MERGE: Meta Reinforcement Learning for Tunable RL Agents at the Edge

Sharda Tripathi
Birla Institute of Technology and Science
Pilani, India

Carla Fabiana Chiasserini
Politecnico di Torino and CNIT
Torino, Italy

***Abstract*—The efficient allocation of radio resources is an essential trait of 5G/6G radio access networks (RANs), as they are called to meet diverse QoS requirements of highly demanding applications. To equip RANs with such an ability and, at the same time, meet their function split constraints, we envision a distributed learning approach for radio resource allocation that makes the most out of the Central Unit (CU) and Distributed Unit (DU) components by effectively exploiting their synergy. On the one hand, our solution, named MERGE, leverages the knowledge of the radio connectivity dynamics that each DU can acquire through the local use of a deep reinforcement learning radio agent. On the other hand, it lets the CU collect such agents in a crowdsourcing fashion, and, then, thanks to a meta-learning policy, properly select and aggregate them to create up-to-date radio agents of the right size (hence, complexity level) to fit the computing constraints of the individual DUs. Our results show that MERGE can match the performance of the highest-complexity radio model with 25% less computational requirements, and, for a given computational resource, it outperforms a single pruned model with a 19% increase in QoS.**

***Index Terms*—Meta reinforcement learning, Edge computing, Virtual RAN, Resource orchestration, ML model compression.**

I. Introduction

The commercial deployment of 5G networks has made available to the users a plethora of advanced services that need massive data rates, high reliability, low latency, and ubiquitous connectivity. To meet such diverse and stringent performance indices, the concept of a virtualized radio access network (vRAN) at the edge is a key enabler, as it brings together the best of network service virtualization and edge computing. Another essential advancement in this respect is the introduction of the gNB functional split architecture, which, by splitting the RAN stack across distinct components, allows for increased flexibility, making the RAN configuration significantly easier. Such disaggregated RAN includes Central Unit (CU), Distributed Unit (DU), and Radio Unit (RU) [1]. While the RU is the radio hardware unit, the CU handles such control plane functions as resource allocation, scheduling, and mobility management, and the DU implements the CU's policies and manages the data plane for data radio transfer. By separating the control and data planes, the split gNB can facilitate the deployment of virtualized network functions, thereby increasing resource handling efficiency.

Considering large-scale connectivity with diverse channel fading characteristics and service requirements, the application of standard optimization frameworks at the CU for solving the problem of radio resource allocation is often intractable. Learning-based resource allocation algorithms [2], instead, have the potential to effectively cope with the system complexity for providing automated network control. However, also in the case of learning-based approaches, several hurdles exist in developing scalable, efficient solutions. [3] has shown that conventional centralized approaches may scale poorly, while distributed radio resource allocation can be much more effective. It is worth observing that the latter can also be a better match for the architecture of a typical disaggregated vRAN, which consists of a number of DUs. On the other hand, a DU has typically limited computing and storage resources, as compared to the CU, which, being in charge of controlling multiple DUs, is computationally more capable. Moreover, implementing learning-based resource allocation algorithms at the DUs would imply that the learning agents are exposed only to local network dynamics, thereby limiting their generalization capability and thus impacting the quality of resource allocation decisions. To address the above issues, it is imperative to envision solutions for the learning of distributed agents that suit both the availability of compute resources at the DUs and the need for model generalization.

In this work, we develop a solution, named MEta Reinforcement learning framework for tunable RL agents at the edGE (MERGE), that, leveraging both the CU and DU, effectively realizes radio resource allocation at the vRAN. MERGE builds on the intuition that, as noted above, a learning approach for radio resource allocation cannot be handled at the DU alone. Instead, the DU can host and execute a *pre-trained, reduced-size (or pruned) version of a learning agent* that matches its computational constraints and has been properly created at the CU. Thanks to its architectural role and larger storage capability, the CU can indeed collect knowledge from all the DUs and use it to improve the capability of radio agents to deal with unseen scenarios (e.g., sudden increase in network traffic, or SINR variations). Further, to guarantee a timely delivery of up-to-date radio agents, the CU can create in advance model versions of different size, hence complexity level, store them, and deliver an agent of the *right size* upon a DU's request.

MERGE thus aims at *learning to learn* optimal resource allocation strategies in computationally constrained environ-

This work was supported by the Horizon Europe project CENTRIC (Grant No. 101096379) and by the EU under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future" (PE00000001 - program "RESTART").

