Horizon Europe Framework Programme
Call: HORIZON-JU-SNS-2022 (HORIZON-JU-SNS-2022)
Type of action: HORIZON JU Research and Innovation Action

CENTRIC

Towards an AI-native, user-centric air interface for 6G networks

Grant Agreement Number: 101096379

WP4–AI-AI Protocols and Radio Resource Management

D4.3–AI-based Techniques for Sustainable & Human-friendly RRM

Contractual Delivery Date:	June 30, 2025
Actual Delivery Date:	June 3, 2025
Responsible Beneficiary:	CNIT
Contributing Beneficiaries:	NNF, CNIT, CNR, KCL, KCL, KEYSIGHT
Dissemination Level:	Public
Version:	Final

PROPRIETARY RIGHTS STATEMENT
This project has received funding from the European Union's Smart Networks and Services Joint Undertaking (SNS JU) under grant agreement No 101096379.

Document Information

Document ID:	WP4/D4.3
Version Date:	June 3, 2025
Total Number of Pages:	123
Abstract:	This document summarizes the achievements of the CENTRIC consortium during the two and a half years of project duration. In particular, it describes the research outcomes of Task 4.2 focused on developing AI-driven techniques for sustainable and human-centric radio resource management. Indeed, as emerging mobile services require to collect increasingly large volumes of data through the radio link and process them via AI/ML models at the network edge, it is imperative to limit the demand for communication and computational resources. To address this challenge, we designed innovative techniques to minimize resource, hence energy, consumption while fulfilling performance requirements and system constraints. Specifically, we created and tested novel solutions for (i) the efficient management of radio interfaces and radio traffic, despite the highly dynamic operational environments, (ii) the energy-aware training and execution of AI/ML models at the network edge, and (iii) the configuration of innovative network architectures that besides network performance metrics, account for EMF exposure. Our solutions often leverage data-driven approaches, which can tackle the complexity and the scale of new generation network systems more effectively when compared to traditional optimization methods. Overall, the solutions and methods we propose represent an effective set of tools to make the support of AI/ML-based mobile services sustainable.
Keywords:	machine learning, reinforcement learning, decentralized learning, random access, energy efficiency, Age of Information (AoI) optimization, EMF exposure

Authors

Full name	**Beneficiary / Organisation**	**e-mail**	**Role**
Carla Fabiana Chiasserini, Claudio Casetti, Roberto Garello	CNIT	carla.chiasserini@polito.it	Deliverable coordinator
Stefano Buzzi	CNIT	buzzi@unicas.it	Individual contributor
Septimia Sarbu	UOULU	sarbu.septimia@uoulu.fi	Individual contributor
Bryan Liu	NNF	bryan.liu@nokia-bell-labs.com	Individual contributor
Petteri Kela	Nokia, Finland	petteri.kela@nokia.com	Individual contributor
Alvaro Valcarce	NNF	alvaro.valcarce_rial@nokia-bell-labs.com	Individual contributor
Francesco Malandrino	CNR	francesco.malandrino@cnr.it	Individual contributor
Matteo Zecchina	KCL	matteo.1.zecchin@kcl.ac.uk	Individual contributor

Reviewers

Carles Navarro Manchon	KEYSIGHT	carles.navarro-manchon@keysight.com	Internal reviewer
Ramoni Adeogun	Aalborg	ra@es.aau.dk	Internal reviewer
Ramoni Adeogun	University	ra@es.aau.dk	Internal reviewer

Executive Summary

This document reports the main research achievements of Task 4.2 within Work Package 4 of the CENTRIC project. Directly contributing to CENTRIC's Objective 3 (*"To develop AI methods for the discovery of customized lightweight communication protocols"*), this work focuses on developing novel Artificial Intelligence (AI) based techniques for sustainable and human-friendly Radio Resource Management (RRM) in future 6G networks. Specifically, we target 6G Key Value Indicators (KVIs) such as energy efficiency and human exposure to Electromagnetic Fields (EMF), alongside traditional performance metrics, ensuring network operation is powerful, sustainable, and responsible. The innovations presented span foundational prediction methods, intelligent scheduling, energy-efficient AI execution at the network edge, and EMF-aware network design and operation, particularly within user-centric cell-free architectures.

According to Task 4.2.1 objective (*Develop techniques to optimize some of the novel 6G sustainability KVIs, such as energy- efficiency and power consumption*), the document presents effective techniques tfor energy efficiency at the radio interface, while meeting Quality of Service (QoS) requirements such as coverage and capacity. To achieve the task goals, we investigate multi-objective optimization techniques by leveraging AI/ML-based methodologies. The first fundamental AI/ML-based optimization we tackle is **packet scheduling**. The proposed solution is 3GPP-compliant, has low complexity, and provides fair throughput across users. The scheme, leveraging deep RL, also exploits a novel training technique for Proximal Policy Optimization (PPO) tailored for radio resource allocation. Results demonstrate the superiority of the proposed scheduler with respect to heuristic alternatives in realistic and standard-compliant 5G system-level simulations. The above work highlights how important and complex it may be to adhere to system and services constraints while optimizing a radio interface. Indeed, achieving sustainable and human-friendly RRM relies on robust AI decision-making, which in turn necessitates reliable predictions of the complex network dynamics. To this end, we introduce **novel methodologies for reliable predictions**. Notably, governing prediction uncertainty is of utmost importance ito adhere to services safety standards. We then propose a calibration technique that can cope with complex dynamic environments and enhances the reliability of probabilistic forecasters by providing accurate error bounds; thus, it is a powerful methodology to handle constraints in control policies for optimal task execution.

Beyond optimizing RRM decisions, it is critical to ensure sustainable AI/ML-based prediction and decision-making processes at the network edge. In accordance with the objectives of Task 4.2.2 (*Develop energy-efficient model training and inference algorithms for AI-AI radio resource management*), we thus investigate how to execute training and execution of AI/ML models in an energy-efficient way at the network edge. Firstly, we formulate the innovative problem of **optimal inference task offloading and execution at the network edge**, through ML models composed of shareable blocks of layers. The solution we develop yields substantial gains with respect to the state of the art, in terms of efficiency of the inference process, bandwidth occupation, and energy savings. Secondly, as crucial to sustainable AI/ML is not only efficient processing but also data selection, we introduce a method that aims at **minimizing the number of data samples to use for ML model training**. Results obtained using state-of-the-art datasets and neural network models, show that the proposed technique dramatically reduces the required data, and network nodes, that need to get involved in the training of ML models, while preserving privacy and learning

performance. Thirdly, whenever distributed learning, such as Federated Learning (FL), is needed to preserve data privacy, we propose decentralized optimization frameworks that reduce communication costs and energy consumption. To this end, we employ second-order optimization techniques and we present different ways to reduce the communication and energy resources for training ML models and executing them for inference.

Finally, addressing the 'human-friendly' aspect requires explicit consideration of the **impact of radio connectivity on human exposure to EMFs**. As foreseen in Task 4.2.3 (*Design of AI-based algorithms for optimizing cell-free networks from a service and EMF viewpoint*), we have explored both traditional and ML-based techniques to reduce the EMF exposure, while focusing on the emerging cell-free networks paradigm. In this context, we first tackle the fundamental issue of **making human-centric decisions on the point-of-access management and on which users should be associated with which points of access**. We face these issues by defining an efficient solution concept that optimally balances network performance and against metrics like energy consumption and EMF exposure. Importantly, using detailed models for EMF exposure estimation and standard-specified signal propagation models, we demonstrate that our solution reduces energy consumption by over 80% with respect to state-of-the-art network alternatives. Then we further address the **suitability of cell-free deployments to provide EMF-compliant wireless coverage** by focusing on massive multiple input multiple output (MIMO). We define the problem of power control in user-centric cell-free massive MIMO systems under EMF constraints, aiming at maximizing the minimum data rate across users on both the uplink and the downlink, showing that EMF safety restrictions can be easily met without jeopardizing data rate.

In summary, the research we conducted has provided a set of solutions that fully meet the objectives of Task 4.2 and of CENTRIC Objective #3. Specifically, our research results offer substantial improvements in performance reliability, energy efficiency, and EMF control, addressing the dual challenge of the increasing resource management complexity and of the stringent operational constraints characterizing next-generation wireless networks. In accordance with Task 4.2.1, the methods presented in Chapter 2 show that AI/ML-based multi-objective optimization techniques and approaches for uncertainty control are highly effective tools to increase the effectiveness of power allocation and RRM as well as to make the system scalable establishing the best trade-off between throughput performance and computational efficiency.

Then, as specifically targeted by Task 4.2.2, our work presented in Chapter 3 demonstrates that, when compared to the state of the art, our ML model caching and intelligent strategies for collaborative training and inference can substantially reduce not only the communication overhead (by 25%), but also memory usage and computing time (by 80% and 77%, resp.). Additionally, they can dramatically decrease energy consumption, requiring just 6.7%–20% of the energy consumed by existing paradigms.

Finally, as reported in Chapter 4, the activities within Task 4.2.3 have provided paradigms for cell-free networks that meet both Qos and EMF constraints. Notably, our approach significantly outperforms conventional ones in terms of energy consumption and EMF exposure —achieving up to 80% savings in some scenarios. Furthermore, our solutions ensure that network performance metrics such as data rate requirements are still met, effectively striking a balance between user QoS and societal health concerns. In this respect, it is important to remark that our study has taken advantage of advanced computational models, both statis-

tical and deterministic, of human body EMF exposure. Specifically, with the aim to carefully assess the impact of 6G network deployments on the human body, we have adopted specific metrics such as electric field strength and dosimetric quantities (i.e., the EMF dose absorbed by biological tissues).

At last, based on the insights provided by our study, we argue that the following methods should be further leveraged and investigated for the realization of 6G systems.

- Enhanced RL algorithms (Section 2.1) have great potential in efficiently scheduling massive radio resources, paving the way for more sophisticated and resource-aware MAC-layer scheduling in next-generation cellular networks. In particular, algorithms need to be further improved to ensure the fulfillment of QoS constraints during the entire operational phase. Additionally, to address the latency on scheduling, a potential approach is to eliminate the loop over the spatial layers.
- Calibration methods (Section 2.2) can be extended to tackle key challenges in wireless networks, such as non-stationary data distributions, communication noise, and decentralized data processing. Addressing these issues will be essential for the widespread adoption of AI-driven techniques and the development of safe AI-native networks;
- The solution created for ML models caching and execution at the mobile edge (Section 3.1) underscores the potential of this innovative paradigm for fusion of multi-modal sensor data at the edge – an application that is gaining increasing attention. Distributed sensor fusion and multi-data processing can indeed be provided through dynamic neural networks that operate according to context and semantic information, and optimize the transmission of such information depending on its relevance level.
- The volume of data collected and processed by ML models (Section 3.2) and the communication overhead for distributed ML model training (Section 3.4) should be minimized – especially in the case of the emerging Large Language Models (LLMs) –, while still ensuring high quality levels of learning and inference. Ways to do so when LLMs are applied require additional study while accounting for data heterogeneity.
- The metrics we used for assessing human-exposure to EMF (Section 4.1) represent a valuable reference for the investigation of this important KVI in the design of next-generation systems. Further, the deep unfolding method (Section 4.2) could be applied in the DL setting to further reduce the system's complexity. Expert knowledge can indeed be incorporated into the design, leading to higher practical feasibility and further simplifying computational resources.

Table of Contents

List of Figures

List of Acronyms and Abbreviations

1LDS Single Loop Deep Scheduler

2LDS Double Loop Deep Scheduler

3GPP 3rd Generation Partnership Project

5G Fifth Generation

5G NR 5G New Radio

AB Action Branching

AC Actor-Critic

AI Artificial Intelligence

AoI Age of Information

BLER Block Error Rate

BTS Base Transceiver Station

CP Conformal Prediction

CPU Central Processing Unit

CQI Channel Quality Indicator

CSI Channel State Information

DQN Deep Q-Network

DDQN Double Deep Q-Network

DL Downlink

DP Dynamic Programming

DRL Deep Reinforcement Learning

DSACD Distributional Soft Actor-Critic Discrete

DNN Deep Neural Network

EMF Electromagnetic Field

FB Full Buffer

FC Fully Connected

fps Frames per Second

FD Frequency Domain

FDS Frequency Domain Scheduling

FR1 Frequency Range 1

GPU Graphics Processing Unit

GoB Grid of Beams

gNB gNodeB

KPI Key Performance Indicator

KVI Key Value Indicator

KNN K-Nearest Neighbors

MPC Model Predictive Control

MARL Multi-Agent Reinforcement Learning

MCS Modulation and Coding Scheme

MCTS Monte Carlo Tree Search

MIMO Multiple Input Multiple Output

ML Machine Learning

MRC Maximal-ratio combining

MSE Mean Square Error

MU-MIMO Multi-User MIMO

PF Proportional Fair

PMI Precoder Matrix Indicator

PPO Proximal Policy Optimization

PoA Point-of-access

QoS Quality of Service

RB Resource Block

RBG Resource Block Group

RI Rank Indicator

RL Reinforcement Learning

RR Round Robin

RRM Radio Resource Management

RZF Regularized Zero Forcing

SAC Soft Actor-Critic

SACD Soft Actor-Critic Discrete

SCO Successive Convex Optimization

SE Spectral Efficiency

SNR Signal-to-noise Ratio

SotA State of the Art

SDS Spatial Domain Scheduling

SU-MIMO Single-User MIMO

TD Time Domain

TDS Time Domain Scheduling

TTI Transmission Time Interval

UE User Equipment

UL Uplink

UPT User Perceived Throughput

vRAN virtualized Radio Access Network

ZF Zero Forcing

1. Introduction

This deliverable describes the radio resource management innovations discovered by CENTRIC within Task 4.2. Our innovations leverage Artificial Intelligence (AI) for improving energy efficiency and better controlling Electromagnetic Field (EMF) exposure in 6G networks while fulfilling the QoS requirements for novel specialized services. They therefore contribute to CENTRIC's Objective 3 (*"To develop AI methods for the discovery of customized lightweight communication protocols"*), by developing techniques for sustainable and human-friendly Radio Resource Management (RRM). 6G is indeed expected to provide not only ubiquitous coverage and high-data rate connectivity, but also to facilitate increasingly sophisticated cyber-physical systems. To this end, 6G systems need to intertwine communication and computing capabilities, posing significantt challenges to operational efficiency. The effective orchestration of such complex and dynamic environments requires innovative methods that fulfill performance requirements while optimizing energy usage, reliability, and QoS. Central to addressing these challenges are advanced techniques in resource allocation and management. Specifically radio resource management, necessitates solutions capable of balancing competing objectives — such as maximizing throughput, minimizing latency, and conserving energy — under fluctuating channel conditions and traffic demands. As traditional model-based approaches often struggle to cope with dynamic and complex scenarios, data-driven, AI/ML-based methods have emerged as a relevant approach, leveraging their ability to adaptively optimize system performance in highly variable environments. The adoption of AI/ML approaches however comes with its own set of challenges.

A critical issue is the reliable quantification and management of prediction uncertainty, addressed in Chapter 2 of this document. Predictions play an essential role in system monitoring and control decisions within cyber-physical contexts, where uncertainty can directly impact safety, operational continuity, and performance fulfillment. Conventional forecasting models often fail to adequately address scenarios characterized by branching future trajectories or rapidly shifting dynamics, leading to overly conservative or insufficiently protective operational decisions. To tackle this issue, the emerging field of Conformal Prediction has gained attention, since it provides a method for post-hoc calibration of predictive models, ensuring reliability and clearly defined risk boundaries. In particular, by integrating probabilistic models with Conformal Prediction (CP), the proposed Probabilistic Time Series-Conformal Risk Prediction (PTS-CRC) method can significantly enhance the capability to make informed control decisions in environments with high uncertainty, thus achieving better safety guarantees and more efficient resource utilization. Unlike conventional methods, PTS-CRC constructs predictive sets using ensembles of prototype trajectories, effectively capturing uncertainties. PTS-CRC has been integrated into a new Model Predictive Control (MPC) framework designed for both open-loop and closed-loop control problems. Notably, the method has been applied to predictive power control for interference mitigation and to energy-efficient HARQ-based schemes. The proposed strategies are validated through wireless network experiments, demonstrating superior predictive accuracy and safer, higher-performance control policies across the diverse tasks.

The second major theme addressed in Chapter 2 is about RL-based schedulers for efficient radio resource allocation. Advancements in RL have indeed transformed scheduling mechanisms in wireless communications. Traditional methods for packet scheduling, while

efficient for simpler scenarios, exhibit significant limitations in scalability and adaptability when faced with the multidimensional problem of assigning resources in time, frequency, and space, thus making ML, and RL in particular, essential tools. Deep RL techniques, such as Proximal Policy Optimization (PPO), have proved to be able to allocate resource while optimally balancing user fairness, throughput, and computational efficiency. These methods inherently adapt to dynamic channel conditions and user demands, offering real-time and scalable solutions. The approach proposed in CENTRIC includes novel enhancements like expert policy guidance, entropy learning, and experience replay methods, which significantly accelerate training and improve scheduler performance. In detail, two primary training methodologies—on-policy and off-policy deep scheduler training—are investigated. On-policy training methods benefit from incorporating expert guidance, reducing convergence time and improving real-time responsiveness. Conversely, off-policy training methods leverage replay buffers to substantially enhance learning efficiency, particularly beneficial in dynamic, high-dimensional network scenarios. Through rigorous system-level simulations adhering to realistic and 3GPP-compliant standards, these deep schedulers consistently outperform their heuristic counterparts, offering robust generalization capabilities across varying bandwidths, MU-MIMO configurations, and traffic patterns.

Chapter 3 presents a set of strategies for the support of scalable and sustainable AI/ML at the radio access network and the network edge, designing innovative methods for minimizing the energy consumption associated with ML models training and execution. The key ideas of the poroposed strategies are intelligent caching and sharing of ML models (or sections thereof) at the edge, selective data usage for ML model training, and communication-efficient protocols for distributed learning — all contributing to a more sustainable use of AI/ML at the network edge. More in detail, we first present OffloaDNN, an optimization framework that addresses scalable offloading of inference tasks at the edge. In ligh of the problem NP-hardness, we design a weighted-tree-based heuristic that intelligently caches and shares DNN layers to reduce memory and compute usage without compromising task accuracy. Key innovations include freezing shared DNN blocks, fine-tuning task-specific layers, and pruning them to match resource constraints. OffloaDNN also optimizes task offloading rates from mobile users to the network edge, thus tackling not only computing but also communication resource allocation. Simulation and emulation results demonstrate that OffloaDNN provides significant gains over the state of the art in terms of memory usage (reduced by $>$80%), computing time (dropped by 77%), and task admission rates (increased by $>$25%).

To further reduce computing time during ML model training, techniques to reduce the volume of used data can be adopted. However, in order not to degrade the ML model performance, data selection generated by distributed sources must be carefully performed. We thus define a data-driven source selection algorithm that constructs cooperation graphs among the network nodes that can act as training clients in a decentralized learning process. The approach essentially allows only the most relevant datasets to participate in collaborative model training, thereby conserving energy. Three metrics — dataset distance, singular value decomposition-based similarity, and a hybrid one — are explored for graph construction. Then an iterative hill-climbing algorithm uses these metrics to balance model accuracy and training cost, achieving close-to-optimal performance even in highly heterogeneous and decentralized environments.
Focusing on cooperative decentralized learning tasks, it is essential to make them as energy-efficient as possible, non only from the computing but also also the communication viewpoint.

To this end, we explore the role of second-order optimization methods in federated learning. We also leverage the superposition properties of the wireless medium for efficient model aggregation, drastically reducing energy and latency. In comparison to popular methods like FedAvg and FedProx, the proposed solution consumes only a fraction of the energy and a fraction of the CPU time.

Finally, Chapter 4 explores the integration of artificial intelligence and advanced architectural paradigms in next-generation wireless networks to address one of the most pressing public concerns: EMF exposure. This is achieved by addressing two primary themes: human-centric decision-making for network configuration of cell-free networks and EMF-compliant resource allocation in cell-free massive MIMO systems. The first one delves into the management of cell-free networks—an emerging architecture where the traditional concept of cellular coverage is replaced by a flexible, dense mesh of Points of Access (PoAs). This flexible network architecture offers a path toward more efficient and responsive networks, but it also introduces complexity in user association, power control, and beam management. Most critically, it raises the issue of how to mitigate EMF exposure not just for active users but for all individuals present in the network area. To address these challenges, we propose a human-centric network management paradigm for cell-free networks, embodied in the Cluster-then-Match heuristic consisting of two stages. It first clusters users based on spatial proximity and then assigns these clusters to specific PoAs using the Hungarian optimization method. Besides throughput, the strategy explicitly considers SAR (Specific Absorption Rate) and IPD (Incident Power Density) as main performance metrics. Performance evaluations show that this approach significantly outperforms conventional approaches, including some that rely on ML, in terms of reducing energy consumption and EMF exposure—achieving up to 80% savings in some scenarios. Furthermore, the envisioned scheme ensures that network performance metrics such as data rate requirements are still met, striking a balance between user QoS and societal health concerns.
In a cell-free massive MIMO system, instead, we propose both a model-based and a data-driven method to let the system effectively operate while meeting EMF constraints. The model-based approach involves maximizing the minimum data rate across users, subject to strict EMF constraints in both uplink and downlink transmissions. Simulation results demonstrate that such a solution outperforms conventional multi-cell MIMO systems in maintaining user fairness and meeting EMF safety thresholds. However, the high computational complexity of the theoretical solution makes it unsuitable for real-time use, which leads to the use of AI-enabled methods. These include feedforward deep neural networks and deep unfolding techniques, trained on data generated from the theoretical solution. Performance evaluations reveal that these ML-based schemes closely match the performance of model-based solutions in terms of data rate and EMF exposure while enabling practical real-time implementations. Both downlink and uplink performance are again examined using SAR and IPD as primary metrics. Notably, while the uplink is more sensitive to EMF due to proximity of the transmitting device to the user, the cell-free architecture's ability to distribute load among nearby APs mitigates this issue.

Below, we summarize how the work we performed contributes to the objectives of WP4 of the CENTRIC project, and of Task 4.2 in particular, as extracted from the formal task description.

Table 1.1: Work adherence to Task 4.2 objectives

Task Objective	**Document Chapter**	**Contribution**
To optimize 6G KPIs and KVIs through ML models for RRM in novel user-centric wireless topologies such as cell-free networks	Chapter 2	(i) Reliable quantification and management of prediction uncertainty in system monitoring and open/closed-loop control decisions through the Probabilistic Time Series-Conformal Risk Prediction method that integrates probabilistic models with Conformal Prediction. (ii)) RL-based schedulers for efficient multidimensional (time, frequency, space) radio resource allocation. Exploiting expert policy guidance, entropy learning, and experience replay methods, while adhering to 3GPP standards training, and scheduler performance are substantially improved.
To develop caching techniques that minimize the energy consumption of ML models during training and inference at the wireless edge	Chapter 3	Innovative, scalable methods for minimizing energy consumption of ML models training and execution at the edge, leveraging (i) intelligent caching and sharing of ML models (or sections thereof), (ii) selective data usage for training, and (iii) communication-efficient protocols for distributed learning offloading of inference tasks at the edge. Achieved gains in comparison to SotA methods such as SEM-O-RAN, FedAvg, and FedProx: memory usage reduced by over 80%, computing time dropped by 77%, and task offloading admission rates increased by more than 25%
To analyse cell-free massive MIMO deployments to maximize network energy efficiency	Chapter 4	(i) A model-based approach maximizing the minimum user data under energy and power constraints and outperforming conventional multi-cell MIMO systems in user fairness. (ii) Low-complexity, AI-enabled methods trained on data generated from the theoretical solution, closely matching the data rate and energy/power performance of model-based solutions while enabling real-time implementations.
To develop AI-based techniques for EMF reduction in cell-free networks	Chapter 4	Human-centric network management paradigm and EMF-compliant resource allocation in cell-free networks through low-complexity heuristics and data-driven solutions. The strategies minimize energy consumption while meeting throughput and EMF constraints, with EMF accounted for through SAR (Specific Absorption Rate) and IPD (Incident Power Density) as performance metrics (up to 80% energy savings w.r.t. conventional approaches)

2. Multi-objective RRM optimization for performance-energy trade-offs

2.1. Practical Deep Schedulers for Radio Resource Allocation

ML methods are increasingly proposed to optimize wireless network functions, such as radio packet scheduling. However, a feasible 3GPP-compliant scheduler capable of delivering fair throughput across users, while keeping a low computational complexity for 5G and beyond has yet to be accomplished. To address this, we enhance State-of-the-Art (SoTA) deep Reinforcement Learning (RL) algorithm and adapt it to train our deep scheduler. On top of developing the deep scheduler framework, we introduce novel training techniques for Proximal Policy Optimization (PPO) tailored for radio resource allocation, which outperformed other tested variants. Besides showing the framework of an on-policy method, we present an off-policy method of training a deep scheduler, which demonstrates outstanding performance gains over the baseline and on-policy approaches. These improvements were achieved while maintaining minimal actor network complexity, making them suitable for real-time computing environments. Our proposed deep scheduler demonstrates strong generalization across different bandwidths, numbers of MU-MIMO layers, and traffic models. Ultimately, we show that our pre-trained deep schedulers outperform their heuristic rivals in realistic and standard-compliant 5G system-level simulations.

2.1.1. Motivation and State of the Art

Wireless systems have evolved, capable of serving vast amounts of data to many users over large areas. This progress stems from splitting wireless channels into resources across multiple dimensions: code, frequency, time, antennas, beams, and spatial layers. The radio packet scheduler dynamically allocates these resources to users, optimizing network performance, fairness, and spectral efficiency.

In 4G, 5G, and, likely, 6G, radio packet schedulers allocate the time, frequency, and spatial radio resources to the users that need them. We refer to these functions as Time Domain Scheduling (TDS), Frequency Domain Scheduling (FDS), and Spatial Domain Scheduling (SDS). They are NP-hard combinatorial problems, often solved with heuristics that lack optimality guarantees and scale poorly. As we transition into 6G, the number of radio resources at our disposal will continue to grow (e.g. larger Multiple Input Multiple Output (MIMO) arrays, larger bandwidths, diverse user devices, etc.). This increasing complexity strains traditional scheduling methods, creating a compelling need for new approaches. Machine Learning (ML) offers a promising path forward for several reasons:

- **Scalability**: ML models, once trained, often make decisions faster than complex heuristics.
- **Flexibility**: ML models naturally consider multi-modal factors, such as Channel State Information (CSI), UE priorities, Quality of Service (QoS) profiles, etc.

- **Adaptability**: ML models learn to adapt to changing channel conditions, traffic patterns, and user demands, a critical capability as networks become more dynamic.

The ability of packet schedulers to react quickly to load changes while considering all other inputs is hard to balance and build into heuristic algorithms. For instance, some radio schedulers may select the best beam based on beam-specific CSI reports. Grounding this decision also on the per-beam traffic load (even if the beam has worse RF conditions) may sometimes yield net capacity gains [4]. Such load changes can happen very fast (especially in the higher FR2 frequencies at 24.25 GHz to 52.6 GHz) when UEs move across beams very quickly. Detecting such rapid changes requires additional sub-routines built into heuristic schedulers, which increase their computational complexity and make debugging even harder. Instead, ML-based deep schedulers trained on real scenarios excel at detecting such hidden patterns in the data, often before the event occurs. While beam selection is a crucial aspect of scheduling in Grid of Beams (GoB) systems, our current proposal addresses the broader challenge of resource allocation in a dynamic Zero Forcing (ZF)-based beamforming context, particularly well-suited for Frequency Range 1 (FR1) (below 6 GHz) and the emerging 6G mid-bands (7-15 GHz).

Building on the established practice of splitting radio scheduling into TDS and FDS as described in [5], our reference scheduler architecture adds an SDS step to support MU-MIMO. The TDS step shortlists candidate UEs for scheduling, and the FDS step allocates frequency resources (optionally non-contiguous) to these candidates. With MU-MIMO, the SDS step pairs candidate UEs on additional Downlink (DL) MU-MIMO user layers for each Resource Block Group (RBG). We use two iterative heuristic SDS methods similar to those in [6], modifying them to first allocate Single-User MIMO (SU-MIMO) resources with FDS and then add spatial user layers until no suitable candidates remain or no additional MU-MIMO gain is expected with Regularized Zero Forcing (RZF)-based beamforming. We describe both SDS variants next:

(**Baseline SDS**) The baseline SDS algorithm operates on a pre-sorted list of UE candidates using the Proportional Fair (PF) metric from the TDS phase. It iterates through candidates in PF-sorted order, selecting the first UE that increases the total throughput of the RBG when co-scheduled with UEs already assigned to previous MU-MIMO user layers. This approach can be computationally demanding for real-time environments, as it may need to iterate through all candidates to find one that improves sum throughput, posing a challenge for practical implementation.

(**PF Greedy SDS**) To explore further gains, we implemented a modified SDS scheduler, *PF Greedy* SDS. For each RBG and MU-MIMO layer, this scheduler exhaustively searches for the candidate UE that maximizes the PF sum metric and increases the MU-MIMO sum throughput estimate for each RBG and spatial layer. This approach increases computational complexity, requiring recalculating beamforming weights and throughput estimations for every scheduled UE and new candidate for each RBG and spatial layer. While an exhaustive search could theoretically find the optimal combination, it is computationally infeasible even within our system-level simulations, let alone in a real-time environment.

The field of ML-based radio resource allocation has a rich history, with numerous designs

proposed over the years [7]. However, the complexity of wireless scheduling makes optimal solutions computationally challenging, hindering the acquisition of reliable labeled data for supervised learning models. The time-sequential nature of scheduling decisions lends itself to hidden Markov chains, leading most deep schedulers to employ Reinforcement Learning (RL).

In [8], a DL Frequency Domain (FD) deep scheduler based on Double Deep Q-Network (DDQN) was proposed, trained to maximize a composite reward for bitrate and fairness via non-contiguous allocation of 5G-compliant RBGs. However, it ignored subband-specific Channel Quality Indicator (CQI), missing frequency selectivity opportunities. Its Q network had two Fully Connected (FC) hidden layers of size 128 with ReLU activations, performing comparably to a PF baseline. The report did not discuss inference time or computational complexity, which is crucial for 5G schedulers making decisions every slot with sub-ms latency.

The deep scheduler proposed in [9] addresses SDS in Uplink (UL) MU-MIMO settings using Soft Actor-Critic (SAC) and K-Nearest Neighbors (KNN), aiming to maximize Spectral Efficiency (SE) and fairness among users. It incorporates UE grouping based on channel correlations and is evaluated with simulated and real-world data. However, the proposed method deviates from 5G New Radio (5G NR) uplink allocation requirements, which mandate contiguous or almost contiguous Resource Block (RB) allocations with RBG granularity [10]. This discrepancy in uplink allocation strategy limits the direct applicability of the approach within standard-compliant 5G NR systems.

Other efforts to reduce the action space size of deep schedulers in MU-MIMO settings include Deep Q-Networks (DQNs) with Action Branching (AB) [11]. However, this method does not scale effectively to practical massive MIMO systems, as it requires a large action space even with just two MIMO layers, involving N_a actions and 3.6 million trainable parameters. Additionally, RL-based methods incorporating Monte Carlo Tree Search (MCTS) have also been explored [12].

More recently, a hybrid FD deep scheduler was proposed [13], where the UEs are selected heuristically according to a weighted priority metric. A Proximal Policy Optimization (PPO) agent is trained to learn the heuristic's priority weights to minimize delay and packet loss. The PPO agent used five hidden layers of size 64. By leveraging the priority-based heuristic, the priority weights can change slowly and the inference pass does not need to be executed in each slot. Such an approach overcomes most implementation challenges but sacrifices some dynamicity and may lead to suboptimal decisions due to the inherent limitations of the priority-based heuristic.

Recognizing the large combinatorial action space that radio schedulers confront, the authors in [14] have also opted for a policy-gradient method such as DDPG, which uses an Actor-Critic (AC) architecture. The aim is to train a ML model for aiding the scheduler to estimate an adequate allocation size N_{RBs} and Modulation and Coding Scheme (MCS) for each UE under virtualized Radio Access Network (vRAN) computational constraints. This design considers the UE-specific uplink Signal-to-noise Ratio (SNR) and the congestion of the cloud

platform's CPUs, and it rewards the model for maximizing successful data decoding, thus promoting high bitrate. The authors claim significant gains in spectral efficiency over a Round Robin (RR) baseline.

2.1.2. Proposed Solution

To develop a practical deep radio resource scheduler that meets real-time Base Transceiver Station (BTS) constraints while maintaining or exceeding legacy scheduler Key Performance Indicators (KPIs), we devised two ML-based scheduler architectures. The first, a 1LDS, optimizes execution time by providing scheduling decisions for all RBGs per MU-MIMO user layer in a single forward pass, as illustrated in Fig. 2.1. In contrast, the second 2LDS approach decides the user allocation of one single RBG per inference pass, requiring loops over both RBGs and MU-MIMO user layers. In both schemes, looping over the user layers is necessary.

Despite using larger input and final neural layers, the 1LDS design achieves a significant speedup due to fewer forward passes while managing to still keep ML model dimensions rather small. Concretely, 1LDS reduces the number of forward passes from $N_{\text{RBG}} \times |L|$ down to $|L|$, where $|L|$ denotes the maximum number of spatially co-scheduled users per RBG, and N_{RBG} is the total number of DL RBGs to be allocated. This reduction is key in cellular systems with large bandwidths. However, performing a single forward pass for all RBGs and MU-MIMO layers would require a model that is too large for practical massive MIMO implementation.

2.1.2.1. Action Space

For each RBG and MU-MIMO user layer, our deep schedulers must select a user. The action space is, therefore, $A = 1, ..., |U| + 1$, where $|U|$ is the number of candidate UEs presented by the Time Domain (TD) scheduler. The additional action $a = |U|+1$ allows for no allocation, useful when candidates do not need more resources or further co-scheduling would degrade performance. Masking can be used to invalidate actions, such as already scheduled UEs for RBG or empty candidate input blocks.

2.1.2.2. State Space

The deep scheduler inputs are grouped into so-called state vectors, whose features have been chosen for their sufficiency in scheduling decision-making and low computational demands. Since the features are collected per UE, they effectively construct what we call a *UE feature segment*, and the state vector is therefore built out of $|U|$ UE feature segments. We describe the features next, although for clarity, and omit the Transmission Time Interval (TTI) index as follows: A prime denotes the variable at the next state, e.g., x', and a double prime denotes the variable at the previous state, e.g., x''.

- **Normalized Past Averaged Throughput ($\hat{R}_u$):** The past averaged throughput for user u is calculated using exponential smoothing:

$$R_u = (1 - \epsilon)p_u + \epsilon R''_u$$

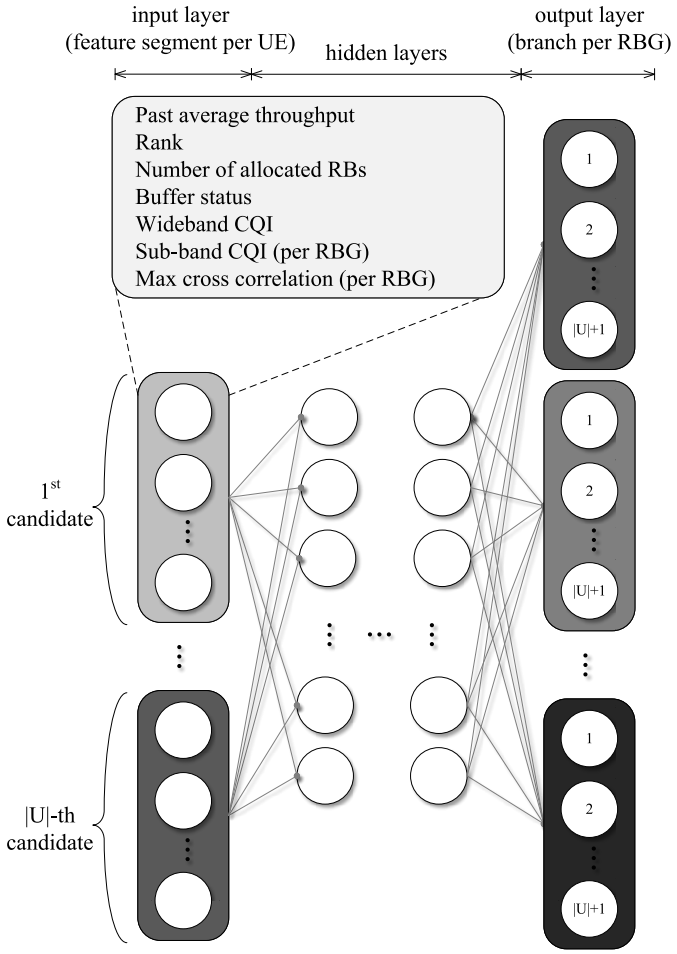

Figure 2.1: 1LDS baseline neural network architecture, with feature segment blanking for variable UE support. $|U|$ indicates maximum schedulable UEs per MU-MIMO user layer.

where p_u is the instantaneous throughput, R''_u is the previous averaged throughput, and ϵ is the forgetting factor. The normalized past averaged throughput is then:

$$\hat{R}_u = \frac{R_u}{R_{\max}}$$

with $R_{\max}$ being the maximum observed past averaged throughput.

- **Normalized Rank of UE ($\hat{h}_u$):** The rank of user u, denoted by h_u, is normalized as:

$$\hat{h}_u = \frac{h_u}{2}$$

The 2 in the denominator stems from the maximum UE rank (see Table 2.1).

- **Normalized Number of Already Allocated RBGs ($\hat{d}_u$):** d_u denotes the number of RBGs allocated to user u during the previous iteration of the scheduling loop across the MU-MIMO layers. This is then normalized as:

$$\hat{d}_u = \frac{d_u}{d_{\max}}$$

where $d_{\max}$ is the bandwidth-dependent total number of RBGs across the full bandwidth.

- **Normalized Downlink Buffer Status ($\hat{b}_u$):** The downlink buffer status for user u, denoted by b_u, is normalized as:
$$\hat{b}_u = \frac{b_u}{b_{\max}}$$
where $b_{\max}$ is a predefined maximum possible buffer size.

- **Normalized Wideband CQI ($\hat{o}_u$):** Min-max scaling is used to scale the wideband CQI for user u to the $[0, 1]$ range as follows:
$$\hat{o}_u = \frac{o_u - o_{\min}}{o_{\max} - o_{\min}}$$
where o_u is the current wideband CQI, and $o_{\min}$ and $o_{\max}$ are the minimum and maximum possible CQI values, respectively.

Per-RBG User-Specific Input Features:

- **Normalized Sub-band CQI ($\hat{g}_{m,u}$):** The sub-band CQI for user u on RBG m, denoted by $g_{m,u}$, is normalized as:
$$\hat{g}_{m,u} = \frac{g_{m,u}}{\bar{g}_{m,u}}$$
where $\bar{g}_{m,u}$ is a predefined normalization scalar.

- **Max Cross-Correlation in User Pairing ($\rho_{m,u}$):** The maximum user pair cross-correlation between already scheduled users and the new candidate u on that RBG m is computed. This pre-calculated value is based on the precoder matrices $P_{m,u}$ and $P_{m,c}$ (for users u and c co-scheduled on RBG m) and is given by:
$$\rho_{m,u} = \max\{\kappa_{m,u,1}, \kappa_{m,u,2}, \dots, \kappa_{m,u,N_{m,l}}\}$$
where $\kappa_{m,u,c} = \max\{\sum_i |[P_{m,u}^H P_{m,c}]_{i,j}|$, for $j \in \{1, 2, \dots, h_c\}\}$, h_c is the rank of user c, $N_{m,l}$ is the number of co-scheduled users on RBG m at layer l, and $[\cdot]_{i,j}$ denotes the element at the i-th row and j-th column of a given matrix.

Having defined the input features, we can now specify the state vectors for the different architectural variants. For the fast 1LDS architecture (Fig. 2.1), the state vector includes five features ($\hat{R}_u$, $\hat{h}_u$, $\hat{d}_u$, $\hat{b}_u$, $\hat{o}_u$) and two RBG-specific features ($\hat{g}_{m,u}$, $\rho_{m,u}$) for each candidate UE. The size of this state vector is $|U| \times (5 + 2N_{\text{RBG}})$.

The 2LDS scheduler, designed for throughput performance, executes one forward pass per RBG and receives RBG-specific inputs. These inputs include the seven previously defined features plus an eighth feature for the mean precoder cross-correlation, as well as the number of co-scheduled users so far. The input vector size for the 2LDS scheduler is $|U| \times 8 + 1$.

Having defined these neural architectures and input feature vectors, there exist multiple ways to train them. In the following sections, we describe different training algorithms and their performance.

2.1.3. On-Policy Deep Scheduler training

While pre-trained Deep Reinforcement Learning (DRL) schedulers may be sufficient in most cellular scenarios, niche applications like military deployments require rapid adaptation to unpredictable radio environments. In these safety-critical situations, the stability and cautious exploration of on-policy RL methods are advantageous.

2.1.3.1. On-policy framework with expert guidance

On-policy RL algorithms like PPO are favored for their ease of implementation and stability. However, their reliance on the current policy for environment interaction can lead to slower convergence due to limited sample diversity. To address this, our proposed 1LDS-based framework integrates an *expert policy* to guide the PPO agent during training (similar to the guide policy described in [15]). This expert policy, a PF scheduler in our case, acts as a source of additional training data, and is illustrated in Fig. 2.2.

