This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2024.3480136

Accurate Prediction of Multi-Dimensional Required Resources in 5G via Federated Deep Reinforcement Learning

Haojun Huang, Qifan Wang, Weimin Wu, Miao Wang, and Geyong Min

Abstract—The accurate prediction of required resources in terms of storage, computing and bandwidth is essential for 5G to host diverse services. The existing efforts illustrate that it is more promising to efficiently predict the unknown required resources with a third-order tensor compared to the 2D-matrix-based solutions. However, most of them fail to leverage the inherent features hidden in network traffic like temporal stability and service correlation to build a third-order tensor for the multi-dimensional required resource prediction in an intelligent manner, incurring coarse-grained prediction accuracy. Furthermore, it is difficult to build a third-order tensor with rate-varied measurements in 5G due to different lengths of measurement time slots. To address these issues, we propose an Accurate Prediction of Multi-Dimensional Required Resources (APMR) approach in 5G via Federated Deep Reinforcement Learning (FDRL). We first confirm the resource requests originated from different Base Stations (BSs) at varied measurement rates have similar features in service and time domains, but cannot directly form a series of regular tensors. Built on these observations, we reshape these measurement data to form a series of standard third-order tensors with the same size, which include many elements obtained from measurements and some unknown elements needed to be inferred. In order to obtain accurately predicted results, the FDRL-based tensor factorization approach is introduced to intelligently utilize multiple specific iteration rules for local model learning, and the accuracy-aware and latency-based depreciation strategies are exploited to aggregate local models for resource prediction. Extensive simulation experiments demonstrate that APMR can accurately predict the multi-dimensional required resources compared to the state-of-the-art approaches.

Index Terms—Tensor factorization, federated deep reinforcement learning, resource prediction, radio access networks, tensor reshaping.

1 Introduction

5G networks characterized by high speed, low latency, large capacity and ubiquitous connectivity have prompted the emergence of various innovative applications, including autonomous driving, virtual reality and Industrial Internet of Things (IIoT) [1], [2]. Generally, these applications require on-demand provisioning of computing, storage and bandwidth resources at access networks with Service Level Agreements (SLAs) between Internet Service Providers (ISPs) and users. Once user requests arrive at the same time, network workloads will burst such that there are not enough resources available to users. Contrarily, if network workloads stay at an under-loaded level, there will be too many idle resources, causing resource waste. The required-resource variations of such services will incur the over/under-provisioning of resources, and thus result in additional costs or poor SLAs in 5G [3], [4]. Consequently, it is necessary to implement the accurate prediction of the required multi-dimensional resources for hosting these services of 5G in a cost-efficient manner.

Recent work has largely leveraged model-driven approaches to predict 5G required resources by modeling it as matrix or tensor factorization to improve Quality of Service (QoS) [5], [6]. The core idea of them is to predict the unknown multi-dimensional required resources with the partial measurements observed from applications and users, enabling underlying resources to cope with the diverse services. All of these efforts primarily concentrate on QoS prediction [4], [6], [7], [8], energy consumption [5], resource availability [9], and workloads [1], [10], [11], [12], [13], [14], [15], built on specific regularities and time-variant correlations of consumed resources.

Despite significant progress has been made [3], there are two issues to be tackled for efficient prediction of required resources in 5G. First, how to design a standard tensor model under different measurement rates to describe the multi-dimensional resource requirements in 5G, which includes a great number of Base Stations (BSs). The current efforts are always concerned with the single resource prediction and are built on the same measurement rate, which is featured with the same number of time slots, to build the prediction model [10], [13], [14], [16]. However, in reality, 5G networks will consume more multi-dimensional resources in terms of computing, storage and bandwidth, rather than only one of them, to host diverse services. Furthermore, the real-world measurements are usually conducted at different measurement rates. This means that the original resource tensor in each BS is characterized by different sizes in computing, storage and bandwidth, and thus cannot be included in the same global model to predict required resources with high accuracy [17]. Hence, there is an urgent need to design a novel tensor reshaping algorithm which can reshape

H. Huang, Q. Wang and W. Wu are with the School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan 430074, China. Email: {hjhuang, wqf, wuwm}@hust.edu.cn.
W. Miao and G. Min are with the Department of Computer Science, University of Exeter, Exeter, EX44QF, UK. Email: {Wang.miao, g.min}@exeter.ac.uk.
Corresponding author: W. Wu.