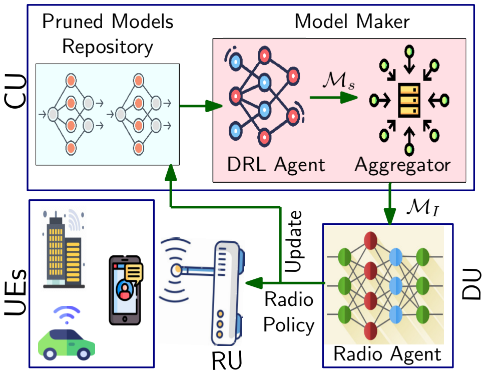


Fig. 1: Structure of the MERGE framework

ments. To this end, it exploits a *meta-learning, crowdsourcing approach*: as depicted in Fig. 1, it delivers pre-trained pruned versions of deep reinforcement learning (DRL) radio agents to the DUs, with size properly tuned to match the individual DUs' computing constraints. Each DU then uses its agent and returns the updated version to the CU, along with an indication of the model performance. In this way, the CU can learn, and refine its ability, to create right-sized radio models that can output high-quality decisions on radio resource allocation. It does so by using the model maker, in which another DRL agent properly selects a subset of the radio agents, $\mathcal{M}_s$, that the CU has collected from the DUs. By aggregating such agents, the model maker generates an inference model of suitable size, $\mathcal{M}_I$, to be used at a DU for radio resource allocation.

To summarize, we make the following contributions:

- We propose MERGE, a meta RL framework for radio resource allocation that effectively leverages the synergy between CU and DUs (Sec. II);
- We design a DRL agent (i.e., the *radio agent*) for determining at the DU the modulation and coding scheme and the allocation of resource blocks for the served users such that the target values of key performance indicators (KPIs) are met (Sec. II-A);
- Next, we devise the *model maker*, which comprises designing (i) a DRL agent (to run at the CU) to select a subset of radio models based on the availability of computing resources at the requesting DU, and (ii) an Aggregator, to combine selected radio models for creating an inference model of suitable size (Sec. II-B);
- Finally, we compare the performance of MERGE against two benchmarks, and show that the decision-making quality of MERGE matches the ideal benchmark with 25% less computational requirements, while, for a given computational resource, its QoS is 19% better than that of a single compressed model (Sec. III).

We remark that, as discussed in Sec. IV, the existing meta-learning approaches deal with the edge compute constraints to some extent, however, none of them addresses radio resource allocation in disaggregated vRANs. Also, MERGE tackles the model localization issue of pruning methods, while creating agents suitable for resource-constrained environments.

II. The MERGE Framework

Our reference scenario comprises a vRAN infrastructure at the edge that leverages the functional split of the gNB. The proposed MERGE framework comprises two modules, (i) the *radio agent*, operating at the DUs and providing radio policies for user equipments (UEs), and (ii) the *model maker*, running at the CU and designing meta learning policies for the radio agents. Since a DU allocates radio resources to the UEs as the network load and the edge-UE link quality vary over time, we design the radio agent as a DRL model. However, due to their limited compute resources, the DUs have to rely on the CU for obtaining a suitably trained radio agent that can make effective decisions. On the other hand, a radio agent running at a DU gets further trained while being executed, thus resulting into continuous updates of the DRL model. We therefore consider that the DUs periodically provide their updated radio models to the CU, which stores them for future use.

Let the CU locally store N versions of the radio model. Since they have been previously used, hence trained, at different DUs, such models provide versions of the same DRL agent, which, may differ in the ability to deal with different domains. Furthermore, they may be *pruned versions* of the same DRL model that has been compressed through different pruning factors for reducing size and complexity to suit the computing capability of specific DUs.

Below, we detail the MERGE framework components, namely, the radio agent and the model maker.

A. Radio agent

We design the radio agent using a deep Q-learning network (DQN) to enable radio resource allocation in a vRAN that meets the KPIs target values. Further, we account for the fact that the DUs may be computationally constrained, due to the limited capability of the edge servers hosting the vRAN and the presence of other services that compete for the available computing resources [3]. Without loss of generality, we focus on a single DU allocating radio resources to U UEs, and let γ_u and σ_u denote (resp.) the SNR and the buffer state reported by UE u to the DU. Further, let t be the throttle time[1] of the vRAN implemented at the DU. Then the context vector observed at the radio agent for making a decision in window p is denoted by $\mathbf{x}_r^{(p)} := \{\gamma_1^{(p)}, \dots, \gamma_U^{(p)}, \sigma_1^{(p)}, \dots, \sigma_U^{(p)}, t^{(p)}\}$. Note that the SNR and the buffer state account for the link and network dynamics, while the radio throttle time indicates the vRAN computational load.