The training phase involves four key stages:

- **State and reward observation**: Following traditional reinforcement learning, the input state for the actor and critic networks is updated at each new TTI. Additionally, the reward from the previous TTI is collected and saved into an experience buffer for the PPO agent to update its policy.
- **Expert label generation**: Based on system requirements and computational resources, an expert policy generates labels to guide the PPO agent.
- **Policy update**: Once the experience buffer reaches its capacity, the PPO agent updates its policy using the collected data and expert labels. This is achieved through back-propagation and optimization of a pre-defined loss function.
- **Resource Allocation**: The deep scheduler returns a vector of UE candidate indices for each RBG. After that, the candidates vector is sampled and the resource allocator assigns the RBG to the chosen UE. Modulation and coding schemes are selected after all RBGs have been allocated.

This approach leverages the strengths of both on-policy RL and expert knowledge, leading to faster convergence and improved scheduling performance in a practical setting. The other fundamental components of training a 1LDS-PPO are detailed next.

2.1.3.2. Fundamental Components

Action space

The final layer of the 1LDS-PPO actor network outputs a vector $\mathbf{z} \in \mathbb{R}^{(|U|+1)N_{\text{RBG}}}$, which is then reshaped into $\hat{\mathbf{z}} \in \mathbb{R}^{N_{\text{RBG}} \times (|U|+1)}$. Let softmax$(X, d)$ be a softmax function to be applied to the d-th dimension of an input matrix X. The softmax function is applied along the second dimension (representing UE candidates) to obtain a probability distribution $\tilde{\mathbf{z}} \in \mathbb{R}^{N_{\text{RBG}} \times (|U|+1)}$ over the UEs for each RBG: $\tilde{\mathbf{z}} = \text{softmax}(\hat{\mathbf{z}}, 2)$, where $\sum_{u=1}^{|U|+1} \tilde{z}_{m,u} = 1$ for $m \in \{1, 2, ..., N_{\text{RBG}}\}$.

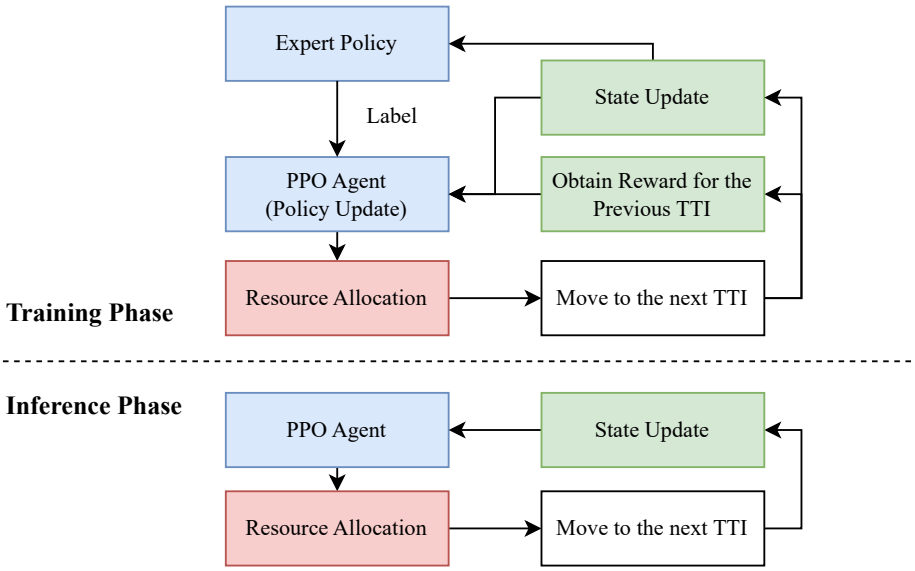

Figure 2.2: Overall Framework of 1LDS - PPO

From this distribution, the action vector $\mathbf{a} = [a_1, a_2, ..., a_{N_{\text{RBG}}}]^{\text{T}}$ is sampled, with each element $a_m \in \{1, 2, ..., |U| + 1\}$ representing the index of the chosen UE for RBG m. If the selected index does not correspond to a valid user, the RBG remains unassigned.
In summary, the action determination process involves:

- Obtaining the output vector $\mathbf{z}$ from the actor network.
- Reshaping $\mathbf{z}$ into $\hat{\mathbf{z}}$.
- Applying the softmax function to obtain the probability distribution $\tilde{\mathbf{z}}$.
- Sampling the action vector $\mathbf{a}$ from $\tilde{\mathbf{z}}$.

Reward function

The reward function is designed to balance two key objectives: maximizing the geometric mean of user throughput (for fairness) and maximizing the performance gain from MU-MIMO. This approach encourages the scheduler to learn a policy that efficiently allocates resources while exploiting the benefits of MU-MIMO. In a 1LDS architecture, the deep scheduler loops over the MU-MIMO user layers, and the reward for the l-th user layer, denoted by r_l, is defined as follows:

$$r_l = \frac{\left(\prod_{u=1}^{|U|} p_u\right)^{1/|U|}}{\bar{r}_l} + G \tag{2.1}$$

where:

- $G = \frac{\sum_{m=1}^{N_{\text{RBG}}} v_m}{N_{\text{RBG}}}$ is the MU-MIMO gain, calculated as the average number of RBGs where MU-MIMO gain is achieved.
- $\bar{r}_l$ is a normalization scalar to keep the reward within a suitable range.

The MU-MIMO gain for a given RBG m is determined as follows:

$$v_m = \begin{cases} 1 & \text{if MU-MIMO gain is obtained by } a_m \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

We consider MU-MIMO gain achieved if allocating the user a_m to RBG m increases the total PF metric across all users. This metric captures the trade-off between overall throughput and individual user fairness. By incorporating throughput fairness and MU-MIMO gain into the reward function, the scheduler is guided to learn a policy that efficiently allocates resources while leveraging the spatial multiplexing capabilities of MU-MIMO.

2.1.3.3. Training data augmentation

To enhance generalization and accelerate training, we shuffle the UE feature segments within the input state (see Fig. 2.1). This generates additional experience samples for the scheduler model to learn from while preserving the reward and swapping the UE candidate indices according to the permutation order. The number of permutations, N_Π, is a pre-set hyperparameter. Each new experience tuple saved in the buffer undergoes N_Π permutations, effectively multiplying the available training samples.

An experience replay buffer with ample memory is also employed to store input state-action pairs from the expert policy. This buffer provides the learning agent with diverse expert demonstrations to learn from, further improving its performance. Unlike traditional off-policy replay buffers, which store rewards, next states, and termination flags, this buffer focuses solely on current states and corresponding expert actions.

2.1.3.4. Loss Function with expert guidance

To accelerate convergence and benefit from expert knowledge, we introduce an additional loss term to the standard PPO loss function, encouraging the on-policy DRL agent to mimic the expert policy. The expert policy generates the labels for the scheduling decisions that the PPO agent can learn from. This term is based on the Jensen-Shannon Divergence (JSD), which measures similarity between probability distributions. JSD is symmetric, robust to zero probabilities (possible in radio scheduling), and bounded between 0 and 1, offering a normalized dissimilarity metric.

Specifically, we calculate the JSD loss, $\mathcal{L}_{\text{JSD}}$, between the actor network's output and the expert policy's label:

$$\begin{aligned} \mathcal{L}_{\text{JSD}} = \frac{1}{2} \sum_{a_u}^{\{1,2,\ldots,|U|+1\}} \pi_{\theta_{\text{PPO}}}(a_u|s) \log \frac{\pi_{\theta_{\text{PPO}}}(a_u|s)}{\pi_{\text{expert}}(a_u|s)} \\ + \frac{1}{2} \sum_{a_u}^{\{1,2,\ldots,|U|+1\}} \pi_{\text{expert}}(a_u|s) \log \frac{\pi_{\text{expert}}(a_u|s)}{\pi_{\theta_{\text{PPO}}}(a_u|s)} \end{aligned} \tag{2.3}$$

where:

- $\pi_{\theta_{\text{PPO}}}(a_u|s)$ is the probability of the PPO agent taking action a_u given state s.

- $\pi_{\theta_{\text{expert}}}(a_u|s)$ is the probability of an expert policy taking action a_u given state s.

To help the agent explore and converge to a policy better than the expert, the traditional PPO loss is employed and defined as:

$$\mathcal{L}_{\text{PPO}} = L_{\text{value}} - L_{\text{policy}} - \xi L_{\text{entropy}}, \tag{2.4}$$

$$L_{\text{value}} = \frac{1}{2}\mathbb{E}\{(V(s) - \hat{J})^2\}, \tag{2.5}$$

$$L_{\text{policy}} = \mathbb{E}\left\{\min\left(\varrho(\theta)\hat{A}, \text{clip}(\varrho(\theta), 1-\epsilon, 1+\epsilon)\hat{A}\right)\right\}, \tag{2.6}$$

$$L_{\text{entropy}} = H(\mathbf{z}|s; \pi_{\theta_{\text{PPO}}}) \tag{2.7}$$

where:

- ξ is a coefficient controlling the strength of entropy regularization, promoting exploration.
- $H(\cdot)$ is the statistical entropy of the current policy.
- $\hat{A}$ is the advantage value, estimated using a generalized advantage estimator [16].
- $\varrho(\cdot) = \frac{\pi_{\theta_{\text{PPO}}}}{\pi_{\theta'_{\text{PPO}}}}$ is the ratio of probabilities between the updated policy ($\pi_{\theta_{\text{PPO}}}$) and the old policy ($\pi_{\theta'_{\text{PPO}}}$). The overall policy is calculated as the product of individual action probabilities across all RBGs: $\pi_{\theta_{\text{PPO}}} = \prod_{i=1}^{N_{\text{RBG}}} \pi_{\theta_{\text{PPO}},i}$.
- ϵ is the clipping parameter to constrain policy updates and maintain stability.
- $V(\cdot)$ is the estimate of the state value function.
- $\hat{J}$ is the target return, computed from the cumulative discounted rewards.

During training, we employ an alternating optimization approach, iteratively minimizing $\mathcal{L}_{\text{PPO}}$ using a batch of experiences and minimizing $\mathcal{L}_{\text{JSD}}$ using state-action pairs from the experience replay buffer.

2.1.4. Off-Policy Deep scheduler training

Off-policy RL methods offer a distinct advantage over their on-policy counterparts by allowing agents to learn from experiences generated by a different policy. This enables efficient utilization of past scheduling decisions stored in a replay buffer, facilitating faster and more robust learning. In radio resource scheduling, off-policy methods are particularly appealing when exploration and stability are critical.

This section investigates several off-policy RL algorithms for radio resource allocation. We first consider DDQN [17] with soft target network updates [18] due to their proven effectiveness in discrete action spaces. In addition to DDQN, we consider SAC [19], which is proven to be a rather effective and sample efficient algorithm due to its entropy regularization. However, as the original SAC algorithm is designed for continuous action spaces, we adopt a modified version, Soft Actor-Critic Discrete (SACD) [20], instead. This variant

is better suited for our discrete action space, with its critic networks estimating Q-values for all actions simultaneously, allowing the actor to learn from the entire Q-value distribution for a given state. Furthermore, since nearby scheduling decisions are not necessarily correlated (especially with shuffled state feature segments and a correspondingly branched output layer, see Fig. 2.1), the Gaussian policy of the original SAC is less suitable than the individual action outputs of SACD, similar to traditional DQN-based algorithms.

Finally, we explore distributional RL, by extending SACD, with distributional critic networks, resulting in Distributional Soft Actor-Critic Discrete (DSACD). With both SAC variants, we learn a policy π_θ and two Q-functions $Q_{\phi 1}$ and $Q_{\phi 2}$, along with their soft-updated targets $\bar{Q}_{\bar{\phi}1}$ and $\bar{Q}_{\bar{\phi}2}$.

2.1.4.1. Distributional Soft Actor-Critic Discrete

A key problem in distributional RL [21] is that it increases the ML model size and computational complexity. This is not desired for models intended for real-time systems such as BTSs. To capture the benefits of distributional learning and State of the Art (SotA) maximum entropy RL for discrete action spaces, we modify the online and target critic networks to be distributional while keeping the actor (policy) network unchanged. This allows us to train a model that combines maximum entropy RL with distributional RL and yield an actor model (policy network) that is no more complex to execute than the original DQN [22].

To transform the critic networks into distributional ones, we apply two key modifications proposed in [23] to the SACD critic networks. First, we expand the critics' output layers by a factor of N, a hyperparameter that determines the number of quantiles used to approximate the distribution of Q-values. This results in an output layer size of $|A| \times N$, where $|A|$ is the size of the action space. This expansion allows the network to represent a distribution of Q-values for each action, providing a richer representation of the value function. Second, we replace the Mean Square Error (MSE) loss used in the original SACD critics with the quantile Huber loss. This loss is more robust to outliers than MSE, and is given by:

$$L(x) = \begin{cases} \frac{1}{2}x^2 & \text{for } |x| < k \\ k(|x| - \frac{1}{2}k) & \text{otherwise} \end{cases}, \tag{2.8}$$

where k is a hyperparameter we set to $k = 1$.

The quantile Huber loss, an asymmetric variant of the Huber loss is given by:

$$\rho_\tau^k(x) = |\tau - \delta_{\{x<0\}}|L(x), \tag{2.9}$$

where δ denotes a Dirac delta function and τ represents the quantile level, with N quantiles evenly spaced between 0 and 1 as $\tau_n = \frac{n}{N}$, for $n = 1, ..., N$. This loss function creates N quantiles per action in the output layers of the critic networks, allowing them to represent a distribution of Q-values instead of a single point estimate. This distributional representation captures uncertainty in the value estimation and can lead to improved performance in RL

tasks.

For our DSACD algorithm, the error x_t for each quantile $n = 1, ...N$ is defined as:

$$x_{\phi,n} = q_{\phi,n}(s, a) - y_{\bar{\phi}}(r(s, a), s'), \tag{2.10}$$

where $q_{\phi,n}(s, a)$ is the n-th quantile output of the critic network for state s and action a, and the target value is calculated as:

$$\begin{aligned} y_{\bar{\phi}}(r(s, a), s') = r(s, a) + \\ \gamma\pi_\theta(s', a')\big(\bar{Q}_{\bar{\phi}}(s', a') - \alpha \log(\pi_\theta(s', a'))\big), \\ a' \sim \pi_\theta(\cdot|s'). \end{aligned} \tag{2.11}$$

Here, γ is the discount factor, $r(s, a)$ is the reward, α is the learned entropy regularization coefficient, and $\pi_\theta(s', a')$ is the probability of taking action a' in state s' according to the actor network. The term $\bar{Q}_{\bar{\phi}}(s', a')$ represents the expected value from the target critic network for state s' and action a'. Note that action a' for the next state s' is drawn from the probability distribution given by the actor-network. The target value $\bar{Q}_{\bar{\phi}}(s', a')$ is obtained by averaging over all N quantiles in the next state s' for the drawn action a', as follows:

$$Q_\phi(s, a) := \sum_{n=1}^{N} \frac{1}{N} q_{\phi,n}(s, a). \tag{2.12}$$

This distributional approach captures the uncertainty in future rewards and allows for a more robust and nuanced learning process.

The actor-network is trained following the principles of SACD [20]. During training, a softmax function is applied to the output layer (separately for each output branch for 1LDS approach) to convert the discrete outputs into a probability distribution over actions. However, to simplify inference and reduce complexity, the softmax operation can be omitted after training, allowing greedy action selection using argmax. The main difference from the original SACD [20] lies in handling the distributional Q-values from the critic networks. Since the critics output a distribution of Q-values for each action, we compute the mean Q-value across all N quantiles for each action. This transforms the quantile outputs into a format comparable to the actor network's probability distribution. To mitigate Q-value overestimation, SAC utilizes two critic networks. The policy objective function, which guides the actor's learning, is defined as:

$$J_\pi(\theta) = \mathbb{E}_{s\sim D}[\pi_\theta(s) \odot [\alpha \log(\pi_\theta(s)) - \min_{j=1,2} Q_{\phi j}(s)]], \tag{2.13}$$

where D is the replay memory and the array of mean Q-values $Q_{\phi j}(s)$ from critic network j are obtained as described previously. It should be noted that the min operation selects the smaller Q-value from the two critics for each action, promoting conservative estimates. This objective function guides the gradient descent process across all actions in the discrete action space. While the Q-networks are updated based on the actions taken, the policy is trained using all action probabilities and their corresponding Q-values, ensuring comprehensive learning from the state-action value distribution.

2.1.4.2. Learning SACD Entropy with Action Masking

To ensure the selection of valid scheduling candidates, we employ action masking to exclude candidates already scheduled for specific RBGs. Additionally, the number of valid actions may vary across TTIs due to the dynamic nature of TD scheduling.

In SACD, the entropy regularization parameter α is learned, and the entropy target depends on the size of the action space. Instead of using a static target value $\bar{H}$ as in [20], we adopt a state-specific target in the entropy objective:

$$J(\alpha) = \mathbb{E}_{s\sim D}[\pi_\theta(s)[\alpha(\log(\pi_\theta(s)) + \bar{H}(s))]], \tag{2.14}$$

where the state-specific target is derived from the number of valid actions $|A(s)|$ in the current state:

$$\bar{H}(s) = -\beta \cdot \log\left(\frac{1}{|A(s)|}\right). \tag{2.15}$$

Here, β is a hyperparameter controlling the target entropy. While the default value of $\beta = 0.98$ from [20] performs well, we further optimized our deep scheduler by tuning α based on the number of valid actions and searching for a more optimal β (see values in Table 2.2).
Masks are stored in the replay memory alongside states, enabling us to determine the number of valid actions during training. Since the actor produces probabilities for all actions, including invalid ones, the probability distribution is adjusted by setting invalid action probabilities to zero and renormalizing the remaining probabilities to sum to one. Finally, the entropy is updated using gradient descent, averaging over all actions:

$$\alpha \leftarrow \alpha - \frac{1}{|A(s)|}\sum_{a}^{A}\pi_\theta(s,a)[\log(\pi_\theta(s,a)) - \bar{H}(s)], \tag{2.16}$$

where the logarithm is applied element-wise to the probability distribution of the discrete action space.

2.1.4.3. Prioritized Experience Replay

Replay memory prioritization, based on the method proposed in [24], originally generated bias toward transitions with higher temporal difference (TD) errors in DQNs. However, in our DSACD algorithm, we have two Q-networks. Therefore, we adapt the prioritization strategy to sample transitions into mini-batches from the replay memory with a probability p based on the average of the error values $x_{\phi j,n}$:

$$p \propto \left[\frac{1}{2}\sum_{j=1}^{2}\frac{1}{N}\sum_{n=1}^{N}|x_{\phi j,n}|\right]^{\omega}, \tag{2.17}$$

where ω is a hyperparameter that determines the shape of the replay memory distribution. New samples are always assigned the maximum priority to ensure they are sampled more frequently, thereby generating a favorable bias toward new experiences. It should be noted that we use the same method for SACD. Naturally, then averaging over the quantiles is not needed.

2.1.4.4. Rewarding Off-Policy Methods

Our training objective is to develop a deep scheduler that allocates scheduling candidates in both the frequency (RBG) and spatial (MU-MIMO) domains in a proportionally fair manner. One approach to achieve this is to maximize the long-term geometric mean of user throughput. However, due to the necessity of making multiple RBG decisions per MU-MIMO user layer at each TTI, we found it more effective to maximize the instantaneous increase in the PF metric resulting from each scheduling decision. Therefore, the reward for the u-th scheduling candidate on the m-th RBG and l-th MU-MIMO user layer is defined as:

$$r_{u,m,l} = \begin{cases} \frac{T_{u,m,l}}{R_u} - \frac{T_{u,m,l-1}}{R_u} & \text{for } l > 1 \\ \frac{T_{u,m,l}}{R_u} & \text{otherwise} \end{cases}, \tag{2.18}$$

where $T_{u,m,l}$ is the achievable data rate for that RBG and user layer, and R_u is the past average throughput of user u.
To prevent the model from learning to artificially minimize past average throughput to inflate rewards, we normalize the reward at each TTI:

$$r_{u,m,l} \leftarrow \begin{cases} \max\left\{\frac{r_{u,m,l}}{\max\limits_u r_{u,m,l}}, -1\right\} & \text{for } \max\limits_u r_{u,m,l} > 0 \\ 1 & \text{for } \max\limits_u r_{u,m,l} < 0 \text{ and } a = |A| \\ -1 & \text{otherwise} \end{cases}, \tag{2.19}$$

This normalization clips the maximum reward to 1 and the minimum to -1, ensuring a stable and balanced learning process.

2.1.5. Performance Evaluation

The performance of the trained models is evaluated in a realistic proprietary Fifth Generation (5G) system-level simulator, which is calibrated against simulation results of other companies in 3rd Generation Partnership Project (3GPP). Both our on-policy and off-policy training methods employed a centralized approach, iterating over TTIs to collect experiences and update a single centralized model. To expedite training, models are initially trained on a reduced system bandwidth with fewer MU-MIMO layers, as detailed in Table 2.1. By evaluating these models under different parameter settings, we can better assess their generalization capabilities. The key parameters of the simulation environment are summarized in Table 2.1. Our focus is on downlink non-contiguous (Type 1) allocation, which enables unrestricted frequency-selective scheduling with RBG granularity. We assume the availability of sub-band CQI and Precoder Matrix Indicator (PMI) reports and limit the maximum Rank Indicator (RI) of users reporting it to 2. We reduce the system bandwidth to expedite the training process while still allowing the training of 2LDS models for 18 RBGs. To promote generalization, we trained the models under mixed traffic conditions, reduced bandwidth, and less MU-MIMO layers. Thus, ensuring their adaptability to diverse real-world scenarios. Carefully tuned hyperparameters for both methods are shown in 2.2. Finally, we show results for two types of traffic models: Full Buffer (FB) and FTP3 bursty.

Table 2.1: System parameters

Evaluation Parameters	
Parameter	**Value**
Deployment	3GPP Dense Urban Macro (see Table 6.1.2-1 in [25])
Number of cells	21
Inter-site distance	200 m
gNB height	25 m
gNB Tx power	44 dBm
Number of UEs	210 uniformly distributed
Random drops	10
Bandwidth	100 MHz (273 RBs, $N_{\text{RBG}} = 18$)
Carrier frequency	4 GHz
Subcarrier spacing	30 kHz
MIMO scheme	MU-MIMO with regularized Zero Forcing
Modulation	QPSK to 256QAM
Link adaptation	10 % Block Error Rate (BLER) target
gNB antenna panel	12x8 and 2 polarizations
Max UE layers	$\lvert L\rvert = 8$
Max UE rank	2
Max MIMO layers	16 (i.e. $\lvert L\rvert \times$ max rank)
CSI	RI and sub-band CQI & PMI
Max scheduled UEs	$\lvert U\rvert = 10$
TD scheduler	PF
UE speed	3 km/h
UE height	1.5 m
UE receiver	Maximal-ratio combining (MRC)
UE antenna	2 dual-polarized Rx antennas
Traffic model	100% Full Buffer, or 100% FTP Model 3 [26]
	(FTP3 with 0.5 MB packets, 20 packets/s)
Parameter Modifications for Training	
Parameter	**Value**
Number of UEs	420 uniformly distributed
Random drops	1
Traffic model	50% of UEs Full Buffer and
	50% of UEs FTP Model 3 [26]
	(FTP3 with 1.5 kB packets, 500 packets/s)
Bandwidth	6.6 MHz (18 RBs, 18 RBGs)
Max UE layers	$\lvert L\rvert = 4$

Table 2.2: Training hyperparameters

Parameter	**Off-Policy**			**PPO**
Discount factor	0			0.7
Epochs per sample	1			1
Mini batch size	32			128
Actor learning rate	0.0001			0.0002
Critic learning rate	0.0001			0.0003
Hidden layers	2			2
Hidden layer size	32			64
Activation function	ReLU			Tanh
Optimizer	Adam [27]			Adam [27]
Replay memory size	$N_{\text{gNB}} \times 1000$			N.A.
Prioritized replay ω	0.5			N.A.
Target smoothing coef.	0.001			N.A.
	DDQN	**SACD**	**DSACD**	**PPO**
Critic quantiles N	N.A.	N.A.	16	N.A.
	2LDS		**1LDS**	
Input size	$\|U\| \times 8 + 1$		$\|U\| \times (5 + 2N_{\text{RBG}})$	
Output size	$\|U\| + 1$		$N_{\text{RBG}} \times (\|U\| + 1)$	
Inferences per TTI	$\|L\| \times N_{\text{RBG}}$		$\|L\|$	
SAC target entropy β	0.4		0.999	

2.1.5.1. Training convergence and hyperparameters

As shown in Table 2.3, where results are given for the modified training parameters in Table 2.1, all off-policy methods performed reasonably well. Notably, DDQN exceeded expectations due to the entropy introduced by shuffling state feature segments. DSACD was the only method to surpass the baseline scheduler in the 1LDS architecture, making it the preferred choice for this configuration. For 2LDS architectures, DSACD provided the highest median user throughput, while SACD achieved the highest geometric mean throughput. The geometric mean throughput best reflects proportional fairness, which aligns with our primary goal.

For final evaluations on the full 100 MHz bandwidth with a higher number of MU-MIMO layers, we selected the pre-trained SACD model for the 2LDS architecture and the DSACD model for the 1LDS architecture as our representative off-policy methods. Due to its on-policy nature, the PPO scheduler requires more TTIs to converge. Therefore, we first compared off-policy training results and selected the best off-policy deep schedulers for the final evaluation.

Fig. 2.3 illustrates smoothed learning curves for the selected off-policy methods. In particular, the optimal hyperparameter of entropy β differed significantly between the models 1LDS and 2LDS. Although 2LDS models performed best with a low β of 0.4, emphasizing the exploitation of learned strategies, 1LDS models required a high β of 0.999, potentially due

Table 2.3: Off-policy throughput gains over the baseline with the modified mixed-traffic training parameters of 2.1

1LDS	**Median**	**Geomean**
DDQN	-3 %	-0.9 %
SACD	-0.9 %	0 %
DSACD	0.9 %	0.7 %
2LDS	**Median**	**Geomean**
DDQN	8.7 %	6.3 %
SACD	8.1 %	6.9 %
DSACD	10.1 %	6.4 %

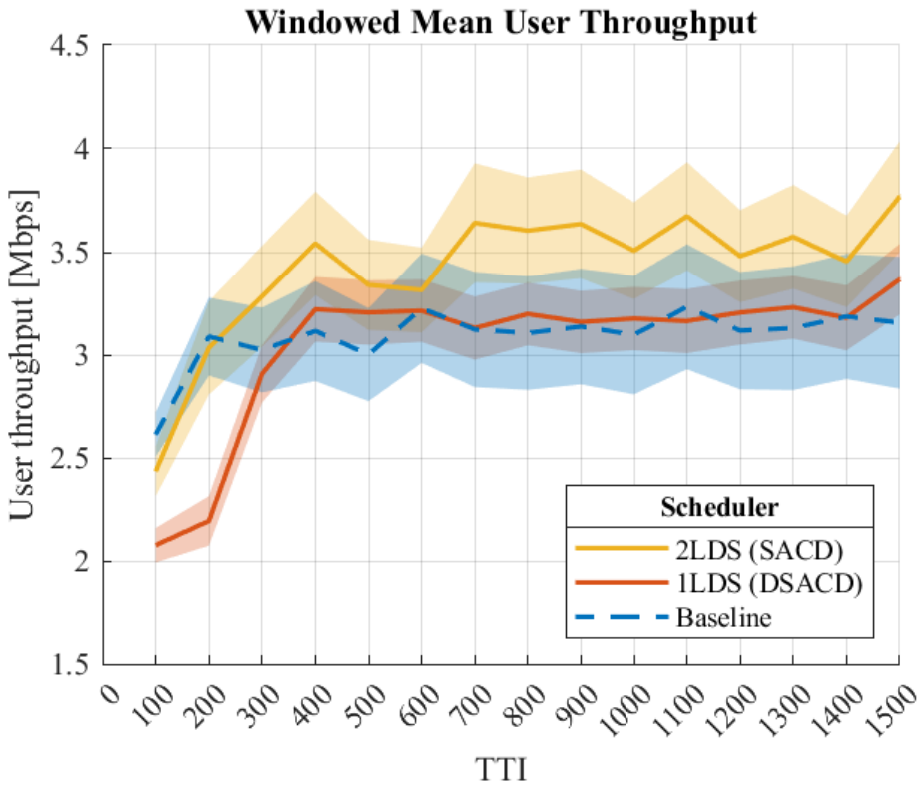

Figure 2.3: Windowed mean user throughputs during the training of best performing 1LDS and 2LDS options. Due to higher entropy required by training 2LDS than 1LDS lower throughputs can be observed during the training.

to their limited capacity to benefit from increased exploration. Additionally, the 1LDS models, with their larger multi-branched output layers, needed more training samples to match the baseline scheduler's performance. This convergence occurred around the 400th TTI, at which point roughly 450,000 training *sars′* tuples had been collected and used for training. Training samples are collected after 100th TTI from each TTI, MU-MIMO user layer, RBG, and gNodeB (gNB).

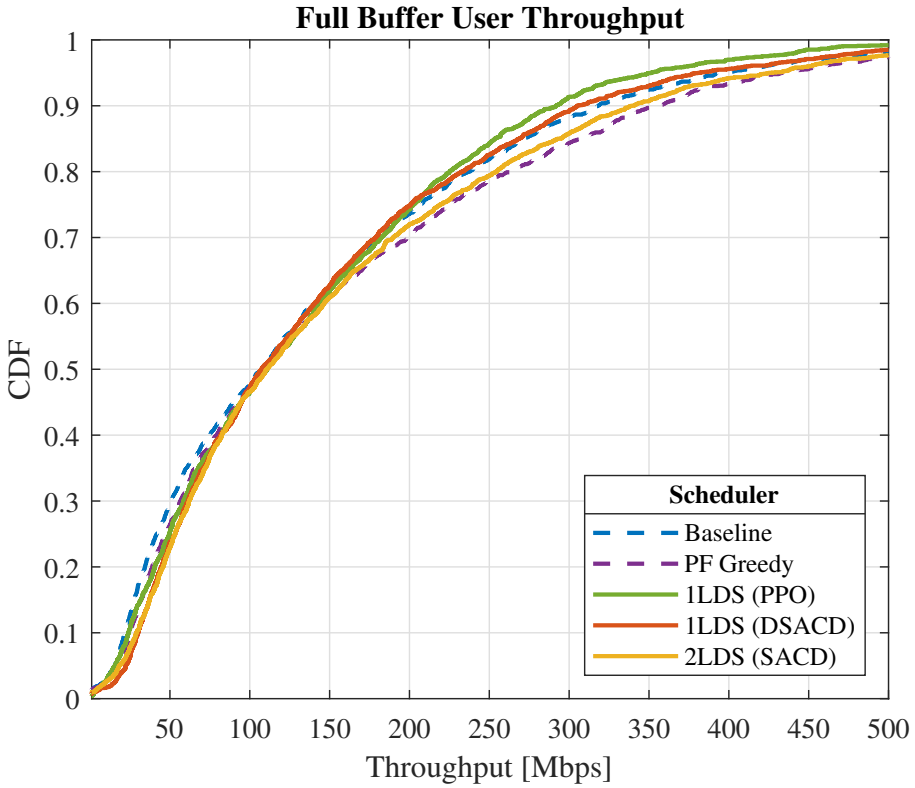

Figure 2.4: User throughput distribution with Full Buffer traffic. Deep schedulers can provide better throughput throughout the user distribution.

2.1.5.2. Assessment of numerical performance

As previously mentioned, trained models are tested in different simulation scenarios than those used for training. Results are collected from 10 different random simulation realizations. To ensure comparable geometric mean values, zero-valued UE throughputs are replaced with ones when calculating geometric means, as some random UE drops result in a few UEs being unable to receive data.

Fig. 2.4 shows that all deep scheduler options perform well with full buffer traffic. Notably, the 2LDS configuration provides better throughput for the majority of UEs. This is explained in Fig. 2.5, where deep schedulers efficiently utilize spatial degrees of freedom by selecting UEs for each RBG and MU-MIMO user layer that can be co-scheduled with others while maintaining fairness. In contrast, the 1LDS (PPO) approach may sometimes over-schedule a few UEs for a single RBG, leading to minor degradation in user throughputs compared to SAC-based options.

Given the discontinuous nature of FTP traffic, we use UPT to evaluate network performance with bursty traffic models. UPT, defined as the throughput measured only when a UE has DL data to receive, is illustrated in Fig. 2.6. The figure shows clear improvements in UPT distribution fairness achieved by the deep schedulers. A similar conclusion can be drawn from the classic user throughput CDF shown in Fig. 2.7.

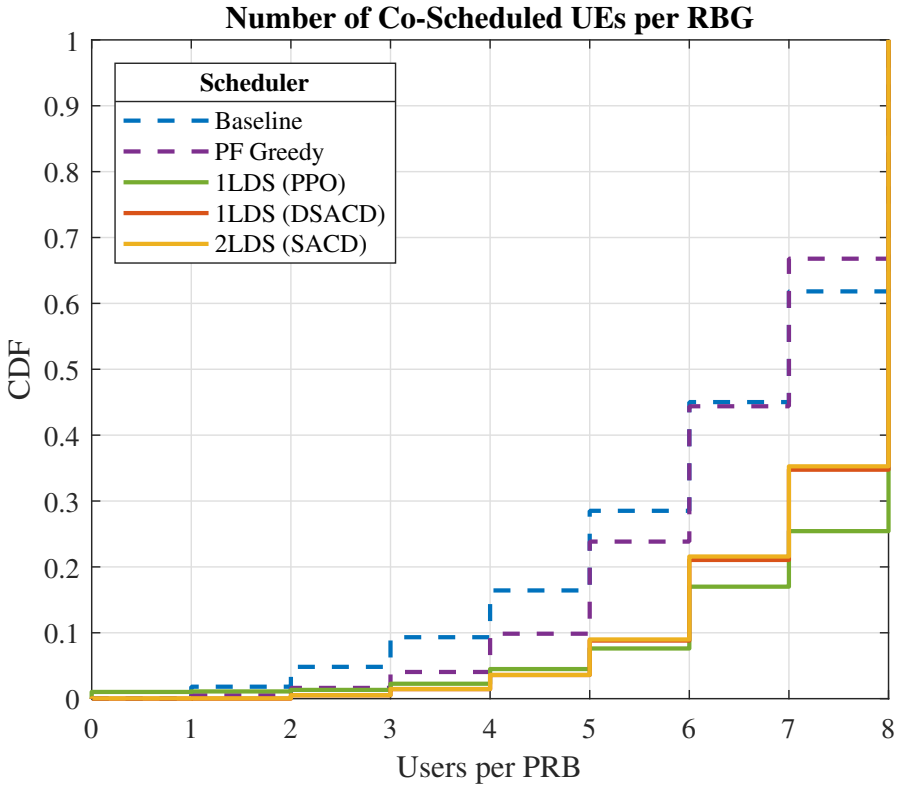

Figure 2.5: Comparison of co-scheduling efficiency across different schedulers.

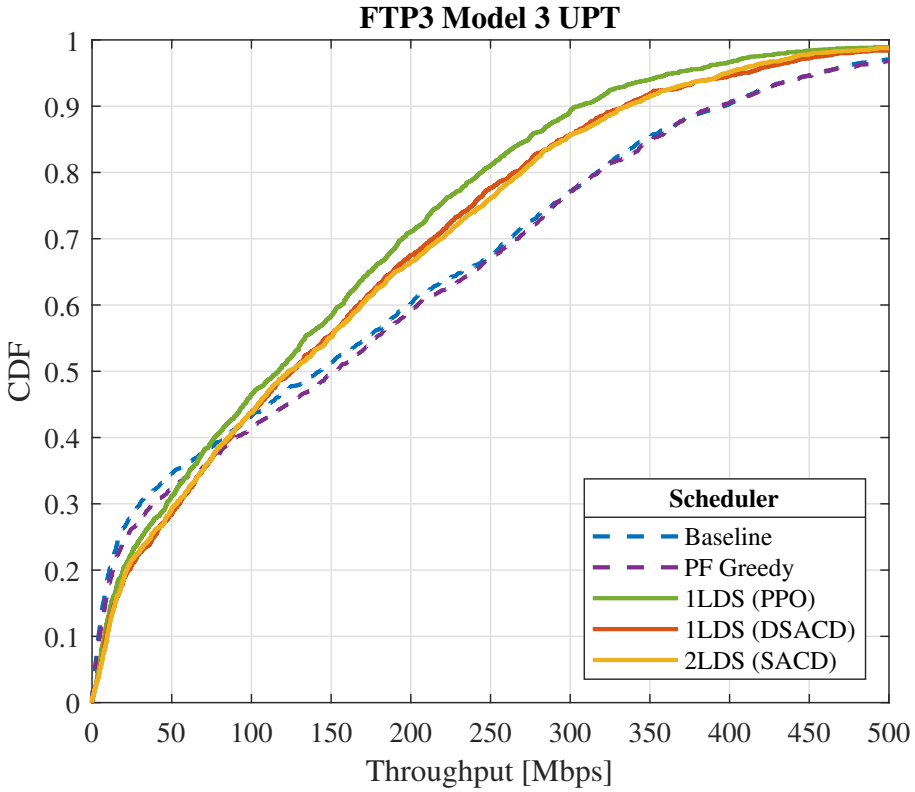

Figure 2.6: FTP Model 3 UPT. With deep schedulers UPT distribution becomes fairer.

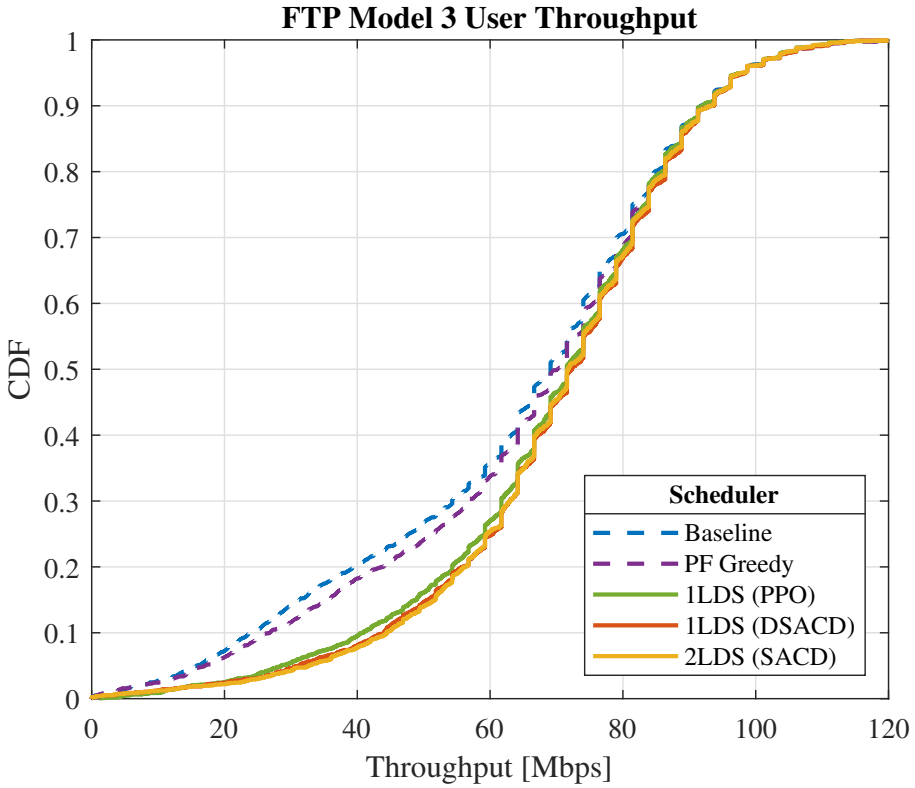

Figure 2.7: User throughput distribution with FTP Model 3 traffic. Most improvements are obtained among cell edge users.

Table 2.4 summarizes the numerical throughput results for separately simulated full buffer and bursty FTP model 3 traffic. All proposed deep schedulers achieve substantial gains, particularly in cell edge throughput and geometric mean throughput. Notably, the least complex variant, 1LDS DSACD, performs almost as well as the more complex 2LDS variant.

Table 2.4: Summary of cell edge (5th %-ile), median, and geometric mean gains over the baseline with full buffer and FTP Model 3 traffic

Full Buffer	**5th %-ile**	**Median**	**Geomean**
PF Greedy	11.3 %	1.3 %	9.7 %
1LDS (PPO)	1.6 %	3.5 %	8.1 %
1LDS (DSACD)	46.8 %	0.2 %	14.2 %
2LDS (SACD)	20.7 %	3.1 %	16.2 %
FTP Model 3	**5th %-ile**	**Median**	**Geomean**
PF Greedy	8.0 %	1.5 %	3.4 %
1LDS (PPO)	80.8 %	3.6 %	17.8 %
1LDS (DSACD)	97.9 %	4.6 %	18.5 %
2LDS (SACD)	106.8 %	3.8 %	18.8 %

2.1.5.3. Computational complexity analysis

Table 2.2 outlines the parameters affecting the computational complexity of both Off- and On-Policy methods. Off-Policy trained models have the fastest single forward pass due to their smaller hidden layer sizes. However, the 2LDS model, despite being the fastest for individual forward passes, requires a separate pass for each RBG and MU-MIMO user layer, resulting in higher overall inference latency.

All input parameters are designed to avoid unnecessary per-TTI calculations, as they can be pre-calculated based on CSI reports. Additionally, UE SU-MIMO precoder cross-correlations can be pre-calculated, with only the maximum value needed during the scheduling of additional MU-MIMO layers. Consequently, execution times are primarily dependent on neural network forward passes. Indicative forward pass times, measured using single-precision floating-point format neural networks on an Intel Xeon Gold 6330 CPU, are provided in Table 2.5. These measurements, obtained from a CPU-optimized in-house C++ implementation, suggest that deep schedulers with 2LDS architectures may struggle to meet 5G inference latency requirements and TTI limits. In contrast, the 1LDS design offers a good balance between throughput performance and computational complexity, with scalability to massive MIMO. We do not provide CPU times for the baselines due to their dependency on specific implementations and libraries, making direct comparisons unsuitable. Additionally, company-internal testing on commercial BTS hardware has been conducted with favorable results, but these findings are confidential and cannot be disclosed.