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on October 28,2024 at 03:10:43 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2024.3480136

resource tensors of all BSs into the reshaped tensors with the same size. Second, how to efficiently predict the ever-changing required resources with the multiple available tensor factorization rules. A number of typical tensor completion solutions [18], [19], [20], [21] have been investigated with the exploration and exploitation of features hidden in measured data. However, the majority of them failed to jointly exploit multiple specific rules to intelligently derive the missing items with high prediction accuracy feedback. For example, the current tensor factorization approaches, i.e., CP-ALS [18], CP-WOPT [19] and GCP [20], always employ one single rule to infer the unknown required resources, which are not well jointly adopted in model iterations with the learned historical experiences of action exploration and the network-related factors behind model computation. This will incur inaccurate prediction results, due to the lack of comprehensive considerations of their superiorities and prediction accuracy feedback. It is required to design new tensor factorization strategies which jointly take into account the superiorities of all available rules to infer the missing elements with more prediction accuracy.

Nowadays, Federated Deep Reinforcement Learning (FDRL), which integrates Federated Learning (FL) with Deep Reinforcement Learning (DRL), has achieved notable advancements and exerted a substantial influence across diverse domains, including edge computing [8], [22], energy management [23] and orchestration of service function chains [24]. FDRL has powerful learning abilities to explore and exploit the hidden regularities and potential network-aware factors in model aggregation and policy optimization in an efficient manner, and thus become a promising paradigm for mitigating the issues outlined above. It can select a better action from multiple candidate actions, each of which follows one tensor factorization rule, to conduct resource prediction with more rewards at each state. Nevertheless, the current FDRL cannot converge quickly due to diverse local learning rewards obtained by the participating nodes. The DRL-based tensor factorization at each BS may train a sub-optimal local model in one episode, which has little positive feedback in model aggregation and thus affects the accuracy of the global model. Moreover, it is difficult for FDRL to aggregate all local models in each episode since every local model will be trained with different training times. It is required to design novel strategies for FDRL to quickly aggregate the local models, thus achieving accurate prediction results. These motivate us to design novel FDRL-based approaches to achieve accurate resource prediction in an efficient manner.

To bridge this gap, in this paper, we propose APMR, a FDRL-based accurate prediction of multi-dimensional required resources approach for 5G. The resource requirements in each BS have been modeled as a third-order tensor which contains measured data and unknown data needed to be predicted. Notice that different rate measurements will incur different numbers of time columns of resource tensor, thus we propose an efficient pre-filling-based tensor reshaping algorithm to establish a series of standard third-order tensors with the equal number of time columns. To obtain accurate prediction results, the FDRL-based framework, which introduces FL and DRL into the global model aggregation and local model training, respectively, has been designed to execute accurate reshaped tensor factorization with a set of available rules. Furthermore, the accuracy-aware and latency-based depreciation strategies have been introduced into FDRL-based model iteration to accelerate the global model convergence in an efficient manner. Distinguishing from the previous efforts, the main contributions of this paper can be summarized as follows:

- Notice that the resource requirements of different Radio Access Networks (RANs) in 5G are measured at different rates in terms of computing, storage and bandwidth, an efficient pre-filling-based tensor reshaping approach is designed to establish standard third-order tensors in all RANs. Built upon the inherited features of temporal stability and service correlation, the accurate prediction of the multi-dimensional required resources in a set of RANs is modeled as an FDRL-based tensor factorization problem.
- The FDRL-based framework is proposed to train an optimal global model for accurate prediction of the multi-dimensional required resources in 5G. In order to accelerate the global model convergence with the accuracy feedback, the accuracy-aware and latency-based depreciation strategies are designed by assigning higher weights to accurate and newly-learned local models. Moreover, to obtain accurate prediction results, the multiple specific iteration rules have been intelligently introduced to factorize resource tensor in parallel.
- Extensive simulation experiments and numerical analysis are implemented to evaluate the performance of APMR for predicting multi-dimensional required resources. Our simulation results demonstrate that APMR outperforms in prediction accuracy compared to other related approaches.

The rest sections of this paper are structured as follows. Section 2 provides an overview of related work. Section 3 presents the problem statement for accurate prediction of required resources, including preliminaries, network model and problem formulation. Subsequently, in Section 4, a FDRL-based framework is proposed for accurate resource prediction. In Section 5, we numerically evaluate the performance of APMR in extensive scenarios. Finally, we conclude the paper in Section 6.

