning one packet at a time to each RU in circular order without prioritization. This ensures that every packet has an equal opportunity to be transmitted by an available RU before moving on to the next packet. However, while this technique is easy to implement and prevents any RU from being consistently favored or neglected, it does not take into account the varying network conditions and user demands, which can result in inefficient resource utilization and reduced network performance.

The exhaustive search algorithm, due to its high computational demands, is impractical for large-scale systems. We assessed its performance in a simplified setting with eight RUs scheduled on a single frequency resource per timeslot. While this does not capture full complexity, it offers insights into the algorithm's effectiveness compared to other methods we studied.

5.4. Performance metrics

The primary performance metrics for evaluating the proposed scheduler include packet loss, throughput, computational complexity, training accuracy, and power usage.

Table 2
Setup parameters values.

Parameter	value
Number of frequency allocations	200
Number of RUs	8
Number of UEs	50
Buffer size	4000
Average TTL	[5:13] distributed uniformly ±5

Packet loss is defined as the percentage of packets that are not successfully delivered to their intended destination. It is calculated by,

$$\mathsf{PacketLoss} = \frac{\text{Number of lost packets}}{\text{Total number of sent packets}} \times 100\%. \tag{16}$$

Throughput measures the rate at which packets are successfully transmitted over the network. In our case, it is calculated as the number of packets successfully transmitted during a timeslot. The average throughput is then calculated by averaging over multiple timeslots.

$$\mathsf{Throughput} = \frac{\text{Total packets successfully transmitted}}{\text{Total number of timeslots}}. \tag{17}$$

Computational complexity is evaluated based on the time complexity and the number of operations required for the algorithm to complete. We compare the computational complexity of the proposed NN approach with that of other well-established algorithms, as described in Section 6.4.

Training accuracy is the measure of how well the NN model fits the training data. It is calculated as follows:

$$\mathsf{TrainingAccuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \times 100\%. \tag{18}$$

Power usage is another critical metric, particularly relevant in energy-constrained environments. It is evaluated by measuring the total power consumed during the transmission process. These metrics provide a comprehensive evaluation of the proposed scheduler's performance, ensuring that all critical aspects are considered.

6. Results and insights

In this section, we present the numerical performance evaluation of our proposed NN model and compare it with various model-driven algorithms in different experiments with identical data and setup. We present our results in terms of packet loss rate, average power consumption, and throughput over average TTL and frequency ranges. To conduct our experiments, we uniformly draw the TTL for a defined average, step, and range.

6.1. Model and data-driven methods evaluation

An offline training was conducted on the proposed data-driven NN model using various sizes of training data. The best and most stable performance was achieved with 5 million training samples generated from the outcomes of the perfect match model-driven algorithm. The specific values of the setup parameters used in the proposed evaluation are presented in Table 2.

Figs. 7(a) and 7(b) present the training and validation accuracies of the proposed supervised model-driven perfect match based method.

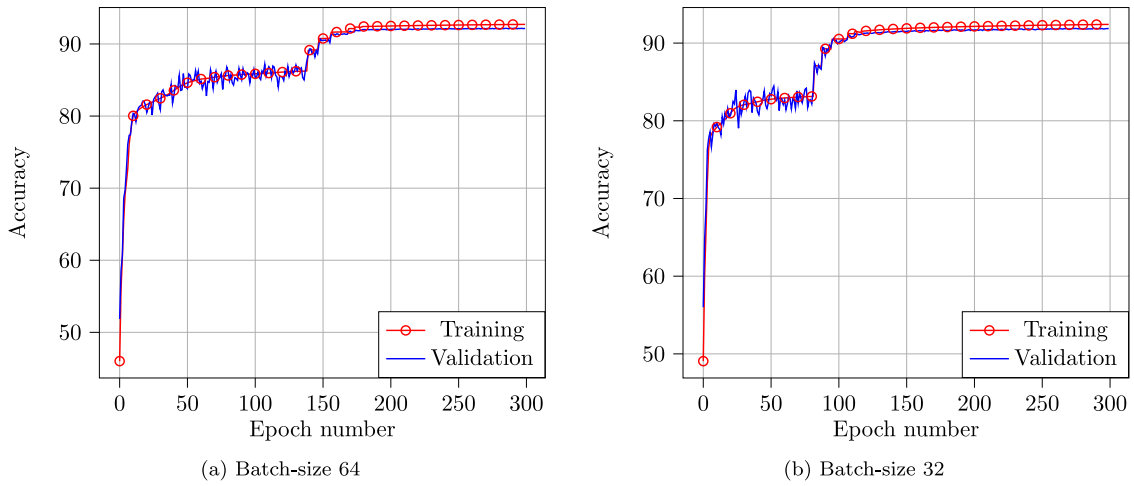


Fig. 7. NN training and validation accuracies using a dataset of 5M samples.