Table 2.5: CPU times of deep scheduler forward passes with 18 RBGs and 8 MU-MIMO UE layers by using ReLU activation

Model architecture	**2LDS**	**1LDS**	
Neurons per Hidden Layer	**32**	**64**	**32**
CPU time per forward pass	2 us	5.6 us	3.4 us
Sum CPU time per TTI	288 us	44.8 us	27.2 us

The execution times shown do not include state feature vector filling or other simulator or product-specific scheduling procedures. However, due to the pre-calculated inputs, such preprocessing consumes an insignificant amount of CPU time. Forward pass times can be further reduced when models are used with CPUs that support native half-precision floating-point operations.

Unlike heuristic baseline schedulers that require extensive throughput estimations with updated RZF precoders and re-selected MCSs, deep schedulers calculate RZF precoders and select MCS only once, after obtaining scheduling decisions from the actor networks. This efficiency makes ML models particularly appealing for scheduling tasks.

2.1.6. Conclusions

In this report, we reviewed the applicability of machine learning for radio resource allocation. We demonstrated that RL can be effectively used to train deep schedulers that outperform exhaustive heuristic alternatives We did not rely solely on SotA RL algorithms; instead, we

modified them to better suit our deep scheduler training tasks. For the PPO-based scheduler, we incorporated expert knowledge into the training procedure, guiding and updating the policy for enhanced performance. Additionally, we derived a new version of the SAC algorithm, termed DSACD. DSACD was further optimized with entropy learning for variable action spaces and prioritized experience replay specific to DSACD. By employing a multi-branch output layer to schedule each MU-MIMO layer separately, DSACD provided the best tradeoff between PF throughput and computational complexity for frequency-selective massive MIMO scheduling.

Our analysis also revealed that while deep schedulers based on 2LDS architectures may struggle to meet 5G TTI time limits, 1LDS-based designs offer a good balance between throughput performance and computational efficiency. This makes the 1LDS architectures a viable option for real-time applications, ensuring scalability to massive MIMO without compromising on performance.

2.2. Reliable Prediction for Wireless Networks Control

In real-world applications, predictions play a crucial role in monitoring and controlling cyber-physical systems, demanding strict reliability and safety standards. Managing prediction uncertainty is particularly challenging in complex, dynamic environments with branching trajectories. This chapter introduces Probabilistic Time Series-Conformal Risk Prediction (PTS-CRC), a post-hoc calibration technique that enhances any probabilistic forecaster by providing accurate error bounds. Unlike existing methods, PTS-CRC constructs predictive sets using an ensemble of prototype trajectories, effectively capturing branching uncertainties. It also extends beyond conventional reliability metrics like coverage, enabling broader reliability guarantees. Integrated into a new Model Predictive Control (MPC) framework, PTS-CRC addresses both open-loop and closed-loop control problems under general quality and safety constraints. We validate its effectiveness through wireless networking experiments, where it consistently delivers more informative predictions and safer control policies with higher returns across various tasks.

2.2.1. Motivation and State of the Art

Quantifying uncertainty in time series prediction and ensuring the safe operation of complex systems are fundamental challenges in statistics and control theory. Traditional model-based approaches, such as Kalman filters, control Lyapunov theory, and robust MPC [28–30], offer theoretical guarantees but rely on the validity of assumed models [31] and computational feasibility, especially for nonlinear dynamics [32].
Data-driven methods, including RNNs [33], LSTMs [34], LLMs [35], learning-based MPC [36], and reinforcement learning (RL) [37–39], offer greater flexibility but lack the strong guarantees of model-based techniques. These methods often struggle with reliable uncertainty quantification [40, 41], particularly under distribution shifts or limited data. While Bayesian approaches provide a principled way to estimate uncertainty, they rely on approximations [42] and can be sensitive to model misspecification and outliers [43–45].
Conformal prediction (CP) has emerged as a promising post-hoc calibration technique that enhances any predictive model with finite-sample reliability guarantees [46–48]. CP typ-

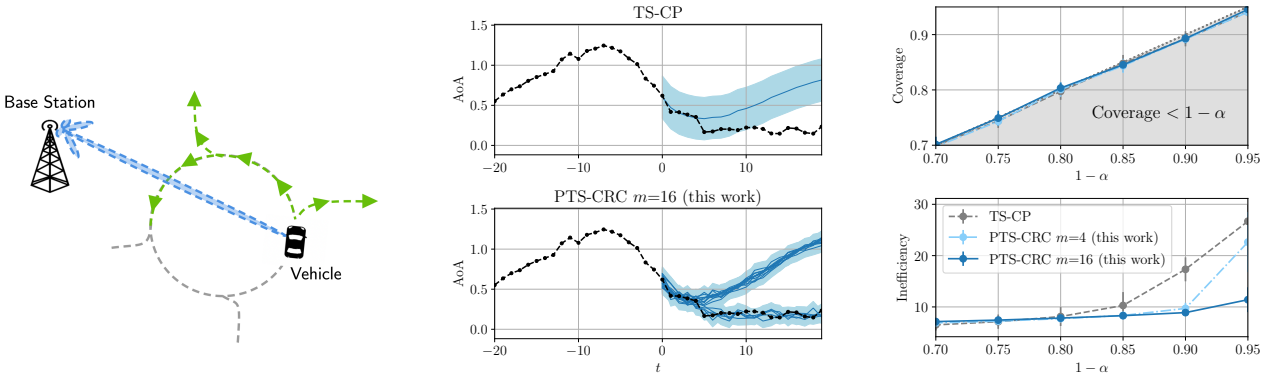

Figure 2.8: Illustration of an exemplifying application of the proposed method: (left) The base station wishes to predict the evolution of the angle of arrival y_t for the line-of-sight radio propagation path from a vehicle moving in a roundabout. (center) Given knowledge of the sequence y_t from at times $t = -20, \dots, -1$, the goal is to produce prediction intervals for the future evolution of the sequence y_t at times $t = 0, \dots, 19$. The true trajectory is shown as a dashed black line, with the predictive intervals (blue shaded areas) produced by the state-of-the-art TS-CP [1] provided in the upper part, while the proposed set-predictor (PTS-CRC) with $m = 4$ and $m = 16$ prototypes (see Fig. 2.9) are plotted in the lower part. (right) Both TS-CP and PTS-CRC are guaranteed to cover the true trajectory with probability at least $1 - \alpha$ (top), but PTS-CRC significantly reduces the inefficiency, i.e., the average predicted set size (bottom).

ically requires a separate calibration dataset [2] and can be combined with probabilistic predictors [49]. It has been applied to various uncertainty quantification tasks, including LLM-based planning [50, 51], policy evaluation in Markov decision processes [52], and run-time verification of dynamic systems [53, 54]. CP-based error bars have also been integrated into MPC for safe planning in shared environments [55, 56]. Concurrently, [57] explored probabilistic CP for explainable decision-making.

Despite its success, CP-based methods face two key limitations that we address in this work:

- *Unimodal vs. forking uncertainties*: Standard time series CP (TS-CP) [1] constructs unimodal error bars around a single trajectory, failing to capture branching uncertainties in dynamic environments. For instance, in Fig. 2.8, a base station predicts the angle of arrival of a moving vehicle, which can exit a roundabout at multiple points. TS-CP cannot model these multiple future paths, leading to overly conservative predictions.
- *Coverage vs. risk*: CP ensures reliability in terms of coverage probability—the likelihood that the predicted interval contains the true trajectory. However, many control applications require broader risk-based constraints. For example, ensuring a minimum average service quality in Fig. 2.8 goes beyond simple coverage probability, which existing methods cannot address.

2.2.2. Proposed Solution

In this section, we describe first the setting and performance criteria for the problem of reliable prediction, and then we detail the class of MPC problems under study.

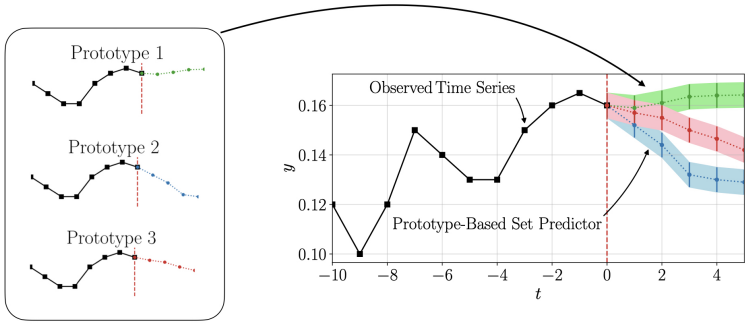

Figure 2.9: A prototype-based set predictor: Given the past evolution of time series $y_{-T:-1}$ (solid black curve), the prototype-based set predictor $\Gamma(y_{-T:-1})$ in (2.23) is constructed based on the set $\mathcal{P}^m(y_{-T:-1})$ containing m prototypical sequences (here $m = 3$) by including all sequences $\hat{y}_{0:\tau-1}$ whose maximum distance to one of the prototypes is bounded by λ (here $\lambda = 0.05$). The prototype-based set predictor $\Gamma(y_{-T:-1})$, reported on the right, includes all the sequences that are either in the red, green, or blue-shaded areas. While prior work is limited to the case of a single prototype ($m = 1$), this paper leverages the use of probabilistic sequence models to allow for $m > 1$ prototypes.

2.2.2.1. Prediction

Given the first T samples of a time series $y_{-T}, ..., y_{-1}$, our goal is to predict the next τ samples $y_0, ..., y_{\tau-1}$ with reliable error bars (see Fig. 2.8). The full trajectory $y_{-T:\tau-1} = [y_{-T}, ..., y_{\tau-1}] \in \mathcal{Y}^{T+\tau}$ is a time series where each sample y_t belongs to a set $\mathcal{Y}$, which may be discrete or a subset of a real-valued vector space. The unknown distribution of the trajectory, $p(y_{-T:\tau-1})$, can be factorized as

$$p(y_{-T:\tau-1}) = p(y_{-T}) \prod_{t=-T+1}^{\tau-1} p(y_t|y_{-T:t-1}), \tag{2.20}$$

where $p(y_t|y_{-T:t-1})$ denotes the conditional distribution of y_t given past samples.
We assume access to a *probabilistic predictor* $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$, which approximates the true conditional distribution $p(y_{0:\tau-1}|y_{-T:-1})$ from (2.20). This predictor may be a pre-trained machine learning model or a physics-based simulator [58–60]. Unlike prior work [1, 52, 53, 55, 61, 62], which focused on deterministic predictors, we consider both implicit and explicit probabilistic models.
• *Deterministic predictors*: Previous approaches assumed that $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$ is concentrated at a single predicted trajectory $\hat{y}_{0:\tau-1} \in \mathcal{Y}^\tau$, making it a deterministic function of $y_{-T:-1}$. That is,

$$\hat{p}(y_{0:\tau-1}|y_{-T:-1}) = \delta\left(y_{0:\tau-1} - \hat{y}_{0:\tau-1}\right), \tag{2.21}$$

where $\delta(\cdot)$ is the Kronecker or Dirac delta function, depending on whether $\mathcal{Y}$ is discrete or continuous.

• *Implicit probabilistic predictors*: These models generate conditionally independent samples,

$$\hat{y}_{0:\tau-1} \sim \hat{p}(y_{0:\tau-1}|y_{-T:-1}), \tag{2.22}$$

but do not explicitly provide $\hat{p}(\hat{y}_{0:\tau-1}|y_{-T:-1})$. Examples include time-GANs [63], diffusion models [64], probabilistic RNNs [65], and spiking neural networks [66, 67].
• *Explicit probabilistic predictors*: These models not only generate samples as in (2.22) but also explicitly compute $\hat{p}(\hat{y}_{0:\tau-1}|y_{-T:-1})$, enabling likelihood-based uncertainty quantification.

2.2.2.2. Set Prediction

We aim to use the probabilistic predictor $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$—either implicit or explicit—to generate a prediction set over future sequences. Specifically, we focus on set predictors that depend on past samples $y_{-T:-1}$ through a set $\mathcal{P}^m(y_{-T:-1}) = \{y^i_{0:\tau-1}\}_{i=1}^m$ of m *prototypical sequences*, where $y^i_{0:\tau-1}$ denotes the i-th prototype.
Our predictors include all future sequences $y_{0:\tau-1}$ that are within a given distance λ of at least one prototype:

$$\Gamma(y_{-T:-1}) = \left\{ y_{0:\tau-1} \in \mathcal{Y}^\tau : \min_{\hat{y}_{0:\tau-1} \in \mathcal{P}^m(y_{-T:-1})} d(y_{0:\tau-1}, \hat{y}_{0:\tau-1}) \leq \lambda \right\} \tag{2.23a}$$

$$= \bigcup_{\hat{y}_{0:\tau-1} \in \mathcal{P}^m(y_{-T:-1})} \{ y_{0:\tau-1} \in \mathcal{Y}^\tau : d(y_{0:\tau-1}, \hat{y}_{0:\tau-1}) \leq \lambda \}, \tag{2.23b}$$

where $\lambda > 0$ is a design parameter. The equivalence of (2.23a) and (2.23b) follows from the definition. While prior work [1, 52, 53, 55, 61, 62] focused on the case $m = 1$, we leverage probabilistic predictors to allow for $m > 1$.
Figure 2.9 illustrates a prototype-based predictor $\Gamma(y_{-T:-1})$ with $m = 3$. The predictor models the expected future sequence based on past data $y_{-T:-1}$. The hyperparameter m is key to controlling prediction region size—larger m can improve accuracy but may reduce predictive efficiency if improbable prototypes are included [49, 68]. Explicit predictors can mitigate this issue by filtering out unlikely prototypes, and tuning m can be done using validation data [57].
The *per-time step predicted set* at time t includes all values assumed by any trajectory in $\Gamma(y_{-T:-1})$ at that step:

$$\Gamma_t(y_{-T:-1}) = \{ y \in \mathcal{Y} : \exists\, \hat{y}_{0:\tau-1} \in \Gamma(y_{-T:-1}) \text{ with } \hat{y}_t = y \}. \tag{2.24}$$

2.2.2.3. Reliability

A reliable predictor ensures that the predicted set $\Gamma(y_{-T:-1})$ aligns with the true sequence $y_{0:\tau-1}$. We define reliability using a *loss function* $\mathcal{L}(\Gamma, y_{0:\tau-1})$, which quantifies the discrepancy between the predicted set and the actual sequence.

Assumption 1 (Bounded and monotonic loss function)**.** *The loss function satisfies*

$$\mathcal{L}(\Gamma, y_{0:\tau-1}) \leq B, \tag{2.25}$$

for all $(\Gamma, y_{0:\tau-1})$*, where* $B \leq \infty$ *is a constant. It is also monotonic in set size: if* $\Gamma' \subseteq \Gamma$*, then* $\mathcal{L}(\Gamma, y) \leq \mathcal{L}(\Gamma', y)$*.*

A predictor $\Gamma(\cdot)$ is α-*reliable* if the expected loss does not exceed a maximum unreliability level α:

$$\mathcal{R}(\Gamma) := \mathbb{E}[\mathcal{L}\left(\Gamma(y_{-T:-1}), y_{0:\tau-1}\right)] \leq \alpha, \tag{2.26}$$

where the expectation is over the unknown distribution $p(y_{-T:\tau-1})$ in (2.20).
Different choices of the loss function lead to distinct reliability conditions: • *Sequence coverage probability*: The *miscoverage loss*

$$\mathcal{L}\left(\Gamma(y_{-T:-1}), y_{0:\tau-1}\right) = \mathbb{1}\left\{y_{0:\tau-1} \notin \Gamma(y_{-T:-1})\right\} \tag{2.27}$$

returns 1 if the true sequence $y_{0:\tau-1}$ is not in $\Gamma(y_{-T:-1})$. In this case, (2.26) ensures

$$\Pr[y_{0:\tau-1} \notin \Gamma(y_{-T:-1})] \leq \alpha. \tag{2.28}$$

• *Sample coverage rate*: The *per-sample miscoverage rate* loss

$$\mathcal{L}\left(\Gamma(y_{-T:-1}), y_{0:\tau-1}\right) = \frac{1}{\tau}\sum_{t=0}^{\tau-1} \mathbb{1}\left\{y_t \notin \Gamma_t(y_{-T:-1})\right\} \tag{2.29}$$

computes the fraction of future samples not in $\Gamma_t(y_{-T:-1})$. Here, (2.26) reduces to:

$$\frac{1}{\tau}\sum_{t=0}^{\tau-1} \Pr\left[y_t \notin \Gamma_t(y_{-T:-1})\right] \leq \alpha. \tag{2.30}$$

2.2.2.4. Efficiency

Efficiency measures the informativeness of $\Gamma(\cdot)$. While the trivial predictor $\Gamma(y_{-T:-1}) = \mathcal{Y}^\tau$ satisfies (2.28) for any α, it is uninformative.
To quantify inefficiency, we define a measure $\mu(\cdot)$ over the trajectory space $\mathcal{Y}^\tau$. For discrete $\mathcal{Y}$, $\mu(\Gamma)$ may count trajectory instances; for $\mathcal{Y}^\tau = \mathbb{R}^\tau$, it may be the Lebesgue measure or the time-averaged measure:

$$\mathcal{I}(\Gamma) := \mathbb{E}[\mu\left(\Gamma(y_{-T:-1})\right)]. \tag{2.31}$$

where the expectation is over $p(y_{-T:\tau-1})$.

2.2.2.5. Model Predictive Control

Reliable and efficient set predictors support decision-making processes that must meet reliability or safety requirements. This work focuses on controlling a dynamical system whose *state* is represented by $s_t \in \mathcal{S}$, evolving through a sequence of *actions* $u_t \in \mathcal{U}$ over time t. For example, s_t may represent a drone's trajectory or network queue occupancy, while u_t could correspond to steering decisions or resource allocations.
The state s_t evolves according to the system equation:

$$s_t = f(s_{t-1}, u_t), \tag{2.32}$$

where $f(\cdot,\cdot)$ is a known function, and s_{-1} is the initial state. The *reliability* of the state sequence s_t is assessed relative to a *target process* y_t over $t = 0, 1, \ldots, \tau - 1$. We define a

constraint function $c(s_{0:\tau-1}, y_{0:\tau-1})$, measuring deviations from a reliability requirement based on $y_{0:\tau-1}$.

For example, y_t may represent data distribution, target positions, wireless traffic levels, or agent locations. The function $c(s_{0:\tau-1}, y_{0:\tau-1})$ quantifies mismatches between s_t and these reference variables.

The target process $y_{0:\tau-1}$ follows an unknown distribution (2.20). Given past actions $u_{-T:t-1}$, states $s_{-T:t-1}$, and samples $y_{-T:t-1}$, the next sample y_t depends only on previous y values:

$$p(y_t|y_{-T:t-1}, u_{-T:t-1}, s_{-T:t-1}) = p(y_t|y_{-T:t-1}). \tag{2.33}$$

Thus, as in [55], y_t is independent of actions u_t, following the randomness of $p(y_t|y_{-T:t-1})$. However, controls $u_{-T:t-1}$ can depend on past $y_{-T:t-1}$ in a closed-loop setting.

The decision maker does not observe $y_{0:\tau-1}$ but has access to past samples $y_{-T:-1}$ and a probabilistic predictor $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$. For closed-loop control, the controller also observes $y_{-T:t-1}$ at time t. Since system dynamics (2.32) are deterministic, the controller fully predicts the impact of actions u_t on s_t.

Based on available information, the decision maker selects actions u_t for $t = 0, 1, \dots, \tau - 1$ to minimize the cumulative cost function $J(s_t, u_t)$ while satisfying an average reliability constraint:

$$\begin{aligned}
\underset{u_{0:\tau-1}}{\text{minimize}} \quad & \sum_{t=0}^{\tau-1} J(s_t, u_t) && \text{(2.34a)}\\
\text{s.t.} \quad & s_t = f(s_{t-1}, u_t), \quad t = 0, 1, \dots, \tau - 1, && \text{(2.34b)}\\
& \mathbb{E}[c(s_{0:\tau-1}, y_{0:\tau-1})] \leq \delta, && \text{(2.34c)}
\end{aligned}$$

where $\delta > 0$ defines the *maximum control unreliability level*. Constraint (2.34c) ensures the average constraint function value remains within δ, with expectations taken over the unknown target process distribution $p(y_{-T:\tau-1})$.

To solve problem (2.34), Model Predictive Control (MPC) uses predictions from $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$. We consider two formulations:

- *Open-loop*: The control sequence $u_{0:\tau-1}$ is determined solely from initial state s_{-1}, past samples $y_{-T:-1}$, and predictor $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$.
- *Closed-loop*: After each step t, the controller updates its prediction based on newly observed y_t. The action u_t is selected using (2.34) with past state and target process observations $s_0, \dots, s_{t-1}$, $y_{-T}, \dots, y_{t-1}$, and the updated predictive distribution $\hat{p}(y_{t:\tau-1}|y_{-T:t-1})$.

In both cases, predictions must ensure compliance with constraint (2.34c) based on the true target process distribution, not just the predictive model.

2.2.3. Probabilistic Time Series Conformal Prediction

In this section, we introduce PTS-CRC. Following Section 2.2.2.1, we differentiate between set predictors based on implicit and explicit probabilistic forecasters. As we will see, explicit probabilistic predictors enable the definition of more general prediction schemes that may provide more informative set predictors.

2.2.3.1. PTS-CRC via Implicit Sequence Models

Given past samples $y_{-T:-1}$, an implicit probabilistic predictor produces samples of predicted trajectories $\hat{y}_{0:\tau-1}$ from the predictive distribution $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$, while not explicitly providing the value of the distribution $\hat{p}(\hat{y}_{0:\tau-1}|y_{-T:-1})$ for the generated sequences $\hat{y}_{0:\tau-1}$. Unlike TS-CP, which relies on a single predicted sequence $\hat{y}_{0:\tau-1}$, PTS-CRC leverages the capacity of a probabilistic sequence model to generate $m \geq 1$ trajectories $\mathcal{P}^m = \{\hat{y}^j_{0:\tau-1}\}_{j=1}^m$ sampled i.i.d. from the model $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$.

Based on the predicted trajectories $\mathcal{P}^m$, PTS-CRC applies the prototype-based set prediction (2.23) for a suitably designed threshold $\lambda^{PTS-CRC}$. Specifically, given a loss function $\mathcal{L}(\Gamma, y_{0:\tau-1})$ and calibration data set $\mathcal{D}_{cal} = \{y^i_{-T:\tau-1}\}_{i=1}^n$ generated i.i.d. from the unknown distribution (2.20), the threshold $\lambda^{PTS-CRC}$ is chosen so as to guarantee the reliability constraint (2.26) for the target maximum unreliability level α under any loss function satisfying Assumption 1.

The reliability requirement (2.26) depends on the unknown joint distribution (2.20), and it can be estimated using the calibration data $\mathcal{D}_{cal}$. To this end, we evaluate the loss $\mathcal{L}\left(\Gamma_\lambda\left(y^i_{-T:-1}\right), y^i_{0:\tau-1}\right)$ for each i-th calibration data point by computing the set predictor $\Gamma_\lambda\left(y^i_{-T:-1}\right)$ in (2.23) based on prototype predictions $\mathcal{P}^m_i = \{\hat{y}^j_{0:\tau-1,i}\}_{j=1}^m$ drawn from the sequence model. Note that we have made explicit the dependence of the set predictor (2.23) on the threshold λ. Then, we evaluate the empirical risk $1/n\sum_{i=1}^n \mathcal{L}\left(\Gamma_\lambda\left(y^i_{-T:-1}\right), y^i_{0:\tau-1}\right)$ by averaging over the calibration data set. Intuitively, PTS-CRC chooses the threshold $\lambda^{PTS-CRC}$ in such a way that this empirical estimate is no larger than α. More precisely, we have

$$\lambda^{PTS-CRC} := \inf\left\{\lambda : \frac{1}{n+1}\left(\sum_{i=1}^{n} \mathcal{L}\left(\Gamma_\lambda\left(y^i_{-T:-1}\right), y^i_{0:\tau-1}\right) + B\right) \leq \alpha\right\}, \tag{2.35}$$

where the empirical estimate of constraint (2.26) is corrected by adding a fictitious $(n+1)$-th data point with maximal loss value B (see Assumption 1). As explained before, this correction follows the CRC framework.

The PTS-CRC procedure, producing PTS-CRC predicted set as

$$\Gamma^{PTS-CRC}(y_{-T:-1}) = \bigcup_{\hat{y}_{0:\tau-1}\in\mathcal{P}^m} \left\{y_{0:\tau-1} \in \mathcal{Y}^\tau : d(\hat{y}_{0:\tau-1}, y_{0:\tau-1}) \leq \lambda^{PTS-CRC}\right\}. \tag{2.36}$$

PTS-CRC satisfies the following reliability guarantee.

Theorem 1. *Assuming that the samples in the calibration data set $\mathcal{D}_{cal}$ and the test sample $y_{-T:\tau-1}$ are drawn i.i.d. from distribution (2.20), and that the loss function $\mathcal{L}(\Gamma, y_{0:\tau-1})$ satisfies Assumption 1, the PTS-CRC set predictor $\Gamma^{PTS-CRC}(y_{-T:-1})$ in (2.36) satisfies the α-reliability guarantee (2.26), where the expectation is taken with respect to the calibration data set $\mathcal{D}_{cal}$, the test data point $y_{-T:\tau-1}$, and the prototypes $\left\{\mathcal{P}^m, \{\mathcal{P}^m_i\}_{i=1}^n\right\}$, with the latter drawn i.i.d. form the respective predictive distributions $\left\{\hat{p}(y_{0:\tau-1}|y_{-T:-1}), \left\{\hat{p}(y_{0:\tau-1}|y^i_{-T:-1})\right\}_{i=1}^n\right\}$.*

The properties of PTS-CRC stated in Theorem 1 can be proved by leveraging tools from the theory of CRC [2] and probabilistic CP [49]. Evaluating the PTS-CRC predictor in (2.36) generally requires an exhaustive search in the set of all sequences $\mathcal{Y}^\tau$. However, for distance measures of the form $d(y_{0:\tau-1}, \hat{y}_{0:\tau-1}) = \max_{t=0,\ldots,\tau-1} d_t(y_t, \hat{y}_t)$, with $d_t(y_t, \hat{y}_t)$ being any

weighted distance measure, the PTS-CRC predictor can be efficiently represented as the union of m balls centered at the prototypical sequences $\mathcal{P}^m$ and radius $\lambda^{PTS-CRC}$, i.e.,

$$\Gamma^{PTS-CRC}(y_{-T:-1}) = \bigcup_{\hat{y}_{0:\tau-1}\in\mathcal{P}^m} \left\{ y_{0:\tau-1} \in \mathcal{Y}^\tau : d_t(y_t, \hat{y}_t) \leq \lambda^{PTS-CRC},\ \forall t = 0, \dots, \tau-1 \right\}. \quad (2.37)$$

Furthermore, PTS-CRC can be employed with any parameterized distance measure in (2.23). The free parameters of the distance measure can be possibly optimized based on an additional calibration data set in order to ensure other desirable properties. For example, as studied in [62], one may target larger uncertainties for predictions that are further ahead into the future.

2.2.3.2. PTS-CRC via Explicit Sequence Models

In this subsection, we propose E-PTS-CRC, a variant of PTS-CRC that leverages *explicit* probabilistic predictors. As detailed in Section 2.2.2.1, explicit forecasters not only allow a trajectory $\hat{y}_{0:\tau-1}$ to be sampled from the model distribution $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$, but they also provide the value of the distribution $\hat{p}(\hat{y}_{0:\tau-1}|y_{-T:-1})$ for the synthesized sample. We take inspiration from the literature on language models [69], with the aim of generating sets of predicted trajectories that are better representatives of the plausible evolutions of the input sequence $y_{0:\tau-1}$.
This objective is accomplished via a biased sampling procedure that generates samples $\hat{y}_{0:\tau-1}$ from a distribution $\hat{q}(y_{0:\tau-1}|y_{-T:-1})$ that is generally distinct from the sequence model $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$, satisfying additional desirable properties. For instance, in the context of text generation, which corresponds to a time series forecasting problem over a sequence of words, trajectories with the largest likelihood are often nonsensical [70], and hence one may wish to filter out sequences by typicality rather than likelihood [71]. Furthermore, it may be desirable to explicitly avoid the generation of sequences that are too unlikely. As exemplary strategies, we elaborate here on sequence-level filtering [49] and autoregressive filtering [69–71].

2.2.3.3. Sequence-level filtering

Sequence-level filtering aims at obtaining samples from high-density regions of the predictive distribution, while reducing the occurrence of unlikely trajectories. This is done by filtering out samples with low likelihood from a set of trajectories sampled from the predictive distribution $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$ [49]. Specifically, given an explicit model $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$, one samples a set $\mathcal{P}^{\lceil m(1+\kappa)\rceil}$ of $\lceil m(1+\kappa)\rceil$ trajectories obtained i.i.d. from $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$ given some $\kappa > 0$, and then obtains a subset $\mathcal{P}^m \subset \mathcal{P}^{\lceil m(1+\kappa)\rceil}$ by selecting the m trajectories with the largest distribution value from the set $\mathcal{P}^{\lceil m(1+\kappa)\rceil}$.

2.2.3.4. Autoregressive filtering

In autoregressive filtering, sample selection is done on a per-time step basis. For example, in *top-k sampling* [69], which applies to discrete sets $\mathcal{Y}$, the next sample $\hat{y}_t$ is constrained to lie within the set $\mathcal{S}_k$ of samples y_t with the top-k largest distribution value $\hat{p}(y_t|\hat{y}_{0:t-1}, y_{-T:-1})$. Sampling is hence done from a truncated probability $\hat{q}(y_t|\hat{y}_{0:t-1}, y_{-T:-1}) \propto$

$\hat{p}(y_t|\hat{y}_{0:t-1}, y_{-T:-1})\mathbb{1}(y \in \mathcal{S}_k)$. Other examples include p-nucleus sampling [70] and locally typical sampling [71], which respectively apply to continuous and discrete sets $\mathcal{Y}$.

2.2.4. PTS-CRC Model Predictive Control

This section presents open-loop and closed-loop policies for the MPC problem (2.34), leveraging PTS-CRC to predict target process trajectories. Control policies based on TS-CP were introduced in [55] and emerge as a special case of the broader framework proposed here. Extending MPC from TS-CP to PTS-CRC offers two advantages: (i) it accommodates tighter constraints to capture forking uncertainties, and (ii) it supports a wider range of safety requirements expressed as expectation constraints (2.34c).

2.2.4.1. Open-Loop MPC

We first consider open-loop MPC, where the action u_t depends only on the initial state s_{-1}, past samples $y_{-T:-1}$, and the probabilistic predictor $\hat{p}(y_{0:\tau-1}|y_{-T:-1})$. Since δ in (2.34c) can be absorbed into the constraint function $c(s_{0:\tau-1}, y_{0:\tau-1})$, we set $\delta = 0$ without loss of generality. Meeting the average cost constraint (2.34c) is challenging due to the unknown distribution $p(y_{0:\tau-1}|y_{-T:-1})$. To analyze the impact of prediction errors, we introduce the following assumption.

Assumption 2 (Constraint Sensitivity)**.** *For some $L > 0$, the constraint function $c(s_{0:\tau-1}, y_{0:\tau-1})$ is L-Lipschitz in $y_{0:\tau-1}$ with respect to a metric $m : \mathcal{Y}^\tau \times \mathcal{Y}^\tau \to \mathbb{R}$, i.e.,*

$$|c(s_{0:\tau-1}, y'_{0:\tau-1}) - c(s_{0:\tau-1}, y''_{0:\tau-1})| \leq Lm(y'_{0:\tau-1}, y''_{0:\tau-1}) \tag{2.38}$$

for all state sequences $s_{0:\tau-1}$ and target process trajectories $y'_{0:\tau-1}, y''_{0:\tau-1}$.

This assumption ensures that constraint variations are bounded by $Lm(y'_{0:\tau-1}, y''_{0:\tau-1})$. For example, an ℓ_p distance constraint,

$$c(s_{0:\tau-1}, y'_{0:\tau-1}) = \left(\sum_{t=0}^{\tau-1} |s_t - y'_t|^p\right)^{1/p}, \tag{2.39}$$

satisfies Assumption 2 with $L = 1$ when $m(\cdot,\cdot) = c(\cdot,\cdot)$. Another example relevant to wireless systems is

$$c(s_{0:\tau-1}, y'_{0:\tau-1}) = \sum_{t=0}^{\tau-1} \log(1 + s_t y'_t), \tag{2.40}$$

which satisfies Assumption 2 with $L = \max_{s\in\mathcal{S}} |s|$ and $m(y'_{0:\tau-1}, y''_{0:\tau-1}) = \sum_{t=0}^{\tau-1} |y'_t - y''_t|$, assuming $s_t y_t \geq 0$.
Under Assumption 2, we derive a *surrogate MPC problem* whose feasibility set is a subset of the original MPC problem (2.34). This ensures that solving the surrogate problem yields feasible solutions for the original problem.

Theorem 2 (PTS-CRC-based surrogate open-loop MPC)**.** *Given the PTS-CRC predictor* $\Gamma^{PTS-CRC}(y_{-T:-1})$ *from (2.35)-(2.36) with distance measure* $d(\cdot,\cdot) = m(\cdot,\cdot)$ *and loss function*

$$\mathcal{L}(\Gamma, y_{0:\tau-1}) = \min_{\tilde{y}_{0:\tau-1}\in\Gamma} m(y_{0:\tau-1}, \tilde{y}_{0:\tau-1}), \tag{2.41}$$

any solution to

$$\begin{aligned} \underset{u_0,\dots,u_{\tau-1}}{\text{minimize}} \quad & \sum_{t=0}^{\tau-1} J(s_t, u_t) && \text{(2.42a)}\\ \text{s.t.} \quad & s_t = f(s_{t-1}, u_t), \quad t = 0, \dots, \tau-1, && \text{(2.42b)}\\ & c(s_{0:\tau-1}, \hat{y}_{0:\tau-1}) \le -L\alpha \quad \forall \hat{y}_{0:\tau-1} \in \Gamma^{PTS-CRC}(y_{-T:-1}) && \text{(2.42c)} \end{aligned}$$

yields feasible solutions for the original MPC problem (2.34).

The surrogate problem (2.42) introduces a stricter constraint (2.42c) based on the predictor's uncertainty. The Lipschitz constant L determines the constraint's sensitivity, and adjusting α controls the balance between feasibility and conservatism.
Problem (2.42) is a semi-infinite program [72], requiring feasibility verification through the lower-level problem:

$$\max_{\hat{y}_{0:\tau-1}\in\Gamma^{PTS-CRC}(y_{-T:-1})} c(s_{0:\tau-1}, \hat{y}_{0:\tau-1}). \tag{2.43}$$

Using the definition of PTS-CRC (2.36), this simplifies to:

$$\max_{y^i_{0:\tau-1}\in\mathcal{P}^m} \max_{\substack{\hat{y}_{0:\tau-1}\in\mathcal{Y}^\tau:\\ d(y^i_{0:\tau-1},\hat{y}_{0:\tau-1})\le\lambda^{PTS-CRC}}} c(s_{0:\tau-1}, \hat{y}_{0:\tau-1}). \tag{2.44}$$

When $c(\cdot,\cdot)$ is concave and $d(\cdot,\cdot)$ convex, (2.44) is convex and solvable via standard methods. Certain choices of $c(\cdot,\cdot)$ and $d(\cdot,\cdot)$ yield closed-form solutions.
The TS-CP-based MPC controller in [55] is a special case of Theorem 2, addressing constraints of the form:

$$\mathbb{E}\left[\mathbb{1}\left\{c(s_{0:\tau-1}, y_{0:\tau-1}) > 0\right\}\right] \le \delta. \tag{2.45}$$

2.2.5. Closed-Loop MPC

In the closed-loop setting, at every time step t, as detailed in Section 2.2.2.5, the controller receives a feedback signal providing the current value of the state of the target process y_t. As such, the control u_t at time t is allowed to depend on the observed sequence $y_{-T:t-1}$, the sequence of state $s_{-1:t-1}$, and the probabilistic predictor $\hat{p}(y_{t:\tau-1}|y_{-T:t-1})$.
In this setting, the control sequence $u_0, ..., u_{\tau-1}$ is designed by following a *receding horizon* strategy. Accordingly, at every time step $t > 0$, the control action u_t is obtained by optimizing the future control sequence $u_t, ..., u_{\tau-1}$ and then retaining only the first action. As we describe next, this optimization leverages leverages the output of the PTS-CRC predictor $\Gamma^{PTS-CRC}(y_{-T:t-1})$ and the feedback sequence $y_{0:t-1}$ in a manner similar to Theorem 1.
Under Assumption 2, at each time step $t = 0, ..., \tau-1$, based on the observed sequence $y_{-T:t-1}$ we define a *surrogate MPC problem* whose feasibility set is guaranteed to be a subset of the feasibility set of the control problem (2.34) for the time interval $t, ..., \tau-1$. The surrogate problem imposes more conservative constraints, which will be addressed using the available predictor $\Gamma^{PTS-CRC}(y_{-T:t-1})$.

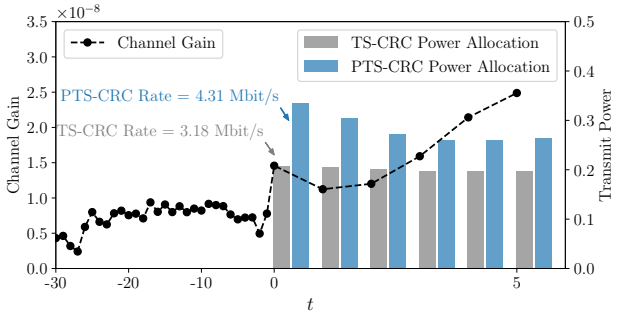

Figure 2.10: Power allocation example obtained solving the model predictive power control problem based on the TS-CRC [2] and the proposed PTS-CRC predictor. The true channel realization (unknown) is shown as a dashed black line. Both the TS-CRC (in gray) and the PTS-CRC (in blue) power allocations ensure that the maximum cumulative interference over $k = 3$ slots does not exceed the safety threshold γ. However, the larger efficiency of the PTS-CRC predictor translates into a rate that is 33% larger than the one obtained using the TS-CRC predictor.

Theorem 3 (PTS-CRC-based closed-loop MPC)**.** *For each time step $t = 0, ..., \tau-1$, consider the PTS-CRC predictor $\Gamma^{PTS-CRC}(y_{-T:t-1})$ obtained with the distance measure $d(\cdot,\cdot) = m(\cdot,\cdot)$ for some target reliability level $\alpha > 0$ and calibrated via (2.35) on the loss*

$$\mathcal{L}(\Gamma, y_{0:\tau-1}) = \min_{\tilde{y}_{0:\tau-1}\in\Gamma} m(y_{0:\tau-1}, \tilde{y}_{0:\tau-1}). \tag{2.46}$$

Then, the sequence of actions $u_0, ..., u_{\tau-1}$, in which u_t is obtained as a solution of

$$\begin{aligned}
\underset{u_t...,u_{\tau-1}}{\text{minimize}} \quad & \sum_{k=t}^{\tau-1} J(s_k, u_k) && \text{(2.47a)}\\
s.t. \quad & s_k = f(s_{k-1}, u_k), \quad \text{for} \quad k = 0, \dots, \tau-1 && \text{(2.47b)}\\
& c(s_{0:\tau-1}, \hat{y}_{0:\tau-1}) \leq -L\alpha, \quad \text{for all} \quad \hat{y}_{0:\tau-1} \in \Gamma^{PTS-CRC}(y_{-T:t-1}), && \text{(2.47c)}
\end{aligned}$$

yields feasible solutions also for the original MPC problem (2.34) in which the average reliability constraint is evaluated on average with respect to prediction $\{\Gamma^{PTS-CRC}(y_{-T:t-1})\}_{t=0}^{\tau-1}$ and the evolution $y_{0:\tau-1}$.

In a manner similar to the open-loop control case studied in the previous subsection, the closed-loop control policy based on TS-CP [55] can be recovered as a special instantiation of the PTS-CRC policy of Theorem 3.