Based on the observed context, the radio agent makes decisions on the maximum modulation and coding scheme (MCS) that should be used to control the vRAN computational requirements while aiming at maximizing the spectral efficiency, and the number of resource blocks (RBs) to be allocated to each UE. Thus, the action vector chosen in decision window p is represented as $\mathbf{a}_r^{(p)} = \{\omega^{(p)}, \nu_1^{(p)}, \dots, \nu_U^{(p)}\}$, with ω and ν_u denoting (resp.) the MCS policy and the number of RBs

[1] CPU throttling is used, e.g., in Kubernetes, to enforce CPU limit. In the throttle time, an application that exceeds the limit gets fewer CPU cycles.

allocated to UE u. The KPIs packet loss (ζ) and latency (λ) are expected to meet their target values as identified by the 3GPP standard [4]. The KPI satisfaction for each UE u is measured at the end of every decision window p using the value of the reward function, defined as [5]

$$r^u_{r(\mathrm{KPI})}(\mathbf{x}^{(p)}_r, \mathbf{a}^{(p)}_r) = \begin{cases} 1 - \mathrm{erf}(\mathrm{KPI}_t - \mathrm{KPI}^u_o(\mathbf{x}^{(p)}_r, \mathbf{a}^{(p)}_r)) \\ \hfill \text{if KPI is met} \\ \mathrm{erf}(\mathrm{KPI}_t - \mathrm{KPI}^u_o(\mathbf{x}^{(p)}_r, \mathbf{a}^{(p)}_r)), \qquad \text{else} \, . \end{cases}$$

Therein $\mathrm{KPI}^u_o(\mathbf{x}^{(p)}_r, \mathbf{a}^{(p)}_r)$ is the observed KPI value in response to action $\mathbf{a}^{(p)}_r$ and context $\mathbf{x}^{(p)}_r$ for UE u, and KPI_t denotes the corresponding target KPI value. The total reward of the radio agent in decision window p is given by,

$$r^{(p)}_r = \frac{1}{U} \sum_{u=1}^{U} \{ r^u_{r(\zeta)}(\mathbf{x}^{(p)}_r, \mathbf{a}^{(p)}_r) + r^u_{r(\lambda)}(\mathbf{x}^{(p)}_r, \mathbf{a}^{(p)}_r) \} \, .$$

The choice of the reward function is motivated by the need to meet the KPI targets *and* keep an observed KPI as close as possible to its target value to minimize resource consumption.

B. Model maker

The model maker is a DQN-based model located at the CU, which implements meta learning for improving the training of the radio agents. Specifically, its objective is to build pre-trained radio models that meet the constraints imposed by the computing-constrained edge while providing high-quality MCS and RB allocations when used for inference at the DUs. The elements of the model maker are as follows.

Context space. Given the radio model, the selection of pruned versions thereof is primarily governed by two factors, namely, (i) the availability of computing resources at the DU, and (ii) the learning quality-related statistics of the pruned radio models owned by the CU. It may be noted that the radio agent and the model maker do not make decisions with the same periodicity. Only when the learning at the radio agent fails to make decisions of acceptable quality, the model maker is invoked to provide a new inference model. Thus, let τ_p be the duration of one decision window at the radio agent and $C > 1$ any arbitrary integer, then $\tau_q = C\tau_p$ is the duration of the decision window at the model maker. Let $\mathcal{M} = \{M_1, \dots, M_N\}$ be the set of N pruned versions of the radio model available in the CU with $z_{i_n}, z_{o_n}, z_{h_n}, h_n, w_{\mu_n}, w_{v_n}, w_{s_n}$ denoting (resp.) the size of input layer, output layer, and hidden layers, number of hidden layers, mean, variance, and sparsity of the weight matrix of the n-th pruned radio model. Further, let φ_a be the available computational resource at the edge; then we define the context vector at the model maker in its decision window q as $\mathbf{x}^{(q)}_m := \{\varphi^{(q)}_a, z^{(q)}_{i_n}, z^{(q)}_{o_n}, z^{(q)}_{h_n}, h^{(q)}_n, w^{(q)}_{\mu_n}, w^{(q)}_{v_n}, w^{(q)}_{s_n}\}$.