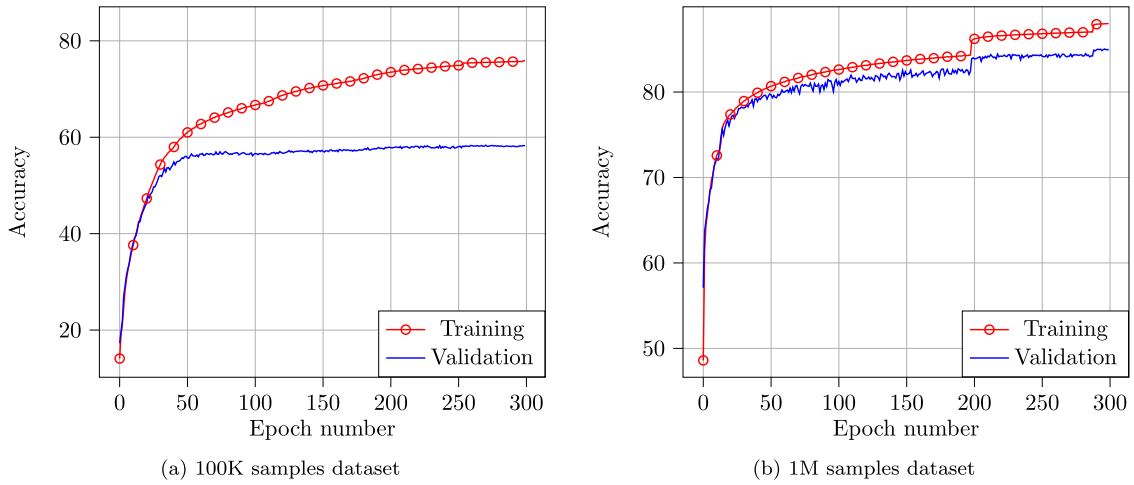


Fig. 8. NN training and validation accuracies, using a batch-size of 64.

As shown, the proposed NN achieves around 92% accuracy for both training and validation. This indicates that in 92% of the cases, the NN made the exact match with the perfect match, while in the other 10%, other matches were proposed. These other matches may also be good ones but are simply different from the ones offered by the perfect match.

6.2. Training data size

Larger datasets can improve the accuracy and generalization of ML models by providing more information about the patterns and relationships within the data, which helps the model learn more robust features and make more accurate predictions. In our training procedure, we recognized a generalization problem for small training data sets. To address this issue, we gradually increased the size of the training data set until we achieved the desired performance. Figs. 8a, 8(b), and 7(a) compare the training and inference accuracy as a function of the epoch number for training sets of size 100K, 1M, and 5M, respectively. The figures demonstrate that both training and inference accuracy behavior significantly improved for larger training sets. Furthermore, the model's robustness and convergence were also enhanced.

Another examined hyperparameter was the batch size. The training and validation accuracies of the proposed NN training for a batch of size 64 is illustrated in Fig. 7(a), and for batch of size 32 in Fig. 7(b). The results show that a similar level of performance can be achieved for both batch sizes, although the accuracy for the 32 batch size network improves earlier in terms of epoch number. However, each epoch duration is twice as long as the 64 batch size network.

6.3. Scheduling methods performance comparison

We aim to evaluate and compare the performance of the various suggested model-driven and data-driven scheduling methods. The inference performance is examined after the post-processing step. In Fig. 9 the inference results of the average packet loss as a function of the average TTL used for generating packets in the buffer are compared between the proposed methods. Furthermore, Figs. 10(a) and 10(b) provide a comparative analysis of the average throughput as a function of both the average TTL and the number of frequency-time allocation, respectively, for the evaluated methods. The calculations of packet loss and throughput are performed for each timeslot and subsequently averaged over a span of 10,000 timeslots.