2.2.6. Performance Evaluation

2.2.7. Open-Loop Model Predictive Power Control for Interference Mitigation

In this subsection we leverage the reliable channel gain set predictors evaluated above to derive model predictive power control policies that satisfy interference constraints. More specifically, we consider the scenario in which licensed users (LU) and unlicensed users (UU) coexist within the cell, and the BS is tasked with the problem of modulating its transmit power $P_0, ..., P_{\tau-1}$ over the next τ communication slots in order to maximize the sum-rate of

the UU, while controlling the interference experienced by the LU. This formulation is motivated by the fact that UUs are typically served in a best-effort fashion whereas higher priority LUs have strict reliability requirements (see, e.g., [73, 74]).
The BS observes the past evolution of the channel gain $g^{LU}_{-T:-1}$ of the LU, as well as the past evolution of the channel gain $g^{UU}_{-T:-1}$ of the UU. The future sum-rate of the UU is estimated based on a forecast $\hat{g}^{UU}_{0:\tau-1}$ of the evolution of the UU channel gain as

$$R^{UU}(P_{0:\tau-1}, \hat{g}^{UU}_{0:\tau-1}) = \frac{1}{\tau}\sum_{t=0}^{\tau-1} B\log_2\left(1+\frac{\hat{g}^{UU}_t P_t}{N_0 B}\right), \tag{2.48}$$

where we recall that P_t is the power allocated by the BS at time t.
The interference constraint for the LU is formulated as an upper bound on the expected maximum cumulative interference over k subsequent communication slots. For a power allocation $P_0, ..., P_{\tau-1}$, the *maximum k-step cumulative interference* at the LU over the future time horizon $0, 1, ..., \tau$, is measured as

$$\phi\left(g^{LU}_{0:\tau-1}, P_{0:\tau-1}\right) = \max_{t'\in\{0,\dots,\tau-k-1\}} \frac{1}{k}\sum_{t=t'}^{t'+k} g^{LU}_t P_t, \tag{2.49}$$

where the maximization ranges over all possible periods of k times slots. Therefore, for a safety threshold γ, the LU interference constraint amounts to the inequality

$$\mathbb{E}\left[\phi\left(g^{LU}_{0:\tau-1}, P_{0:\tau-1}\right)\right] \leq \gamma, \tag{2.50}$$

where the expectation is over the unknown LU channel evolution $g^{LU}_{0:\tau-1}$. We note that the constraint (2.50) can be also expressed in the language of robust signal temporal logic (STL) [75].
The interference threshold γ in (2.50) is determined based on the observed past LU channel evolution $g^{LU}_{-T:-1}$. Accordingly, it is set to a fraction $\beta \in [0, 1]$ of the maximum k-step cumulative interference over the past T communication slots, i.e.,

$$\gamma = \beta\left(\max_{t'\in\{-T,\dots,-1-k\}} \frac{P_{\max}}{k}\sum_{t=t'}^{t'+k} g^{LU}_t\right). \tag{2.51}$$

By (2.51), a smaller value of β imposes a stricter interference constraint. The power control problem can then be formalized as the following *open-loop* problem

$$\underset{P_0,\dots,P_{\tau-1}}{\text{maximize}} \quad \frac{1}{\tau}\sum_{t=0}^{\tau-1} B\log_2\left(1+\frac{\hat{g}^{UU}_t P_t}{N_0 B}\right) \tag{2.52a}$$

$$\text{s.t.} \quad 0 \leq P_t \leq P_{\max}, \quad \forall t \in \{0, \dots, \tau-1\} \tag{2.52b}$$

$$\mathbb{E}\left[\phi\left(g^{LU}_{0:\tau-1}, P_{0:\tau-1}\right)\right] \leq \gamma. \tag{2.52c}$$

The constraint (2.52c) is not directly tractable due to the expectation over the unknown distribution of the future evolution of the LU channel gain $g^{LU}_{0:\tau-1}$. However, the constraint function satisfies Assumption 2 in Section 2.2.4 by virtue of the inequality

$$\left|\phi\left(g^{LU}_{0:\tau-1}, P_{0:\tau-1}\right) - \phi\left(\tilde{g}^{LU}_{0:\tau-1}, P_{0:\tau-1}\right)\right| \leq \frac{P_{\max}}{k}\max_{t'\in\{0,\tau-1-k\}}\sum_{t=t'}^{t'+k}\left|g^{LU}_t - \tilde{g}^{LU}_t\right|. \tag{2.53}$$

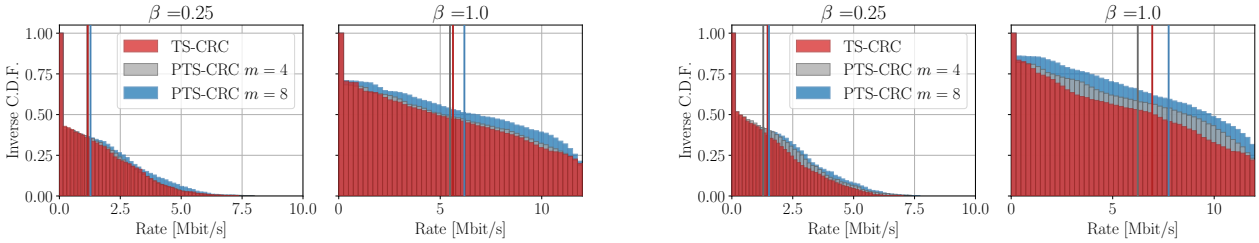

Figure 2.11: Inverse empirical cumulative distribution function (C.D.F.) of the rate obtained by the different model predictive power control policies as a function of the maximum LU interference level β. We consider two different interference windows of size $k = 1$ (left) and $k = 3$ (right).

As stated in Theorem 2, it is then possible to replace constraint (2.52c) with a stricter constraint that depends on the output of a reliable set predictor.
To elaborate on this point, define the distance

$$d^k(\hat{g}_{0:\tau-1}, g_{0:\tau-1}) = \frac{1}{k} \max_{t' \in \{0,\tau-1-k\}} \sum_{t=t'}^{t'+k} |g_t - \hat{g}_t| \,. \tag{2.54}$$

For a given given power allocation $P_{0:\tau-1}$ the lower-level problem (2.44) associated with the surrogate constraint function is

$$\max_{g^i_{0:\tau-1} \in \mathcal{P}^m} \max_{\substack{\hat{g}_{0:\tau-1} \in \mathcal{Y}^\tau: \\ d^k(\hat{g}_{0:\tau-1}, g^i_{0:\tau-1}) \leq \lambda^{PTS-CRC}}} \phi\left(\hat{g}_{0:\tau-1}, P_{0:\tau-1}\right), \tag{2.55}$$

whose solution can efficiently evaluated as

$$\max_{g^i_{0:\tau-1} \in \mathcal{P}^m} \max_{t' \in \{0,\tau-1-k\}} \frac{1}{k} \sum_{t=t'}^{t'+k} g^i_t P_t + \lambda^{PTS-CRC} \max_{t=t',\ldots,t'+k} P_t. \tag{2.56}$$

The solution of the resulting MPC problem yields a communication rate that lower bounds the optimal rate of the original problem (2.52), which is not attainable due to the lack of knowledge of future channel realizations. We benchmark the performance of different set predictors by addressing the associated MPC control problem.
In Fig. 2.10, we illustrate the power control solutions obtained by the proposed PTS-CRC predictor with $m = 8$ prototypes, and by a TS-CRC predictor, which is obtained by applying the same steps as PTS-CRC with $m = 1$ to the predictive mean of the DeepAR model. This approach is adopted here as a benchmark since TS-CP [1] cannot address the average constraint in (2.52). We set $T = 30$ and $\tau = 6$. While both power allocations meet the interference requirement, the PTS-CRC power control policy has larger transmit power, and therefore it attains a higher communication rate. This improvement is attributed to the higher efficiency of the PTS-CRC predictor, which leads to surrogate constraints that are less conservative as compared to those given by TS-CRC.
The performance gain of PTS-CRC is further validated in Fig. 2.11, in which we provide the inverse empirical cumulative distribution function (C.D.F) of the communication rate obtained

by solving 1000 instances of the surrogate control problem for $\beta \in \{0.25, 1\}$ and $k \in \{1, 3\}$. As β and k increase, the interference constraint (2.52c) is relaxed, and all power control policies yield larger communication rates. However, for fixed values of β and k, the empirical C.D.F. of the PTS-CRC power control policy has a heavier tail and a larger mean, indicating that these power allocations are able to serve the UU with larger rates. The performance gain becomes more evident for larger values of the number of predictor's samples m. For example, for $k = 3$ and $\beta = 1$, the 50th percentile of the PTS-CRC-based power control policy with $m = 8$ prototypes is 80% larger as compared to TS-CRC.

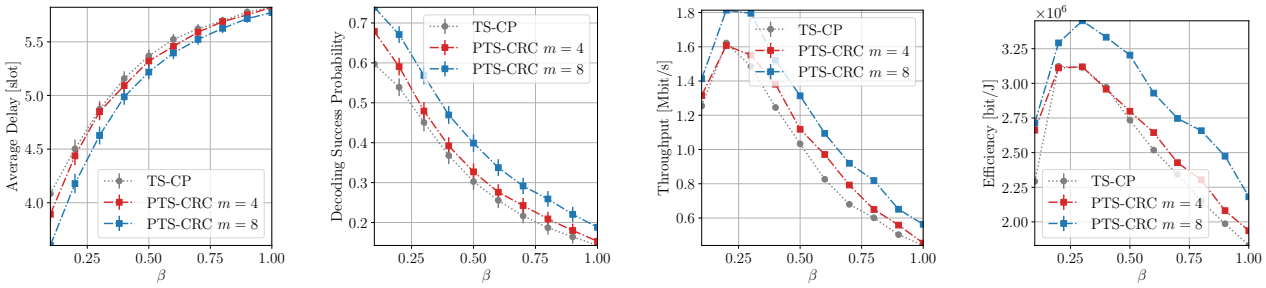

Figure 2.12: Average delay, decoding probability, throughput, and energy efficiency of HARQ-IR schemes based on the TS-CP predictor [1] and the proposed PTS-CRC predictor for $m = 4$ and $m = 8$ prototypes.

2.2.8. Energy Efficient HARQ-IR via Closed-Loop Model Predictive Power Control

In this subsection, we address the problem of designing energy-efficient hybrid automatic repeat request with incremental redundancy (HARQ-IR) protocols [76] by leveraging reliable channel state information forecasting.

As a brief review, given a sequence of random channel gains $g_{t:\tau-1}$ and transmit powers $P_{t:\tau-1}$, $\tau - 1 - t$ retransmissions of a packet encoded with rate R [bit/s/Hz] yields successful decoding at the receiver with probability [77]

$$P_{dec}(P_{t:\tau-1}, R) = \Pr[\sum_{t'=t}^{\tau-1} \log_2\left(1 + \frac{P_{t'} g_{t'}}{BN_0}\right) > R], \tag{2.57}$$

where the probability is over the future evolution of the channel gain $g_{t:\tau-1}$.

At each time t, the BS has access to the feedback sequence $g_{-T:t-1}$ of past channel gains fed back by a user equipment and it must modulate the transmit power P_t for the current time slot. The goal is to minimize energy expenditure while ensuring a minimum HARQ-IR decoding probability. The target communication rate R in constraint (2.57) is set to the rate achieved during the τ communication slots $-\tau - 1, ..., -1$ prior to their start of the HARQ process for a transmit power $\beta P_{\max}$ with $\beta \in \{0, 1\}$, i.e.,

$$R = \sum_{t=-\tau-1}^{-1} \log_2\left(1 + \frac{\beta P_{\max} g_t}{BN_0}\right). \tag{2.58}$$

By (2.58), a larger value of β indicates a more stringent constraint (2.57).

Overall, at every time step $t = 0, 1, ..., \tau - 1$ the BS optimizes the transmit power level P_t by addressing the *closed-loop* MPC problem

$$\underset{P_t,\dots,P_{\tau-1}}{\text{minimize}} \quad \sum_{k=t}^{\tau-1} P_k \tag{2.59a}$$

$$\text{s.t.} \quad t \leq P_k \leq P_{\max}, \quad \forall k \in \{t, \dots, \tau - 1\} \tag{2.59b}$$

$$P_{dec}(P_{t:\tau-1}, R - R_t) > \delta. \tag{2.59c}$$

where $0 < \delta < 1$ is the target decoding probability and

$$R_t = \sum_{t'=0}^{t-1} \log_2 \left(1 + \frac{P_{t'} g_{t'}}{BN_0}\right) \tag{2.60}$$

is the achieved rate decodable based on the past $t - 1$ retransmissions. If the problem (2.59) is not feasible, the BS does not transmit, while if it is feasible the power P_t is used for transmission. If $R - R_t < 0$ the HARQ-IR message is successfully decoded, and hence the transmission process stops. The protocol also stops at the maximum number of retransmissions τ irrespective of whether the decoding was successful or not.
Constraint (2.59c) cannot be evaluated, since the distribution of the future channel gain sequence $g_{t:\tau-1}$ is unknown. However, the constraint function satisfies Assumption 2 in Section 2.2.4, hence we have the following inequality

$$\left| \frac{1}{\tau} \sum_{t=0}^{\tau-1} \log_2 \left(1 + \frac{P_t g_t}{BN_0}\right) - \frac{1}{\tau} \sum_{t=0}^{\tau-1} \log_2 \left(1 + \frac{P_t \tilde{g}_t}{BN_0}\right) \right| \leq \frac{P_{\max}}{BN_0} \frac{1}{\tau} \sum_{t=0}^{\tau-1} |g_t - \tilde{g}_t|. \tag{2.61}$$

Thus, by Theorem 3, it is possible to replace the original constraint (2.59c) with a stricter constraint based on the output of reliable set predictors. Specifically, for a given power allocation $P_{0:\tau-1}$, the lower-level problem (2.44) associated to the surrogate constraint function is

$$\min_{g^j_{0:\tau-1} \in \mathcal{P}^m} \quad \min_{\substack{\hat{g}_{0:\tau-1} \in \mathcal{Y}^\tau: \\ |g^j_{0:\tau-1} - \hat{g}_{0:\tau-1}|_1 \leq \lambda^{PTS-CRC}}} \sum_{t'=0}^{t-1} \log_2 \left(1 + \frac{P_{t'} \hat{g}_{t'}}{BN_0}\right), \tag{2.62}$$

which can be efficiently evaluated using a water-filling algorithm [78]. By solving the resulting optimization problem, we can obtain a power control policy that satisfies the safety constraint. For a numerical example, we set the observed feedback sequence of length $T = 30$ and a maximum number of retransmissions $\tau = 6$ steps using the TS-CP and the proposed TS-CRC set predictor with $m = 4$ and $m = 8$ prototypes. In Fig. 2.12, we present key performance indicators of the resulting HARQ-IR transmission protocol for different values of β. Specifically, we solve 1000 problem instances and compute the average number of retransmissions, the probability of decoding the HARQ packet, the average throughput, and the average energy efficiency expressed as the number of decoded bits per Joule of transmit energy.
As the value of β increases, the target information rate becomes larger, resulting in an increase in average delay and a decrease in decoding success probability for all schemes.

However, when the PTS-CRC predictor is employed to predict the evolution of the future channel gain, the resulting HARQ-IR protocol can decode a larger fraction of information packets with shorter delays as compared to the TS-CP-based HARQ-IR scheme. Consequently, the PTS-CRC-based HARQ-IR scheme achieves an average throughput and average energy efficiency up to 25% higher than that of the TS-CP-based scheme.

2.2.9. Conclusions

We have addressed the problem of monitoring and controlling cyber-physical systems based on set predictors that provide reliable uncertainty estimates. To this end, we have proposed PTS-CRC, a novel post-hoc calibration technique that leverages pre-trained probabilistic sequence models, like language models, to obtain predictive intervals with finite-sample reliability guarantees. PTS-CRC leverages an ensemble of prototype trajectories sampled from the sequence model to effectively capture forking uncertainties, while satisfying reliability guarantees beyond the conventional coverage criterion. Furthermore, we have demonstrated an application of PTS-CRC to open-loop and closed-loop model predictive control problems under general average constraints on the quality or safety of the control policy.

3. Context-based caching for sustainable AI at the wireless edge

This chapter reports the research activity performed within Task 4.2.2, titled “Develop energy-efficient model training and inference algorithms for AI-AI radio resource management". To achieve the task’s objectives, our work has tackled the main aspects determining the carbon footprint of ML models training and execution. *First*, energy efficiency can be greatly increased by caching ML models at the edge so that they can be cleverly shared among different inference tasks. *Secondly*, the training data to use, as, by reducing their amount, we can substantially decrease the consumption of computing and networking, hence, energy, resources. *Third,* whenever distributed learning, such as Federated Learning (FL), needs to be used to preserve data privacy, communication costs can be reduced by adopting information compression techniques.
Accordingly, this chapter introduces the solutions we developed that, leveraging on the above three approaches, allow for substantial energy saving. Importantly, they do so while meeting the limited edge capability as well as the learning and inference quality requirements of mobile services and applications. Specifically,

1. Section 3.1 presents an architectural and algorithmic framework for intelligently caching ML models (or parts thereof) at the edge nodes in a way that minimizes power consumption of energy-intensive training and inference operations;

2. Section 3.2 proposes a methodology to select the right data to train the right ML models at the network nodes;

3. Section 3.4 describes a technique for limiting the communication costs ion distributed learning, whenever training data should be kept at the learning nodes for privacy preservation.

3.1. Caching and Sharing ML Models at the Edge

As the demand for intelligent mobile services and applications continues to rise, the need for executing tasks with resource- and compute-intensive deep neural networks (DNNs) is also increasing. Offloading these tasks from user devices to edge servers has become a widely researched approach to reduce the computational burden on mobile user equipment (UEs). However, this strategy faces the challenge of managing multiple tasks on edge servers with limited computing and memory resources. It is thus crucial to design innovative solutions for the efficient caching and sharing of DNN models at the network edge that optimize (i) the utilization of edge resources, particularly memory – which has been largely overlooked in prior work – and the radio resources used for task offloading; (ii) the selection and number of offloaded tasks for execution; (iii) the quality of task offloading, and (iv) the configuration of the DNNs. Below, we first formulate the DNN for Scalable Offloading of Tasks problem, demonstrate that it is NP-hard, and propose a weighted-tree-based heuristic solution, named OffloaDNN, which efficiently resolves the problem. Further details can also be found in the paper led by CNIT in [79].

3.1.1. State of the Art and Motivation

Several recent works have addressed task offloading, which is generally formulated as a mixed integer non-linear problem (MINLP) and, due to its complexity, in most cases, cannot be solved to optimality. A MINLP formulation of the joint optimization of the offloading strategy and the allocation of edge computing resources can be found in [80], where the authors solve it by quasi-convex/convex optimization methods and an efficient heuristic algorithm. [81] formulates the MINLP problem of joint offloading, content caching, and resource allocation and solves it by tree-search and branch and bound.
Another body of work aims to solve the above problems, and variants thereof, through machine-learning techniques. For instance, [82] develops a deep learning method for multi-label classification, while minimizing the system overhead in a single-user, single-cell scenario. [80] proposes a feedforward neural network to jointly optimize the offloading decision and the allocation of computing resources at the edge, considering latency constraints. Latency constraints are also the focus of [83], which deals with statistical rather than deterministic latency guarantees. In particular, [83] formulates a model to correlate QoS guarantees with task offloading strategies, and proposes an algorithm to offload tasks with statistical QoS using convex optimization.
In the context of CV tasks execution, [84] presents a solution for real-time CV applications that aims to achieve the best DNN accuracy and delay according to the hardware type for which the DNN workload is intended. [85] uses a feature specific for CV, i.e., adaptive quality optimization, to offload tasks by selecting a suitable execution version according to task latency constraints. A similar concept of adaptive task quality is employed by SEM-O-RAN [86], which maximizes the number of offloaded tasks by (i) applying semantic compression to task input images, and (ii) allocating edge resources of different types in a balanced manner, as to avoid resource starvation. Contrary to ours, the above approaches do not leverage DNN blocks sharing, optimizations of the DNNs structure, or fine-tuning and pruning, and only consider binary task admission decisions.
Novelty. In summary, no existing work specifically addresses the scalability of CV tasks offloading, or looks at the CV DNN structures as a way to optimize the execution of such tasks at the edge. In contrast, we address the challenges presented by task offloading in edge computing environments in a holistic manner, optimizing radio and computational resources for efficient DNNs deployment and inference. Both input data acquisition for inference execution and transmitting results back to UEs indeed require careful management of radio resources to ensure optimal usage and prevent communication bottlenecks. Moreover, apart from the memory occupied by the deployed DNNs, adjusting the DNN architectures for specific CV tasks and compressing them prior to deployment, as well as conducting inference on the offloaded data, necessitates substantial computational work from the CPUs and GPUs within edge servers.
To further motivate our work and the strategy adopted by our proposed solution, we highlight below three experiment-driven facts.

- Training cost optimization: In our experiments [79], we showed that the training computational cost (i.e., CPU/GPU usage) can be effectively traded-off with the desired level of performance, depending on the training objectives. This can be achieved by properly freezing some portions of a DNN and sharing them among different tasks, and by fine-tuning others.

In particular, training a DNN starting with random parameter initialization leads to slow convergence to the desired accuracy. Instead of training a DNN from scratch for each task, some layers can be trained and then frozen and shared across different tasks, while others can be fine-tuned on a task-specific dataset [87]. For instance, DNNs pre-trained on large datasets like ImageNet [88] can be adapted to a new task with minimal re-training. If the new dataset is similar to the pre-training data, freezing the early layers can preserve useful features while reducing computational costs by limiting the number of trainable parameters. We compared several DNN configurations using ResNet-18 as a feature extractor [89]: the configurations differ in terms of *how much of the DNN is shared or frozen, and how much is fine-tuned* with the new dataset (which includes an additional object class, such as grocery items). Different configurations vary in their training speed, accuracy, and GPU memory usage. To begin with, the DNN is initially trained on a subset of the ImageNet dataset, which includes 60 object categories, providing the pre-trained parameters that serve as the backbone. These configurations are then further trained using a new dataset that introduces an additional object class.

When a new task of detecting grocery items (e.g., mushrooms) is emulated, a fully-tuned configuration requires over 200 training epochs to reach approximately 80% testing accuracy for the new task, whereas configurations with shared layers (while others layers are frozen) achieve this accuracy more quickly. Furthermore, configurations with shared layers use lower GPU memory occupancy, highlighting that the shared layer blocks are not consuming processing resources for parameter training. However, after more than 250 epochs, the fully fine-tuned configuration ultimately achieves higher accuracy, while the shared ones tend to overfit. The key takeaway is thus that we can achieve acceptable accuracy for specific tasks using shared configurations with reduced training costs, or we can fully fine-tune DNN parameters for enhanced performance.

- Inference cost optimization: We demonstrate that the time it takes to perform inference on offloaded tasks when DNNs are deployed on resource-constrained edge servers can be decreased by using compressed (namely, pruned) versions of some portions of a DNN. Such time decrease, which also implies a lower computational cost, must however be balanced with the obtained level of inference accuracy.

  DNN pruning is a widely used compression technique designed to reduce the number of parameters in a DNN while striving to maintain its accuracy [90]. We then tested the idea that removing parts of a DNN after fine-tuning it for a specific task will significantly speed up the computation time on edge servers, hence, also reduce the inference computational cost, potentially at the expense of performance. The experiment used ResNet-18 as a feature extractor, with the same configurations as above, fine-tuned for 100 epochs on a new task of detecting 'Musical Instruments.' For a fair comparison among different configurations, the number of fine-tuning epochs is fixed to 100, and the constant pruning ratio is set to 80%. After fine-tuning, the magnitude pruning method from DepGraph [91] is employed on the fine-tuned layer blocks only, which removes the least significant weights in the fine-tuned layer blocks.

  As we showed in [79], the inference compute time for each pruned configuration resulted to be much shorter compared to their non-pruned counterparts. Also, after prun-

ing, all configurations showed a slight decline in performance compared to their original versions. While additional fine-tuning of the pruned model could enhance the accuracy for all configurations, it would also lead to higher training costs. Thus, a trade-off exists between the inference compute time and average class accuracy performance. Additionally, fine-tuning from this stage of the pruned model may increase the accuracy of all the configurations, but it will also lead to higher training costs. **The key takeaway from this experiment is that, to reduce the inference compute time (hence, also the computational cost) for tasks at the edge, we can select from various pruned configurations based on the balance they provide between accuracy and inference compute time. Alternatively, if the task demands high accuracy, we can opt for a DNN configuration that has not been pruned.**

- **Image quality optimization:** In radio resource constrained system, the input data to the inference pipeline may be degraded due to, e.g., compressed images or noisy captures. We show that the performance loss due to lesser quality (e.g., higher compression) of the task input data available during inference can be effectively mitigated. However, finding a configuration that can effectively address this issue is not trivial.

  Previous research suggests that DNN performance is susceptible in the presence of out-of-domain distortion in the inference data. [92] showed that DNN models exhibit degraded performance when exposed to JPEG-compressed images after transfer learning. Their study underlines that models trained on high-quality images from ImageNet [88] lose significant classification accuracy when presented with compressed images. This suggests that fine-tuning the model for different tasks does not always fully adapt the models to handle the compression of inference data during model deployment. Experiments in [86,93] demonstrate that the performance of object detection models like Faster R-CNN and YOLOv5x are affected by JPEG compression artifacts. The accuracy declines sharply as the compression level increases, with performance deteriorating significantly on highly compressed images. This sensitivity arises because DNNs rely on learned features that JPEG compression can obscure or distort, such as edges, shapes, and texture of the object.

  Specifically, we compared the robustness of DNN models fine-tuned for a new task with common layers from a pre-trained model in the presence of degraded images. We employed three different levels of JPEG compression ($\{10, 50, 100\}$) applied on the testing dataset. Moreover, we tested the model robustness in terms of testing accuracy obtained with the pruned versions of the same model, with three pruning ratios ($50\%, 70\%, 85\%$). To conduct this experiment, we used the vanilla ResNet-18 and we train each model for 100 epochs. **The key takeaway from our experiments [79] is that when performing inference tasks on compressed data, we can select a configuration (pruned/not pruned) of the DNN based on its inference accuracy against different levels of input data compression. Notably, higher-quality input data allows for more pruned versions of the DNN without losing much accuracy.**

3.1.2. Proposed Solution

We consider a scenario where UEs enhance their performance by offloading CV tasks to an edge computing platform linked to the base station covering these devices. A UE offloads

Table 3.1: Notation

Symbol	Description
$\tau \in \mathbb{T}$	Requested inference tasks, with $\lvert\mathbb{T}\rvert$=T
$d \in \mathbb{D}$	Set of possible dynamic DNN structures, each possibly able to serve multiple tasks
$s^d \in \mathbb{S}^d$	Block belonging to the dynamic DNN structure d
p_τ	Priority of task τ
$\pi_\tau^d \in \Pi_\tau^d$	Sequence of blocks $[s^d]_d$ belonging to the dynamic DNN structure d, suitable for executing task τ
λ_τ	Request rate of task τ
A_τ	Minimum accuracy tolerable for task τ
L_τ	Maximum latency tolerable for task τ
$\mathbb{Q}_\tau$	Set of auxiliary variables denoting the possible quality levels for the data that are input to task τ
R	Number of available RBs
C	Available compute time (CPU/GPU)
C_T	Maximum training cost (CPU/GPU time in seconds)
M	Available memory (RAM/VRAM)
σ_τ	SNR of UEs requesting task τ
b_w	Bandwidth per RB
$B(\sigma_\tau)$	No. of bits carried by an RB assigned to a UE generating requests and input for task τ
$\beta(q_\tau)$	No. of bits associated with transferring data with quality level q_τ as input to task τ
$c(s^d)$	Compute time required by block s^d
$\mu(s^d)$	Memory required by block s^d
$c_t(s^d, \cdot)$	Cost of training s^d
x_τ^d	Binary decision variable taking on 1 when DNN structure d is used for task τ
$y_{\pi_\tau^d}$	Binary decision variable, taking on 1 when π_τ^d is selected for task τ
z_τ	Real-valued decision variable representing the task requests admission ratio
r_τ	Integer decision variable indicating the no. of RBs assigned to UEs offloading task τ
q_τ	Decision variable indicating the quality level of the input data for task τ
$m(s^d)$	Binary auxiliary variable, indicating whether s^d is used by at least one task

its task to an edge server using a dedicated radio network slice. The UE's data, to be provided as input to the inference task, is characterized by a quality level, which is determined by the OffloaDNN controller. However, given its scarce resource capacity and the limited available bandwidth, the edge platform must decide which tasks to accept and how to manage them efficiently. We therefore introduce below the optimization problem to be solved at the OffloaDNN controller. The notations used here and in the following are summarized in Table 3.1.

3.1.2.1. The DOT Problem

For each type of inference task, the OffLoaDNN controller has to make the following decisions: (i) the ratio of task requests that can be offloaded to the edge; (ii) the quality level of the data to be provided as input to an inference task and (iii) the corresponding RB allocation; (iv) the best DNN to handle such tasks, (v) and its configuration. The goal is to do so while minimizing the costs associated with radio resources for offloading admitted tasks and edge resources for training and inference of DNNs handling these tasks, as well as the rejection rate of tasks, considering their priority. We thus define the following decision variables:

• the task admission vector $\mathbf{z}$=$[z_1, \ldots, z_T]$ where $z_\tau \in [0, 1]$ indicates the fraction of task τ request rate that is admitted for offloading;

• the task-DNN mapping vector $\mathbf{x}^d=[x_1^d, \dots, x_T^d]$, $\forall d \in \mathbb{D}$, where x_τ^d is a binary variable indicating whether task τ is served by the DNN d or not;
• the DNN path selection vector $\mathbf{y}_{\pi^d}=[y_{\pi_1^d}, \dots, y_{\pi_T^d}]$, $\forall d \in \mathbb{D}$ and path on d, where $y_{\pi_\tau^d}$ is a binary variable that takes on 1 if π_τ^d is used to execute task τ;
• the radio resource allocation vector $\mathbf{r}=[r_1, \dots, r_T]$ where r_τ is the number of RBs allocated for offloading task τ;
• the quality index vector $\mathbf{q}=[q_1, \dots, q_T]$, where $q_\tau \in \mathbb{Q}_\tau$ is a variable that indicates the specific quality level assigned to the input data for task τ.
Accordingly, we formulate the DOT problem as presented in the colored box below.

DNNs for scalable Offloading of Tasks (DOT)

$$\min_{\substack{z, x^d, \\ \mathbf{y}_{\pi^d}, \mathbf{r}, \\ \mathbf{q}}} \sum_{\tau \in \mathbb{T}} \alpha(1-z_\tau)p_\tau + (1-\alpha)\left[\sum_{s^d \in \mathbb{S}} \frac{c_t\left(s^d, \{y_{\pi_\tau^d}\}_{\pi_\tau^d : s^d \in \pi_\tau^d}\right)}{C_t} + \sum_{\tau \in \mathbb{T}} z_\tau \lambda_\tau \left(\frac{r_\tau}{R} + \sum_{d \in \mathbb{D}} \sum_{\pi_\tau^d \in \Pi_\tau^d} \sum_{s^d \in \pi_\tau^d} x_\tau^d y_{\pi_\tau^d} \frac{c(s^d)}{C} \right)\right] \tag{3.1a}$$

s.t.

$$\sum_{d \in \mathbb{D}} \sum_{s^d \in \mathbb{S}^d} m(s^d) \cdot \mu(s^d) \leq M, \tag{3.1b}$$

$$\sum_{\tau \in \mathbb{T}} z_\tau \lambda_\tau \sum_{d \in \mathbb{D}} \sum_{\pi_\tau^d \in \Pi_\tau^d} x_\tau^d y_{\pi_\tau^d} \sum_{s^d \in \pi_\tau^d} c(s^d) \leq C, \tag{3.1c}$$

$$\sum_{\tau \in \mathbb{T}} z_\tau r_\tau \leq R, \tag{3.1d}$$

$$z_\tau \lambda_\tau \cdot \beta(q_\tau) \leq B(\sigma_\tau) \cdot r_\tau, \ \forall \tau \in \mathbb{T}, \tag{3.1e}$$

$$a_\tau(q_\tau, y_{\pi_\tau^d}) \geq A_\tau \mathbb{1}_{z_\tau > 0}, \ \forall \tau \in \mathbb{T}, \tag{3.1f}$$

$$l_\tau(q_\tau, c(s^d, \tau), r_\tau, y_{\pi_\tau^d}, \sigma_\tau)\mathbb{1}_{z_\tau > 0} \leq L_\tau, \ \forall \tau \in \mathbb{T}, \tag{3.1g}$$

$$r_\tau \leq J \cdot z_\tau; \forall \tau \in \mathbb{T}, \tag{3.1h}$$

$$r_\tau \geq z_\tau; \forall \tau \in \mathbb{T}, \tag{3.1i}$$

$$\sum_{\tau \in \mathbb{T}} \mathbb{1}_{z_\tau > 0} \sum_{\pi_\tau^d \in \Pi_\tau^d} y_{\pi_\tau^d} \mathbb{1}_{s^d \in \pi_\tau^d} \leq K \cdot m(s^d), \ \forall s^d \in \mathbb{S}_d, \tag{3.1j}$$

$$\sum_{\tau \in \mathbb{T}} \mathbb{1}_{z_\tau > 0} \sum_{\pi_\tau^d \in \Pi_\tau^d} y_{\pi_\tau^d} \mathbb{1}_{s^d \in \pi_\tau^d} \geq m(s^d), \ \forall s^d \in \mathbb{S}_d \tag{3.1k}$$

The objective function in (3.1a) balances the task admission term and the resource allocation term using the parameter α. The resource allocation term includes: (i) the cost of training each block $s^d \in \mathbb{S}$ selected for one or more admitted tasks, normalized to the total DNN training cost (where $\mathbb{S}$ represents the set of all possible DNN blocks); (ii) the fraction of total radio resources allocated for offloading admitted tasks; and (iii) the cost of CPU/GPU time used by each DNN block for inference of admitted tasks, normalized to the full DNN cost. The training cost (c_t) of a block s^d depends on the subset of tasks using that block (indicated by

decision variables $y_{\pi_\tau^d}$), reflecting the potential savings from sharing a block among different tasks [94]. This cost is zero if no task uses s^d. Both training (c_t) and inference (c) costs can be calculated offline, so given the set of tasks $\mathbb{T}$ and possible subsets, these costs are inputs to the DOT problem.
The computing resource budget requirements are enforced by constraints (3.1b) for memory and (3.1c) for CPU/GPU time, ensuring these do not exceed the available capacity. Notably, when multiple tasks use the same DNN block s^d, the memory utilization of s^d is counted only once by introducing the binary auxiliary variable $m(s^d)$, which is set to 1 if s^d is used by at least one task. On the other hand, the compute time consumed is summed over all tasks using the DNN block s^d, scaled by the admission task rate $z_\tau \cdot \lambda_\tau$. Similarly, radio resource requirements are described by constraints (3.1d) and (3.1e). Constraint (3.1d) ensures that the number of RBs assigned to the radio network slices serving a task does not exceed the available capacity. Constraint (3.1e) requires that, for the selected admission task rate $z_\tau \cdot \lambda_\tau$, each task is provided a radio slice with enough bandwidth to transmit task input data of quality q_τ from a UE having channel quality σ_τ. Constraints (3.1f) and (3.1g) enforce compliance with task requirements. Constraint (3.1f) mandates that the accuracy function associated with task τ must satisfy the minimum accuracy requirement A_τ for tasks admitted with a non-zero ratio z_τ. Likewise, constraint (3.1g) requires that for every task admitted with a non-zero ratio, the task latency function must meet the maximum tolerable end-to-end latency requirement. The indicator function $\mathbb{1}_{z_\tau>0}$ in these constraints equals 1 when $z_\tau>0$ and 0 otherwise. Constraints (3.1h) and (3.1i) are employed to enforce $r_\tau=0$ when $z_\tau=0$ and $r_\tau>0$ when $z_\tau>0$, respectively, where J is a large value. Finally, constraints (3.1j) and (3.1k) are employed to force the values of the auxiliary binary variable $m(s^d)\in[0,1]$. To be more specific, constraint (3.1j) employs the Big M method (where K is a large value) to ensure that $m(s^d)$ equals 1 when DNN block s^d is utilized by at least one task. Conversely, as stated by (3.1k), $m(s^d)$ must be 0 when s^d is not utilized by any task. Also, the indicator function $\mathbb{1}_{s^d\in\pi_\tau^d}$ in (3.1j) and (3.1k) takes on value 1 when DNN block s^d is part of path π_τ^d and 0 otherwise.
By showing that the NP-hard binary multidimensional Knapsack problem (0/1 d-KP) can be transformed into an instance of the DOT problem within polynomial time, it can be easily proved that also the latter is NP-hard.

3.1.2.2. The OffloaDNN Solution Strategy

In light of the problem, complexity we present a heuristic algorithm, named scalable Offloading of DNN tasks (OffloaDNN), for serving inference tasks through efficient sharing and configuration of dynamic DNN structures. In our OffloaDNN approach, we address the characteristics and challenges of DOT as follows:

(i) *Graph-based model:* We tackle the problem complexity by modeling the solution space through a graph that is *built by processing the tasks sequentially (instead of in parallel)* according to their priority level, from highest to lowest. Each vertex represents a possible decision for a task, i.e., a path on a dynamic DNN structure that can be used for that task execution, and edges connect different deployment options when transitioning between tasks. To reduce the solution space and explore it efficiently, only the vertices corresponding to feasible solutions, i.e., honoring accuracy and inference latency constraints, are included upon processing a new task.

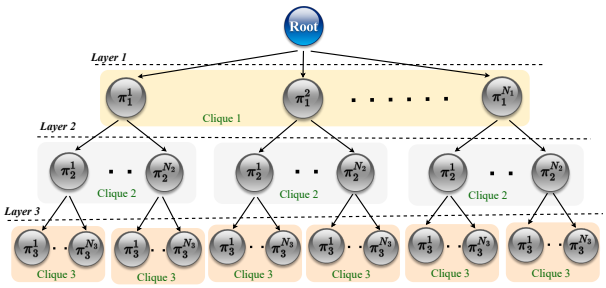

Figure 3.1: Hierarchical tree representation for T=3 tasks. Here, task τ=1 has highest priority with N_1 siblings in clique 1, τ=2 has second highest priority with N_2 siblings in clique 2, and τ=3 has third highest priority with N_3 siblings in clique 3. Each vertex v_j=π_τ^j in the clique at t-th layer corresponds to a possible path on a DNN that can serve the task of priority t.

(ii) *Assigning attributes to vertices:* Vertices carry multi-dimensional and heterogeneous attributes capturing the diverse nature of the resources to be allocated: total memory consumption, training and inference compute time, and radio resource allocation for data transfer. Additionally, attributes represent the task admission ratio, and the accuracy and latency associated with a given path. Attributes also provide flexibility in modeling the system, in that their values can vary in dynamism: they can be fixed, dynamic, or subject to optimization upon graph traversal.

(iii) *Tree structure:* The graph we build has a tree structure, with each layer corresponding to the decisions that are possible for a given task, from the highest-priority task to the lowest-priority one (the tree root represents the process starting point). Every layer accommodates “sibling" groups of vertices, i.e., groups of vertices that are replicated almost the same but for the value taken by the total memory utilization and the training cost attributes. The group of siblings at layer t is referred to as clique t. Each repetition of a clique connects to a single parent vertex, enabling us to track the dependencies of memory and training cost from previously selected paths during graph traversal (i.e., the choices concerning the higher-priority tasks). Indeed, the vertex total memory consumption and training cost attributes update dynamically during traversal. Further, by properly ordering the vertices within each clique according to their inference compute time, we dramatically reduce the complexity of tree exploration by selecting the tree branch that minimizes this metric.

The tree construction is depicted in Fig. 3.1.
Given the constructed tree, we define a cost function representing the total cost associated with a tree branch as the DOT objective function in (3.1a). In so doing, the optimal solution can be found by selecting the least-cost branch. Traversing the tree to compute the branches cost involves employing a Depth-First Search (DFS) approach to track branch costs and update the dynamic attributes of the vertices within each branch.
More specifically, while traversing the vertices of a branch, the memory consumption and training cost at each subsequent vertex at layer t and task τ are updated, considering the additional costs incurred by employing new blocks $\{s^d\}$ compared to those used by the preceding vertices within the same branch up to layer $t-1$. If the memory consumption exceeds the threshold M at any vertex on a branch, exploration of that branch halts. Thus, after exploring all branches, the cost for each branch can be computed by solving an optimization

problem with (i) the objective function as in (3.1a) but with $\mathbf{x}_\tau^d$, $\mathbf{y}_\tau^d$, π_τ^d given (note that the latter are the vertices on the considered branch); (ii) the constraints are as in (3.1c)–(3.1e) and (3.1g), since while building the tree we already made sure that those related to memory, accuracy, and latency were honored. Finally, the branch with the least cost, provides the optimal task admission ratio and resource allocation. Within this branch, the vertex information π_τ^j at layer t identifies the optimal DNN d and the corresponding DNN path for task τ.
In this new optimization problem, the objective function is a linear combination of known constants and decision variables $\mathbf{z}_\tau$ and $\mathbf{r}_\tau$. The first two and the fourth constraints are linear and, hence, convex. The third constraint involves linear combinations of z_τ and r_τ. Also, $B(\sigma_\tau)$ is positive and $b(q_\tau)$ is a convex function. Hence, the problem is convex in $\mathbf{z}_\tau$ and $\mathbf{r}_\tau$ and can be solved to the optimum by using any convex optimizer. Nevertheless, the above solution strategy has exponential complexity; further, the worst case complexity for solving the linear optimization problem for $\mathbf{z}_\tau$ and $\mathbf{r}_\tau$ with $2(T+1)$ constraints and maximum $2T$ variables for a branch is $\mathcal{O}(4T\times(T+1))$.
We therefore leverage the method followed while constructing the tree, i.e., the fact that at each layer vertices appear in increasing order w.r.t. their inference compute time, and, while traversing the tree from the root to the leaves, we select the first branch. The rationale is that, in (3.1a), the total inference cost is minimized when the corresponding compute time of all tasks is minimal. Notably, though the inference compute cost of the branches increases from left to right, the value of the DOT objective function may not follow the same trend, as it includes further terms. However, OffloaDNN exhibits now a *polynomial, and indeed dramatically reduced complexity*, at the expense of achieving a sub-optimal solution. We will show below that, in spite of this, OffloaDNN matches the optimum very closely.