2 Related Work

The resource prediction in 5G is of great importance to host various ongoing and upcoming services [16], [25]. More recently, there are different prediction approaches reported in the literature, which are interested in the prediction of resource requirements in energy consumption [5], QoS [6], [7], [8], mobility [22], resource availability [9], and workloads [4], [10], [11], [13], [14], [15].

In order to enhance prediction accuracy, many machine-learning-based resource prediction approaches have been developed to fulfill the QoS requirements of real-world applications. To further improve the failure prediction accuracy of the previous Deep-Learning (DL)-based methods, a failure prediction algorithm based on multi-layer Bidirectional Long Short Term Memory (BiLSTM) is proposed in [4], to identify task and job failures in the cloud. In

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on October 28,2024 at 03:10:43 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2024.3480136

[7], the Adaptive Matrix Factorization (AMF) is designed by introducing data transformation, online learning and adaptive weights to conduct online QoS prediction for candidate services. Considering various mobile edge computing (MEC) service scheduling scenarios, an adaptive QoS prediction strategy has been designed in [8] to select the optimal QoS-aware services. The efforts in [14], [15] have investigated the ensemble learning framework involving base-learners and decision making to obtain accurate prediction outcome. A novel method, Evolution Graph for Workload Prediction (EvoGWP), proposed in [13], employs a delicately designed graph-based evolution learning algorithm to predict long-term dynamic changes. In [26], a DL-based model is designed to predict available throughput for the non-standalone 5G. In [27], an Automated Model for Prediction (AMP), including bandwidth prediction and model auto-selection, is proposed to achieve low-latency communications in mobile networks. In [16], an Accurate Prediction of Required virtual Resources (APRR) approach via DRL is proposed, which adopts specific rules alternately to predict required resources. However, these investigations nearly cannot well capture the correlation among different types of resources, and thus acquire coarse-grained prediction results.

Notice that tensor completion with a certain amount of measured data can obtain the missing data, three efficient tensor-completion-based derivatives [21], [28], [29] have been developed. A novel localized tensor completion model (LTC) is presented in [21], to enhance recovery accuracy through exploiting the stronger local correlation of data to form and recover low-rank sub-tensors. In [28], a novel Neural Tensor Completion (NTC) scheme is proposed to capture the high-order and nonlinear correlations across different feature dimensions by modeling third-order interaction among data features and building a 3D interaction map. In [29], the data reconstruction ability of adversarial training with Generative Adversarial Networks (GANs) is exploited to infer the missing data. Distinguishing from the above-mentioned solutions, APMR first collects consumed resources and adopts pre-filling-based tensor reshaping to form a series of standard reshaped tensors. Then, the FDRL-based tensor factorization is applied by APMR, which introduces multiple rules specified by the existing tensor factorization, to conduct accurate prediction of multi-dimensional required resources in 5G. Finally, the predicted resource tensors with different sizes will be obtained through conducting the reverse procedure of pre-filling-based tensor reshaping.

TABLE 1
The Important Parameters and Notations

Notations	Descriptions
$\boldsymbol{\mathcal{X}}/\boldsymbol{\mathcal{D}}$	The original/reshaped resource tensor
$\boldsymbol{M}/\boldsymbol{N}/\boldsymbol{T}$	R-rank factor matrices of $\widehat{\boldsymbol{\mathcal{D}}}$ with the sizes of $M\times R$, $N\times R$, and $3\times R$, respectively
f_i/t_j	The i-th service and j-th time slot
$d(i,j,k)/\widehat{d}(i,j,k)$	The required/predicted resource of service f_i at t_j for resource type k
$\omega_{i,j,k}$	Binary variable to describe whether $d(i,j,k)$ is contained in $\boldsymbol{\mathcal{X}}$
Q_p/Q_t	The action-value function of predictive/target Q-network
$\boldsymbol{\theta}(i)/\theta_t^z/\theta_t'^z$	The global model of cloud server and predictive/target Q-network model of local client c_z
β_i^z	Aggregation weight of θ_t^z at global episode i
$<\mathcal{S},\mathcal{A},\mathcal{R}>$	MDP-based three-tuple containing state, action and reward
G_t	The return from state S_t to terminal state
π^*	The optimal policy trained by agent

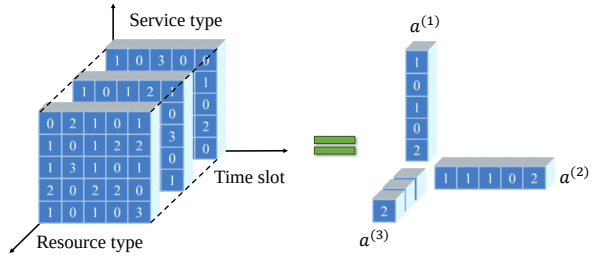


Fig. 1. Third-order rank-one tensor factorized into the outer product of three vectors.