The results demonstrate that the proposed NN model attains performance comparable to the perfect-match and Hungarian deterministic methods, highlighting its ability to compete effectively. As anticipated, there is a general decline in performance with the reduction of the average TTL. Notably, the NN model achieves its performance while significantly reducing both computational and temporal complexity. This highlights the benefit of the proposed data-driven approach in achieving efficient scheduling while maintaining high performance.

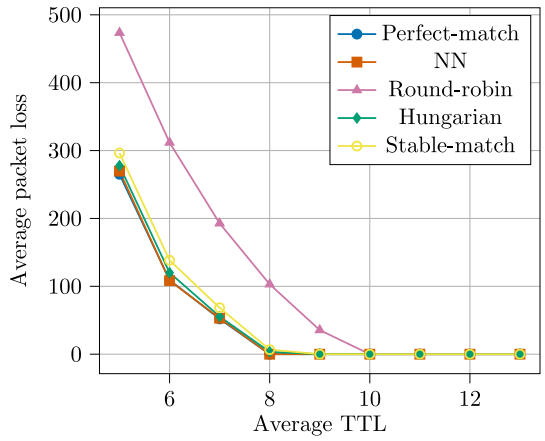


Fig. 9. Average packet loss per average TTL.

The analysis also reveals that the round-robin method falls short in terms of both packet loss and throughput. We also observe a decline in throughput as the number of frequency bands increases. This can be attributed to the increased complexity of packet switching as the number of available packets in the buffer decreases with rising frequency resources. Given the non-uniform spatial distribution of UEs, an expected variation in UE concentration around different RUs leads to an asymmetric load distribution. Consequently, this uneven distribution necessitates increased packet switching demand at those RUs experiencing higher UE concentrations. Furthermore, examining the throughput as a function of an average TTL, as depicted in Fig. 10(b), we see the gradual decrease in throughput for higher TTLs. This can be explained by the process of packet replacement in the buffer after each timeslot, where all expired packets are replaced with new ones. As a result, for a lower average TTL, a larger number of new packets are available for switching, thus increasing the probability of finding a replacement during packet switching. This subsequently leads to a higher throughput.

Expanding on our comparative analysis, we further examined the average total transmit power consumption per resource element across all the scheduled RUs. Both the Hungarian and perfect match methods demonstrated equivalent levels of power consumption. Taking the perfect match required average transmit power as the benchmark, the NN approach resulted in an approximate 2% increase in power use. The round robin algorithm led to a substantial increase, consuming nearly double the power. This marked a significant deviation from the power efficiency exhibited by the other methods. Meanwhile, the stable match method indicated a more efficient use of power, outperforming the other methods by demonstrating a reduction in consumption of about 12%.

Table 3
Complexity comparison.

Algorithm	Calculation
Exhaustive search	$\mathcal{O}\left((K!)^P\right)$
Stable match	$\mathcal{O}\left(K^2P\right)$
Perfect match	$\mathcal{O}\left(K^{3.5}\log(K)\right)$
Hungarian	$\mathcal{O}\left(K^3\right)$
NN	$\mathcal{O}(KP)$
Round robin	$\mathcal{O}(KP)$

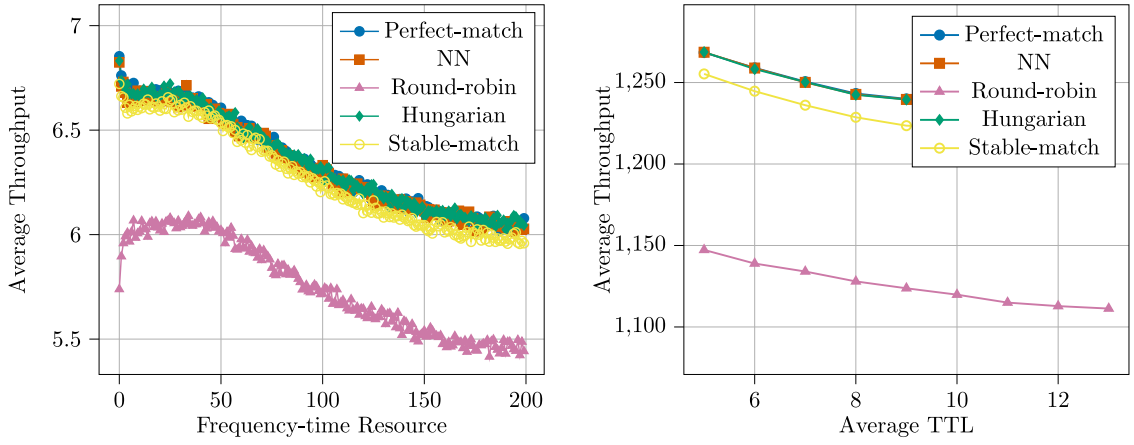
Due to the exponential time complexity of the exhaustive search algorithm, we conducted a limited analysis by considering a single frequency per time resource. Figs. 11(a) and 11(b) compare the average packet loss and average throughput as a function of the average TTL, respectively. The exhaustive search exhibits superior performance compared to all other examined methods. However, as already explained, this advantage comes at the cost of extremely high time and computational complexity, making it impractical for real-time implementation. These findings highlight the trade-off between performance and computational efficiency in scheduling algorithms.