Table 3.2: Scenarios Parameters

Parameter	**Small scenario**	**Large scenario**
T	$\{1,\dots,5\}$	20
$\mathbb{T}$	$\{1\},\dots\{1,\dots,5\}$	$\{1,\dots,20\}$
λ_τ [req./s]	$5\,\forall\tau$	$\forall\tau$ low: 2.5; medium: 5; high: 7.5
A_τ [top-1]	$[0.9, 0.8, 0.7, 0.6, 0.5]$	$0.8 - 0.015\cdot\tau$, $\tau\in\mathbb{T}$
L_τ [ms]	$[200, 300, 400, 500, 600]$	$200 + 20\cdot\tau$, $\tau\in\mathbb{T}$
$\lvert\mathbb{D}\rvert$	3	125
$\lvert\Pi_\tau^d\rvert$	5	10
C [s]	2.5	10
C_t [s]	1000	1000
M [GB]	8	16
$\beta(\sigma_\tau)$ [Kb]	350	350
$B(\sigma_\tau)$ [Mbps]	0.35	0.35
α	0.5	0.5
p_τ	$[0.8, 0.7, 0.6, 0.5, 0.4]$	$[1, 0.95, \dots, 0.1, 0.05]$
R [RBs]	50	100

3.1.3. Performance Evaluation

Small-scale scenario. We crafted a small-scale scenario to compare OffloaDNN to the

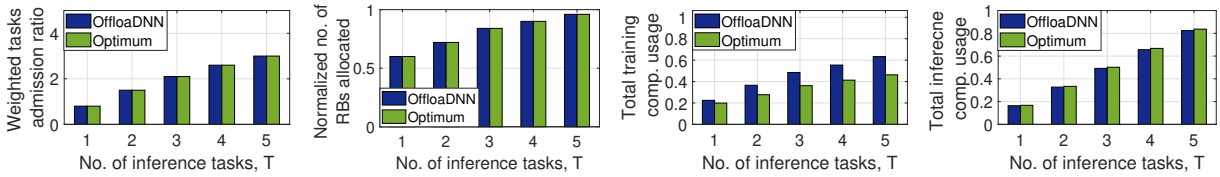

Figure 3.2: Small-scale scenario: comparison between OffloaDNN and the optimum as a function of the number of inference tasks T: (left) average task admission ratio weighted by the tasks' priority; (center-left) total number of RBs allocated to the tasks' slices, normalized to the maximum available; (center-right) total compute usage for the training of the active DNNs; (right) total compute usage for the admitted inference tasks, normalized to the maximum available.

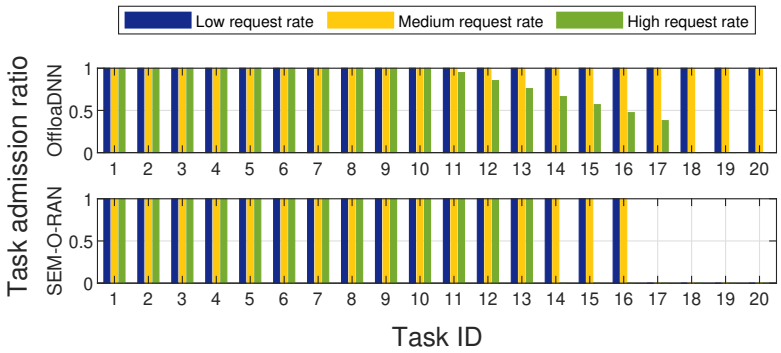

Figure 3.3: Large-scale scenario: admission rate of each task for OffloaDNN (top) and SEM-O-RAN (bottom).

optimum, which can be practically derived only for problem instances with few tasks, from 1 to 5. Tasks are ordered according to decreasing priority, while their request rate is fixed at 5 req/s for every task. Each task is also assigned a distinct latency and accuracy requirement as specified in Table 3.2. For the DNNs and paths reported in the table, we remark that each DNN path is composed of four blocks.

The cost breakdown achieved by OffloaDNN and the optimum is reported in Fig. 3.2. The weighted tasks admission ratio (Fig. 3.2(left)) is computed as the summation over all tasks of the product of the task admission ratio and the task corresponding priority. We observe that OffloaDNN allows for the same weighted task admission ratio as the optimum. Similarly, looking at the total number of RBs allocated to the tasks offloading (normalized to the available bandwidth R) in Fig. 3.2(center-left), one can observe that OffloaDNN performs as good as the optimum. Fig. 3.2(center-right) reveals that the slightly higher DOT cost under OffloaDNN relatively to the optimum is due to an increased cost of training the selected DNNs structures. However, remarkably, Fig. 3.2(right) underlines that OffloaDNN requires a lower inference compute usage than the optimum, with such usage accounting for the aggregate compute resource fraction necessary for executing all inference tasks admitted at the edge. The reduction of this relevant metric is attained thanks to the way OffloaDNN has been designed. Specifically, it is due to the combination of sorting the vertices within each clique based on the required compute time while building the weighted-tree graph, *and* the selection of the first branch of the tree performed by OffloaDNN.

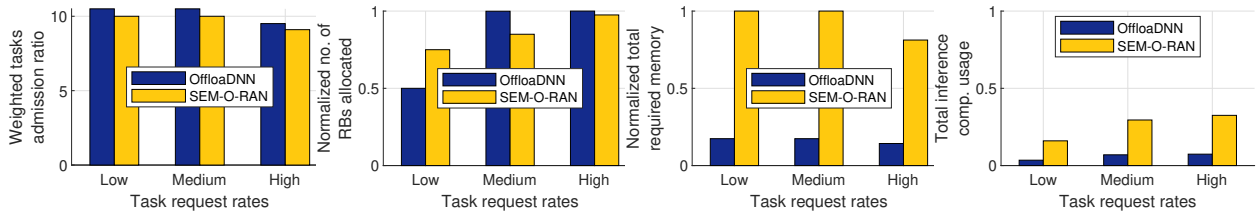

Figure 3.4: Large-scale scenario: OffloaDNN vs SEM-O-RAN as the tasks rate varies: (left) average task admission ratio weighted by task priority; (center-left) total no. of allocated RBs, normalized to the maximum available; (center-right) total memory utilization; (right) total compute usage for the admitted tasks, normalized to the maximum available.

3.1.4. Conclusions

We tackled the offloading and execution of multiple computer vision tasks at the edge, leveraging three main innovations: (1) a proper set of DNN layers can be shared among diverse offloaded tasks to save memory resources at the edge; (2) a tailored number of common layers can be “frozen" while task-specific layers can be fine-tuned, to limit training costs, preserve previously acquired knowledge, and, at the same time, fulfill tasks accuracy requirements; (3) task-specific layers can be pruned in a customized manner to further save memory and computing time at the edge while meeting accuracy requirements. To best apply such innovations, we formulated an optimization problem that determines the most efficient DNNs configurations, the tasks offloading rate that the edge can support, and the radio and computing resources to be allocated. We then envisioned a low-complexity solution strategy that effectively solves the above (NP-hard) problem. Using ResNet-18 and extensive numerical results, we validated our solution and demonstrated that it matches the optimum very closely in small-scale scenarios, and substantially outperforms state-of-the-art alternative (with 82.5% reduction in memory usage, 77.3% decrease in inference compute time, and 26.9% more admitted offloaded tasks) in larger-scale scenarios. The solution we developed thus reached a TRL 3.

3.2. Selection of Training Data in Distributed Learning

We now introduce the solution we developed to effectively and efficiently select the data to be used for ML model training in when different network nodes are willing to contribute to a learning task. Specifically, as the larger the amount of used training data, the higher the computational and energy cost of model training, and the longer the training convergence time, it is important to determine which data are essential to achieving the desired learning target and limit the training to such data. To this end, we considered a network system where each node owns a dataset and designed a data-driven node selection scheme. In other words, our solution determines the nodes that should cooperate towards the training of a machine learning model, hence, tweaking a *cooperation graph* connecting the nodes themselves. The *data-driven* approach we propose relies on three metrics to choose the edges to activate in the cooperation graph, and on an efficient iterative algorithm exploiting them. Through our performance evaluation, which leverages state-of-the-art datasets and neural network architectures, we find that privacy-preserving metrics accounting for the dif-

ference between local datasets are very effective in identifying the best edges to activate to improve the efficiency of model training without hurting performance. Further details can be found in our conference paper in [95].

3.2.1. Motivation and State of the Art

The capabilities of ML models keep growing at a very fast pace and, as mentioned in the previous section, a relevant aspect is the *training* of such models, which requires ever-increasing amounts of data and computational resources, e.g., CPU and GPU cores. As a consequence, the training of large-scale ML models may take advantage of multiple learning nodes, sharing both their resources and their local datasets. The cooperation between nodes typically follows one of two main paradigms, namely, distributed learning or decentralized learning.

In distributed learning, all nodes train one model, each using its local data, and nodes are assisted by a coordinator (a.k.a. aggregator). At each iteration, each node sends its local model to the aggregator, which combines them (e.g., by averaging) and sends the global model it obtains back to the learning nodes. Federated learning [96] and variants thereof [97,98] all follow such distributed training paradigm. Conversely, *decentralized learning* [99,100], exemplified in Fig. 3.5, adopts altogether different cooperation and communication strategies. There is no centralized coordinator, rather nodes directly communicate with each other. Furthermore, nodes only exchange the gradients for the model that is trained, and, most importantly, the cooperation between nodes is *flexible*, i.e., nodes are not required to exchange such gradients with all other nodes. Instead, nodes are free to decide the neighbors to cooperate with, building a *cooperation graph* like that formed by the red edges in Fig. 3.5. Thanks to its higher flexibility and the fact that it does not require a coordinator of the training process, decentralized learning is best suited to fully distributed, heterogeneous scenarios, where a coordinator cannot be easily identified (or, it is not convenient to do so) and nodes deemed to cooperate have different capabilities.

Also, as mentioned, choosing the pairs of nodes that cooperate, i.e., deciding the cooperation graph, is critical for the overall learning performance. Accordingly, we envision that learning nodes are assisted by a learning *orchestrator* (depicted as the brain in Fig. 3.5). Importantly, as per the decentralized learning paradigm, learning nodes do not send models to the orchestrator; rather, they share the information needed to identify the most appropriate edges to add to the cooperation graph. The orchestrator combines such information, determines the best cooperation graph, and sends it to the learning nodes, as shown by the green lines in Fig. 3.5. In general, the orchestrator will seek to minimize the *cost* of the overall learning process, subject to learning quality (e.g., accuracy) constraints: adding more edges will, in general, improve the learning quality [99, 100], while also increasing the incurred cost [99].

In the existing literature, decentralized learning has been investigate in, e.g., [101], which compares decentralized learning gossip-based algorithms against FL, showing that the best version of both schemes achieves similar results. In fact, [102] theoretically demonstrates that decentralized learning has the potential to surpass the performance of FL. This clearly underlines the potentiality of decentralized learning, whose optimization represents indeed a highly promising direction towards learning paradigms that aim at leveraging the capabilities of different nodes. Existing works also address the use of non-i.i.d. data in decentralized learning, e.g., by assigning weights to the received information [103], modifying gradients

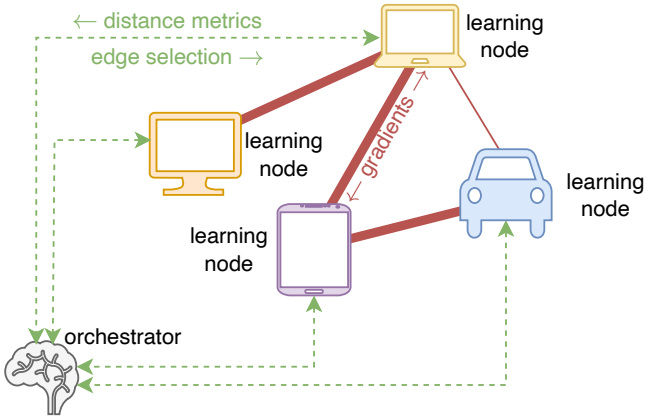

Figure 3.5: A decentralized learning scenario and conceptual representation of the proposed solution. Four *learning nodes* with heterogeneous capabilities and information have the option of cooperating to perform a DNN training task. Cooperation takes the form of exchanging gradients; nodes that can communicate are connected by solid, red edges, which may (thick edges) or may not (thin ones) be used for exchanging gradients (i.e., be active). In the scenario we envision, nodes are assisted by an *orchestrator*: nodes send to the orchestrator information about their dataset (e.g., distance metrics) and the orchestrator determines the most cost-effective cooperation graph. It then sends back to the nodes indications on which of their neighbors they should cooperate with. Such communication is represented by green, dashed edges in the figure.

as the training process approximates low loss values [104], or varying the communication frequency between learning nodes depending on specific data indicators [105]. Optimizing energy, or, more in general, cost and system resources is also critical. From the viewpoint of the network, reducing the number of data transmissions can save a substantial amount of bandwidth [105]. However, it is crucial to analyze the existing trade off between system efficiency and ML model consensus [106]. The use of communication resources has been tackled in [99], which presents an analysis of cooperation among candidate learning nodes in several distributed learning scenarios and provides an algorithm that optimizes time and system resources. Finally, it has been demonstrated analytically that the logical topology may have a significant impact on the performance of decentralized learning [107]. In fact, [108] proves that a ring communication topology allows increasing the number of nodes without damaging the spectral gap, which affect the convergence of a model training. Additionally, [109] presents a hierarchical framework that selects learning nodes by applying a clustering method that accounts for the position of the network nodes.

In our work, we tackle the problem that the orchestrator needs to solve to optimally select the edges of the cooperation graph, accounting for two complementary viewpoints: (i) *how* to make the decisions, and (ii) which *information* to use to make such decisions. Concerning the former, we devise an efficient, iterative algorithm yielding provably near-optimal solutions. For the latter, we propose and evaluate three different *metrics* to assess how useful each edge can be, featuring different trade-offs between the decision quality and the extent to which they preserve the privacy of local datasets. Our approach envisions adding a new entity to the traditional decentralized learning scenario – the orchestrator in Fig. 3.5 –, providing it with information, and having it make some fairly complex decisions about learning

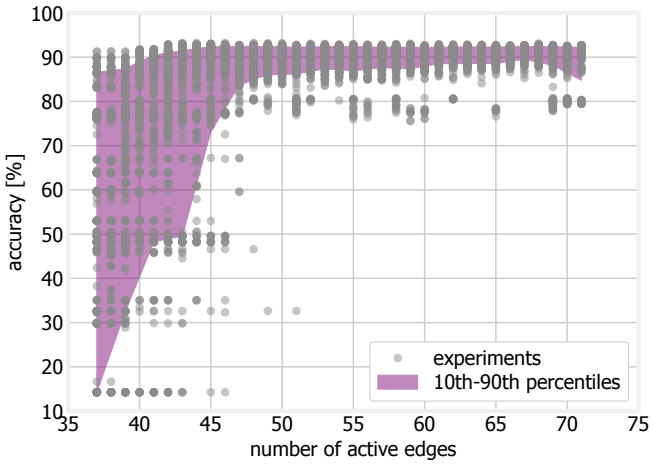

Figure 3.6: We consider an activity classification task based upon [3] and over 9,000 cooperation graphs. Markers in the plot correspond to cooperation graphs, and their locations along the x- and y-axes correspond (respectively) to the number of edges in the graph and the resulting accuracy. The purple area encloses the 10th and 90th percentiles of accuracy.

organization. This is of course worthwhile if good decisions about the cooperation graph can be made that significantly improve the learning quality. Importantly, the orchestrator *does not* aggregate models trained locally and has no part in the training process, i.e., the training procedure still fully meets the principles of decentralized learning.

As a motivating example, let us consider a simple decentralized learning scenario, based upon widely-used, publicly-available datasets and DNN architectures. Specifically, we consider nine learning nodes that have to perform an activity classification task over the HAR dataset [3]. The dataset includes data collected from 30 volunteers aged 19-48, with each person performing one of six activities (WALKING, WALKING_UP, WALKING_DOWN, SITTING, STANDING, LAYING) while wearing a Samsung Galaxy S II smartphone on the waist. The authors collected acceleration and velocity information from the embedded gyroscope, and then manually associated each set of readings with the corresponding activity. The sensor signals (accelerometer and gyroscope) were pre-processed by applying noise filters and then sampled in fixed-width sliding windows of 2.56 sec and 50% overlap (128 readings/window). Each node has a local dataset whose size varies between 400 and 600 samples. Furthermore, the number of samples of each class in each local dataset are assigned randomly, which results in a non-uniform data distribution.

For classification, we use the feed-forward DNN from [110]. We opt for that architecture in view of its simplicity, limited resource utilization, and reproducibility. We implement the DNN using the *de facto* standard `PyTorch` library, and perform training using the `lightning` extension.

In this scenario, we consider a total of 9,450 different cooperation graphs, and track, for each of them, (i) the number of edges therein and (ii) the accuracy it yields. The results are summarized in Fig. 3.6, where each marker corresponds to a cooperation graph, and the marker's locations along the x- and y-axes correspond (respectively) to the number of edges in the graph and the resulting accuracy. Moreover, the purple area in the plot corresponds to the values between the 10th and 90th percentiles of accuracy.

A first observation we can make from the plot is that, as expected, more cooperation (hence, more edges) almost always results in better performance. It is more interesting to remark what happens for less connected cooperation graphs, e.g., with around 40 edges. In these cases, the resulting accuracy can be as low as 10% – the same as randomly guessing – but also as high as 90%, matching that yielded by cooperation graphs with twice the number of edges. Achieving one accuracy value or the other solely depends upon the quality of our decisions about the cooperation graph.
An alternative way of seeing the high-level purpose of our work and the PickEdge framework is looking at the purple area, corresponding to the solution space we can explore. Moving left results in cheaper solutions; moving down results in lower-quality ones. PickEdge aims at moving as much as possible towards the left, while avoiding going down; ideally, we would move across the top edge of the purple area. Notice how the edge of such purple area is relatively straight, i.e., *there are* solutions that are both cheap and effective, hence, it is worthwhile looking for them.

3.2.2. Proposed Solution

Our system model, depicted in Fig. 3.7, includes a set of learning nodes $\mathcal{N}=\{n\}$, representing the nodes that can participate in the learning process. We further know a set of edges $\mathcal{E}=\{(n_1, n_2)\}\subseteq\mathcal{N}^2$, representing the pairs of nodes that *could* cooperate with each other. For each edge, we are given a per-epoch cost $c(n_1, n_2)$ representing, e.g., the energy consumed if nodes n_1 and n_2 cooperate. Our main decision is whether to activate each edge, expressed through binary variables $y(n_1, n_2)\in\{0, 1\}$. Also, we have to decide the number K of epochs to run. Given the vector $\mathbf{y}$ of all y-variables and K, the achieved loss is given by $\ell(K, \mathbf{y})$; we impose that such a loss value must be equal to or smaller than a target value $\ell^{\max}$. Our high-level goal is to minimize the total training cost, subject to the fact that the loss target is met:

$$\min_{K,y} \quad K \sum_{(n_1,n_2)\in\mathcal{E}} y(n_1, n_2)c(n_1, n_2), \tag{3.2}$$

$$\text{s.t. } \ell(K, \mathbf{y}) \leq \ell^{\max}. \tag{3.3}$$

There are, however, two main issues with the problem as it is stated above. Firstly, there is no closed-form expression for the loss $\ell(K, \mathbf{y})$, and even estimating the impact of decisions over the loss is not straightforward. Secondly, the problem as a whole is NP-hard; specifically, it is an integer non-linear programming problem, or INLP, where the non-linearity comes from the definition of $\ell(K, \mathbf{y})$. It follows that it is impractical to solve it directly through, e.g., a solver. We deal with both issues by proposing: (i) three possible metrics to estimate the value of activating a given edge; (ii) an iterative algorithm, based on hill-climbing, which makes near-optimal decisions.
Our solution concept, called PickEdge, features two main components. First, we replace the loss $\ell(K, \mathbf{y})$ with an estimate of the value of enabling each edge. Then, we use such value estimates in our iterative algorithm, prove that it makes near-optimal decisions. Given an inactive edge $(n_1, n_2)\in\mathcal{E}$ (i.e., such that $y(n_1, n_2)=0$), the current values of the y-variables $\mathbf{y}\in\{0, 1\}^{|\mathcal{E}|}$, and the number K of epochs to run, the value $v(n_1, n_2, K, \mathbf{y})$ associated with edge (n_1, n_2) is defined as the loss improvement (i.e., decrease) achieved by enabling the edge itself, i.e.,

$$v(n_1, n_2, K, \mathbf{y}) = \ell(K, \mathbf{y}) - \ell(K, \mathbf{y}'), \tag{3.4}$$

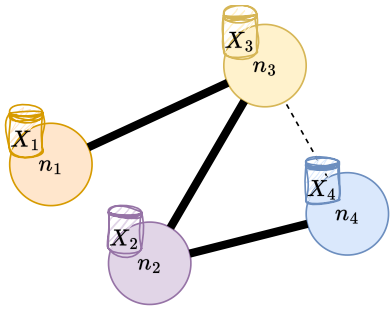

Figure 3.7: An example scenario with four nodes in $\mathcal{N}$ (represented by circles) and four edges in $\mathcal{E}$ (represented by lines). To each node corresponds a local dataset X_n, represented by a cylinder. The style of lines denotes whether or not the corresponding edge is active; as an example, $y(n_1, n_3)$=1, $y(n_3, n_4)$=0, while $(n_1, n_2) \notin \mathcal{E}$.

where $\mathbf{y}'$ is a copy of $\mathbf{y}$, with the value corresponding to (n_1, n_2) set to 1. Here, we consider that, enabling additional edges does not hurt performance, hence, $v(n_1, n_2, K, \mathbf{y})$ is always non-negative. Exact edge values are impractical to compute, given the stochastic nature of the DNN training process. Furthermore, having those values depending on both the number K of epochs to run and the currently-selected edges $\mathbf{y}$ is very cumbersome, and significantly adds to the problem complexity. Accordingly, in the following we present three *metrics* to assess edge values, leveraging different types of information.

Next, we recall that the spectral gap $\gamma(\mathbf{y})$ of a graph is the difference between the moduli of the two largest eigenvalues of its incidence matrix. It has been observed [100, Eq. (5)] that such a quantity has an influence on the learning quality; specifically, a lower spectral gap results in a higher loss. Further, adding an edge to a graph increases the spectral gap[1]. Accordingly, the spectral metric $\widetilde{v}_\gamma(n_1, n_2, \mathbf{y})$ assigns to each edge (n_1, n_2) a value corresponding to how much enabling that edge increases the spectral gap, i.e., $\widetilde{v}_\gamma(n_1, n_2, \mathbf{y}) = \gamma(\mathbf{y}') - \gamma(\mathbf{y})$. The spectral metric is the easiest to compute, as it requires no information whatsoever about the nodes or their datasets. On the negative side, not accounting for such information may lead to sub-optimal performance.

Additionally, it has been observed [111] that including nodes with overly-different local dataset can hurt the overall learning performance. Accordingly, calling X_n the local dataset of node n, the data distance estimate of the value of edge (n_1, n_2) is the opposite of the distance between their datasets: $\widetilde{v}_\mathrm{D}(n_1, n_2) = -\|X_{n_1} - X_{n_2}\|$. Unlike the spectral metric $\widetilde{v}_\gamma(n_1, n_2, \mathbf{y})$, the above definition does not depend upon the current decisions $\mathbf{y}$. Thus, the dataset distance metric can be computed offline, and does not change as decisions evolve. A major disadvantage of the data distance metric is that it requires to share the local datasets X_n. This is problematic for two main reasons: it creates additional overhead, and it may jeopardize privacy. To avoid such issues, we *decompose* the local datasets using singular value decomposition (SVD) decomposition, and represent them as: $X_n = U_n \Sigma_n V_n^\intercal$, where Σ_n is a diagonal matrix, and the matrices V_n and U_n can be interpreted, respectively, as a description of the dataset as a whole (akin to a base) and a representation of the individual samples therein. Accordingly, the distance between the datasets introduced above can be well approximated

[1] A fully-connected topology is associated with a spectral gap of 1; all other topologies have gap values between 0 and 1.

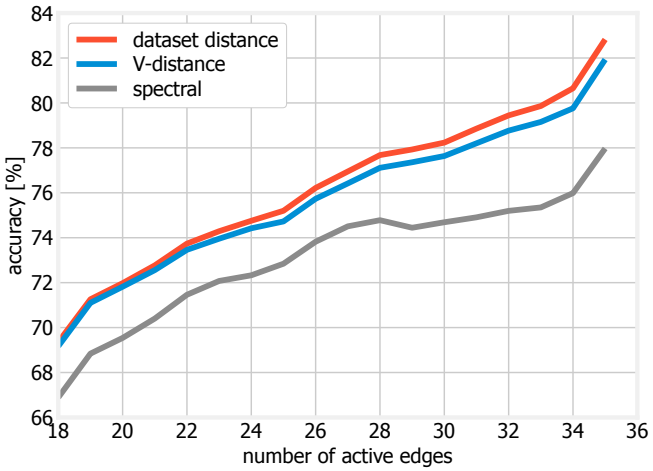

Figure 3.8: Accuracy achieved when different metrics are employed by PickEdge, as a function of the number of edges selected to be included in the cooperation graph.

Algorithm 1 The PickEdge algorithm

Require: $K, \ell^{\max}$ **forall** $(n_1, n_2) \in \mathcal{E}$ **do**
1: **end**
$y(n_1, n_2) \leftarrow 0$
2: **while** $\ell(K, \mathbf{y}) \geq \ell^{\max} \wedge \min \mathbf{y} = 0$ **do**
end
line:check
3: $(n_1^\star, n_2^\star) \leftarrow \arg\max_{\substack{(n_1,n_2)\in\mathcal{E} \\ : y(n_1,n_2)=0}} \frac{v(n_1,n_2,K,\mathbf{y})}{c(n_1,n_2)}$
4: $y(n_1^\star, n_2^\star) \leftarrow 1$
5: **return y** =0

through the distance between the corresponding V-matrices, giving the following metric:

$$\widetilde{v}_{\mathrm{V}}(n_1, n_2) = -\| V_{n_1} - V_{n_2} \|. \tag{3.5}$$

Importantly, *V-matrices are much smaller than the original datasets (i.e., the X-matrices), and can be exchanged between nodes with negligible overhead and without jeopardizing privacy*.

3.2.2.1. The PickEdge algorithm

The PickEdge algorithm (Algorithm 1) follows a straightforward greedy approach. We start with no edges enabled (Line 1); then, until the learning quality is attained, we choose a new edge to activate. Specifically, we identify the inactive edge that maximizes the ratio between its value and the cost (Line 3), and set the corresponding y-variable to 1 (Line 4). Notice that, for ease of presentation, Algorithm 1 assumes K to be given; if that is not the case, multiple instances of the algorithm could be ran with different values of K.
Algorithm 1 has two interesting properties. First, its worst-case computational complexity is polynomial – indeed, linear in the number of edges in $\mathcal{E}$. Second, for all the definitions of

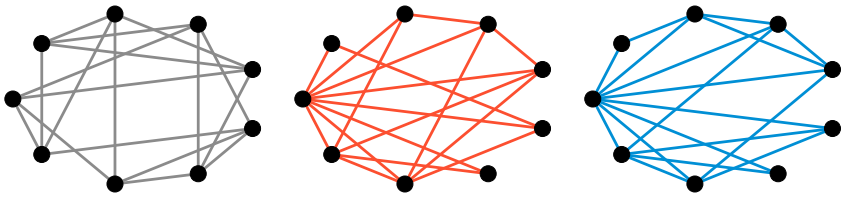

Figure 3.9: Cooperation graphs yielded by PickEdge when the target number of edges is 18 and the used metric is spectral (left), dataset distance (center), V-distance (right).

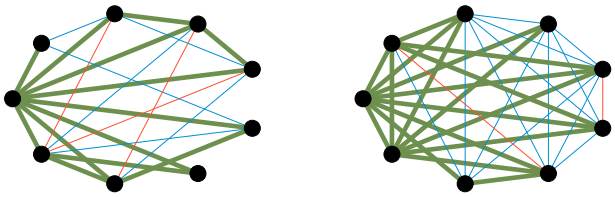

Figure 3.10: 18-edge (top) and 36-edge (bottom) cooperation graphs: edges activated under both the dataset distance and V-distance metrics (green); under dataset distance only (red); and, under V-distance only (blue).

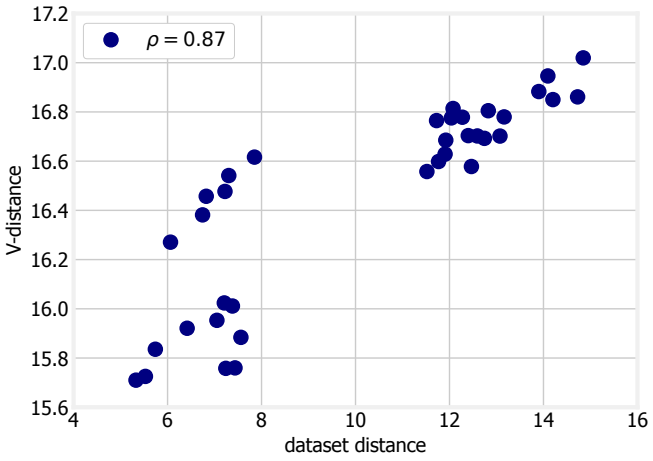

Figure 3.11: Correlation between dataset distance and V-distance: each marker represents a pair of datasets. The marker's position along the x- and y-axes are equal, respectively, to the dataset distance and V-distance between the corresponding datasets.

edge value we presented, it yields a performance (i.e., the total cost in (3.2)), that is within a constant factor from the optimum. From inspection of Algorithm 1, it can be proved that the worst-case computational complexity of the PickEdge algorithm (in Algorithm 1) is linear in the number of edges $|\mathcal{E}|$. The concrete meaning of this result is that the time taken by Algorithm 1 to make its decisions grows linearly with the size of the problem instance to solve, e.g., if the instance is twice as large, Algorithm 1 will take – at most – twice to run. The "at most" part reflects the fact that computational complexity proofs always deal with the worst case, and performance can be (and often is) better in many concrete cases. Also, computational complexity is linked with the resources needed to make the required decisions in the required time, hence, the associated energy consumption and cost.
Concerning the competitive ratio, For all of the metrics value definitions presented above, tthe PickEdge algorithm (in Algorithm 1) has a constant competitive ratio of $1-\frac{1}{e}$. The proof comes from [112, Theorem 4.7], showing that greedy algorithms in the form of Algorithm 1 have a competitive ratio of $1-\frac{1}{e}$, if the value function is submodular. Further, concerning the three functions defined above, two of them are linear (i.e., do not depend upon current decisions $\mathbf{y}$), hence, they are also submodular. As for the third one, we leverage the experiments in [113, 114], showing that the increase in the spectral gap achieved by adding one edge decreases with the spectral gap itself – the very definition of submodularity. Intuitively, the aforementioned property shows that Algorithm 1 make decisions that are not too far away from optimal ones, for all possible inputs. The property deals with worst case scenarios, and decisions can even be optimal in many practical cases. It is however important to remark that the performance of Algorithm 1 can only be as good as the value estimates provided to it. In other words, if the estimates are wrong, then so will the decisions returned by the algorithm.

3.2.3. Performance Evaluation

We consider the same reference scenario as in our motivating example, with nine nodes performing a decentralized learning task. The first aspect we are interested in is the performance achieved by PickEdge when different metrics are used, quantified through the classification accuracy.
The results are summarized in Fig. 3.8. As one might expect, a larger number of edges (hence, more cooperation) is always associated with a better accuracy. Indeed, as also shown in [100], cooperation always improves accuracy – unless very specific distributions of data and computation times are considered. It is more interesting to observe the difference between our metrics of interest: the spectral metric yields the lowest accuracy, with dataset distance and V-distance providing higher, and very similar, performance.
The reason for the relatively poor performance of the spectral metric is in the fact that it does not account for anything other than the topology of the cooperation graph, hence, employing that metric may result in not exploiting properly nodes with more, and/or more useful, data. As far as the dataset distance is concerned, it consistently yields the best performance – which is linked to the fact that the dataset distance metric captures the most information about local datasets and their features. Remarkably, the V-distance metric results in a performance that is only marginally worse, in spite of the fact that this metric makes use of (and transfers between nodes) a much smaller amount of information. Even more importantly, the difference tends to disappear as cooperation graphs become sparser, i.e., as operational conditions for cooperation become more challenging.

In summary, Fig. 3.8 suggests that **it is possible (i) to remove a large fraction of the edges of the graph without significantly affecting performance, and (ii) to use efficient, privacy-preserving techniques to choose the edges to remove without incurring additional performance losses.**

Fig. 3.9 shows the cooperation graphs obtained when the target number of edges is set to 18; this allows us to understand how different metrics influence the decisions made by PickEdge and the overall learning performance. We can observe a striking difference between the decisions yielded by the spectral metric (left plot, in grey) and the other two metrics (center and right plots): only considering the spectral gap always results in graphs that are as close to regular as possible. Indeed, regular graphs result in higher spectral gaps (mesh graphs have the highest possible gap). All other factors, e.g., the characteristics of the local datasets, are ignored; as mentioned earlier, this contributes to the lower performance shown in Fig. 3.8.

Concerning the dataset distance and V-distance metric, the resulting graphs (center and right plots in Fig. 3.9, denoted in red and blue, respectively) are clearly not regular. This is due to the fact that both these metrics seek to improve performance by utilizing network nodes with larger and/or better-quality local datasets, achieving the good performance highlighted in Fig. 3.8. Even more interestingly, the two graphs are quite similar to each other, which suggests that the two metrics lead to similar decisions.

This is confirmed by Fig. 3.10, highlighting that the dataset distance and V-distance metrics result in the same decision (i.e., activate or not) for the vast majority of edges. The resulting cooperation graphs overlap for more than 77% (thick, green edges in the figure) in the 18-edge case (top), and to an even larger degree in the 36-edge case (bottom). This confirms our intuition that the V-distance metric captures much of the same information as the more complete – but more onerous and less privacy-preserving – dataset distance one, and leads to decisions that not only yield similar performance, but are similar to each other. Importantly, we have verified this important behavior in different scenarios, with very different cooperation patterns between nodes.

Last, Fig. 3.11 presents the dataset distance and V-distance metrics themselves, and to which extent they are correlated. In the plot, each marker represents a pair of datasets; the marker's position along the x- and y-axes are equal, respectively, to the dataset distance and V-distance between the corresponding datasets. We can observe that the correlation coefficient between the two metrics (denoted by ρ in the plot) is rather high, namely, 0.87. More importantly, it often happens that, given three datasets X_0, X_1, and X_2, if X_1 is closer to X_0 than X_2 according to one metric, then the same happens according to the other one. Specifically, this happens for 92% of the possible dataset triples. Considering the structure of Algorithm 1, and especially Line 3, this guarantees that the decisions yielded by different metrics are, in fact, the same.

3.3. Conclusions

We have targeted the problem of decentralized learning in mobile networks, where the benefits of cooperation between learning nodes must be balanced against the resulting overhead and cost. In this context, we have proposed an efficient and effective algorithm, called PickEdge, able to generate near-optimal cooperation graphs using any metric expressing

how desirable the cooperation between two nodes is. We have proposed, discussed, and evaluated three different metrics, each exhibiting a different trade-off between the information that is needed and the resulting performance. Through our performance evaluation, we have found that privacy-preserving metrics leveraging singular value decomposition to estimate the distance between local datasets yield essentially the same performance as less-efficient, non-privacy-preserving metrics that directly measure the distance itself.

3.4. Energy efficient machine learning techniques at the Edge

In edge computing applications, the amount of data shared across the networks to train machine learning models is ever increasing, as well as the associated carbon footprint. Distributed applications require high inference accuracy, while maintaining the energy expenditure at sustainable levels and preserving the privacy of the edge user. To address the above needs, we investigate decentralized optimization frameworks via distributed learning schemes, to reduce communication costs and preserve privacy in networked systems. To this end, we employ second-order optimization techniques and we propose ways to reduce the required communication and energy resources of training such machine learning models and deploying them for inference.

3.4.1. Motivation and State of the Art

Internet of Things (IoT) and edge computing applications have witnessed a surge in recent years. Numerous devices are interconnected to form large networks that produce massive amounts of data. Such wide availability of data is crucial for training high-performing machine learning (ML) models for various applications such as image classification [89], speech recognition [115] and wireless resource allocation [116]. The classical approach to training these models involves the edge devices sending their raw data to a parameter server (PS) for centralized training. However, due to privacy constraints and bandwidth limitations, edge devices may not be able to send their data to the PS. To bridge this gap, federated learning has been proposed as a solution to perform distributed model training, alleviating the need to share raw and private data [117]. In first-order FL methods, such as federated averaging (FedAvg) [118], the gradients used to update the model parameters are computed locally and they are aggregated and broadcast by the PS. These methods are computationally less intensive, but suffer from slow convergence. Techniques based on momentum and adaptive learning rates have been proposed in the literature to accelerate their convergence cite. At the same time, quantization and compression schemes have minimized their compression overhead, resulting in better performance in resource-constrained environments [117].

Employing only first-order methods, in the form of the gradient information, in the optimization of the objective function of the application, is insufficient to obtain an accurate solution fast. Second-order methods, which endow the optimization algorithm with curvature information, lead to faster convergence, but at the cost of higher computational load and storage, given by the computation and transmission of the full Hessian matrix at each iteration. Together with the communication overhead and privacy issues, these drawbacks are exacerbated for large-scale models. Sharing the inverse Hessian-gradient product vector represents one approach to address these issues [119]. However, it fails to scale to

large number of devices due to the computational complexity of the order $\mathcal{O}(d^3)$ and storage requirements of the order $\mathcal{O}(d^2)$, where d represents the model size. The iterative approximation of the inverse of the Hessian matrix and its recursive multiplication with the gradient reduce the storage load [120], but the latter operation maintains its complexity of the order $\mathcal{O}(d^3)$. This issue prohibits the adoption of second-order methods as an optimization framework in energy-constrained IoT applications. Other approximations of the Hessian matrix [121] lead to computational overheads and increase the risk of model drift [122].

For large-scale models, the communication of the shared model updates becomes a major bottleneck. Inspired by the superposition of signals in multiple access channels (MAC), analog over-the-air (OTA) aggregation techniques applied in first-order methods have reduced the communication overhead [123]. So far, generally, second-order methods have not been enhanced with OTA due to high computational and storage costs. The only exception is represented by [124], which still requires the computation and storage of the full Hessian matrix.

3.4.2. Proposed Solution

3.4.2.1. Our contributions

We introduce the federated learning algorithm termed OTA Fed-Sophia, where we leverage the reduction in communication overhead provided by the OTA approach, combined with an accelerated convergence property given by second-order methods. Our approach scales well with the model size, employs the optimizer from [125] and reduces bandwidth requirements via OTA aggregation. We summarize our contributions as follows:

- Firstly, by using the local curvature information, we avoid the computation, storage and transmission of the full Hessian matrix. This approach decreases the computation and communication costs, while preserving the user's privacy.
- Secondly, to the best of our knowledge, our study is the first one to combine OTA aggregation with a scalable second-order FL algorithm via channel inversion.
- Thirdly, we validate our proposed approach in simulation experiments and, by needing fewer communication resources and lower energy consumption, we obtain gains over several federated learning baselines, such as FedAvg [118], FedProx [122] and DONE [120].

3.4.2.2. System model and problem formulation

We consider N devices that communicate with a PS to learn the optimal parameters of a model, $\boldsymbol{\theta} \in \mathbb{R}^d$, that minimize the following objective function

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \left\{ f(\boldsymbol{\theta}) \triangleq \sum_{n=1}^{N} f_n(\boldsymbol{\theta}) \right\}, \tag{3.6}$$

where $f_n(\boldsymbol{\theta})$ represents the local loss function of device n. We aim at minimizing the global loss function without utilizing the local datasets of the clients. By aggregating the gradients

received from all the clients, the PS may solve iteratively the problem formulated in equation 3.6 as:

$$\theta^{k+1} = \theta^k - \eta \nabla f(\theta^k) = \theta^k - \eta \sum_{n=1}^{N} \nabla f_n(\theta^k), \tag{3.7}$$

where $\nabla f(\theta^k)$ is the gradient at the k^{th} round and η an appropriately chosen learning rate.

Instead of using a uniform step size, in the form of the Hessian matrix, clients may employ second-order information about the behaviour of their local loss function. The Hessian measures the local curvature around the current point and refines the gradient direction, resulting in faster convergence. To solve problem 3.6, the update at iteration $(k+1)$ given by Newton's method is written as

$$\theta^{k+1} = \theta^k - \left(\sum_{n=1}^{N} \nabla^2 f_n(\theta^k) \right)^{-1} \left(\sum_{n=1}^{N} \nabla f_n(\theta^k) \right), \tag{3.8}$$

where $\nabla^2 f_n(\theta^k) \in \mathbb{R}^{d \times d}$ is the local Hessian matrix. Let $\boldsymbol{H}_n^k$ denote the Hessian matrix and $\boldsymbol{g}_n^k$ the gradient vector $\nabla f_n(\theta^k$ at the k^{th} iteration. For each iteration k until convergence, every client n computes $\boldsymbol{H}_n^k$ and $\boldsymbol{g}_n^k$ and sends them to the PS, which aggregates the values and updates the global model as given by equation 3.8. The size of the Hassian matrix is very large and sending it to the PS causes communication bottlenecks. Due to its quadratic complexity, for large models, such as neural networks (NNs), computing and storing the full Hessian is infeasible. Moreover, sharing raw Hessians and gradients exposes the information about the datasets to inverse attacks and creates privacy concerns.