Action space. Upon receiving a new request for a radio model from one of the DUs under its control, the CU selects K ($K \leq N$) radio agents. The design objectives of the model maker are to identify *how many* (namely, the value of K) and *which* pruned radio models to select for creating an aggregated inference model. To jointly meet these objectives, we formulate the model maker's actions as vectors, each denoting a probability distribution over the choice of models to be selected. Specifically, by denoting with α_n the probability of choosing pruned radio model n, we define the action vector selected in decision window q as $\mathbf{a}^{(q)}_m := \{\alpha^{(q)}_1, \dots, \alpha^{(q)}_N\}$, w Given N pruned radio models in the CU, the number of possible action vectors is $A_{tot} = \sum_{n=1}^{N} \binom{N}{n}$. Further, if $K \leq N$ pruned radio models are to be selected through $\mathbf{a}^{(q)}_m$, it will have exactly K non-zero elements corresponding to the selected models. To avoid any bias towards selecting particular pruned radio models, we define $\alpha^{(q)}_n$ for $n = 1, \dots, N$ as,

$$\alpha^{(q)}_n = \begin{cases} 1/K, & \text{if pruned model } n \text{ is selected,} \\ 0, & \text{otherwise.} \end{cases}$$

Reward. Since the inference model needs to run using the limited resources at the DU without compromising the quality of radio decisions, we assign two KPIs to the model maker, (i) amount of computing resources consumed by the inference radio model, φ_c, and (ii) reward of the radio agent at its convergence, ρ, to assess the satisfaction of the radio KPIs for the served UEs. Then the total reward of the model maker in decision window q is given by,

$$r^{(q)}_m = r_{m(\varphi)}(\mathbf{x}^{(q)}_m, \mathbf{a}^{(q)}_m) + r_{m(\rho)}(\mathbf{x}^{(q)}_m, \mathbf{a}^{(q)}_m),$$

where $r_{m(\varphi)}(\mathbf{x}^{(q)}_m, \mathbf{a}^{(q)}_m), r_{m(\rho)}(\mathbf{x}^{(q)}_m, \mathbf{a}^{(q)}_m)$ are the reward components pertaining to KPIs (resp.) φ_c and ρ when action $\mathbf{a}^{(q)}_m$ is taken on the observation of context $\mathbf{x}^{(q)}_m$. We define these reward components as,

$$r_{m(\varphi)}(\mathbf{x}^{(q)}_m, \mathbf{a}^{(q)}_m) = \frac{(1 - \varphi^{(q)}_c / \varphi^{(q)}_a)}{(1 - \varphi_{min} / \varphi^{(q)}_a)},$$

$$r_{m(\rho)}(\mathbf{x}^{(q)}_m, \mathbf{a}^{(q)}_m) = \rho^{(q)} / r_{r(max)},$$

where φ_{min} is the minimum computing resource corresponding to the case when the radio model pruned by the maximum compression factor is executed, and $r_{r(max)}$ is the maximum possible reward of the radio agent. From the reward function for the model maker, it is clear that $0 \leq r^{(q)}_m \leq 2$.

DQN learning and pruned models selection. Q-learning is a model-free RL technique in which the agent learns to choose actions for a given state of the environment by maximizing its long-term reward. For the model maker, we adopt the definition of long term reward given as the discounted expected return [6], $G_m = \sum_{i=0}^{\infty} \gamma^i r^{(q+i)}_m$, where $\gamma \in [0, 1]$ is the discount factor to account more for the immediate next rewards than the longer-term future rewards. The value of taking an action in decision window q of the model maker, following meta-learning policy π, is quantified using the Q-value as, $Q_\pi(\mathbf{x}^{(q)}_m, \mathbf{a}^{(q)}_m) = \mathbb{E}_\pi[G_m | \mathbf{x}^{(q)}_m, \mathbf{a}^{(q)}_m]$. In practice, since the optimal Q-values follow the Bellman's recursive equation $Q^\star_\pi(\mathbf{x}^{(q)}_m, \mathbf{a}^{(q)}_m) = \mathbb{E}[r^{(q)}_m + \gamma \max_{\mathbf{a}^{(q+1)}_m} Q^\star_\pi(\mathbf{x}^{(q+1)}_m, \mathbf{a}^{(q+1)}_m) | \mathbf{x}^{(q)}_m, \mathbf{a}^{(q)}_m]$, they are iteratively learned over successive decision windows by minimizing the difference between $Q^\star_\pi(\mathbf{x}^{(q)}_m, \mathbf{a}^{(q)}_m)$ and its estimate

$\hat{Q}_\pi(\mathbf{x}_m^{(q)}, \mathbf{a}_m^{(q)})$. In a DQN, $Q_\pi^\star(\cdot)$ and $\hat{Q}_\pi(\cdot)$ are approximated using two separate deep neural networks, called the target and the prediction network, denoted (resp.) by $Q(\cdot, W^{tgt})$ and $Q(\cdot, W^{pred})$, with W^{tgt} and W^{pred} as the corresponding parameters. The target network output behaves as the ground truth for the prediction network, while the output of the latter governs the choice of selected actions. Thus, in decision window q, the DQN is trained by minimizing the squared error loss function given by [6],

$$L^{(q)}(W^{tgt}, W^{pred}) = [r_m^{(q)} + \gamma \max_{\mathbf{a}_m^{(q+1)}} Q^\star(\mathbf{x}_m^{(q+1)}, \mathbf{a}_m^{(q+1)}, W^{tgt}) \\ -\hat{Q}(\mathbf{x}_m^{(q)}, \mathbf{a}_m^{(q)}, W^{pred})]^2$$