3 Preliminaries and Problem Statement

In this section, we first present the preliminary knowledge and definitions of tensors, followed by the network model of resource requirements. Then, we illustrate the problem formulation of required resource prediction. For clarity, the main notations and their descriptions are listed in TABLE 1 to promote comprehension.

3.1 Preliminaries

Definition 1: Tensors. A tensor is defined as a multi-dimensional array and is a higher-order generalization of a vector (first-order tensor) and a matrix (second-order tensor). An $\mathbb{N}$-way or $\mathbb{N}$th-order tensor is denoted by $\boldsymbol{\mathcal{X}}\in\mathbb{R}^{I_1\times I_2\times\cdots\times I_{\mathbb{N}}}$, where $\mathbb{N}$ is the order of $\boldsymbol{\mathcal{X}}$, also called way or mode. The elements of $\boldsymbol{\mathcal{X}}$ are represented by $x_{i_1,i_2,\ldots,i_{\mathbb{N}}}$, $i_n\in[1,\ldots,I_{\mathbb{N}}]$ with $1\leq n\leq\mathbb{N}$.

Definition 2: Rank-One Tensor and Tensor Rank. An $\mathbb{N}$-way tensor $\boldsymbol{\mathcal{X}}\in\mathbb{R}^{I_1\times I_2\times\cdots\times I_{\mathbb{N}}}$ is rank one if it can be calculated by the outer product of $\mathbb{N}$ vectors, i.e., $\boldsymbol{\mathcal{X}}=a^{(1)}\circ a^{(2)}\circ\ldots\circ a^{(\mathbb{N})}$, where $a^{(n)}(1\leq n\leq\mathbb{N})$ denotes a vector in $\mathbb{R}^{I_n}$ and the symbol $\circ$ represents the outer product. Let $a_{i_n}^{(n)}$ represent the vector elements of $a^{(n)}$. Therefore, the element $x_{i_1,i_2,\ldots,i_{\mathbb{N}}}$ of $\boldsymbol{\mathcal{X}}$ can be calculated by the product of the corresponding vector elements, given by

$$x_{i_1,i_2,\ldots,i_{\mathbb{N}}}=a_{i_1}^{(1)}\cdot a_{i_2}^{(2)}\cdot\cdots\cdot a_{i_{\mathbb{N}}}^{(\mathbb{N})},1\leq i_n\leq i_{\mathbb{N}}. \quad (1)$$

For ease of understanding, a rank-one tensor $(\mathbb{N}=3)$ can be factorized into the outer product of three vectors as shown in Fig. 1. The tensor rank R of $\boldsymbol{\mathcal{X}}$ also known as CP-rank, measures the least number of rank-one tensors which can be combined to $\boldsymbol{\mathcal{X}}$, denoted by $R=rank(\boldsymbol{\mathcal{X}})$.

Definition 3: Kronecker and Khatri–Rao product. The matrix resulting from the $Kronecker$ product of matrices $A\in\mathbb{R}^{I\times J}$ and $B\in\mathbb{R}^{K\times L}$ is denoted as $A\otimes B$. The resulting

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on October 28,2024 at 03:10:43 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2024.3480136

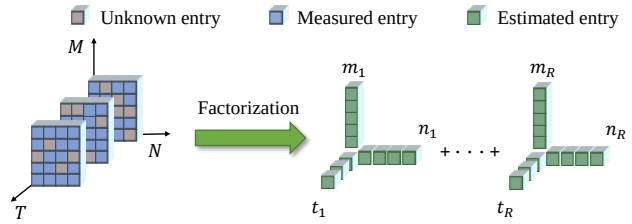


Fig. 2. CP decomposition of a third-order tensor.