3.4.2.3. Proposed algorithm

3.4.2.4. Fed-Sophia

If we employ an NN model and a cross-entropy local loss function, $f_n(\theta^k)$, then, via the Gauss-Newton decomposition, $\boldsymbol{H}_n^k$ may be approximated as

$$\boldsymbol{H}_n^k \approx \boldsymbol{J}_{\theta_n^k} \phi(\theta_n^k, \boldsymbol{x}) \cdot \boldsymbol{S} \cdot \boldsymbol{J}_{\theta_n^k} \phi(\theta_n^k, \boldsymbol{x})^T, \tag{3.9}$$

where ϕ represents the logits function, that is the mapping from input values, $\boldsymbol{x}$, to raw output values, $\phi(\theta_n^k, \boldsymbol{x})$, represents the Jacobian of ϕ with respect to (w.r.t.) θ_n^k and $\boldsymbol{S}$ the second-order derivative of the loss function w.r.t. the logits.

Let $\hat{\boldsymbol{h}}_n^k$ denotes the diagonal elements of the Hessian matrix, that is $\hat{\boldsymbol{h}}_n^k \triangleq \text{diag}(\boldsymbol{H}_n^k)$. They can be efficiently estimated using the Gauss-Newton-Bartlett (GNB) method [125], Algorithm 2. The curvature information along each direction is preserved in the diagonal elements of the Hessian, which provide the benefits of a cost-effective, memory-efficient and privacy-preserving approach to solving the optimization problem 3.6. The diagonal elements $\hat{\boldsymbol{h}}_n^k$ can be expressed as

$$\hat{\boldsymbol{h}}_n^k = B \cdot \hat{\boldsymbol{g}}_n^k \odot \hat{\boldsymbol{g}}_n^k, \tag{3.10}$$

where $\odot$ denotes the element-wise product, B represents the mini-batch size and $\hat{\boldsymbol{g}}_n^k$ is the gradient vector of the loss function, evaluated at the points from the mini-batch and with labels $\hat{\boldsymbol{y}}_j$ sampled from the logits function $\phi(\boldsymbol{\theta}_n^k, \boldsymbol{x}_j)$ on the inputs $\boldsymbol{x}_j$. To reduce the computational burden of the Hessian, we only compute it every τ iterations and we employ mini-batches. We mitigate the effect of the noise introduced by these operations with an exponential moving average (EMA) of the gradients, denoted as $\boldsymbol{m}_n^k$:

$$\boldsymbol{m}_n^k = \beta_1 \boldsymbol{m}_n^{k-1} + (1-\beta_1)\hat{\boldsymbol{g}}_n^k \tag{3.11}$$

and for the Hessian

$$\boldsymbol{h}_n^k = \begin{cases} \beta_2 \boldsymbol{h}_n^{k-1} + (1-\beta_2)\hat{\boldsymbol{h}}_n^k, & k \bmod \tau = 0 \\ \boldsymbol{h}_n^{k-1} & \text{otherwise}, \end{cases} \tag{3.12}$$

where β_1 and β_2 are tuning parameters. The PS receives the EMA of the gradients and that of the Hessians from the clients and aggregates them as

$$\overline{\boldsymbol{m}}^k = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{m}_n^k \tag{3.13}$$

$$\overline{\boldsymbol{h}}^k = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{h}_n^k \tag{3.14}$$

and computes the update direction as the element-wise division $\overline{\boldsymbol{m}}^k / \overline{\boldsymbol{h}}^k$. When this update direction is employed with non-convex functions, the Hessian matrix may exhibit rapid changes and inaccuracies, particularly in the early stages of the optimization process. This leads the algorithm to converge to a maximum or saddle point. To address this issue, we use element-wise clipping to retain values from $\overline{\boldsymbol{h}}^k$ only when the curvature is positive. We also limit the step size when the curvature is steep. We define a clipping function, $\hat{\boldsymbol{z}} = \text{clip}(\boldsymbol{z}, \gamma)$, of the vector $\boldsymbol{z} \in \mathbb{R}^d$ using a threshold $\gamma > 0$ as

$$\hat{z}_i = \max\left(\min\left(z_i, \gamma\right), -\gamma\right). \tag{3.15}$$

Now, the update in 3.8 becomes

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k - \eta \cdot \text{clip}\left(\frac{\overline{\boldsymbol{m}}^k}{\max(\gamma \cdot \overline{\boldsymbol{h}}^k, \epsilon)}, 1\right). \tag{3.16}$$

Here, $\epsilon > 0$ provides the guarantee of positivity for the curvature and avoids the division by zero. Very small or negative $\overline{\boldsymbol{h}}^k$ entries reduce the pre-conditioned gradient to $\overline{\boldsymbol{m}}^k/\epsilon$. This triggres clipping and makes $\boldsymbol{\theta}^k - \boldsymbol{\theta}^{k+1} = \eta \cdot \text{sign}(\overline{\boldsymbol{m}})$, which protects against worst-case updates when the Hessian estimate is small or negative.

3.4.2.5. Analog over-the-air direction aggregation Fed-Sophia (OTA Fed-Sophia)

At each iteration k, each of the N devices sends their computed EMAs, $\boldsymbol{m}_n^k$ and $\boldsymbol{h}_n^k$, to the PS through a wireless fading MAC. They utilize b subcarriers over a total of $\tilde{T} \in \{T, 2T\}$ time

slots, where $b \leq d$ and $T = \lceil \frac{d}{b} \rceil$. The clients perform slot-level synchronization, which yields signal synchronization at the PS and exploits the superposition property of the channel. The employ timing advance in LTE, which represents a technique to adjust the transmission timing, thus, mitigating misalignment and synchronizing the signal arrival at the PS [126]. For each iteration k, at each time slot t, each device n transmits a vector of length b, which we denote

$$\boldsymbol{S}_n(t) = [S_{n,1}(t), \dots, S_{n,b}(t)]^T \in \mathbb{C}^b. \tag{3.17}$$

Over the i^{th} subcarrier, the PS receives the channel output described as

$$y_i(t) = \sum_{n=1}^{N} h_{n,i}(t) s_{n,i}(t) + z_i(t), \tag{3.18}$$

where $h_{n,i}(t) \in \mathbb{C}$ represents the fading channel of the i^{th} subcarrier between the n^{th} client and the PS and $z_i(t) \sim \mathcal{CN}(0,1)$ the additive zero-mean and unit-variance white Gaussian noise (AWGN) at the PS at time slot t. We assume that the channel state information (CSI) is known to the PS and to the clients for each time slot t. Therefore, each client utilizes $h_{n,i}(t)$ and performs the channel inversion operation.

For each client n, during every transmission round, the average transmit power constraint is upper bounded by the maximum power, P_n:

$$\frac{1}{d} \sum_{i=1}^{d} |s_{n,i}(t)|^2 \leq P_n. \tag{3.19}$$

To avoid violating the power constraint imposed on each client, we set the transmitted signal $s_{n,i}(t)$ as

$$s_{n,i}(t) = \begin{cases} \alpha(t) \frac{\boldsymbol{m}^k_{n,i}}{h_{n,i}(t)}, & \text{if } |h_{n,i}(t)| \geq h_{th} \\ 0, & \text{otherwise} , \end{cases} \tag{3.20}$$

where $\alpha(t)$ represents a scaling factor and h_{th} represents a predefined threshold. The above equation indicates that, when the channel conditions are poor for client n, subchannel i and time slot t, that is $|h_{n,i}(t)| < h_{th}$, the client does not transmit any signal to the PS. Thus, not all model parameters reach the PS. Let $e_n(t) = \{i \in [d] : |h_{n,i}(t)| \geq h_{th}, \forall n \in [N]\}$ denote the set of elements that satisfy equation 3.20, that is that can be transmitted. Based on the exact CSI information, each client computes the scaling factor $\alpha(t)$, which satisfies the power constraint as follows

$$\frac{\alpha_n(t)^2}{|e_n(t)|} \sum_{i=1}^{d_n(t)} \left| \frac{\boldsymbol{m}^k_{n,i}(t)}{h_{n,i}(t)} \right|^2 \leq P_n. \tag{3.21}$$

After this computation, it sends the result to the PS through an error-free channel. The PS determines the minimum scaling factor satisfying all the constraints as $\alpha(t) = \min\{\alpha_i(t) : i \in e_n(t)\}$. Then, it broadcasts it to all the clients through a control channel. The PS receives the $\alpha(t) \sum_{n \in N_i(t)} \boldsymbol{m}^k_{n,i}(t) + z^k_i(t)$, where $N_i(t) = \{n \in [N] : |h_{n,i}(t)| \geq h_{th}\}, \forall i \in [d]$, represents the set

of clients that can transmit over the i^{th} subcarrier. Then, the PS divides the received expression by $\alpha(t)$ and applies matched filtering. After processing, the received signal becomes $\overline{\boldsymbol{m}}_i = \sum_{n\in N_i(t)} \boldsymbol{m}_{n,i}^k + \hat{z}_i^k$, where $\hat{z}_i^k$ is the matched filtered AWGN. The PS also receives the new approximate Hessian $\overline{\boldsymbol{h}}_i^k = \sum_{n\in N_i(t)} \boldsymbol{h}_{n,i}^k + \hat{z}_i^k$. Based on this information, the PS computes the update direction according to equation 3.16 and broadcasts the resulting vector to the clients via an error-free channel. They will update their model locally using this information received from the PS. The AWGN introduces errors when $\alpha(t)$ is small, amplifying the noise. The best approach to mitigate this issue is to perform a joint optimization of the transmission power and the scaling factor that minimizes $\sum_{n=1}^{N} \left(\boldsymbol{m}_n^k - \overline{\boldsymbol{m}}^k\right)$. But, due to its complexity, we leave this aspect for future work.

The OTA Fed-Sophia algorithm employs the parameter η and the hyperparameters $\beta_1, \beta_2, \epsilon, \gamma$, initializes the model parameters $\boldsymbol{\theta}^0$, $\boldsymbol{m}_n^0 = 0$, $\boldsymbol{h}_n^0 = 0$ and operates as follows: for each communication round k from 0 to K, for each client $n \in [N]$, it receives the global model $\boldsymbol{\theta}^k$ and sets its local model equal to it. It computes the local stochastic gradient $\hat{\boldsymbol{g}}_n^k$ and the update $\boldsymbol{m}_n^k$. Then, it calculates the scaling factor $\alpha_n(t)$ according to 3.21 and receives the global value $\alpha(t)$ from the PS via a control channel. For each component $i \in [d]$, if $|h_{n,i}(t)| \geq h_{th}$, then the device sends $\alpha(t)\boldsymbol{m}_{n,i}^k / h_{n,i}(t)$ to the PS. If $k \bmod \tau = 0$, then it computes its local Hessian estimate $\hat{\boldsymbol{h}}_n^k$ and update $\boldsymbol{h}_n^k$. It calculates the scaling factor $\alpha_n(t)$ according to 3.21 and receives the global value $\alpha(t)$ from the PS via a control channel. For each component $i \in [d]$, if $|h_{n,i}(t)| \geq h_{th}$, then the device sends $\alpha(t)\boldsymbol{h}_{n,i}^k / h_{n,i}(t)$ to the PS. It also sends the updated local Hessian $\hat{\boldsymbol{h}}_n^k$ to the PS. If $k \bmod \tau \neq 0$, then $\boldsymbol{h}_n^k = \boldsymbol{h}_n^{k-1}$. The PS computes the update $\overline{\boldsymbol{m}}_i$ and $\overline{\boldsymbol{h}}_i^k = \sum_{n\in N_i(t)} \boldsymbol{h}_{n,i}^k + \hat{z}_i^k$, $\forall i \in [d]$. The global model is updated according to equation 3.16.

3.4.3. Performance Evaluation

3.4.3.1. Training settings

We evaluate the performance of our proposed solution on a distributed classification problem and compare it with that of FedAvg [118], FedProx [122], DONE [120] and Fed-Sophia [121]. The datasets used for training and evaluation are: MNIST, Sent140, CIFAR-10 and CIFAR-100. The neural network models used for training and evaluation are: a multilayer perceptron (MLP), an LSTM model, a convolutional neural network (CNN) and a ResNet architecture. The data is independently and identically distributed (IID) among $N = 32$ clients, with 75% used for training and 25% used for testing. All models are trained using the cross-entropy loss function. Each communication round comprises 1 local iteration for Fed-Sophia and 10 for FedAvg, FedProx. In the case of DONE, additional local iterations are performed because it requires multiple updates to reach convergence. The gradient is computed using a batch size of 64 for Fed-Sophia, FedAvg and FedProx and the entire dataset for DONE. By introducing data heterogeneity in the form of a maximum of 3 labels allowed per client, we investigate the non-IID scenario.

3.4.3.2. Communication network settings

For each client, the transmission power is equal to $P_n = 1\text{mW}$, $\forall n \in \{1, 2, \dots, N\}$, the available channel bandwidth is equal to $W_{ch} = 20\,\text{MHz}$, the total number of subcarriers is equal to $b = 1200$ and each subcarrier has a bandwidth of $W_{sub} = 15\,\text{KHz}$. Aligning with LTE standards, the symbol duration is equal to $\tau = 1\,\text{ms}$. We assume the coefficients of the channels between the clients and the PS follow a Rayleigh distribution with zero mean and unit variance $\mathcal{CN}(0, 1)$. The signal-to-noise ratio is equal to SNR = 25 dB.

For OTA Fed-Sophia, in the uplink, we utilize analog communication to reduce the communication bottleneck and, in the downlink, digital communication, to broadcast the updated model to the clients. Each element of the update vector is transmitted by means of a subcarrier. Therefore, all clients need to transmit the entire vector over $\lceil \frac{d}{b} = \{67, 144, 353, 9350\} \rceil$ time slots for the datasets: MNIST (with the associated MLP model), Sent140 (with the associated LSTM model), CIFAR-10 (with the associated CNN model) and CIFAR-100 (with the associated ResNet model), respectively. For the digital version of Fed-Sophia, DONE and FedAvg, the elements of the update directions are represented and transmitted using 32 bits. The required number of transmissions becomes $32d$, which leads to the total number of time slots τ_n equal to

$$\sum_{s=1}^{\frac{N_s}{N}} \sum_{t=0}^{\tau_n} \tau R_{s,n}(t) \geq 32d, \tag{3.22}$$

where N_s is the total number of available subcarriers, N is the number of clients and $R_{s,n}(t)$ represents the maximum achievable bit rate at the t^{th} time slot and is equal to

$$R_{s,n}(t) = BW \log_2 \left(\frac{1 + P_n |h_{s,n}(t)|^2}{N_0 BW} \right), \tag{3.23}$$

where N_0 is the power spectral density of the noise and is equal to $N_0 = 10^{-9}\text{W/Hz}$.

Throughout training, the total energy expenditure of each device has two components:

(i) *the communication energy*, E_t, which measures how much energy has been consumed during the data transmission and reception,

(ii) *the computational energy*, E_c, which measures how much energy has been spent by hardware components, such as CPUs, GPUs and memory, to perform computations.

After k global iterations, the total energy comsumed by device n is equal to

$$E_{n,T}(k) = E_{n,c}(k) + E_{n,t}(k) = \sum_{j=1}^{k} e_n^j + 32d \sum_{j=1}^{k} e_{n,\text{PS}}^j, \tag{3.24}$$

where e_n^j denotes the energy consumed by the n^{th} client, during the j^{th} communication round, to train and update the model and $e_{n,\text{PS}}^j$ denotes the energy consumed by the transmission of one bit from the 32-bit representation to the PS.

3.4.3.3. Performance comparison with baselines

We employ the test accuracy in a distributed classification task as our performance measure. Using the benchmark datasets of MNIST, Sent140, CIFAR-10 and CIFAR-100, we compare the test accuracy of OTA Fed-Sophia against that of Fed-Sophia, FedAvg, FedProx and DONE, as a function of the number of communication uploads computed in 3.22.

Illustrated in Figure 3.12, our method provides the best test accuracy on the MNIST dataset at convergence, slightly surpassing Fed-Sophia, DONE and FedProx, in this order, with a more pronounced lead over FedAvg. It also requires the smallest number of communication uploads to reach this convergence value over all the baselines. In our algorithm, even though the Hessian estimates are unreliable in the beginning of the training, thanks to its clipping operation that addresses inaccuracies, it does not decrease the test accuracy. For example, in order to reach a target accuracy of 80%, OTA Fed-Sophia requires 1.6×10^2 uploads, Fed-Sophia 8.1×10^2 uploads, DONE 9.4×10^2 uploads, FedProx 20×10^2 uploads and FedAvg 46×10^2 uploads. DONE utilizes a better estimate of the Hessian matrix, but requires the entire dataset and performs 2 communication rounds to compute the global model gradient. It also estimates the inverse of the Hessian.

Figure 3.13 depicts the test accuracy performed on a sentiment analysis task using the Sent140 dataset and an LSTM model. Due to its divergence with the number of local iterations, we exclude the algorithm DONE from the comparison with the baselines. Our method outperforms Fed-Sophia by a large margin and FedProx and FedAvg by an even larger one. The last two algorithms represent first-order methods, which cannot compete in terms of accuracy with the second-order ones, Fed-Sophia and OTA Fed-Sophia. Similar conclusions may be drawn from Figure 3.14, where we conduct the analysis on the CIFAR10 dataset using a CNN model. The first-order methods exhibit much lower performance compared to the second-order ones, Fed-Sophia and OTA Fed-Sophia, with a slight advantage of FedAvg over FedProx. Compared with the analysis shown in Figure 3.13, here Fed-Sophia reaches the same accuracy as OTA Fed-Sophia, but after a much larger number of communication uploads. Fed-Sophia reaches a 60% accuracy after 8×10^4 uploads, which represents a six-fold increase over the FedAvg requirement of 4.9×10^5 uploads. At 4.1×10^3, OTA Fed-Sophia achieves the same accuracy with the lowest number of uploads. It yields over 2 orders of magnitude speedup compared to FedAvg and FedProx.

In Figure 3.15, we evaluate the ResNet architecture on the CIFAR100 dataset, to demonstrate the scalability of OTA Fed Sophia. The model contains over 11.2×10^6 parameters, which challenges the first-order algorithms. They struggle to overcome the 40% accuracy mark, with FedProx remaining slightly behind FedAvg. But, OTA Fed-Sophia, at 80% reaches the highest accuracy of all the methods, followed by Fed-Sophia at around 75%. Our algorithm also requires the least number of uploads to reach its convergence value.

For all the methods, we conduct an additional comparative analysis of how the test accuracy varies with the type of data distributions. We compare the maximum accuracy for IID and non-IID data distributions after 1.5×10^4 communication uploads. For IID data, OTA Fed-Sophia reaches a test accuracy of 97.3%, Fed-Sophia 95.8%, DONE 94.2%, FedProx 93.6% and FedAvg 89%. For non-IID data, OTA Fed-Sophia reaches a test accuracy of 87%,

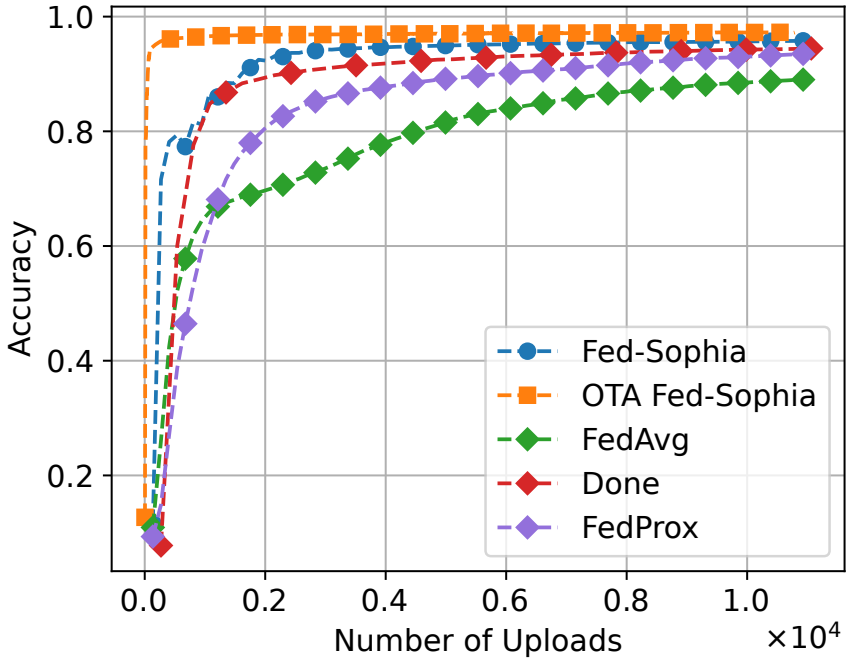

Figure 3.12: Test accuracy as a function of the number of communication uploads for Fed-Sophia and OTA Fed-Sophia against several baselines for the MNIST dataset using an MLP model.

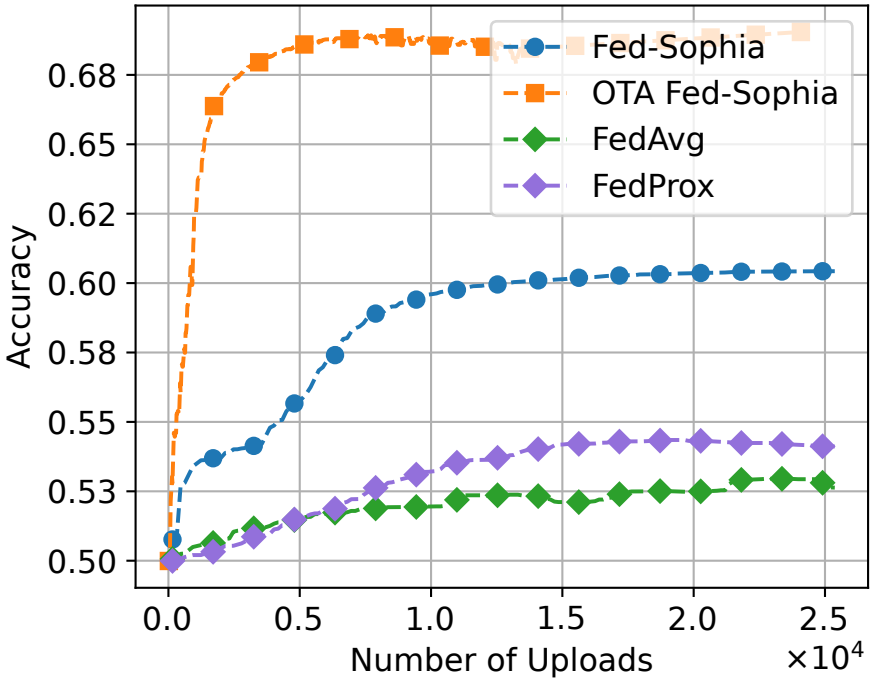

Figure 3.13: Test accuracy as a function of the number of communication uploads for Fed-Sophia and OTA Fed-Sophia against several baselines for the Sent140 dataset using an LSTM model.

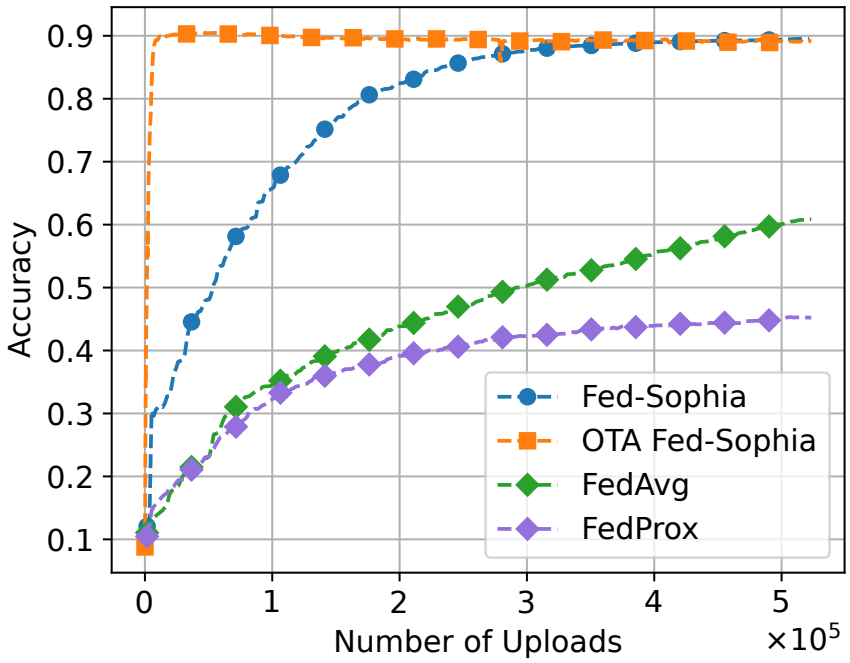

Figure 3.14: Test accuracy as a function of the number of communication uploads for Fed-Sophia and OTA Fed-Sophia against several baselines for the CIFAR10 dataset using a CNN model.

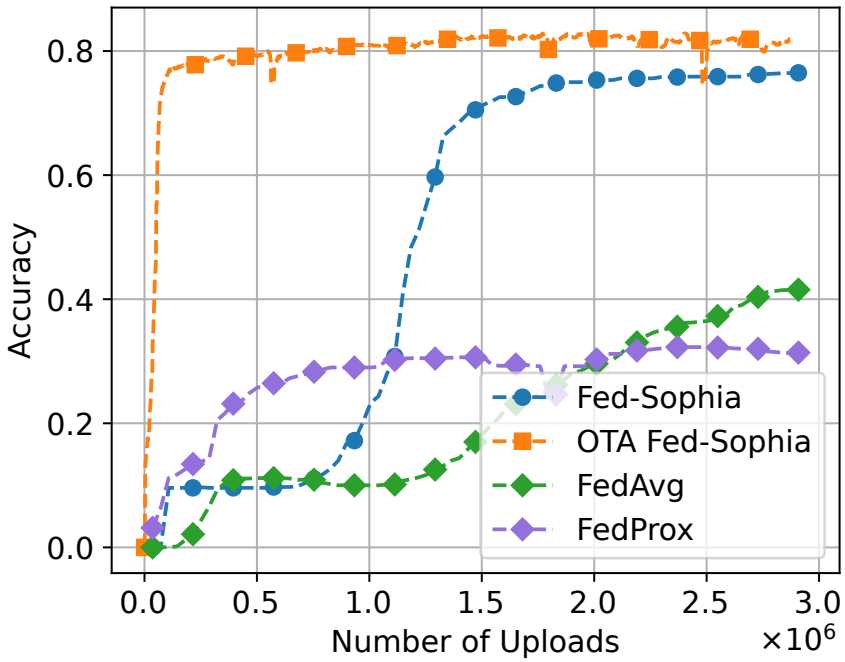

Figure 3.15: Test accuracy as a function of the number of communication uploads for Fed-Sophia and OTA Fed-Sophia against several baselines for the CIFAR100 dataset using a ResNet architecture.

FedProx 86.2%, Fed-Sophia 85.8%, DONE 83.5% and FedAvg 62.1%. OTA Fed-Sophia outperforms all methods for both IID and non-IID data distributions. For FedAvg, in the case of non-IID data distribution, the client updates diverge during local training to minimize communication costs. This leads to model drift, which degrades its performance. By penalizing the deviations form the global mode, FedProx mitigates this issue and stabilizes its performance. By avoiding multiple local updates, OTA Fed Sophia tackles client deviation and through fast convergence and the efficient use of over-the-air aggregation, compensates for the extra communication overhead.

3.4.3.4. Computation time and energy comparison

For the analysis of the computation time and energy expenditure of all the above methods, we calculate the CPU time and emitted carbon footprint required to reach an 80% accuracy for the MNIST dataset using the MLP model. FedAvg requires 7.52 sec of CPU time and produces a carbon footprint (g-CO2-eq) equal to $1.84E-3$. FedProx requires 5.37 sec of CPU time and produces a carbon footprint (g-CO2-eq) equal to $1.31E-3$. DONE requires 3.26 sec of CPU time and produces a carbon footprint (g-CO2-eq) equal to $4.04E-3$. Fed-Sophia requires 1.76 sec of CPU time and produces a carbon footprint (g-CO2-eq) equal to $0.27E-3$. OTA Fed-Sophia requires 0.17 sec of CPU time and produces a carbon footprint (g-CO2-eq) equal to $0.26E-3$. FedAvg and FedProx require more time and energy to achieve the same accuracy as ours. The efficiency of (OTA)Fed-Sophia is due to the GNB estimator, which estimates the diagonal elements of the Hessian using gradients. Our approach reaches the target using 85% less CPU time than DONE and 3 times less than FedAvg/FedProx. FedAvg features the lowest computational complexity, but suffers from low convergence. DONE exhibits high complexity per communication round, but requires fewer rounds than FedAvg.

3.4.4. Conclusions

In the form of OTA Fed-Sophia, we have provided a scalable second-order FL algorithm with OTA aggregation, to achieve savings in communication resources and energy expenditure and preserve the privacy of clients. Using local curvature information, we avoid computing, storing and transmitting the full Hessian matrix. Transmitting updates in digital form introduces communication delays, which become exacerbated for large models. To alleviate this problem, OTA Fed-Sophia leverages the superposition property of wireless channels. It drastically reduces the transmission delays and achieves the target accuracy in one-tenth of the CPU time needed by the digital approach. Another benefit of (OTA) Fed-Sophia is its energy expenditure that is the lowest among all methods, while that of DONE is the largest, due to its complex computations. Overall, Fed-Sophia and OTA Fed-Sophia are the most energy-efficient, consuming only 15% of the energy required by FedAvg, 20% of that of FedProx and 6.7% of that of DONE. They save substantial communication resources compared with the digital version. We validated our methods in numerical simulations. Therefore, we estimate the technology readiness level (TRL) of our solution at 2.

4. EMF reduction via AI-enabled cell-free networking

This chapter presents the work done within Task 4.2.3, "Design of AI-based algorithms for optimizing cell-free networks from a service and EMF viewpoint". In the context of the task, CNR and CNIT have explored both traditional algorithms and ML-based techniques to reduce the EMF exposure in cell-free, next-generation networks. Unlike many current approaches, we have widened our focus to include the *management* of cell-free networks, as opposed to the mere planning thereof. The next sections describe in more detail the work led by CNR on decision-making for human-centric networking (Sec. 4.1) and the work led by CNIT on resource and power allocation in massive MIMO systems (Sec. 4.2).

4.1. Human-centric decision-making in 6G

CNR-led paper [127] addressed the scenario of cell-free networks, where the notion of cells is be replaced by a collection of points-of-access (PoAs), with overlapping coverage areas and/or different technologies. Along with a promise for greater performance and flexibility, this creates further pressure on network management algorithms, which must make joint decisions on (i) PoA-to-user association and (ii) PoA management. We solve this challenging problem through an efficient and effective solution concept making *human-centric* decisions, where pure network performance is balanced against metrics like energy consumption and electromagnetic field exposure, which concern all humans in the network area – including those who are not network users. The performance evaluation, which leverages detailed models for EMF exposure estimation and standard-specified signal propagation models, confirms that our approach outperforms state-of-the-art network management schemes, including those utilizing machine learning, reducing energy consumption by over 80%.

4.1.1. Motivation and State of the Art

The motivation of CNR's work lies in two of the main issues investigated by the CENTRIC project, to wit: the emergence of *cell-free* scenarios, with base stations being replaced by heterogeneous PoAs, and the increased attention to *human-centric* networking.
The combined result of the above trends is a much larger *complexity* of the network management decisions to make, stemming from multiple different sources. There are multiple PoAs to manage, using different technologies and, importantly, frequencies. Furthermore, many technologies support beamforming [128], which multiplies the decisions to make. Finally, each end user can be served through multiple PoAs. All decisions impact, at the same time and often in counter-intuitive ways, both the network performance and the associated energy consumption and EMF exposure. Many traditional network management schemes fail to consider all the above factors; at the same time, the complexity of the problem to solve rules out general-purpose optimization approaches.
We fill this gap by proposing *Cluster-then-Match* (CtM), an efficient algorithm able to swiftly make joint, high-quality decisions about end user-to-PoA assignment and PoA management, including beam width, direction, and power. Unlike existing approaches, CtM:

1. jointly makes decisions for all PoAs, thus, can account for their mutual influence;

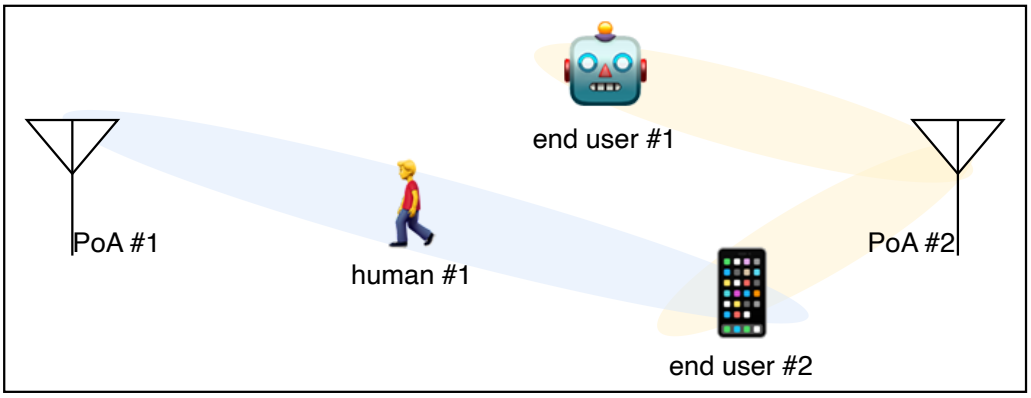

Figure 4.1: A simple indoor cell-free network. Two points-of-access (PoAs) serve their end users through one or more beams; one of the users can be served by two PoAs. A human is irradiated by the beam from PoA #1 to User #2; humans will nonetheless incur EMF exposure whether or not they are users of the network.

2. jointly manages the beam direction, width and power of all beams from all PoAs;

3. jointly accounts for the performance experienced by all end users *and* the resulting energy consumption and EMF exposure, including that incurred by people who are not network users.

To make the aforementioned decisions efficiently and swiftly, CtM uses a two-step approach, predicted on first *clustering* the end users into groups that can be served by the same PoA, and then *matching* the clusters of the end users with the PoAs.
Our main contributions can be summarized as follows:

- we present a concise and comprehensive system model, describing cell-free networks, their entities, the decisions to make therein, their performance, their impact in terms of energy consumption, and the approach used for the estimation of EMF exposure levels;

- we present cluster-then-match (CtM), an efficient approach to make swift, high-quality network management decisions;

- we combine our model and the CtM approach with the standardized channel model [129];

- we compare the performance of CtM against state-of-the art approaches that only focus on performance, showing how CtM matches their performance while incurring much lower energy consumption and EMF exposure.

4.1.2. Proposed Solution

In the following, we first describe how to model cell-free, human-centric networks; then, we describe our CtM solution.

4.1.2.1. System model

Our system model aims at representing the main elements of cell-free networks, their features, and the main decisions to make therein.

Model elements and parameters. Our system includes the PoAs, represented by elements $p \in \mathcal{P}$, and the end users (either devices such as IoTs, or user terminals), represented by elements $d \in \mathcal{D}$. Each PoA $p \in \mathcal{P}$ is associated with one or more *beams* $b \in \mathcal{B}$. Further, we consider a set of *humans* $h \in \mathcal{H}$, representing people who might incur the EMF exposure resulting from the network. Humans in $\mathcal{H}$ include both users of the network and simple passers-by, as in Fig. 4.1.

End users, humans, and PoAs, are associated with a position and height, described through parameters: $x_{\mathrm{M}}(m)$, $y_{\mathrm{M}}(m)$, and $z_{\mathrm{M}}(m)$ for humans $h \in \mathcal{H}$; $x_{\mathrm{D}}(d)$, $y_{\mathrm{D}}(d)$, and $z_{\mathrm{D}}(d)$ for end users $d \in \mathcal{D}$; $x_{\mathrm{P}}(p)$, $y_{\mathrm{P}}(p)$, and $z_{\mathrm{P}}(p)$ for PoAs $p \in \mathcal{P}$. People with a user terminal (e.g., smartphone, tablet, laptop, wearable) are represented by *both* a end user in $\mathcal{D}$ and a human in $\mathcal{H}$, sharing the same location. IoT devices like robots, on the other hand, are only represented by an element in $\mathcal{D}$.

Concerning PoAs $p \in \mathcal{P}$, they operate at a frequency $f(p)$ (which is known) and feature a set of beams $B(p) \subseteq \mathcal{B}$, each of which must have a minimum width $\omega^{\min}(p)$. Furthermore, we indicate with $\pi(b) \in \mathcal{P}$ the PoA originating beam b. If a single PoA can operate at different frequencies, then two distinct elements in $\mathcal{P}$ are created to represent it, sharing the same location.

Decisions and their effects. Our decisions concern the joint aspects of (i) end user-to-PoA assignment, and (ii) PoA and beam management. The former is addressed through binary variables $y(b, d) \in \{0, 1\}$, expressing whether beam $b \in \mathcal{B}$ (hence, PoA $\pi(b) \in \mathcal{P}$) serves device $d \in \mathcal{D}$. Concerning the latter, we have to make four decisions, namely:

- the transmission power $P_{\mathrm{tx}}(p)$ of PoAs $p \in \mathcal{P}$;
- the azimuth angle $\phi(b)$ of beam $b \in \mathcal{B}$;
- its elevation angle $\theta(b)$;
- its width $\omega(b)$.

All the above decisions are then fed as input to the following three functions:

- Rate$(y, P_{\mathrm{tx}}, \phi, \theta, \omega, d)$, computing the data rate obtained by each end user $d \in \mathcal{D}$;
- Energy$(y, P_{\mathrm{tx}}, \phi, \theta, \omega)$, computing the total energy consumption;
- Exposure$(y, P_{\mathrm{tx}}, \phi, \theta, \omega, h)$, estimating the EMF exposure levels incurred by each human $h \in \mathcal{H}$, quantified through any relevant metric.

These functions are then combined to express the objective we seek to optimize and the system constraints. From the viewpoint of our scheme, they are considered to be given and known. It is important however to remark how rate, energy, and exposure can be computed through different techniques – and, in the case of exposure, even be quantified through different metrics.

As an example, the received power can be determined through simple path-loss formulas or through much more complex channel models, accounting for clutter and mobility. Similarly,

EMF exposure can be quantified through metrics that focus on the electromagnetic field at a given location (e.g., electric field strength or power density [130]), or through more complex metrics like the specific absorption rate (SAR), which account for the interaction between electromagnetic fields and biological tissues [131]. Different scenarios and conditions call for different metrics, hence, there is no *right* way to compute the rate, energy, and exposure. Accordingly, our system model and problem formulation – as well as the CtM solution strategy – can accommodate *any* technique to do so, from the simplest to the most realistic ones.

4.1.2.2. Proposed approach

Our CtM heuristic strategy is predicated on (i) sequentially considering the main elements of our system model (end users, PoAs, humans), and (ii) at each step, restricting our attention to the most promising possible decisions. In so doing, CtM can explore a *subset* of the original solution space that, nonetheless, contains most of (often, all) the highest-quality solutions.
Specifically, as summarized in Fig. 4.2, the CtM strategy consists of three main steps:

1. clustering the end users in $\mathcal{D}$, so as to make subsequent decisions on a per-cluster, rather than per-end user, basis;

2. assigning clusters to PoAs in $\mathcal{P}$, thus determining beam steering;

3. optimizing beam width and transmission power levels.

In the following, we describe each of the steps separately.
Step 1: Clustering. A major reason for the complexity of our decisions is the many-to-many relationship between end users in $\mathcal{D}$ and PoAs in $\mathcal{P}$. On the one hand, we need to take this relationship into account to exploit the potential of cell-free networks; on the other hand, considering *all* possible end user-to-PoA associations is unnecessarily complex. Our intuition is to leverage the beams in $\mathcal{B}=\bigcup_{p\in\mathcal{P}} B(p)$, and make the key observations that:

- since there will be more end users than beams, each beam will serve multiple end users, and

- to keep beams as narrow as possible, it is desirable that end users served by the same beam are close to each other.

The latter is motivated by the fact that wider beams result in more interference, as well as higher energy consumption and exposure.
Following the observations above, in Step 1 of the CtM strategy we *cluster* the end users in $\mathcal{D}$ into as many clusters as the number of possible beams in $\mathcal{B}$, exploiting their position information (i.e., the x_{P} and y_{P} parameters) to make clusters as small – in terms of area occupied by their end users – as possible. Although any clustering algorithm can be used in this step, for concreteness we adopt the k-means algorithm [132]. k-means indeed yields very good results in scenarios like ours, also thanks to the fact that (unlike hierarchical clustering and later approaches such as DBSCAN) it takes as an input the target number k of clusters, which corresponds to the number of beams in our case. End users in the same

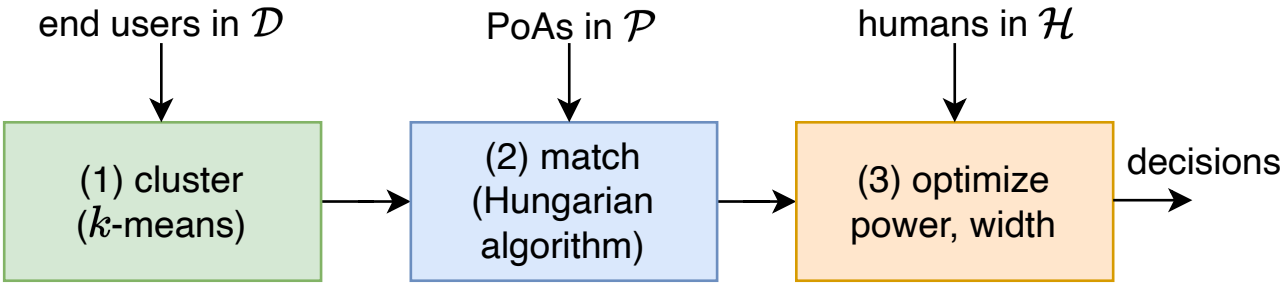

Figure 4.2: The three main steps of the CtM strategy: *clustering* of the end users, cluster-to-PoA *assignment*, *optimizing* beam width and transmission power levels.

cluster can be served either via pairing in multi-user MIMO (MU-MIMO) [133], or via time-division multiplexing (TDM) if MU-MIMO is not used in the current scenario.