The Q-value estimation is followed by an ϵ-greedy action selection policy. Notice that the cardinality of the action space is given by A_{tot}, which quickly scales with increase in N, in turn increasing the complexity of the DQN implementation. To this end, we limit the size of the output layer of the prediction network to N, and augment it with a softmax layer. Thus, in the decision window q, the prediction network output is a probability distribution $\beta^{(q)} = \{\beta_1, \dots, \beta_N\}$, where β_n indicates the preference of choosing pruned radio model n. Subsequently, we evaluate the Hellinger's distance between the probability distribution predicted by the DQN and the possible actions as, $H^{(q)} = \{h_i^{(q)}\}, h_i^{(q)} = \frac{1}{\sqrt{2}}\sqrt{\sum_{j=1}^{N}(\sqrt{\alpha_j} - \sqrt{\beta_j})^2}$, for $i = 1, \dots, A_{tot}$, and select the greedy action $\mathbf{a}_m^{(q)} = \text{argmin}_h H^{(q)}$ with probability $1 - \epsilon$. Over successive decision windows, ϵ decays by a factor of 10^{-4} to allow for more exploitation rather than exploration of the environment. Models corresponding to non-zero elements in $\mathbf{a}_m^{(q)}$ are the desired subset of selected pruned radio models.

Aggregate radio model. Since the selected pruned radio models may be heterogeneous owing to their different pruning factors, and hence different sizes, we follow the hierarchical strategy in [7] to aggregate them into a radio model to be used for inference at the requesting DU. Let the selected subset of pruned radio models in decision window q be $\mathcal{M}_s = \{M_1, \dots, M_K\}$, parameterized by corresponding weight matrices $\{\ \theta_1, \dots, \theta_K\ \}$. We consider each of these models to belong to one of the L complexity levels, each level signifying a different pruning factor of the radio model. That is, a model belonging to a higher complexity level has been pruned by a smaller pruning factor, hence will require more computational resources for its implementation. Further, let $\{c_1, \dots, c_L\}$ be the number of selected pruned radio models at each complexity level. Then, at complexity level l, we perform the aggregation as, $\theta^l = \frac{1}{K}\sum_{k=1}^{K}\theta_k^l$, and proceed up the hierarchy of complexity levels in a similar manner, but excluding the aggregated weights at the immediate lower complexity level, i.e., $\theta^{l+1}\backslash\theta^l = \frac{1}{K-c_l}\sum_{k=1}^{K-c_l}\theta_k^{l+1}\backslash\theta_k^l$. Finally, the weight matrix of the aggregated inference model $\mathcal{M}_I$ is evaluated as $\theta_I = \bigcup_{l=1}^{L-1}\theta^{l+1}\backslash\theta^l$.

Once the aggregated inference model is ready, it is shared with the requesting DU for making resource allocation decisions in the vRAN. The performance of $\mathcal{M}_I$ is observed at the DU in terms of reward of the radio agent at convergence, ρ, which in turn is shared back with the model maker for its learning. The workflow of MERGE in a generic decision window q of the model maker is summarized in Alg. 1.

Algorithm 1: Workflow in the MERGE framework

```
1  Given: $\mathcal{M}$, a set of $N$ pruned versions of radio model locally stored at the CU
2  DU reports $\varphi_a^{(q)}$ to the CU, requests for a new radio model
3  Observe context $\mathbf{x}_m^{(q)}$, input to the model maker
4  Obtain probability distribution $\beta^{(q)}$ using DQN in the model maker
5  Evaluate Hellinger's distance $H^{(q)}$
6  Apply $\epsilon$-greedy policy for the selection of action $\mathbf{a}_m^{(q)}$
7  Obtain $\mathcal{M}_I$ through hierarchical aggregation of selected pruned models,
     deliver $\mathcal{M}_I$ to the requesting DU
8  For decision window $p$ of DU, $p = 1, 2, \cdots$ do
9  Observe context $x_r^{(p)}$ at the DU, input to the radio agent
10 Evaluate $a_r(p)$ using $\mathcal{M}_I$, take action in vRAN
11 Observe KPIs $\zeta^{(p)}$, $\lambda^{(p)}$, evaluate $r_r^{(p)}$
12 Update parameters of radio agent
13 Repeat 9-11 until convergence of radio agent, observe $\rho^{(q)}$
14 Observe KPIs $\rho^{(q)}$, $\varphi_c^{(q)}$ at the CU, evaluate $r_m^{(q)}$
15 Update parameters of the model maker
```

III. Performance evaluation

In this section, we first assess the performance of the proposed MERGE framework by showing the convergence of the model maker, followed by the behavior of the KPIs at the CU and DU. We further highlight the trends observed during the meta learning process, and finally present a comparison of the MERGE framework against two baseline schemes.