matrix has dimensions $IK \times JL$ and is defined by

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \dots & a_{1J}B \\ a_{21}B & a_{22}B & \dots & a_{2J}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}B & a_{I2}B & \dots & a_{IJ}B \end{bmatrix} \tag{2}$$
$$= \begin{bmatrix} a_1 \otimes b_1 & \dots & a_1 \otimes b_L & a_2 \otimes b_1 & \dots & a_I \otimes b_L \end{bmatrix},$$

where $a_j (j \in [1, J])$ and $b_l (l \in [1, L])$ denote the j-th column of A and the l-th column of B, respectively. The $Khatri–Rao$ product is the matching column-wise $Kronecker$ product [18]. Therefore, the $Khatri–Rao$ product of matrices $A \in \mathbb{R}^{I \times K}$ and $B \in \mathbb{R}^{J \times K}$ is denoted as

$$A \odot B = \begin{bmatrix} a_1 \otimes b_1 & a_2 \otimes b_2 & \dots & a_K \otimes b_K \end{bmatrix}, \tag{3}$$

with the size of $IJ \times K$. Especially, if A and B are vectors, their $Kronecker$ and $Khatri–Rao$ products are identical.

3.2 Network Model

It is considered that there are K RANs in 5G, each of which contains a BS and a number of User Equipments (UEs). Each UE can access to a BS, which is associated with a coverage area to provide different services. All of such services include high-speed data transmission, low-latency communications, highly reliable network connection, etc. The whole network is divided into K regions, determined by the location of BS and its UEs. Each BS can be regarded as a local client that implements multiple services, with a number of consumed resources, including computing, storage, and bandwidth. The network operators can monitor the resource requirements of existing services at each time slot. Each service consumes different quantities of resources depending on its role in the whole operation, and the working time of services has been divided into a series of time slots with different numbers due to different rate measurements. Thus, the required resources of various services can be counted at discrete time intervals, and denoted as a third-order tensor. The required resources of service f_i at time slot t_j in the z-th $(1 \le z \le K)$ RAN are recorded as a vector $(d^z(i,j,1), d^z(i,j,2), d^z(i,j,3))$ which represent resources in terms of computing, storage, and bandwidth, respectively. Each type of resource requirement in the z-th can be regarded as an $M \times N^z$ matrix $X_k^z (1 \le k \le 3)$ with $d^z(i,j,k)$, whose rows and columns represent different services and time slots, respectively. Therefore, a third-order tensor $\boldsymbol{\mathcal{X}}^z$ in the z-th RAN can be constructed with M kinds of service $f_1, ..., f_M$ at N time slots $t_1, ..., t_N$.

3.3 Problem Formulation

Built on the above-mentioned preliminaries and low-rank feature identified in our previous work [16], the accurate prediction of required resources in each RAN can be formulated as a low-rank tensor completion problem, which is proposed to estimate the unknown data of tensor $\boldsymbol{\mathcal{X}}^z (1 \le z \le K)$ through learning massive observed data. Wherein, tensors $\boldsymbol{\mathcal{X}}^z$ with different numbers of time columns are reshaped as $\boldsymbol{\mathcal{D}}^z$ with equal numbers of time columns.

For brevity, we will omit the superscript z of $\boldsymbol{\mathcal{D}}^z$ and $\boldsymbol{\mathcal{X}}^z$ when it cannot impact reading comprehension. As illustrated in Fig. 2, we can factorize $\boldsymbol{\mathcal{D}}$ into R rank-one tensors, which can be further factorized into the outer product of three vectors, given by

$$\boldsymbol{\mathcal{D}} \approx \widehat{\boldsymbol{\mathcal{D}}} = \sum_{r=1}^{R} m_r \circ n_r \circ t_r = [\![\boldsymbol{M}, \boldsymbol{N}, \boldsymbol{T}]\!], \tag{4}$$

where $\boldsymbol{M}=[\boldsymbol{m}_1, \dots, \boldsymbol{m}_R]$, $\boldsymbol{N}=[\boldsymbol{n}_1, \dots, \boldsymbol{n}_R]$, and $\boldsymbol{T}=[\boldsymbol{t}_1, \dots, \boldsymbol{t}_R]$ are matrices of dimensions $M \times R$, $N \times R$, and $3 \times R$, respectively. Wherein, the matrices $\boldsymbol{M}$, $\boldsymbol{N}$, and $\boldsymbol{T}$ can be used to compute the outer product, yielding the tensor $\widehat{\boldsymbol{\mathcal{D}}}$, which represents the predicted tensor with the size of $M \times N \times 3$ for approximating the tensor $\boldsymbol{\mathcal{D}}$. Let $m_{ri} (1 \le i \le N, 1 \le r \le R)$ be an element at row i of vector $\boldsymbol{m_r}$, likewise for n_{rj} and t_{rk} of vector $\boldsymbol{n_r}$ and $\boldsymbol{t_r}$. Based on Eq. (1), the element $d(i,j,k)$ can be estimated by $\widehat{d}(i,j,k)$, namely

$$d(i,j,k) \approx \widehat{d}(i,j,k) = \sum_{r=1}^{R} m_{ri} \cdot n_{rj} \cdot t_{rk}. \tag{5}$$