Step 2: Matching. We have now as many clusters of end users as the number of beams in $\mathcal{B}$; our next task is to *match* the beams and the clusters they serve. This is a *bipartite matching* problem; importantly, thanks to the clustering performed in Step 1, the matching to perform is one-to-one instead of many-to-many. To efficiently solve to optimality our one-to-one bipartite matching problem, we apply the Hungarian algorithm [134]. The input to the algorithm consists of a square matrix $\mathbf{C}=\{c_{ij}\}$, where element c_{ij} is the *cost* of serving the i-th cluster of devices through the j-th beam; the output is the assignment minimizing the sum of the costs.

As our cost metric, we simply consider the *distance* from the PoA originating the beam to the centroid of the cluster of devices to serve. Making decisions based upon the distance has a twofold advantage:

- shorter distances between PoAs and devices make it possible to use lower power levels and narrower beams;
- using distances (as opposed to, e.g., the achievable rate) does not require any knowledge of the specific implementation of the Energy,Rate, or Exposure functions.

The latter item also implies that Step 2 of CtM is easy and quick to perform even in scenarios and cases where evaluating the rate, energy, or exposure is costly and/or time consuming (e.g., if those quantities are evaluated through simulations).

Step 3: Optimizing beam width and power. In this step, we set the width of each beam and the transmission power of each PoA, keeping into account both the objective and the constraints. Specifically, we set the width of beams to the minimum value necessary to serve the end users assigned to them; further, we reduce their power as much as possible to decrease the energy consumption and the exposure, while ensuring the required performance. Concerning beam widths, let $D(b) \subseteq \mathcal{D}$ be the set of end users belonging to the cluster assigned to beam b. Then the angle $\alpha(b,d)$ from each end user $d \in U(d)$ is given by: $\alpha(b,d) = \arctan \frac{y_{\mathrm{P}}(\pi(b)) - y_{\mathrm{D}}(d)}{x_{\mathrm{P}}(\pi(b)) - x_{\mathrm{D}}(d)}$, and the width of the beam is set to the difference between the maximum and minimum of said angles, i.e.,

$$\omega(b) \leftarrow \max\left\{\omega^{\min}(\pi(b)), \max_{d \in D(b)} \alpha(b,d) - \min_{d \in D(b)} \alpha(b,d)\right\}.$$

Table 4.1: Simulation parameters

Parameter	**Value**
Scenario size	80×20 m^2
No. of PoAs	8
Frequency	3 GHz (PoAs 1–2) 5 GHz (PoAs 3–8)
Bandwidth	20 MHz
No. of end users	100
No. of humans	200
Required rate	100 Mbit/s
SAR_{wb} limit	80 mW/kg [136]

Notice how the above expression also accounts for the minimum beam width $\omega^{\min}$, reflecting the fact that there are technological limits – specific to each PoA – preventing beams from being as narrow as one might desire.
The last part of Step 3 concerns power levels. Our key observations are as follows:

- reducing power might endanger feasibility (by resulting in a violation of the rate constraint), but will never affect adversely energy consumption or EMF exposure;
- reducing the power of a PoA will not impact feasibility for end users served by another PoA.

Accordingly, we process one PoA at a time, as follows:

(a) set an amount of power δ;

(b) consider one of the PoAs $p \in \mathcal{P}$;

(c) reduce the power $P_{tx}(p)$ by δ;

(d) repeat Step (c) so long as that results in a feasible solution;

(e) go back to Step (b) and consider a different PoA.

The above procedure can be repeated, in a Newton-like fashion, setting a smaller δ and starting afresh from Step (a). In so doing, we can get even more fine-grained (hence, higher quality) decisions, at the cost of a longer running time.

4.1.3. Performance Evaluation

We compare our CtM strategy against a benchmark called MaxRate, which uses simulated annealing [135] to find a solution that maximizes the minimum among the data rates achieved by devices. The main parameters of our reference scenario are summarized in Tab. 4.1
We begin by investigating the most basic aspect of CtM's and MaxRate's performance, i.e., the incurred energy consumption. Concerning this critical aspect, Fig. 4.3(left) shows that CtM outperforms MaxRate, consuming almost an order of magnitude less power. Groups of

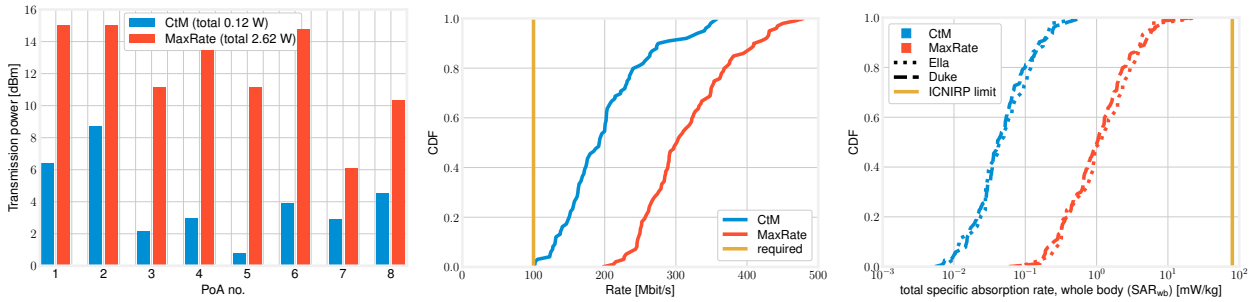

Figure 4.3: CtM and the MaxRate benchmark: transmitted power (left), distribution of data rates (center), and SAR_{wb} values (right).

bars in the plot correspond to different PoAs, and confirm that CtM can reduce the transmission power of all PoAs compared to MaxRate. Also notice how, for both strategies, PoAs 1 and 2, operating at 3 GHz and typically serving farther-away end users, get assigned higher values of transmission power than the other PoAs that operate at 5 GHz.

Striking as Fig. 4.3(left) is, one might rightfully wonder whether such a reduced energy consumption comes at a cost in terms of data rate and service quality. The answer, as shown in Fig. 4.3(center), is both positive and negative. Indeed, under the CtM strategy (blue line in the plot) end users get lower data rates than under MaxRate (red line in the plot); however, such data rates are always (and sometimes significantly) above the required level (yellow line in the plot). This is consistent with the way data rates are included in our problem: so long as all end users are guaranteed $\text{Rate}^{\min}(d)$, there is no reason to further increase data rates. At a more general level, Fig. 4.3(left) and Fig. 4.3(center) suggest how adopting a human-centric approach, hence, using data rate performance as a constraint and not an objective, can bring substantial energy savings without jeopardizing service requirements.

In Fig. 4.3(right), we move to the other human-centric metric we consider, i.e., EMF exposure as quantified through SAR_{wb}. Once can immediately see that CtM yields a much lower SAR_{wb} than MaxRate; further, SAR_{wb} levels for either strategies are significantly below the limit values recommended in [136], to wit, 80 mW/kg. It is perhaps even more interesting to remark how there are *two* lines in the plot for each strategy, one dashed and one dotted, corresponding (resp.) to the Ella and Duke models. This further confirms that our approach can account for individual characteristics when assessing SAR_{wb}.

We now check whether there is any clear space pattern in the distribution of rate by plotting, in Fig. 4.4, the location of each end user and the rate they get under the MaxRate and CtM strategies. Consistently with Fig. 4.3(center), MaxRate ensures to virtually all end users a data rate that is much higher than required, hence, all markers in the left-hand side map are deep blue. In the right map, instead, we can see many lighter markers, though no red ones – highlighting the fact that CtM yields rate values that are closer to, but above, the required one. As one might expect, nodes with lower rate tend to be farther away from the PoA serving them, hence, experience higher attenuation and/or interference.

Similarly, Fig. 4.5 presents the SAR_{wb} levels experienced by humans in $\mathcal{H}$, under the MaxRate and CtM solutions. As we can expect from Fig. 4.3(right), CtM results in uniformly low SAR_{wb} values, hence, dark green markers. MaxRate, on the other hand, results in higher SAR_{wb} levels, hence, slightly lighter markers. As per Fig. 4.3(right), however, all SAR_{wb} values are significantly lower than the limit, thus, there are no purple markers on the plot. Also notice

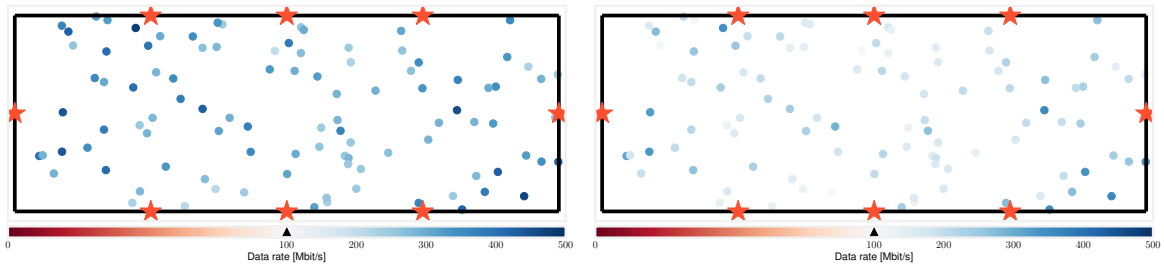

Figure 4.4: Data rate experienced by different end users under the MaxRate (left) and CtM (right) strategies. The black marker on the color bar corresponds to the required rate. Red stars represent PoAs.

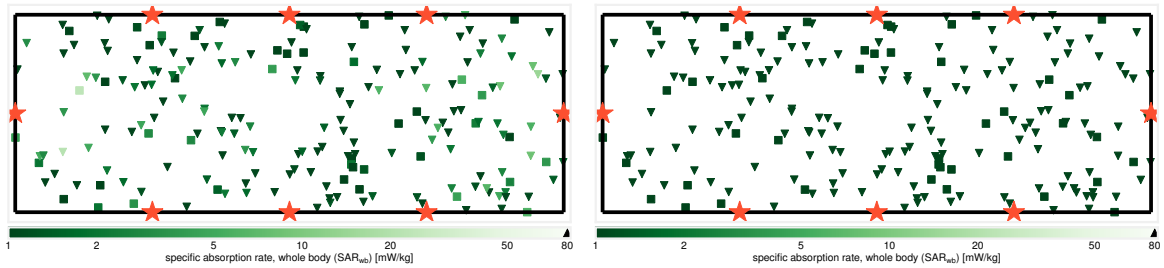

Figure 4.5: SAR_{wb} experienced by different humans under the MaxRate (left) and CtM (right) strategies. The black marker on the color bar corresponds to the ICNIRP limit. Square and triangle markers correspond (resp.) to humans associated with the Duke and Ella models. Red stars represent PoAs.

how the shape of the marker identifies the model (Ella or Duke) used to determine the SAR_{wb} of that specific human.

4.1.4. Conclusions

We addressed cell-free networks where devices can be served by multiple PoAs, each with different features and capabilities. In this context, we have formulated the problem of human-centric network management with the objective to minimize energy consumption subject to data rate requirements and EMF limits. We proposed an efficient and effective heuristic called cluster-then-match (CtM), predicated on first clustering the end users, and then assigning one beam from a PoA to each cluster. CtM considers the data rate requirements of all end users and the EMF exposure incurred by all humans in the topology, including humans who are not users of the network but are still exposed to the wireless signals. Our results show that solutions with a lower energy consumption tend to yield lower EMF exposure, and that CtM can decrease energy consumption by over 80% with respect to solutions that maximize the data rate.
Current research will deal with additional decision-making strategies to complement CtM. Likely candidates include Markov decision processes and deep reinforcement learning. At the same time, further channel models and EMF exposure estimation approaches can be considered.

4.2. EMF-compliant resource allocation in CF-mMIMO systems

CNIT researchers have produced paper [137], which has been presented at the 2024 IEEE WCNC Conference, held in April 2024. The paper addresses the suitability of cell-free massive MIMO (CF-mMIMO) deployments to provide EMF-compliant wireless coverage. Indeed, the impressive growth of wireless data networks has recently led to increased attention to the issue of electromagnetic pollution, as witnessed also by the activities carried out within WP1 of this project. Specific absorption rates and incident power densities have become popular indicators for measuring EMF exposure. The paper [137] thus tackles the problem of power control in user-centric CF-mMIMO systems under EMF constraints. Specifically, the power allocation maximizing the minimum data rate across users is derived for both the uplink (UL) and the downlink (DL) under EMF constraints. The developed solution is also applied to a cellular mMIMO system and compared to other benchmark strategies. Simulation results prove that EMF safety restrictions can be easily met without jeopardizing the minimum data rate, the cell-free mMIMO architecture outperforms the multi-cell mMIMO deployment, and the proposed power control strategy greatly improves the system's fairness.

Current activities are focused on the AI-based implementation of the resource allocation from [137], whose model-based solution is used as a database to train the ML algorithms. Two architectures are considered: classical feed-forward deep neural networks (DNNs) and deep unfolding. Both data-driven approaches open the door to real-time implementations (provided that enough resources are available for offline learning). This means that, in practice, they could easily replace the theoretical counterpart from [137], which fails to enable (feasible) online updates due to its prohibitive complexity. The outcome of this research is to be submitted to the Open Journal of the Communications Society.

4.2.1. Motivation and State of the Art

Given the impressive growth of wireless communications, EMF-related health risks are becoming a growing concern worldwide [138]. To comply with the increasing demand for broadband services, denser deployments and higher frequencies are mandatory [139], which calls for the need to impose EMF exposure constraints in the design of wireless networks [140].

According to the most recent studies reported by the International Commission on Non-Ionizing Radiation Protection (ICNIRP) and the United States Federal Communications Commission (FCC), several criteria can be established to limit user radiation to avoid any alleged medical condition (cf. [141]). This radiation is generally modeled through (i) SAR and (ii) IPD [142]. All these metrics capture the characteristics of the propagation environment and measure the EMF radiation perceived (i) by the human body (or certain parts) in W/kg and (ii) over a specific coverage area in W/m^2. For short distances (less than 20 cm) and low frequencies (typically below 6 GHz), deeper skin penetrations are experienced. That is why SAR usually dominates the radiation exposure in the UL [143]. Contrarily, IPD restrictions are commonly applied in DL transmissions, where the absorption is mostly superficial [144].

One of the key 5G wireless technologies has been the usage of base stations (BSs) equipped with a large number of transmitting antennas, generally referred to as *massive multiple-input-multiple-output* (mMIMO) [145]. In such systems, edge users suffer from inter-cell interference and poor channel conditions that rapidly degrade performance [146]. Due to such attenuation, the DL contribution to EMF exposure is often omitted [147]. To alleviate the poor performance experienced by cell-edge users, it has been proposed to deploy the antennas in a distributed manner, i.e., through access points (APs), and to design the network in a user-centric fashion [148]. This new paradigm, entitled *cell-free* mMIMO (CF-mMIMO), eliminates cell borders and guarantees good quality of service (QoS) for all users [149]. CF-mMIMO is one of the prominent technologies for future 6G data networks.

In CF-mMIMO, however, DL radiation might no longer be negligible since APs are closer to users (the coverage area is reduced), and the resulting path loss might be small. At the same time, though, less power is needed in the UL (users are served by various APs), and, thus, the perceived exposure can be potentially reduced as well. This trade-off makes the performance of CF-mMIMO under safety constraints unclear and naturally raises the comparison with previous centralized scenarios. Such analysis is still unexplored and will be covered in the sequel, where we configure DL/UL power controls to satisfy QoS and EMF requirements.

Unfortunately, owing to the inherent nature of the signal model in CF architectures, the formulation of these power control policies becomes nonconvex. As discussed below, this intricacy makes the problem only tractable via iterative procedures such as *successive convex optimization* (SCO) [150]. Such methodologies entail an immense complexity that rapidly escalates with the size of the networks, which poses the question of their feasibility for real-time implementations [151, 152]. To overcome this challenge, in the second part of this section, we use machine learning techniques for the network design and reformulate the problem using (fully connected) *deep neural networks* (DNNs) [153]. Accordingly, we employ model-based solutions to generate a comprehensive database for the DNNs. Two different strategies are contemplated: conventional "black-box" training [154] and novel deep unfolding [155, 156]. Both supervised methods will undergo a comparative analysis against their theoretical counterparts, evaluating their performance in terms of QoS and execution time. At the time of writing, only some preliminary results have been obtained for the former that serve as proof of concept. The latter will be presented as a future research line.

4.2.2. Problem Formulation

We consider a user-centric CF-mMIMO deployment with K single-antenna users served by a set of M APs equipped with L antennas and connected to a central processing unit (CPU) through fronthaul links with unlimited capacity. As discussed below, only a subset of N APs might be associated with each user. An example is depicted in Fig. 4.6.

The goal is to design the power control to maximize the minimum data rate in the DL and UL (equivalent to ensuring a certain QoS for all users) under EMF constraints. In both cases, the optimization problem can be formulated as

$$\max_{\mathcal{P}} \min_{k} R_k(\mathcal{P}) \quad \text{s.t.} \quad \mathcal{C}(\mathcal{P}), \tag{4.1}$$

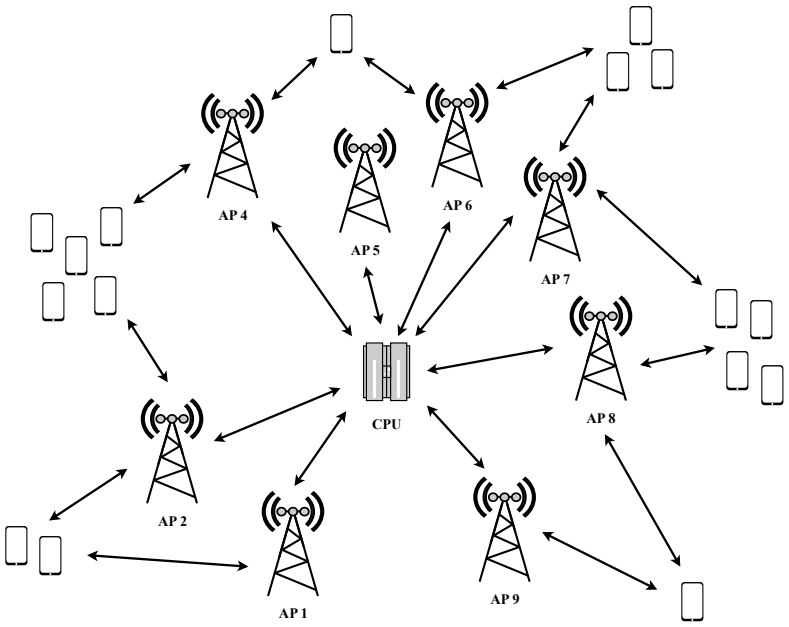

Figure 4.6: CF-mMIMO deployment where $K = 18$ UEs are connected to subsets of $N = 2$ APs, out of a total of $M = 9$, each with $L = 3$ antennas.

where $\mathcal{P}$ is the set of transmit powers, $R_k(\mathcal{P})$ is the data rate, and $\mathcal{C}(\mathcal{P})$ is the constraint set. Problem (4.1) will be solved through model-based and data-driven techniques.

4.2.2.1. Downlink Scenario

In this scenario, $\mathcal{P} = \{p_{k,m}\}$ is the set of AP power coefficients, with $|\mathcal{P}| = KM$. The rate is

$$R_k(\mathcal{P}) = \frac{\tau_d}{\tau_c} B \log_2\left(1 + \gamma_k(\{p_{k,m}\})\right), \tag{4.2}$$

where $\gamma_k(\{p_{k,m}\})$ is the DL signal-to-interference-plus-noise ratio (SINR):

$$\gamma_k = \frac{\left|\sum_{m=1}^{M} a_{k,m}\sqrt{p_{k,m}}\mathbf{h}_{k,m}^{\mathrm{H}}\mathbf{b}_{k,m}\right|^2}{\sum_{j\neq k}\left|\sum_{m=1}^{M} a_{j,m}\sqrt{p_{j,m}}\mathbf{h}_{k,m}^{\mathrm{H}}\mathbf{b}_{j,m}\right|^2 + \sigma_k^2}, \tag{4.3}$$

with $a_{k,m} \in \{0,1\}$ and $\mathbf{h}_{k,m} \in \mathbb{C}^L$ the association and channel between user k and AP m, respectively. In addition, $\mathbf{b}_{k,m} \in \mathbb{C}^L$ is the unit-norm beamforming vector used by AP m to direct the information signal towards user k, and σ_k^2 is the variance of the thermal noise.

Note that, to calculate the rate in (4.2), we assumed a common bandwidth B and a coherence time τ_c that splits into τ_d (DL) and τ_u (UL) such that $\tau_d + \tau_u \leq \tau_c$ [157].

Finally, the constraint set $\mathcal{C}(\mathcal{P})$ can be represented by

$$C1 : p_{k,m} \geq 0 \quad \forall k, m; \quad C2 : \sum_{k=1}^{K} p_{k,m} \leq P_m \quad \forall m; \quad C3 : \xi_k \leq I_k \quad \forall k, \tag{4.4}$$

where $C1$ ensures non-negative coefficients (true by definition), $C2$ introduces the set of power budgets P_m for the different transmitters (i.e., APs), whereas $C3$ limits the radiation perceived at the user side to I_k. Accordingly, ξ_k is the IPD at user k's location, including the contribution of the desired and interfering signals [143, 158]:

$$\xi_k = \frac{4\pi}{(c/\kappa)^2} \sum_{j=1}^{K} \left| \sum_{m=1}^{M} a_{j,m} \sqrt{p_{j,m}} \mathbf{h}_{k,m}^{\mathrm{H}} \mathbf{b}_{j,m} \right|^2, \tag{4.5}$$

with κ the carrier frequency and c the speed of light.

4.2.2.2. Uplink Scenario

Similarly, here $\mathcal{P} = \{q_k\}$ (containing the set of UE transmit powers such that $|\mathcal{P}| = K$),

$$R_k(\mathcal{P}) = \frac{\tau_u}{\tau_c} B \log_2 (1 + \rho_k (\{q_k\})), \tag{4.6}$$

and, thus, $\mathcal{C}(\mathcal{P})$ can be defined with the constraints

$$C1 : 0 \leq q_k \leq Q_k \quad \forall k; \quad C2 : \varepsilon_{k,n} \leq E_{k,n} \quad \forall k, n, \tag{4.7}$$

where $C1$ bounds the support of the powers of all the users between 0 and Q_k, while $C2$ fixes the maximum value of the EMF exposure corresponding to the n-th body part at $E_{k,n}$.

In this case, ρ_k is the UL SINR:

$$\rho_k = \frac{q_k \left| \sum_{m=1}^{M} a_{k,m} \mathbf{f}_{k,m}^{\mathrm{H}} \mathbf{h}_{k,m} \right|^2}{\sum_{j \neq k} q_j \left| \sum_{m=1}^{M} a_{k,m} \mathbf{f}_{k,m}^{\mathrm{H}} \mathbf{h}_{j,m} \right|^2 + \sum_{m=1}^{M} a_{k,m} \eta_m^2 \|\mathbf{f}_{k,m}\|_2^2}, \tag{4.8}$$

and $\varepsilon_{k,n} = b_{k,n} q_k$ is the associated SAR [138, 140], with $b_{k,n}$ the coefficient associated to the n-th body part of user k (e.g., head, chest, etc.).

4.2.3. Proposed Solutions

4.2.3.1. Model-Based Power Control

The optimization problem's convexity varies depending on the type of data transmission, either DL or UL. The following subsections differentiate between these configurations.

4.2.3.1.1 Downlink

In the DL, given the nature of (4.2), we must resort to suboptimal approaches for finding a feasible solution. That is why, in the upcoming subsection, we present an iterative procedure that relies on SCO (successive convex approximation) and leads to a stationary point [150]. More specifically, at the i-th iteration, the power set $\mathcal{P} \equiv \mathcal{P}^{(i)}$ is updated from the previous solution $\mathcal{P}^{(i-1)}$ until convergence is reached. To do so, we first transform the original formulation by moving the objective function to a new constraint and, after that, approximate the resulting set $\mathcal{C}(\mathcal{P})$ with suitable convex functions.

Defining $d_{k,m} \triangleq \sqrt{p_{k,m}}$ as the new design variable, the problem (4.1) is equivalent to (cf. [159])

$$\begin{aligned}
\max_{\{d_{k,m}\},\delta} \quad \delta \quad \text{s.t.} \quad & C1: d_{k,m} \geq 0 \quad \forall k, m \\
& C2: \sum_{k=1}^{K} a_{k,m} d_{k,m}^2 \leq P_m \quad \forall m \\
& C3: \frac{4\pi}{\lambda^2} \sum_{j=1}^{K} \left| \sum_{m=1}^{M} a_{j,m} d_{j,m} \mathbf{h}_{k,m}^{\mathrm{H}} \mathbf{b}_{j,m} \right|^2 \leq I_k \quad \forall k \\
& C4: \underbrace{\delta \left(\sum_{j \neq k} \left| \sum_{m=1}^{M} a_{j,m} d_{j,m} \mathbf{h}_{k,m}^{\mathrm{H}} \mathbf{b}_{j,m} \right|^2 + \sigma_k^2 \right)}_{\triangleq f_k([\mathbf{d}_1, \ldots, \mathbf{d}_{k-1}, \mathbf{d}_{k+1}, \ldots, \mathbf{d}_K])} - \underbrace{\left| \sum_{m=1}^{M} a_{k,m} d_{k,m} \mathbf{h}_{k,m}^{\mathrm{H}} \mathbf{b}_{k,m} \right|^2}_{\triangleq g_k(\mathbf{d}_k)} \leq 0 \quad \forall k,
\end{aligned} \tag{4.9}$$

where we introduced an auxiliary variable δ to transform the non-concave objective function as constraint $C4$, which is still nonconvex. In light of that, we proceed like so.

With the help of SCO, we can find a local optimum. In a nutshell, for a given δ, the optimization will be decomposed into a sequence of subproblems solved iteratively. Each of them needs to have a global optimum for guaranteeing convergence, which means the second term in $C4$ must be approximated by a surrogate function [160]. On top of that, a bisection search can be applied to derive the optimal δ [161].

Among others, a widely employed strategy is to linearize the function $g_k(\mathbf{d}_k)$ so that the constraint $C4$ in (4.9) is convexified. In particular, when applying the first-order Taylor expansions at the previous feasible point, i.e., $\mathbf{d}_k^{(i-1)}$, we can obtain the following lower bound:

$$g_k(\mathbf{d}_k) \geq g_k\left(\mathbf{d}_k^{(i-1)}\right) + 2\left(\mathrm{Re}\left\{\mathbf{c}_k \mathbf{c}_k^{\mathrm{H}}\right\} \mathbf{d}_k^{(i-1)}\right)^{\mathrm{T}} \left(\mathbf{d}_k - \mathbf{d}_k^{(i-1)}\right), \tag{4.10}$$

with $\mathbf{c}_k \triangleq [a_{k,1}\mathbf{h}_{k,1}^{\mathrm{H}}\mathbf{b}_{k,1}, \ldots, a_{k,M}\mathbf{h}_{k,M}^{\mathrm{H}}\mathbf{b}_{k,M}]^{\mathrm{T}}$.

Accordingly, at each iteration, we will end up with a subproblem that is a worst-case scenario (more restrictive constraint) but can be globally solved with standard numerical routines, e.g., CVX [162]. Finally, the whole procedure is repeated until we converge to a local optimum.

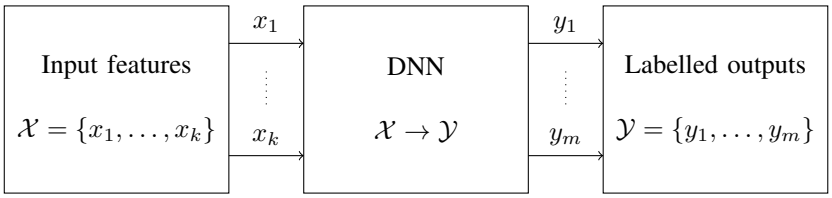

Figure 4.7: Fully-connected DNN scheme. The set of inputs $\mathcal{X}$ is directly mapped onto the set of outputs $\mathcal{Y}$.

4.2.3.1.2 Uplink

Contrarily, the resulting problem is convex in the UL setup. As a result, a globally optimal solution can be found via classic optimization methods [161]. More precisely, following similar steps, the power control becomes

$$\begin{aligned} &\max_{\{q_k\},\delta} \ \delta \\ &\text{s.t.} \quad C1 : 0 \le q_k \le Q_k \quad \forall k; \quad C2 : b_{k,n} q_k \le E_{k,n} \quad \forall k, n \\ &\qquad C3 : \delta \left(\sum_{j \neq k} q_j \left| \sum_{m=1}^{M} a_{k,m} \mathbf{f}_{k,m}^{\mathrm{H}} \mathbf{h}_{j,m} \right|^2 + \sum_{m=1}^{M} a_{k,m} \eta_m^2 \left\| \mathbf{f}_{k,m} \right\|_2^2 \right) - q_k \left| \sum_{m=1}^{M} a_{k,m} \mathbf{f}_{k,m}^{\mathrm{H}} \mathbf{h}_{k,m} \right|^2 \le 0 \quad \forall k. \end{aligned} \tag{4.11}$$

Unlike before, the newly added constraint $C3$ is quasiconvex (lower bound on the logarithm of a linear fraction; thus, quasiconcave). Accordingly, since $C1$ and $C2$ are linear, for every δ, this will result in a quasilinear problem whose (global) optimum can be found. Once again, the optimal value of δ can be obtained via the bisection method [159].

4.2.3.2. Data-Driven Power Control

We aim to derive data-driven implementations to address previous power control optimizations feasibly. Indeed, these approaches will learn from the past theoretical values, a procedure that can be performed offline while keeping the system complexity low for real-time applications [156].

We opt for the popular fully connected DNN approach, where the final solutions $\{p_{k,m}\}$ (DL) and $\{q_k\}$ (UL) of the model-based approach are considered as the labeled outputs $\mathcal{Y}$ for the DNN [163, 164]. The input features $\mathcal{X}$ also differ depending on the scenario, so we dedicate separate subsections for the DL and UL architectures.

A general overview of this strategy is provided in Fig. 4.7, where $k \triangleq |\mathcal{X}|$ and $m \triangleq |\mathcal{Y}|$ correspond to the input and output sizes, respectively. Note that the features are directly mapped onto the power control coefficients, i.e., a single DNN substitutes the entire optimization.

4.2.3.2.1 Downlink

In line with the rationale in [154], we consider the coefficients provided by the fractional power control (FPC) [165] as inputs, i.e., $p_{k,m} = p_m \alpha_{k,m}^{\kappa}$, where $\alpha_{k,m}$ is the large-scale fading (LSF) coefficient between AP m and UE k, κ is the parameter to tune the power distribution:

- $\kappa = 1$ prioritizes good channel qualities, also known as proportional power control,
- $\kappa = 0$ provides equal service in the network, i.e., uniform power control (UPC),
- $\kappa = -1$ resembles a fairer policy that helps links with poor propagation,

and p_m scales the resulting coefficients to satisfy the power constraints, i.e.,

$$p_m = \frac{P_m}{\sum_{k=1}^{K} a_{k,m}\alpha_{k,m}^{\kappa}}. \tag{4.12}$$

This way, the number of inputs and outputs is the same $|\mathcal{X}| = |\mathcal{Y}| = MK$ coefficients.

Regarding the structure of the DNN, we employ five layers of sizes 1024, 4096, 2048, 1024, and MK. The first two layers use linear and ELU activation functions, the last layer applies ReLU, and tanh is introduced for the rest. Such choices are found by trial and error.

The DNNs are trained following the MSE loss between the model-based solution and the network's output with l_2 regularization and the Adam optimizer. The learning rate is first set to 10^{-3} for the initial epochs to ensure convergence and later decreased by 0.1 to fine-tune.

4.2.3.2.2 Uplink

Likewise, in the UL, we consider as K input features the equivalent FPC coefficients:

$$q_k = Q_k \frac{\left(\sum_{m=1}^{M} a_{k,m}\alpha_{k,m}\right)^{\varkappa}}{\max_j \left(\sum_{m=1}^{M} a_{j,m}\alpha_{j,m}\right)^{\varkappa}}, \tag{4.13}$$

where $\varkappa$ is defined as κ. In this case, the structure of the DNN follows that in the DL but with fewer units per hidden layer: 32, 256, 128, 64, 32. Training is also performed similarly.

4.2.4. Performance Evaluation

Numerical simulations should compare the proposed setting with the following two baseline schemes: (a) a CF-mMIMO system optimized according to problem (4.1) without constraint $C3$ (DL) or $C2$ (UL); this permits understanding the impact, on the system's performance, of the EMF constraint; and (b) a multi-cell mMIMO (MC-mMIMO) system where each user is connected to only one macro BS; in this case, we will be able to evaluate the possible advantages of the CF-mMIMO deployment in fulfilling EMF constraints.

For fairness, the cellular setup will comprise L BSs with M antennas. This is illustrated in Fig. 4.7, where the numerology is consistent with Fig. 4.6.

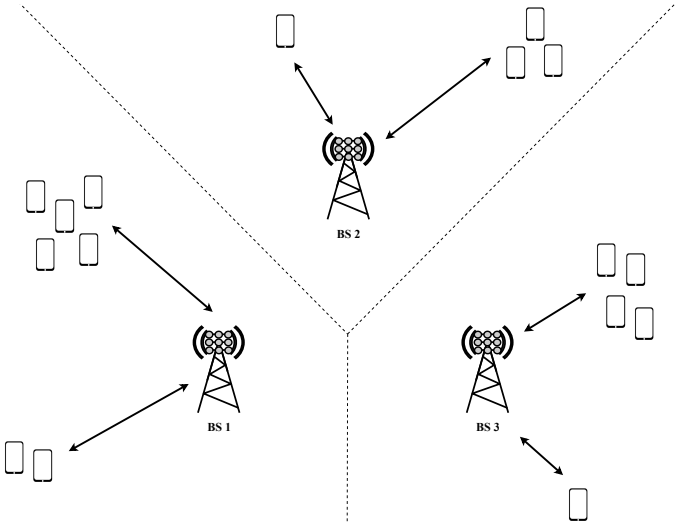

Figure 4.8: Illustrative example of a MC-mMIMO setup with K=14 users and L=3 multi-antenna BSs. Unlike cell-free, users are connected to only one BS (dashed lines indicate cell borders).

4.2.4.1. Channel Propagation

The link between user m and AP k is [157, (1)]

$$\mathbf{h}_{k,m} = \sqrt{\frac{\alpha_{k,m}}{1+\beta_{k,m}}} \left(\bar{\mathbf{h}}_{k,m} + \sqrt{\beta_{k,m}} e^{j\psi_{k,m}} \mathbf{v}_m(\theta_{k,m}) \right), \tag{4.14}$$

where $\alpha_{k,m}$ is the LSF coefficient including the path loss, $\beta_{k,m}$ is the Rician factor [166, Table B.1.2.1-2], $[\bar{\mathbf{h}}_{k,m}]_l \sim \mathcal{CN}(0,1)$ are the uncorrelated Rayleigh distributed non-line-of-sight (NLoS) components, $\psi_{k,m} \sim \mathcal{U}[0, 2\pi]$ is the phase offset, $\mathbf{v}_m(\cdot) \in \mathbb{C}^L$ is the steering vector (generated according to a uniform linear array), and $\theta_{k,m}$ is the (LoS) angle of arrival.

Regarding the channels between users and macro BSs, we adopt a Rician model. However, to avoid redundancy, their derivation is omitted.

4.2.4.2. Channel Estimation

Perfect channel state information might be an unrealistic assumption in practical systems. Instead, we must acquire this knowledge locally at the APs through UL orthogonal pilots. This will allow us to characterize the sufficient statistics of the channels [161]. More precisely, the linear minimum mean-squared error estimates $\hat{\mathbf{h}}_{k,m}$ can be constructed based on UL training sequences. For more details, please refer to [167, Subsection II-C].

4.2.4.3. System Parameters

Throughout all experiments, we consider a deployment area of 1 km^2, wrapped around the edges to avoid boundary effects. The scenario follows the micro-urban configuration described in [166] with $P_m = P = 23$ dBm $\forall m$, $Q_k = \mu_k = 20$ dBm $\forall k$, $\sigma_k^2 = \eta_m^2 = N_o B$ $\forall k, m$, $N_o = -174$ dBm/Hz, and $B = 20$ MHz. This means $(M/L)P$ will be the power budget of each BS. Besides, we set $N = 5$ for the user-AP association.

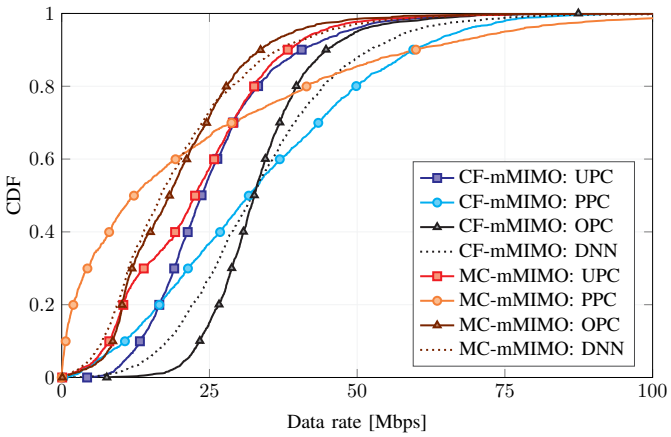

Figure 4.9: DL rate in (a) CF-mMIMO and (b) MC-mMIMO with UPC, PPC, OPC, and DNN.

In line with [141], we focus on whole-body EMF constraints applied to the general public: $I_k = 10$ W/m^2 $\forall k$ for the IPD limit and $E_{k,n} = 0.08$ W/kg $\forall k, n$ for the SAR (single) metric, with coefficient $b_{k,n} = 8$ kg^{-1} $\forall k, n$.

Finally, we assume a coherence time and bandwidth of 1 ms and 200 kHz, respectively [157]. Therefore, $\tau_c = 200$ symbols or (time-frequency samples) are available for communication. Accordingly, the first $\tau_p = K/2$ symbols will be dedicated to channel estimation, i.e., two users will be sharing the same pilots, and $\tau_d = \tau_u = (\tau_c - \tau_p)/2$ samples will be assigned to both DL and UL phases.

4.2.4.4. Downlink Results

Along with the optimal power control (OPC), we include uniform power control (UPC) and FPC with $\kappa = 1$, i.e., proportional power control (PPC), as benchmark schemes [165]. This will help to emphasize the performance of our proposal. Recall that although all designs are based on CSI uncertainties, they are ultimately applied to the true channels.

The rate CDF for $K = 20$, $L = 4$, and $M = 40$ is depicted in Fig. 4.9. As expected, CF-mMIMO outperforms the multi-cell approach in all cases, especially for unlucky users. In addition, our power control mechanism also ensures a higher QoS than the other two strategies. This improvement is more notable in the cell-free deployment: around 50% and 80% of the users w.r.t. PPC and UPC, respectively. Besides, one can see that the performance of the DNN-based solution approaches that of the OPC, meaning it could safely substitute that solution.

The CDF of the radiation per user (measured as IPD) is presented in Fig. 4.10. Although larger distances in the MC-mMIMO lead to lower values, we are still far from the 10 W/m^2 limit, and the OPC generally yields smaller IPDs. Therefore, the EMF limitation does not seem detrimental to the QoS for this particular setting. This is due to the high propagation losses in the DL, which again stresses the safety of both mMIMO architectures. Remarkably, the data-driven solution yields values even closer to those of the model-based allocation.

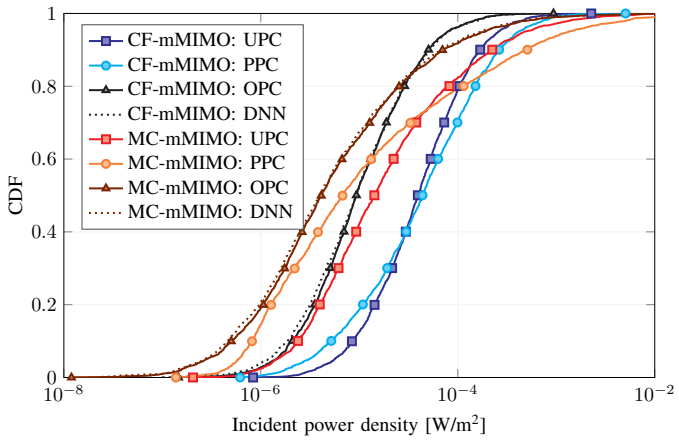

Figure 4.10: IPD in (a) CF-mMIMO and (b) MC-mMIMO with UPC, PPC, OPC, and DNN.

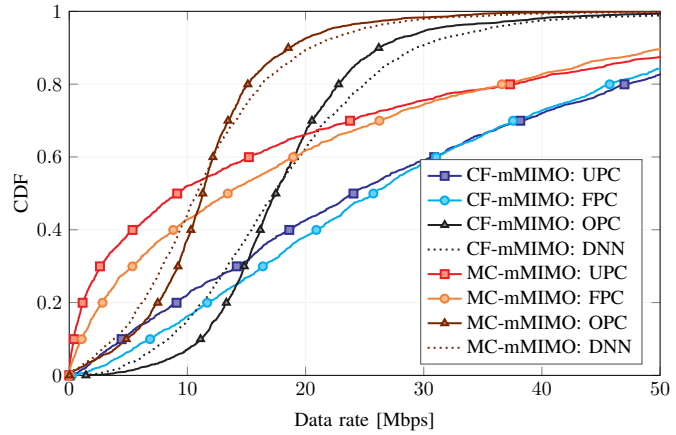

Figure 4.11: UL rate in (a) CF-mMIMO and (b) MC-mMIMO with UPC, FPC, OPC, and DNN.