6.4. Complexity analysis

Our supervised NN model, leveraging scheduling results from the perfect match algorithm as training labels, demonstrates competitive performance against traditional scheduling methods. The detailed computational complexities for each discussed method, presented in Table 3, are calculated per single frequency-time resource, as packet scheduling is independent between different frequency-time resources. The table ranks the algorithms from the highest to the lowest computational complexity, providing a foundational understanding of each method's theoretical demand.

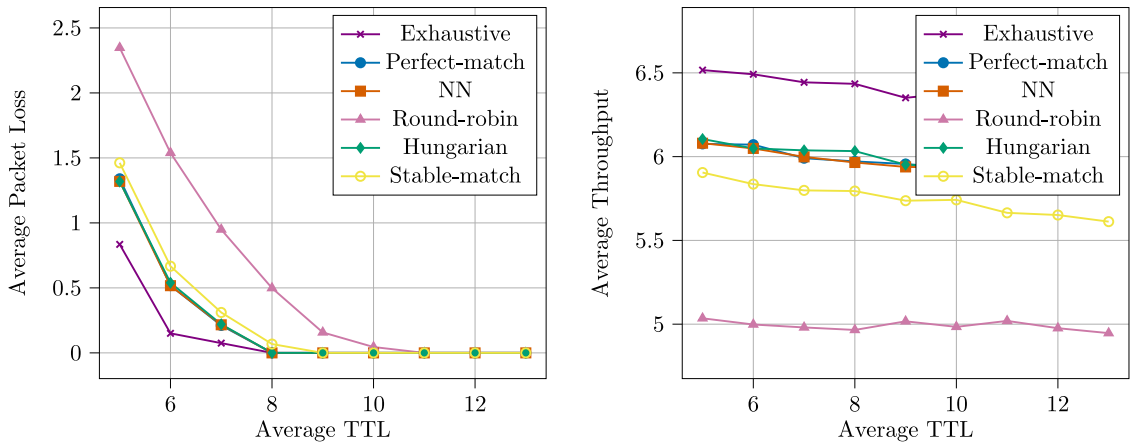
Here, P is the number of packets to be scheduled per resource. As was mentioned earlier, the number of scheduled packets is equal to the number of RUs at each iteration, meaning $P = K$.

While theoretical computational complexity provides insight into an algorithm's efficiency, it may not always reflect the actual performance. Notably, NNs can surpass the traditional methods in runtime efficiency due to their parallel processing capabilities, especially when leveraging GPUs. Moreover, the proposed NN models can be adapted to scale different network sizes without proportionally increasing its complexity



(a) per frequency allocations (b) per average TTL

Fig. 10. Average throughput (Packets per timeslot).



(a) Average packet loss (b) Average throughput

Fig. 11. Performance metrics per average TTL across one frequency resource per timeslot.

or size, further validating its practicality. This feature, pivotal in the rapidly evolving landscape of wireless networks, ensures the model's performance remains robust without necessitating significant architectural modifications. Based on the experimental setup detailed in Section 5.2, our empirical findings validate the NN model's efficiency. The model demonstrated execution speeds at least five times faster than the perfect match and Hungarian methods on our NVIDIA GeForce RTX 2080 Ti Generation hardware, a testament to its efficiency and suitability for large-scale scheduling environments.