To accurately predict the multi-dimensional required resources in the z-th RAN, the DRL-based tensor factorization will be intelligently executed with the available rules, by minimizing the objective function $L(\boldsymbol{M}^z, \boldsymbol{N}^z, \boldsymbol{T}^z) = \|\boldsymbol{\mathcal{D}}^z - [\![\boldsymbol{M}^z, \boldsymbol{N}^z, \boldsymbol{T}^z]\!]\|_F^2$ between $\boldsymbol{\mathcal{D}}^z$ and $\widehat{\boldsymbol{\mathcal{D}}}^z$. Therefore, the accurate multi-dimensional required resource prediction in K 5G RANs can be formulated as the FDRL-based tensor factorization problem, which can be given by

$$\begin{aligned} &\min_{\{\boldsymbol{M}^z, \boldsymbol{N}^z, \boldsymbol{T}^z\}_{z=1}^K} \sum_{z=1}^{K} L(\boldsymbol{M}^z, \boldsymbol{N}^z, \boldsymbol{T}^z) \\ &= \sum_{z=1}^{K} \|\boldsymbol{\mathcal{D}}^z - [\![\boldsymbol{M}^z, \boldsymbol{N}^z, \boldsymbol{T}^z]\!]\|_F^2 \\ &= \sum_{z,i,j,k=1}^{K,M,N,3} \|d^z(i,j,k) - \sum_{r=1}^{R} m_{ri}^z n_{rj}^z t_{rk}^z\|_F^2. \end{aligned} \tag{6}$$

Here, m_{ri}^z, n_{rj}^z and t_{rk}^z represent the elements of factor matrices $\boldsymbol{M}^z$, $\boldsymbol{N}^z$ and $\boldsymbol{T}^z$, respectively, defined in Eq. (5).

4 FDRL-based 5G Resource Prediction

This section mainly proposes the APMR in detail for the required resource prediction via FDRL. We first outline the FDRL-based tensor factorization framework of APMR, and introduce the establishment of resource tensor. Then, the design of the Markov Decision Process (MDP)-based three-tuple and the specific process of DRL-based tensor factorization are illustrated, followed by the recovery of resource tensor.

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on October 28,2024 at 03:10:43 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2024.3480136