4.2.4.5. Uplink Results

The CDFs of the data rate and the perceived EMF (in terms of SAR) for $K = 20$, $L = 4$, and $M = 40$ are depicted in Figs. 4.11 and 4.12, respectively. Here, FPC is used with $\varkappa = -0.5$. Once more, the cell-free solution provides larger rates than its centralized counterpart, and the OPC outperforms the rest of the power control mechanisms. Likewise, the DNN provides similar results, especially in terms of radiation. However, unlike before, SAR values are not far away from the ultimate 0.08 W/kg limit, and a similar exposure is perceived in both mMIMO scenarios. This highlights the interest in EMF analysis for the UL.

4.2.5. Conclusions

We designed the power control scheme to boost QoS under EMF constraints in a user-centric CF-mMIMO network. Based on model-based solutions, data-driven approaches have been implemented. The radiation in the DL and UL has been modeled via IPD and SAR metrics, respectively. Numerical results show that cell-free can outperform the MC-mMIMO architecture and that, although the UL is more sensitive to EMF exposure, the impact is less prominent in the DL. Finally, it is also proven that the proposed DNNs can suitably replace the theoretical methodologies, which entail a prohibitive complexity for real-time applications.

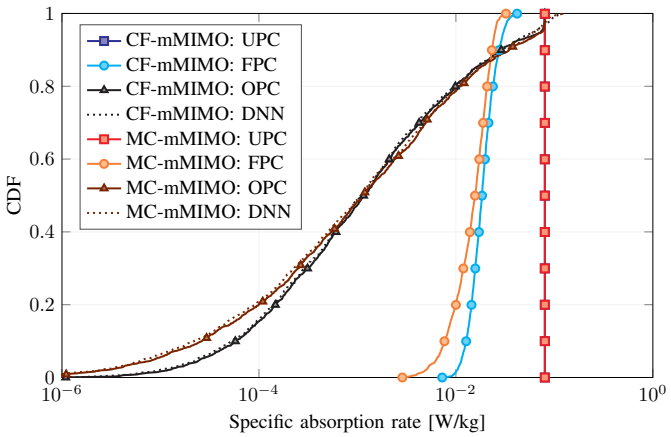

Figure 4.12: SAR in (a) CF-mMIMO and (b) MC-mMIMO with UPC, FPC, OPC, and DNN.

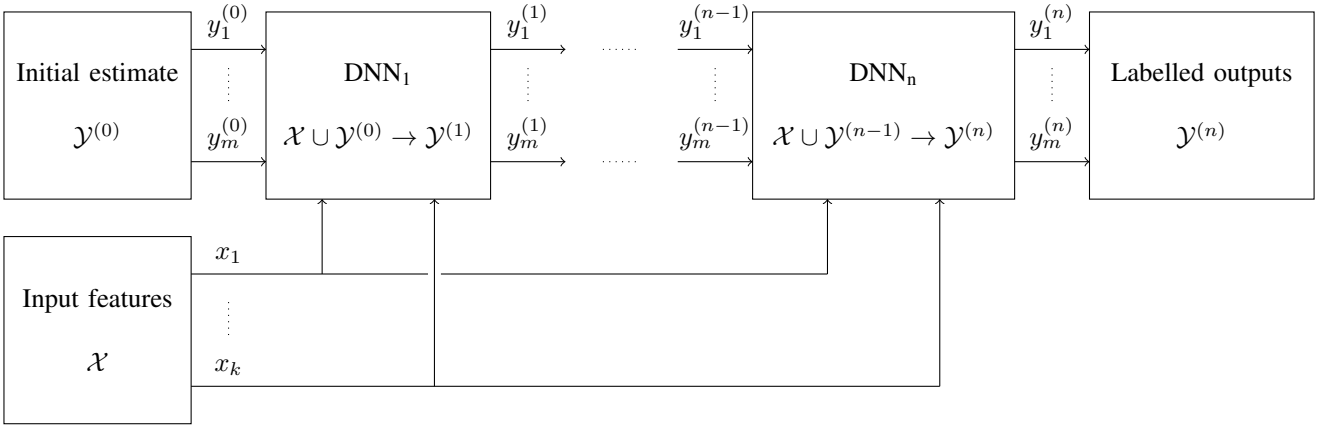

Figure 4.13: Deep unfolding scheme. The inputs $\mathcal{X}$ are mapped onto the outputs $\mathcal{Y} \equiv \mathcal{Y}^{(n)}$ through n DNNs. Each DNN models each iteration of the SCO.

On the other hand, deep unfolding could be applied in the DL setting to further reduce the system's complexity. Briefly, given the inherent sequential nature of the SCO procedure, we could mimic the structure of the model-based algorithm by replacing the iterations with small DNNs [155]. In other words, expert knowledge would be incorporated into the design, leading to higher practical feasibility and further simplifying computational resources. We then would address each subproblem in (4.9) via separate concatenated DNNs. Accordingly, the i-th DNN would take as input the initial FPC values $\mathcal{X}$ together with the previous power coefficients $\mathcal{Y}^{(i-1)}$. The working principle is shown in Fig. 4.13, with n the number of iterations.

Unfortunately, the proposed bisection search cannot be directly modeled through deep unfolding, given that feasibility problems are ill-defined in DNN architectures [168]. To circumvent that, an alternative is to approximate the minimum function by the log-sum-exp [169, Subsection VI-B]. This way, the method for finding a feasible solution could be unfolded.

As mentioned before, all this serves as a guideline for the object of study in future works. Deep unfolding is merely presented here as a possible extension of the present task.

5. Conclusions

This deliverable presents the research activities performed within the CENTRIC T4.2 aimed at the application of intelligent optimization techniques for energy efficiency, performance improvement, and human safety assurance in next-generation wireless communication systems. By addressing the complex intersection of machine learning, wireless resource management, and societal demands such as environmental sustainability and public health, this document offers a multifaceted view of how wireless networks can evolve in a sustainable manner.

Key aspects investigated and leveraged within T4.2 are the principle of intelligence at the edge and the use of AI/ML approaches as a suite of tools tailored to specific constraints and objectives to be achieved at the radio interface. In Chapter 2, this was exemplified by the introduction of an AI-based model predictive control, which emphasizes the need for dependable AI predictions in wireless networking. To enhance reliability, the activity applied conformal prediction – a post-hoc calibration technique – to pre-trained probabilistic forecasters, enabling the implementation of safe model-predictive power control policies. It is worth remarking that this solution substantially increases the effectiveness of power allocation and RRM and makes the system scalable, establishing the best trade-off between throughput performance and computational efficiency.

In Chapter 3, the focus shifted to the sustainability of AI/ML at the network edge. Notably, given the increasing energy and computational demands of DNN models, we proposed a solution for model caching, task offloading, and execution of inference tasks. The OffloaDNN framework we developed introduces an optimal reuse of blocks of DNN layers cached at the edge of the network and a pruning strategy that dramatically reduce memory and compute demands without sacrificing model performance. This is then combined with an intelligent selection of tasks to be offloaded from mobile devices to the network edge, so as to control the computing load at the edge and data load on the radio link. This contribution is especially relevant, given the growing integration of AI/ML services into the edge infrastructure and the associated energy footprint. When compared to the state of the art, the proposed ML model caching and intelligent strategies for collaborative training and inference can substantially reduce not only the communication overhead (by 25%), but also memory usage and computing time (by 80% and 77%, resp.). Additionally, they can dramatically decrease energy consumption, requiring just 6.7%–20% of the energy consumed by existing paradigms.

Chapter 4 brings to the forefront the issue of EMF exposure, an often-overlooked aspect in the design of radio networks, which instead deserves more and more attention given the increasing densification of wireless networks. The cell-free networking paradigm has been investigated through a human-centric lens, where the goal is not just coverage and high throughput but also minimizing radiation exposure to users. The proposed solutions leverage on the one hand clustering and matching methods, and on the other hand model-based optimization and machine learning. Our approach significantly outperforms conventional ones in terms of energy consumption and EMF exposure, achieving up to 80% savings. Furthermore, our solutions ensure that network performance metrics such as data rate requirements are still met, effectively striking a balance between user QoS and societal health concerns. It is also important to remark that our study has taken advantage of advanced computational models, both statistical and deterministic, of human body EMF exposure, and

that we have adopted specific metrics such as electric field strength and dosimetric quantities. Furthermore, it is worth remarking that each of the proposed techniques is not only based on optimization or learning theoretical methods but was also validated through simulations using realistic radio environments. This enabled a sound validation of the created solutions, indicating a clear path from concept to implementation.

While the developed activities achieved the goals set for T4.2 – in a nutshell, the realization of intelligent, energy efficient, and human-centric radio interfaces and network infrastructure, they also opened relevant directions for future research. Specifically, the calibration methods proposed in Section 2.2 can be extended to tackle key challenges in wireless networks, such as non-stationary data distributions, communication noise, and decentralized data processing. Addressing these issues will be essential for the widespread adoption of AI-driven techniques and the development of safe AI-native networks.

The findings introduced in Section 2.1 underscore the potential of modified RL algorithms in efficiently scheduling massive radio resources, paving the way for more sophisticated and resource-aware next-generation cellular networks. With advancements in RL algorithms and the architecture of the deep scheduler, there are promising research directions to enhance the practicality of MAC layer scheduling in terms of both performance and computational latency. In terms of performance, it is essential to meet specific QoS constraints in a product. This necessitates an improvement in the algorithmic design for the RL agent to ensure a certain performance metrics are met during the inference phase. Additionally, to address the latency on scheduling, a potential approach is to eliminate the loop over the spatial layers. This would enable the RL agent to make scheduling decisions for both FD scheduling and user pairing in a single inference call.

The solution created for ML models caching and execution at the mobile edge in Section 3.1 can be exploited for sensor fusion and processing of multi-modal data – an application that is gaining increasing attention. Notably, this domain highlights gaps in current research, offering new opportunities for optimization in terms of both energy consumption and quality of service. Distributed sensor fusion and multi-data processing can be provided through dynamic neural networks that operate according to context information and semantic hierarchical communication. New ML architectures can be effectively exploited, jointly with novel communication paradigms that better protect the transmission of data that are more relevant to the inference task at hand. Innovative orchestration paradigms can thus be developed to jointly act on the dynamic architecture of neural networks and the use of radio resources, for efficient computing and communication.

As making ML sustainable has become imperative – recently even exacerbated by the increasing popularity of LLMs –, the idea presented in Section 3.2 should also be expanded. Notably, the amount of data collected and processed by ML models should be minimized while still ensuring high quality levels of learning and inference output. Ways to do so when LLMs are applied require additional studies and effort, while accounting for the heterogeneity of the data and goals that the various entities and users may have.

Finally, the metrics we used in Section 4.1 for assessing human-exposure to EMF represent a valuable reference for the investigation of this important KVI in the design of next-generation systems. As for the deep unfolding method in Section 4.2, this could be applied in the DL setting to further reduce the system complexity. Given the SCO procedure's inherent sequential nature, we could mimic the structure of the model-based algorithm by replac-

ing the iterations with small DNNs. In other words, expert knowledge can be incorporated into the design, leading to higher practical feasibility and further simplifying computational resources.

6. References

[1] K. Stankeviciute, A. M Alaa, and M. van der Schaar, "Conformal time-series forecasting," *Advances in neural information processing systems*, vol. 34, pp. 6216–6228, 2021.

[2] A. N. Angelopoulos, S. Bates, A. Fisch, L. Lei, and T. Schuster, "Conformal risk control," *arXiv preprint arXiv:2208.02814*, 2022.

[3] D. Anguita, A. Ghio, L. Oneto, X. Parra, J. L. Reyes-Ortiz, *et al.*, "A public domain dataset for human activity recognition using smartphones," in *ESANN*, 2013.

[4] R. Hashemi, V. Ranasinghe, T. Veijalainen, P. Kela, and R. Wichman, "User throughout optimization via deep reinforcement learning for beam switching in mmwave radio access networks," in *2024 IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC Workshops), to be published*, 2024.

[5] P. Kela, J. Puttonen, N. Kolehmainen, T. Ristaniemi, T. Henttonen, and M. Moisio, "Dynamic packet scheduling performance in UTRA long term evolution downlink," in *International Symposium on Wireless Pervasive Computing*, pp. 308–313, 2008.

[6] J. Fan, G. Y. Li, and X. Zhu, "Multiuser MIMO scheduling for LTE-A downlink cellular networks," in *IEEE 79th Vehicular Technology Conference (VTC Spring)*, pp. 1–5, 2014.

[7] M. Seguin, A. Omer, M. Koosha, F. Malandra, and N. Mastronarde, "Deep Reinforcement Learning for Downlink Scheduling in 5G and Beyond Networks: A Review," in *IEEE 34th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pp. 1–6, 2023.

[8] F. Al-Tam, N. Correia, and J. Rodriguez, "Learn to Schedule (LEASCH): A Deep Reinforcement Learning Approach for Radio Resource Scheduling in the 5G MAC Layer," *IEEE Access*, vol. 8, pp. 108088–108101, 2020.

[9] Q. An, S. Segarra, C. Dick, A. Sabharwal, and R. Doost-Mohammady, "A deep reinforcement learning-based resource scheduler for massive mimo networks," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 1, pp. 242–257, 2023.

[10] "3GPP technical specification 38.101: User equipment (UE) radio transmission and reception."

[11] A. Giovanidis, M. Leconte, S. Aroua, T. Kvernvik, and D. Sandberg, "Online frequency scheduling by learning parallel actions," *arXiv: 2406.05041*, 2024.

[12] D. Sandberg, T. Kvernvik, and F. D. Calabrese, "Learning robust scheduling with search and attention," in *IEEE International Conference on Communications*, pp. 1549–1555, 2022.

[13] M. Roshdi, E. Swistak, R. German, and M. Harounabadi, "QoS-DRAMA: Quality of Service aware DRL-based Adaptive Mid-level resource Allocation scheme," in *The 2nd Workshop on Next-generation Open and Programmable Radio Access Networks (NG-OPERA) at IEEE INFOCOM*, 2024.

[14] N. Apostolakis, M. Gramaglia, L. E. Chatzieleftheriou, T. Subramanya, A. Banchs, and H. Sanneck, "ATHENA: Machine Learning and Reasoning for Radio Resources Scheduling in vRAN Systems," *IEEE Journal on Selected Areas in Communications*, vol. 42, no. 2, pp. 263–279, 2024.

[15] I. Uchendu, T. Xiao, Y. Lu, B. Zhu, M. Yan, J. L. P. Simón, M. Bennice, C. Fu, C. Ma, J. Jiao, S. Levine, and K. Hausman, "Jump-start reinforcement learning," in *International Conference on Machine Learning*, 2022.

[16] J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *International Conference for Learning Representations (ICLR)*, 2016.

[17] D. S. Hado van Hasselt, Arthur Guez, "Deep reinforcement learning with double Q-learning," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, p. 2094–2100, 2016.

[18] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, and Y. Tassa, "Continuous control with deep reinforcement learning," *Computer Science*, no. 6, 2015.

[19] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proceedings of the 35th International Conference on Machine Learning* (J. Dy and A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870, PMLR, 10–15 Jul 2018.

[20] P. Christodoulou, "Soft actor-critic for discrete action settings," *arXiv: 1910.07207*, 2019.

[21] M. G. Bellemare, W. Dabney, and M. Rowland, *Distributional Reinforcement Learning*. MIT Press, 2023.

[22] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, "Playing Atari with deep reinforcement learning," *arXiv: 1312.5602*, 2013.

[23] W. Dabney, M. Rowland, M. G. Bellemare, and R. Munos, "Distributional reinforcement learning with quantile regression," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, p. 2892–2901, 2018.

[24] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," in *International Conference for Learning Representations (ICLR)*, 2016.

[25] "3GPP technical report 38.913: Study on scenarios and requirements for next generation access technologies."

[26] "3GPP technical report 36.889: Study on licensed-assisted access to unlicensed spectrum."

[27] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference for Learning Representations (ICLR)*, 2015.

[28] M. Z. Romdlony and B. Jayawardhana, "Stabilization with guaranteed safety using control lyapunov–barrier function," *Automatica*, vol. 66, pp. 39–47, 2016.

[29] A. Anand, K. Seel, V. Gjærum, A. Håkansson, H. Robinson, and A. Saad, "Safe learning for control using control lyapunov functions and control barrier functions: A review," *Procedia Computer Science*, vol. 192, pp. 3987–3997, 2021.

[30] M. B. Saltık, L. Özkan, J. H. Ludlage, S. Weiland, and P. M. Van den Hof, "An outlook on robust model predictive control algorithms: Reflections on performance and computational aspects," *Journal of Process Control*, vol. 61, pp. 77–102, 2018.

[31] A. Bemporad and M. Morari, "Robust model predictive control: A survey," in *Robustness in identification and control*, pp. 207–226, Springer, 2007.

[32] I. R. Manchester and J.-J. E. Slotine, "Robust control contraction metrics: A convex approach to nonlinear state-feedback H^{∞} control," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 333–338, 2018.

[33] D. E. Rumelhart, G. E. Hinton, R. J. Williams, *et al.*, "Learning internal representations by error propagation," 1985.

[34] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[36] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-based model predictive control: Toward safe learning in control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 269–296, 2020.

[37] W. Chen, D. Subramanian, and S. Paternain, "Probabilistic constraint for safety-critical reinforcement learning," *arXiv preprint arXiv:2306.17279*, 2023.

[38] M. Hasanbeig, D. Kroening, and A. Abate, "Towards verifiable and safe model-free reinforcement learning," CEUR Workshop Proceedings, 2020.

[39] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 1437–1480, 2015.

[40] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International conference on machine learning*, pp. 1321–1330, PMLR, 2017.

[41] L. Tao, Y. Zhu, H. Guo, M. Dong, and C. Xu, "A benchmark study on calibration," *arXiv preprint arXiv:2308.11838*, 2023.

[42] O. Simeone, *Machine learning for engineers*. Cambridge University Press, 2022.

[43] S. G. Walker, "Bayesian inference with misspecified models," *Journal of statistical planning and inference*, vol. 143, no. 10, pp. 1621–1633, 2013.

[44] R. Martinez-Cantin, K. Tee, and M. McCourt, "Practical Bayesian optimization in the presence of outliers," in *International conference on artificial intelligence and statistics*, pp. 1722–1731, PMLR, 2018.

[45] M. Zecchin, S. Park, O. Simeone, M. Kountouris, and D. Gesbert, "Robust PAC^m: Training ensemble models under misspecification and outliers," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[46] V. Vovk, A. Gammerman, and G. Shafer, *Algorithmic learning in a random world*, vol. 29. Springer, 2005.

[47] V. Quach, A. Fisch, T. Schuster, A. Yala, J. H. Sohn, T. S. Jaakkola, and R. Barzilay, "Conformal language modeling," *arXiv preprint arXiv:2306.10193*, 2023.

[48] N. Deutschmann, M. Alberts, and M. R. Martínez, "Conformal autoregressive generation: Beam search with coverage guarantees," *arXiv preprint arXiv:2309.03797*, 2023.

[49] Z. Wang, R. Gao, M. Yin, M. Zhou, and D. M. Blei, "Probabilistic conformal prediction using conditional random samples," *arXiv preprint arXiv:2206.06584*, 2022.

[50] J. Wang, J. Tong, K. Tan, Y. Vorobeychik, and Y. Kantaros, "Conformal temporal logic planning using large language models: Knowing when to do what and when to ask for help," *arXiv preprint arXiv:2309.10092*, 2023.

[51] A. Z. Ren, A. Dixit, A. Bodrova, S. Singh, S. Tu, N. Brown, P. Xu, L. Takayama, F. Xia, J. Varley, *et al.*, "Robots that ask for help: Uncertainty alignment for large language model planners," *arXiv preprint arXiv:2307.01928*, 2023.

[52] T. G. Dietterich and J. Hostetler, "Conformal prediction intervals for markov decision process trajectories," *arXiv preprint arXiv:2206.04860*, 2022.

[53] L. Lindemann, X. Qin, J. V. Deshmukh, and G. J. Pappas, "Conformal prediction for STL runtime verification," in *Proceedings of the ACM/IEEE 14th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2023)*, pp. 142–153, 2023.

[54] F. Cairoli, N. Paoletti, and L. Bortolussi, "Conformal quantitative predictive monitoring of STL requirements for stochastic processes," in *Proceedings of the 26th ACM International Conference on Hybrid Systems: Computation and Control*, pp. 1–11, 2023.

[55] L. Lindemann, M. Cleaveland, G. Shim, and G. J. Pappas, "Safe planning in dynamic environments using conformal prediction," *arXiv preprint arXiv:2210.10254*, 2022.

[56] A. Dixit, L. Lindemann, S. X. Wei, M. Cleaveland, G. J. Pappas, and J. W. Burdick, "Adaptive conformal prediction for motion planning among dynamic agents," in *Learning for Dynamics and Control Conference*, pp. 300–314, PMLR, 2023.

[57] Y. Patel, S. Rayan, and A. Tewari, "Conformal contextual robust optimization," *arXiv preprint arXiv:2310.10003*, 2023.

[58] S. T. Jose and O. Simeone, "Address-event variable-length compression for time-encoded data," in *2020 International Symposium on Information Theory and Its Applications (ISITA)*, pp. 71–75, IEEE, 2020.

[59] G. Revach, N. Shlezinger, X. Ni, A. L. Escoriza, R. J. Van Sloun, and Y. C. Eldar, "KalmanNet: Neural network aided kalman filtering for partially known dynamics," *IEEE Transactions on Signal Processing*, vol. 70, pp. 1532–1547, 2022.

[60] K. Pratik, R. A. Amjad, A. Behboodi, J. B. Soriaga, and M. Welling, "Neural augmentation of kalman filter with hypernetwork for channel tracking," in *2021 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, IEEE, 2021.

[61] S. Sun and R. Yu, "Copula conformal prediction for multi-step time series forecasting," *arXiv preprint arXiv:2212.03281*, 2022.

[62] M. Cleaveland, I. Lee, G. J. Pappas, and L. Lindemann, "Conformal prediction regions for time series using linear complementarity programming," *arXiv preprint arXiv:2304.01075*, 2023.

[63] J. Yoon, D. Jarrett, and M. Van der Schaar, "Time-series generative adversarial networks," *Advances in neural information processing systems*, vol. 32, 2019.

[64] K. Rasul, C. Seward, I. Schuster, and R. Vollgraf, "Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting," in *International Conference on Machine Learning*, pp. 8857–8868, PMLR, 2021.

[65] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "DeepAR: Probabilistic forecasting with autoregressive recurrent networks," *International Journal of Forecasting*, vol. 36, no. 3, pp. 1181–1191, 2020.

[66] H. Jang, O. Simeone, B. Gardner, and A. Gruning, "An introduction to probabilistic spiking neural networks: Probabilistic models, learning rules, and applications," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 64–77, 2019.

[67] B. Rosenfeld, O. Simeone, and B. Rajendran, "Spiking generative adversarial networks with a neural network discriminator: Local training, Bayesian models, and continual meta-learning," *IEEE Transactions on Computers*, vol. 71, no. 11, pp. 2778–2791, 2022.

[68] S. Park and O. Simeone, "Quantum conformal prediction for reliable uncertainty quantification in quantum machine learning," *arXiv preprint arXiv:2304.03398*, 2023.

[69] A. Fan, M. Lewis, and Y. Dauphin, "Hierarchical neural story generation," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, 2018.

[70] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The curious case of neural text degeneration," *arXiv preprint arXiv:1904.09751*, 2019.

[71] C. Meister, T. Pimentel, G. Wiher, and R. Cotterell, "Locally typical sampling," *Transactions of the Association for Computational Linguistics*, vol. 11, pp. 102–121, 2023.

[72] R. Hettich and K. O. Kortanek, "Semi-infinite programming: theory, methods, and applications," *SIAM review*, vol. 35, no. 3, pp. 380–429, 1993.

[73] N. Hoven and A. Sahai, "Power scaling for cognitive radio," in *2005 International Conference on Wireless Networks, Communications and Mobile Computing*, vol. 1, pp. 250–255, IEEE, 2005.

[74] A. Sahai, N. Hoven, S. M. Mishra, and R. Tandra, "Fundamental tradeoffs in robust spectrum sensing for opportunistic frequency reuse," in *Proc First Intl Workshop on Tech. and Policy for Accessing Spectrum*, 2006.

[75] J. V. Deshmukh, A. Donzé, S. Ghosh, X. Jin, G. Juniwal, and S. A. Seshia, "Robust online monitoring of signal temporal logic," *Formal Methods in System Design*, vol. 51, pp. 5–30, 2017.

[76] S. Lin and D. J. Costello, *Error control coding: fundamentals and applications*. Upper Saddle River, NJ: Pearson/Prentice Hall, 2004.

[77] W. Lee, O. Simeone, J. Kang, S. Rangan, and P. Popovski, "HARQ buffer management: An information-theoretic view," *IEEE Transactions on Communications*, vol. 63, no. 11, pp. 4539–4550, 2015.

[78] P. He, L. Zhao, S. Zhou, and Z. Niu, "Water-filling: A geometric approach and its application to solve generalized radio resource allocation problems," *IEEE transactions on Wireless Communications*, vol. 12, no. 7, pp. 3637–3647, 2013.

[79] C. Puligheddu, N. Varshney, T. Hassan, J. Ashdown, F. Restuccia, and C. F. Chiasserini, "OffloaDNN: Shaping DNNs for scalable offloading of computer vision tasks at the edge," in *2024 IEEE 44th International Conference on Distributed Computing Systems (ICDCS)*, pp. 624–634, 2024.

[80] B. Yang, X. Cao, J. Bassey, X. Li, and L. Qian, "Computation offloading in multi-access edge computing: A multi-task learning approach," *IEEE Transactions on Mobile Computing*, vol. 20, no. 9, 2021.

[81] J. Zhang *et al.*, "Joint resource allocation for latency-sensitive services over mobile edge computing networks with caching," *IEEE Internet of Things Journal*, 2019.

[82] S. Yu, X. Wang, and R. Langar, "Computation offloading for mobile edge computing: A deep learning approach," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2017.

[83] Q. Li, S. Wang, A. Zhou, X. Ma, F. Yang, and A. X. Liu, "QoS driven task offloading with statistical guarantee in mobile edge computing," *IEEE Transactions on Mobile Computing*, 2022.

[84] Z. Fang, J.-H. Lin, M. B. Srivastava, and R. K. Gupta, "Multi-tenant mobile offloading systems for real-time computer vision applications," in *Proceedings of the 20th International Conference on Distributed Computing and Networking*, 2019.

[85] A. Toma, J. Wenner, J. E. Lenssen, and J.-J. Chen, "Adaptive quality optimization of computer vision tasks in resource-constrained devices using edge computing," in *2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, 2019.

[86] C. Puligheddu, J. Ashdown, C. F. Chiasserini, and F. Restuccia, "SEM-O-RAN: Semantic O-RAN slicing for mobile edge offloading of computer vision tasks," *IEEE Transactions on Mobile Computing*, 2023.

[87] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014.

[88] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[89] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[90] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," in *ACM ICLR*, 2016.

[91] G. Fang, X. Ma, M. Song, M. B. Mi, and X. Wang, "Depgraph: Towards any structural pruning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[92] S. Dodge and L. Karam, "Understanding how image quality affects deep neural networks," in *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*, pp. 1–6, IEEE, 2016.

[93] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2722–2730, 2015.

[94] R. Aljundi, P. Chakravarty, and T. Tuytelaars, "Expert gate: Lifelong learning with a network of experts," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2017.

[95] F. Malandrino, C. BARROSO FERNANDEZ, C. J. Bernardos Cano, C. F. Chiasserini, A. De La Oliva, and M. Onsori, "Data-driven and privacy-preserving cooperation in

decentralized learning," in *The 49th IEEE Conference on Local Computer Networks (LCN) 2024*, (Caen, France), 2024.

[96] J. Konečnỳ, B. McMahan, and D. Ramage, "Federated optimization: Distributed optimization beyond the datacenter," *arXiv preprint arXiv:1511.03575*, 2015.

[97] J. Ren, G. Yu, and G. Ding, "Accelerating dnn training in wireless federated edge learning systems," *IEEE Journal on Selected Areas in Communications*, 2020.

[98] F. Malandrino, G. Di Giacomo, A. Karamzade, M. Levorato, and C. F. Chiasserini, "Matching DNN compression and cooperative training with resources and data availability," in *IEEE INFOCOM*, 2023.

[99] F. Malandrino, C. F. Chiasserini, N. Molner, and A. De La Oliva, "Network support for high-performance distributed machine learning," *IEEE/ACM Transactions on Networking*, 2022.

[100] G. Neglia, G. Calbi, D. Towsley, and G. Vardoyan, "The role of network topology for distributed machine learning," in *IEEE INFOCOM*, 2019.

[101] I. Hegedűs, G. Danner, and M. Jelasity, "Decentralized learning works: An empirical comparison of gossip learning and federated learning," *Journal of Parallel and Distributed Computing*, 2021.

[102] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu in *NeurIPS* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), 2017.

[103] S. Li, T. Zhou, X. Tian, and D. Tao, "Learning to collaborate in decentralized learning of personalized models," in *IEEE/CVF CVPR*, 2022.

[104] Y. Shi, L. Shen, K. Wei, Y. Sun, B. Yuan, X. Wang, and D. Tao, "Improving the model consistency of decentralized federated learning," in *ICML*, 2023.

[105] K. Hsieh, A. Phanishayee, O. Mutlu, and P. Gibbons, "The non-IID data quagmire of decentralized machine learning," in *ICML*, 2020.

[106] W. Liu, L. Chen, and W. Zhang, "Decentralized federated learning: Balancing communication and computing costs," *IEEE Transactions on Signal and Information Processing over Networks*, 2022.

[107] T. Vogels, H. Hendrikx, and M. Jaggi, "Beyond spectral gap: the role of the topology in decentralized learning," in *NeurIPS*, vol. 35, 2022.

[108] W. Zhang, X. Cui, A. Kayi, M. Liu, U. Finkler, B. Kingsbury, G. Saon, Y. Mroueh, A. Buyuktosunoglu, P. Das, D. Kung, and M. Picheny, "Improving efficiency in large-scale decentralized distributed training," in *IEEE ICASSP*, 2020.

[109] L. Yang, Y. Lu, J. Cao, J. Huang, and M. Zhang, "E-tree learning: A novel decentralized model learning framework for edge ai," *IEEE Internet of Things Journal*, 2021.

[110] A. N. D. Model, "Learning framework for edge ai," *IEEE Internet of Things Journal*, 2020.

[111] Z. Allen-Zhu, Y. Li, and Z. Song, "A convergence theory for deep learning via over-parameterization," in *ICML*, 2019.

[112] R. K. Iyer and J. A. Bilmes, "Submodular optimization with submodular cover and submodular knapsack constraints," in *NeurIPS*, 2013.

[113] V. Vu, "Random discrete matrices," in *Horizons of combinatorics*, Springer.

[114] K. Tikhomirov and P. Youssef, "The spectral gap of dense random regular graphs," *The Annals of Probability*, 2019.

[115] N. Chen, S. Watanabe, J. Villalba, P. Żelasko, and N. Dehak, "Non-autoregressive transformer for speech recognition," *IEEE Signal Processing Letters*, vol. 28, pp. 121–125, 2020.

[116] X. Gao, Y. Liu, X. Liu, and L. Song, "Machine learning empowered resource allocation in IRS aided MISO-NOMA networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 5, pp. 3478–3492, 2021.

[117] P. Kairouz and et al, "Advances and open problems in federated learning," *Foundations and Trends ® in Machine learning*, vol. 14, no. 1-2, pp. 1–210, 2021.

[118] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas, "Communication-efficient learning of deep networks from descentralized data," *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

[119] A. Ghalkha, C. Issaid, A. Elgabli, and M. Bennis, "DIN: A descentralized inexact Newton algorithm for consensus optimization," *IEEE Transactions on MachineLearning in Communications and Networking*, 2024.

[120] C. Dinh, N. Tran, T. Nguyen, W. Bao, A. Balef, B. Zhou, and A. Zomaya, "DONE:distributed approximate Newton-type method for Federated Edge Learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 11, pp. 2648–2660, 2022.

[121] A. Elbakary, C. Issaid, M. Shehab, K. Seddik, T. ElBatt, and M. Bennis, "Fed-Sophia: A communication-efficient second-order federated learning algorithm," *IEEE International Conference on Communications (ICC 2024)*, 2024.

[122] T. Li, A. Sahu, M. Zaheer, M. Sanjabi, A. Talwalker, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine Learning and Systems*, vol. 2, pp. 429–450, 2020.

[123] B. Nazer and M. Gastpar, "Computation over multiple-access channels," *IEEE Transactions on Information Theory*, vol. 53, no. 10, pp. 3498–3516, 2007.

[124] M. Krouka, A. Elgabli, C. Issaid, and M. Bennis, "Computation over multiple-access channels," *IEEE Transactions on Green Communications and Networking*, vol. 6, no. 3, pp. 1862–1874, 2022.

[125] H. Liu, Z. Li, D. Hall, P. Liang, and T. Ma, "Sphia: A scalable stochastic second-order optimizer for language model pre-training," *The* 11^{th} *International Conference on Learning Representations (ICLR 2023)*, 2023.

[126] T. ETSI, "136 213 v8.8.0 (2009-10)," *LTE Evolved Universal Terrestrial Radio Access (E-UTRA) Physical layer procedures, 3GPP TS*, vol. 36, 2009-2010.

[127] E. Chiaramello, C. F. Chiasserini, F. Malandrino, A. Nordio, M. Parazzini, A. Valcarce, *et al.*, "Cluster-then-match: Efficient management of human-centric, cell-less 6G networks," in *IEEE WoWMoM 2024*, IEEE, 2024.

[128] I. Ahmed *et al.*, "A survey on hybrid beamforming techniques in 5g: Architecture and system model perspectives," *IEEE Comm. Surv. & Tut.*, 2018.

[129] ETSI, "5G; study on channel model for frequencies from 0.5 to 100 GHz (3GPP TR 38.901 Release 16)," tech. rep., Nov. 2020.

[130] D. Colombi, B. Thors, and C. Törnevik, "Implications of emf exposure limits on output power levels for 5g devices above 6 ghz," *IEEE Ant. and Wir. Prop. Lett.*, 2015.

[131] B. M. Hochwald and otherse, "Incorporating specific absorption rate constraints into wireless signal design," *IEEE Comm. Mag.*, 2014.

[132] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the royal statistical society. Series C (applied statistics)*, 1979.

[133] S. Sur, I. Pefkianakis, X. Zhang, and K.-H. Kim, "Practical mu-mimo user selection on 802.11 ac commodity networks," in *ACM MobiCom*, 2016.

[134] R. Jonker and T. Volgenant, "Improving the hungarian assignment algorithm," *Operations Research Lett.*, 1986.

[135] J. Schneider and S. Kirkpatrick, *Stochastic optimization*. 2007.

[136] International Commission on Non-Ionizing Radiation Protection, "Guidelines for limiting exposure to electromagnetic fields (100 khz to 300 ghz)," *Health physics*, 2020.

[137] S. Liesegang and S. Buzzi, "EMF-aware power control for massive MIMO: Cell-free versus cellular networks," *IEEE Wireless Communications and Networking Conference*, pp. 1–6, 2024.

[138] A. Zappone and M. D. Renzo, "Energy efficiency optimization of reconfigurable intelligent surfaces with electromagnetic field exposure constraints," *IEEE Signal Process. Lett.*, vol. 29, pp. 1447–1451, 2022.

[139] H. Tataria *et al.*, "6G wireless systems: Vision, requirements, challenges, insights, and opportunities," *Proc. IEEE*, vol. 109, no. 7, pp. 1166–1199, 2021.

[140] M. R. Castellanos *et al.*, "Dynamic electromagnetic exposure allocation for rayleigh fading MIMO channels," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 728–740, 2021.

[141] L. Chiaraviglio, A. Elzanaty, and M.-S. Alouini, "Health risks associated with 5G exposure: A view from the communications engineering perspective," *IEEE O. J. Commun. Soc.*, vol. 2, pp. 2131–2179, 2021.

[142] *Mitigation techniques to limit human exposure to EMFs in the vicinity of radiocommunication stations*, ITU-T, Recommendation K.70 2020.

[143] C. Psomas *et al.*, "Design and analysis of SWIPT with safety constraints," *Proc. IEEE*, vol. 110, no. 1, pp. 107–126, 2022.

[144] International Commission on Non-Ionizing Radiation Protection, "ICNIRP guidelines on limiting the exposure to time-varying electric, magnetic, and electromagnetic fields (100 kHz to 300 GHz)," *Health Phys*, vol. 118, no. 5, p. 483–524, 2020.

[145] L. Lu *et al.*, "An overview of massive MIMO: Benefits and challenges," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 5, pp. 742–758, 2014.

[146] S. Buzzi and C. D'Andrea, "Cell-free massive MIMO: User-centric approach," *IEEE Wireless Commun. Lett.*, vol. 6, no. 6, pp. 706–709, 2017.

[147] H. Ibraiwish, A. Elzanaty, Y. H. Al-Badarneh, and M.-S. Alouini, "EMF-aware cellular networks in RIS-assisted environments," *IEEE Commun. Lett.*, vol. 26, no. 1, pp. 123–127, 2022.

[148] G. Interdonato *et al.*, "Ubiquitous cell-free massive MIMO communications," *EURASIP J. Wireless Commun. Netw.*, vol. 118, no. 1, p. 197, 2019.

[149] C. Wiame, C. Oestges, and L. Vandendorpe, "Joint data rate and EMF exposure analysis in user-centric cell-free massive MIMO networks," *arXiv:2301.11127*, 2023.

[150] S. Liesegang, A. Zappone, O. Muñoz, and A. Pascual-Iserte, "Rate optimization for RIS-aided mMTC networks in the finite blocklength regime," *IEEE Commun. Lett.*, vol. 27, no. 3, pp. 921–925, 2023.

[151] E. Bjornson and P. Giselsson, "Two applications of deep learning in the physical layer of communication systems [lecture notes]," *IEEE Signal Process. Mag.*, vol. 37, no. 5, pp. 134–140, 2020.

[152] J. Hoydis, F. A. Aoudia, A. Valcarce, and H. Viswanathan, "Toward a 6G AI-native air interface," *IEEE Commun. Mag.*, vol. 59, no. 5, pp. 76–81, 2021.

[153] J. Guo and C. Yang, "Learning power allocation for multi-cell-multi-user systems with heterogeneous graph neural networks," *IEEE Trans. Wireless Commun.*, vol. 21, no. 2, pp. 884–897, 2022.

[154] M. Zaher, Ö. T. Demir, E. Björnson, and M. Petrova, "Learning-based downlink power allocation in cell-free massive MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 22, no. 1, pp. 174–188, 2023.

[155] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Process. Mag.*, vol. 38, no. 2, pp. 18–44, 2021.

[156] N. Shlezinger, Y. C. Eldar, and S. P. Boyd, "Model-based deep learning: On the intersection of deep learning and optimization," *IEEE Access*, vol. 10, pp. 115384–115398, 2022.

[157] M. Elwekeil, A. Zappone, and S. Buzzi, "Power control in cell-free massive MIMO networks for UAVs URLLC under the finite blocklength regime," *IEEE Trans. Commun.*, vol. 71, no. 2, pp. 1126–1140, 2023.

[158] Q. Gontier *et al.*, "Joint metrics for EMF exposure and coverage in real-world homogeneous and inhomogeneous cellular networks," *arXiv:2302.03559*, 2023.

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

[160] Y. Sun, P. Babu, and D. P. Palomar, "Majorization-minimization algorithms in signal processing, communications, and machine learning," *IEEE Trans. Signal Process.*, vol. 65, no. 3, pp. 794–816, 2017.

[161] H. Q. Ngo *et al.*, "Cell-free massive MIMO versus small cells," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1834–1850, 2017.

[162] M. Grant and S. Boyd, "CVX: MATLAB software for disciplined convex programming, version 2.2." `http://cvxr.com/cvx`, 2020.

[163] O. Elijah *et al.*, "Intelligent massive MIMO systems for beyond 5G networks: An overview and future trends," *IEEE Access*, vol. 10, pp. 102532–102563, 2022.

[164] Y. Shi *et al.*, "Machine learning for large-scale optimization in 6G wireless networks," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 4, pp. 2088–2132, 2023.

[165] Ö. Demir, E. Björnson, and L. Sanguinetti, *Foundations of User-Centric Cell-Free Massive MIMO*. Now Publishers, 2021.

[166] *Further advancements for E-UTRA physical layer aspects*, 3GPP, Technical Report 36.814 v9.2.0 2017.

[167] C. D'Andrea *et al.*, "Analysis of UAV communications in cell-free massive MIMO systems," *IEEE O. J. Commun. Soc.*, vol. 1, pp. 133–147, 2020.

[168] L. Pellaco, M. Bengtsson, and J. Jaldén, "Matrix-inverse-free deep unfolding of the weighted MMSE beamforming algorithm," *IEEE O. J. Commun. Soc.*, vol. 3, pp. 65–81, 2022.

[169] K. Zhi *et al.*, "Two-timescale design for reconfigurable intelligent surface-aided massive MIMO systems with imperfect CSI," *IEEE Trans. Inf. Theory*, vol. 69, no. 5, pp. 3001–3033, 2023.

CENTRIC

https://centric-sns.eu/