In summary, the NN-based approach not only achieves competitive theoretical computational efficiency but also excels in real-world applications, where its execution speed and scalability offer benefits over traditional algorithms. Whereas the complexity of the Hungarian and perfect match methods scales with the cube of the number of base stations, $\mathcal{O}\left((K^3)\right)$, the effective computational load of our NN approach exhibits behavior of $\mathcal{O}\left((K^2)\right)$, particularly when considering the operational context of large-scale deployments.

7. Conclusions and future work

In this work, we introduce a DL based algorithm specifically designed for a CF scheduler. We propose a data-driven, supervised NN that effectively learns and mimics the behavior of the perfect match algorithm, delivering similar output with significantly reduced complexity. The proposed algorithm was tested and evaluated across multiple resources, encompassing multiple RUs and UEs, utilizing simulated data. Our results demonstrated that the NN model is highly effective, achieving performance that is competitive with the model-driven perfect match algorithm. Additionally, our model outperforms simpler methods like the round-robin algorithm, while maintaining competitive performance when compared to the exhaustive search algorithm.

Although our supervised NN-based approach significantly reduces computational complexity, it is important to recognize that its performance is closely tied to the deterministic method used to generate training labels. However, this comparison is made against deterministic methods with very high performance, such as the perfect match algorithm. As a result, the performance of the NN in terms of packet loss and throughput is comparable to these high-performance classical methods, although the NN can make different, but still effective, scheduling decisions. Future work could focus on reducing or eliminating the dependence on classical methods, allowing the NN to optimize scheduling decisions more independently and effectively. This approach could be particularly beneficial in scenarios that involve a large number of RUs, where traditional methods may struggle with scalability.

Looking ahead, our goal is to broaden the scope of our work to encompass various areas, including massive MIMO, buffer optimization, transmission power optimization, and the inclusion of additional scheduling constraints. These constraints may involve localization, varying channel effects, expanding to different MCS, multiple beams under each RU, and implementing parallel scheduling over serial using frequency-time resources. A key aspect of our future work will be the design of a scheduler that addresses both communication and high-accuracy localization needs, optimizing system performance through enhanced localization accuracy and scheduling efficiency.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the Israel Science Foundation (ISF) under Grant 899/21 and Grant 3211/23, and in part by the NSF-(Israel-USA) Binational Science Foundation (BSF) Grant and by the Israel Innovation Authority.

Data availability

Data will be made available on request.

References

[1] H.Q. Ngo, A. Ashikhmin, H. Yang, E.G. Larsson, T.L. Marzetta, Cell-free massive MIMO: Uniformly great service for everyone, in: 2015 IEEE 16th International Workshop on Signal Processing Advances in Wireless Communications, SPAWC, 2015, pp. 201–205, http://dx.doi.org/10.1109/SPAWC.2015.7227028.

[2] H.Q. Ngo, A. Ashikhmin, H. Yang, E.G. Larsson, T.L. Marzetta, Cell-free massive MIMO versus small cells, IEEE Trans. Wireless Commun. 16 (3) (2017) 1834–1850.

[3] E. Dahlman, S. Parkvall, J. Skold, 5G NR: The Next Generation Wireless Access Technology, Academic Press, 2020.

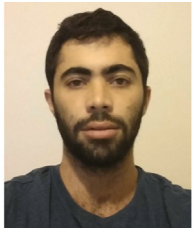
[4] 3GPP, 3rd Generation Partnership Project (3GPP). Study on Channel Model for Frequencies From 0.5 to 100 GHz, Technical specification (ts), 3rd Generation Partnership Project (3GPP), URL https://www.3gpp.org/ftp/Specs/archive/38_series/38.901.

[5] E. Björnson, J. Hoydis, L. Sanguinetti, Massive MIMO networks: Spectral, energy, and hardware efficiency, Found. Trends Signal Process. 11 (3–4) (2017) 154–655.

[6] G. Interdonato, E. Björnson, H.Q. Ngo, P. Frenger, E.G. Larsson, Ubiquitous cell-free massive MIMO communications, EURASIP J. Wireless Commun. Networking 2019 (1) (2019) 1–13.