To begin, we create a full DQN radio model that can make high-quality radio resource allocation decisions at the DUs in the case where there are no computational constraints. Next, we consider 7 pruned versions of the full model to be locally stored at the CU, out of which 2 are pruned by a factor of 0.5, and the rest by 0.1, 0.3, 0.6, 0.8, and 0.9 each. The pruned models are further trained using different context and action scenarios to emulate statistical network diversity. Such pre-trained pruned radio models are then selectively combined using the meta learning policy of the model maker. We focus on a DU requesting a radio model and consider that it is serving 2 UEs. To enable the pre-training and learning of the radio agents, we have used the datasets reporting the CPU constraints and the corresponding KPIs for various contexts and actions, available at https://github.com/corrado113/VERA.

Convergence evaluation. The values of the model maker reward, and of its components, over successive decision windows are shown in Fig. 2. We observe that both reward components, and hence the total reward, saturate close to their maximum value. Also, the variation in $r_{m(\rho)}$ is comparatively smaller than $r_{m(\varphi_c)}$, because ρ, being the radio agent's reward at convergence, takes values closer to $r_{r(max)}$. In summary, these results clearly highlight the effectiveness of the meta learning policy of the MERGE framework.

KPI performance. Next, we present the evolution of the KPIs for the model maker and the radio agent in (resp.) Fig. 3 and Fig. 4. From Fig. 3, one can notice that barring a few initial decision windows when the model maker is still learning, a high value of reward at convergence is maintained at the

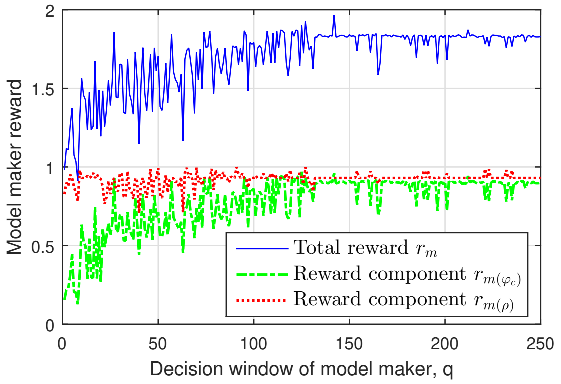


Fig. 2: Convergence of the model maker

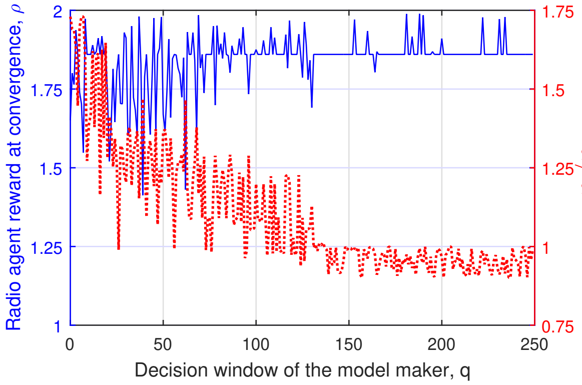


Fig. 3: Model maker KPIs

radio agent. This signifies that the inference model $\mathcal{M}_I$ is making high-quality radio resource allocation decisions at the DU. Besides, Fig. 4 also underlines that, once $\mathcal{M}_I$ is tuned in at the DU, the radio KPIs consistently fall below their respective target values. For these plots, we have considered $\tau_p = 1\,\text{ms}$ and $\tau_q = 2,000\tau_p$, hence the learning of the radio agent is faster compared to the learning of the model maker. The choice of coefficient $2,000$ is motivated from our initial experiments wherein we observed that in most cases, the radio agent converges within $2000\tau_p$. Further, Fig. 3 shows that, as the model maker learns the meta learning policy, the ratio of computational resources consumed by $\mathcal{M}_I$ with respect to the available computational resources at the DU converges close to 1. In other words, the proposed MERGE framework is able to ensure sufficiently high-quality decisions while satisfying the computational constraints of the DU.

Selection of pruned radio models. Fig. 5 presents some statistics related to the selection of pruned radio models. The plots underline that the subset of selected pruned models mostly comprises either 1, 2, or 3 pruned radio models, with preferred pruning factors as 0.5, 0.6, and 0.8. This suggests that, on the one hand, the meta learning policy discourages the selection of a large number of pruned radio models, or heavier models (i.e., with lower pruning factor) in the wake of computational constraints at the DU. On the other hand,

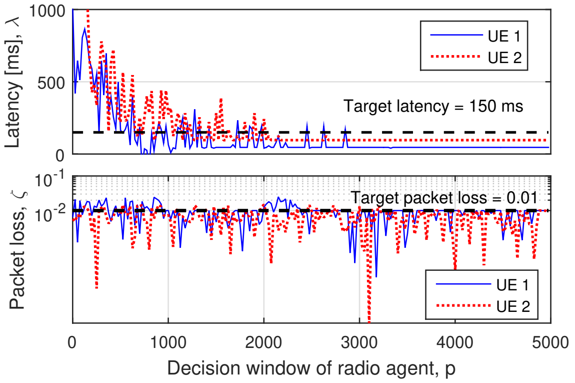
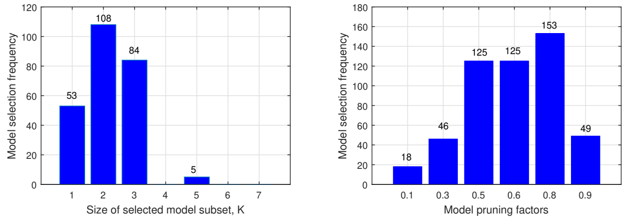


Fig. 4: Radio agent KPIs



(a) (b)

Fig. 5: Model selection statistics from the model maker

the selection of very light models is less likely due their poor decision-making capability. Hence, a combination of a few moderately sized pruned radio models is best to obtain high-quality decisions at the computationally constrained DUs.

Comparison with the baseline approaches. We now compare the performance of MERGE against two baseline approaches. In the first, we preserve the quality of decision making by implementing the full radio model, assuming no computational constraints at the DU: this is the best case of radio resource allocation decisions and acts as the benchmark for the meta-learning policy. From our dataset, we remark that $\varphi_a = 0.8$ is enough to run the full radio model. Here, φ_a, denoting the available computational resources at the DU, is the CPU limit of the containerized implementation of the vRAN in the testbed using which the dataset has been reported. In the second benchmark, we compromise on the quality of decisions while adhering to the computational constraints at the DU, i.e., a single pruned model suiting the available computational resources at the DU makes resource allocation decisions, and there is no meta learning. The comparison in terms of reward of the radio agent (which, in turn, signifies the QoS satisfaction of the UEs) is presented in Fig. 6. We observe that $\mathcal{M}_I$, as derived from MERGE, performs at par with the best case as the learning converges, albeit consuming less computational resources. Also, Fig. 6 shows that, given the same availability of computational resources, MERGE outperforms a single pruned model at the DU, with the latter being selected as the best pruned model that the DU can afford. Specifically, MERGE matches the performance of the full radio model with 25% less computational requirements, and for a given computational resource, its QoS satisfaction is

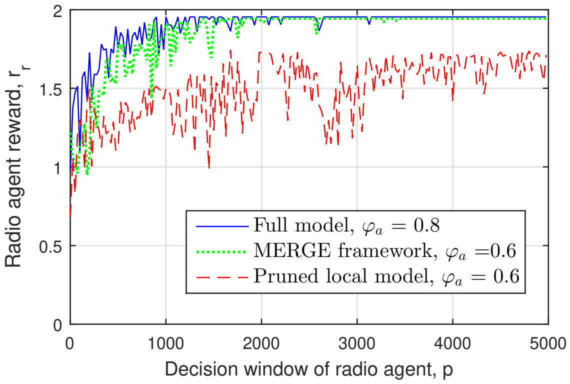


Fig. 6: Comparison of MERGE against two benchmarks

19% better in comparison to the single pruned model.

IV. Related Work

Resource allocation in vRAN through deep learning is fairly common in the literature and is of interest, particularly for large-scale networks [2]. However, owing to the huge computational requirements, their use in resource-constrained edge environments is limited. Thus, exploring techniques to reduce the size of deep learning models, yet maintain their functional accuracy, is of great interest. In this respect, the state-of-the-art can be classified into three broad areas, (i) Pruning, (ii) Federated learning (FL), and (iii) Meta-learning.

Pruning approaches are designed to reduce the size of deep networks subject to the QoS target of the KPIs. For instance, [8] proposes a DRL model to adaptively prune deep networks deployed on IoT devices, in order to minimize their energy consumption.While this approach works independently on stand-alone devices, [9] proposes a collaborative compression scheme to reduce the deep network size used for human activity recognition on mobile devices. [10], instead, partitions the deep network into physical topology aware sub-nets and simultaneously prunes them to minimize the communication costs involved in distributed inference.