[7] S. Eddine Hajri, J. Denis, M. Assaad, Enhancing favorable propagation in cell-free massive MIMO through spatial user grouping, in: 2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications, SPAWC, 2018, pp. 1–5, http://dx.doi.org/10.1109/SPAWC.2018.8445847.
[8] P. Imputato, S. Avallone, An analysis of the impact of network device buffers on packet schedulers through experiments and simulations, Simul. Model. Pract. Theory 80 (2018) 1–18.
[9] S. Panda, Performance optimization of cell-free massive MIMO system with power control approach, AEU-Int. J. Electron. Commun. 97 (2018) 210–219.
[10] E. Björnson, L. Sanguinetti, Cell-free versus cellular massive MIMO: What processing is needed for cell-free to win? in: 2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications, SPAWC, 2019, pp. 1–5, http://dx.doi.org/10.1109/SPAWC.2019.8815488.
[11] A. Haghrah, M.P. Abdollahi, H. Azarhava, J.M. Niya, A survey on the handover management in 5G-NR cellular networks: aspects, approaches and challenges, EURASIP J. Wireless Commun. Networking 2023 (1) (2023) 1–57.
[12] R. Mennes, M. Camelo, M. Claeys, S. Latré, A neural-network-based MF-TDMA MAC scheduler for collaborative wireless networks, in: 2018 IEEE Wireless Communications and Networking Conference, WCNC, 2018, pp. 1–6, http://dx.doi.org/10.1109/WCNC.2018.8377044.
[13] S. Chinchali, P. Hu, T. Chu, M. Sharma, M. Bansal, R. Misra, M. Pavone, S. Katti, Cellular network traffic scheduling with deep reinforcement learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, 2018.
[14] M. Guenach, A.A. Gorji, A. Bourdoux, A deep neural architecture for real-time access point scheduling in uplink cell-free massive MIMO, IEEE Trans. Wireless Commun. 21 (3) (2021) 1529–1541.
[15] M. Guenach, A.A. Gorji, A. Bourdoux, Joint power control and access point scheduling in fronthaul-constrained uplink cell-free massive MIMO systems, IEEE Trans. Commun. 69 (4) (2020) 2709–2722.
[16] F. Moradi, V. Hakami, S.V. Azhari, Learning to optimize power allocation in cell-free massive MIMO networks with hybrid green energy, Comput. Commun. (2024).
[17] Q. Fan, Y. Zhang, J. Li, D. Wang, H. Zhang, X. You, Network-assisted full-duplex cell-free mmwave networks: Hybrid MIMO processing and multi-agent DRL-based power allocation, Phys. Commun. (2024) 102350.
[18] R. Arshad, S. Baig, S. Aslam, User clustering in cell-free massive MIMO NOMA system: A learning based and user centric approach, Alex. Eng. J. 90 (2024) 183–196.
[19] M. Zaher, Ö.T. Demir, E. Björnson, M. Petrova, Learning-based downlink power allocation in cell-free massive MIMO systems, IEEE Trans. Wireless Commun. 22 (1) (2022) 174–188.
[20] A. Lacava, M. Polese, R. Sivaraj, R. Soundrarajan, B.S. Bhati, T. Singh, T. Zugno, F. Cuomo, T. Melodia, Programmable and customized intelligence for traffic steering in 5G networks using open RAN architectures, IEEE Trans. Mob. Comput. 23 (4) (2023) 2882–2897.
[21] T. Zheng, J. Wan, J. Zhang, C. Jiang, Deep reinforcement learning-based workload scheduling for edge computing, J. Cloud Comput. 11 (1) (2022) 3.
[22] Y. Hao, M. Chen, H. Gharavi, Y. Zhang, K. Hwang, Deep reinforcement learning for edge service placement in softwarized industrial cyber-physical system, IEEE Trans. Ind. Inform. 17 (8) (2020) 5552–5561.
[23] L. Sun, T. Zong, S. Wang, Y. Liu, Y. Wang, Towards optimal low-latency live video streaming, IEEE/ACM Trans. Netw. 29 (5) (2021) 2327–2338.
[24] Y.-J. Choi, C.S. Kim, S. Bahk, Flexible design of frequency reuse factor in OFDMA cellular networks, in: ICC, Vol. 6, Citeseer, 2006, pp. 1784–1788.
[25] Y. Li, X. Zhang, C. Cui, S. Wang, S. Ma, Fleet: Improving quality of experience for low-latency live video streaming, IEEE Trans. Circuits Syst. Video Technol. 33 (9) (2023) 5242–5256.
[26] M. Shreedhar, G. Varghese, Efficient fair queuing using deficit round-robin, IEEE/ACM Trans. Netw. 4 (3) (1996) 375–385.
[27] J. Edmonds, Maximum matching and a polyhedron with 0, 1-vertices, J. Res. Natl. Bureau Std. B 69 (125–130) (1965) 55–56.
[28] D.G. McVitie, L.B. Wilson, The stable marriage problem, Commun. ACM 14 (7) (1971) 486–490.
[29] H.W. Kuhn, The hungarian method for the assignment problem, Naval Res. Logist. Q. 2 (1–2) 83–97.
[30] D.J. Bernstein, Understanding brute force, in: Workshop Record of ECRYPT STVL Workshop on Symmetric Key Encryption, ESTREAM Report, Vol. 36, Citeseer, 2005, p. 2005.
[31] D. Bertsimas, J.N. Tsitsiklis, Introduction to Linear Optimization, vol. 6, Athena Scientific Belmont, MA, 1997.
[32] S. Whitehouse, Gale-Shapley algorithm, 2014, URL https://youtu.be/0m_YW1zVs-Q.
[33] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[34] S. Gold, A. Rangarajan, et al., Softmax to softassign: Neural network algorithms for combinatorial optimization, J. Artif. Neural Netw. 2 (4) (1996) 381–399.
[35] MATLAB 5G Toolbox, MATLAB 5G Toolbox, The MathWorks, Natick, MA, USA, 2020, URL https://www.mathworks.com/products/5g.html.