The pruning approaches often suffer from model localization, thereby impacting the learning quality. This issue is addressed using FL, which allows for model training from multiple compute-constrained local learning agents while preserving data privacy [11], [12]. In [13], the compute cost of FL is further reduced using ensembles of pruned deep networks. [14], instead, adapts the model size to the nodes' computing capability and data sets. Conceptually, FL enables learning one task across multiple agents, thereby leading to richer learning even for the agents running on compute constrained devices. However, the output of FL agent is not adapted for each participating agent, and may limit its performance in a compute constrained scenario. Such tuning can be enabled using meta-learning policies, which are the most general among all. Recently, meta-learning in mobile edge networks have been proposed for task offloading, scheduling and resource allocation [15]–[17]. Here, we highlight that the existing meta-learning approaches may address the compute constraints in edge computing to some extent, however, none of them has been explored for radio resource allocation in disaggregated vRANs as we do. Through MERGE, we address in a vRAN environment, the model localization issue of pruning methods, as well as create local agent constraints-specific deep networks for radio resource allocation, which lacks in FL. To our knowledge, such comprehensive, scalable learning architecture for distributed radio resource allocation under computational constraints in vRAN has not been proposed so far.

V. Conclusions

To support decision making based on distributed learning that suits the capability of a disaggregated vRAN, we designed a meta-learning crowdsourcing approach, named MERGE. In MERGE, DUs use radio agents for radio resource allocation decisions. The CU collects such models and properly selects and aggregates them to create up-to-date radio agents that produce high-quality decisions at the DUs, while meeting their computational constraints. Our results show that MERGE matches the best case with 25% less computational requirements, and, for a given computational resource, it outperforms the QoS provided by a single pruned model by 19%.

As future work, we are investigating MERGE in more complex scenarios, with multiple DUs as well as intelligent services that have to be executed at the edge vRAN, thus increasing the system heterogeneity and dimensionality.

References

[1] *3GPP TS 38.470, "F1 general aspects and principles, (Rel. 17)*, 2022.

[2] B. Brik *et al.*, "Deep learning for B5G open radio access network: Evolution, survey, case studies, and challenges," *IEEE Open J. Comm. Soc.*, vol. 3, 2022.

[3] S. Tripathi *et al.*, "Fair and scalable orchestration of network and compute resources for virtual edge services," *IEEE Trans. Mob. Comp.*, 2023.

[4] *3GPP TS 23.501 v16.3.0 Tech. Spec. Group Services and System Aspects; System architecture for the 5G System; Stage 2, (Rel. 16)*, 2019.

[5] S. Tripathi *et al.*, "A context-aware radio resource management in heterogeneous virtual RANs," *IEEE Trans. Cogn. Comm. Net.*, 2022.

[6] N. Sanghi, *Deep Reinforcement Learning with Python*. New York, NY: Apress Berkeley, CA, 2021.

[7] E. Diao *et al.*, "HeteroFL:computation and communication efficient federated learning for heterogeneous clients," in *ICLR*, 2021.

[8] M. Zawish *et al.*, "Energy-aware AI-driven framework for edge-computing-based iot applications," *IEEE Internet Things J.*, 2023.

[9] J. Liang *et al.*, "A collaborative compression scheme for fast activity recognition on mobile devices via global compression ratio decision," *IEEE Trans. Mob. Comp.*, 2023.

[10] T. Jian *et al.*, "Communication-aware DNN pruning," in *INFOCOM*, 2023.

[11] Y. Jiang *et al.*, "Model pruning enables efficient federated learning on edge devices," *IEEE Trans Neural Netw. Learn Syst.*, pp. 1–13, 2022.

[12] Z. Jiang *et al.*, "Computation and communication efficient federated learning with adaptive model pruning," *IEEE Trans. Mob. Comp.*, 2023.

[13] B. Alhalabi *et al.*, "Fednets: Federated learning on edge devices using ensembles of pruned deep neural networks," *IEEE Access*, vol. 11, 2023.

[14] F. Malandrino *et al.*, "Matching DNN compression and cooperative training with resources and data availability," in *INFOCOM*, 2023.

[15] Z. Zhang *et al.*, "MR-DRO: A fast and efficient task offloading algorithm in heterogeneous edge/cloud computing environments," *IEEE Internet Things J.*, vol. 10, no. 4, 2023.

[16] S. Chen *et al.*, "Cache-assisted collaborative task offloading and resource allocation strategy: A metareinforcement learning approach," *IEEE Internet Things J.*, vol. 9, no. 20, 2022.

[17] K. Min *et al.*, "Meta-scheduling framework with cooperative learning towards beyond 5G," *IEEE JSAC*, 2023.