Yara Huleihel received her B.Sc. and M.Sc. degrees in electrical and computer engineering from Ben-Gurion University of the Negev, Israel, in 2011 and 2014, respectively. She is currently pursuing a Ph.D. degree at the same institution. Her research interests include communication networks and machine learning.



Gil Maman completed his B.Sc. in Electrical and Computer Engineering at Ben-Gurion University of the Negev, Israel, in 2019. He is currently pursuing his M.Sc. degree at the same university, focusing on communication networks and machine learning.

Zion Hadad received his Ph.D. in electrical engineering from CUNY, New York. He is an inventor and pioneer in OFDMA technology since 1997, Dr. Hadad has significantly contributed to the commercialization and standardization of communication technologies. As the founder and CEO of Runcom, Dr. Hadad was instrumental in establishing OFDMA as the foundation for key standards, including DVB-RCT in 1999, IEEE 802.16 (2000–2009), and 3GPP 4G LTE beginning in 2004. Throughout his career, Dr. Hadad has authored dozens of patents, significantly advancing the field of wireless communication technologies. In 2015, he founded RunEL, focusing on advanced 5G RAN technologies, including 5G SoCs and base stations (gNBs). His current research explores 6G technologies, integrating cutting-edge advancements such as Generative AI, Quantum Computing, Blockchain, and ISAC. Dr. Hadad aims to design highly efficient, low-power platforms to address the emerging challenges of the next decade.

Eli Shasha received his Ph.D. in Physics from Tel Aviv University and has an extensive background in ASIC RD technologies. Dr. Shasha's career includes roles as a VLSI project manager at BreezeCom and a design engineer at Motorola. Since joining Runcom in 2001, he has led various projects, including the development of algorithms for IEEE 802.16 and the simulation of channel estimation for OFDM channels. Currently, he spearheads the 5G PHY/MAC development at RunEL, focusing on NR 5G LDPC, Polar encoding, and AI-driven MAC scheduling. His multidisciplinary expertise spans physics, electronics, and computer science, with a particular focus on communication systems.

Haim Permuter received the B.Sc. and M.Sc. degrees (summa cum laude) in electrical and computer engineering from Ben-Gurion University of the Negev, Israel, in 1997 and 2003, respectively, and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, USA, in 2008. From 1997 to 2004, he was an Officer with the Research and Development Unit of the Israeli Defense Forces. Since 2009, he has been with the Department of Electrical and Computer Engineering, Ben-Gurion University of the Negev, where he is currently a Professor and the Luck-Hille Chair of electrical engineering. He also serves as the Head of the communication track in his department. He was a recipient of several awards, among them the Fullbright Fellowship, the Stanford Graduate Fellowship (SGF), the Allon Fellowship, and the U.S.–Israel Binational Science Foundation Bergmann Memorial Award. He has served on the editorial boards for the IEEE TRANSACTIONS ON INFORMATION THEORY from 2013 to 2016 and he is serving again since 2023.