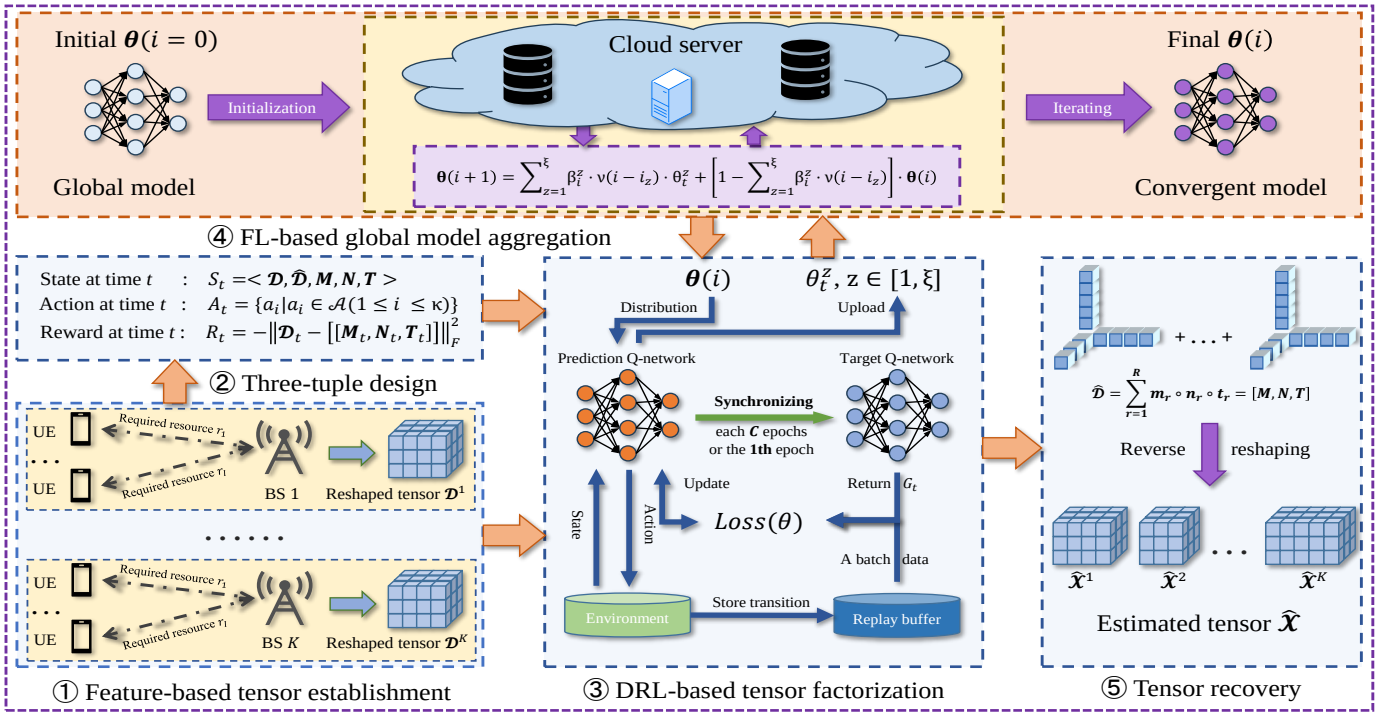


Fig. 3. The framework of APMR, which includes the establishment of resource tensor, MDP-based three-tuple design, DRL-based tensor factorization, FL-based global model aggregation and tensor recovery.

4.1 The Framework of APMR

The core idea of APMR is to design an intelligent framework which takes advantage of FDRL-based exploration and exploitation to accurately predict the required 5G resources as defined in Eq. (6). Specifically, the diverse resource requirements of each RAN are measured under different rate measurements, and established as a resource tensor based on their features, respectively. Then, each resource tensor is reshaped into a standard tensor with the same size. After parameter initialization, each client employs an agent to intelligently select one of the available iteration rules by Q-network to factorize tensors step by step. Next, the cloud server conducts model aggregation built on the local models received from clients within a specific period of time, and broadcasts this model for further training. In this way, a globally optimal model can be obtained via FDRL, meaning that the unmeasured resource requirements can be accurately predicted through conducting tensor factorization and reverse tensor reshaping. APMR can work well as the scale of 5G networks increases. The operations of APMR, as shown in Fig. 3, consist of five steps: ①feature-based resource tensor establishment, ②three-tuple design, ③DRL-based tensor factorization, ④FL-based global model training, and ⑤tensor recovery.

There are K BSs as the client nodes $c_1, c_2, \ldots, c_K$ to execute local training and send local-trained models $\theta_t^z (1\le z\le K)$ at time t to the cloud server. Wherein, the local training combines Deep Q-Networks (DQNs) with a set of tensor factorization rules to update local model θ_t^z at each epoch. Then, the local model is uploaded to the cloud server from client node for the global model aggregation in an asynchronous manner, as shown in Fig. 4. The i-th global episode with the duration of T is designed to decide when the global model $\boldsymbol{\theta}(i)$ is updated, each of them contains a series of

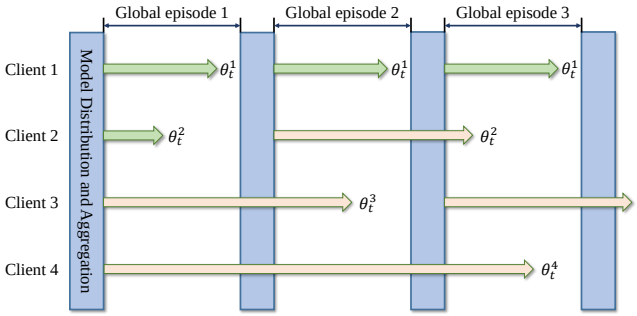


Fig. 4. Federated learning in asynchronous pattern.

epochs. It is assumed that there are some local clients like c_z finished local training and got a converged model θ_t^z in the i-th global episode, respectively. Then these local models like θ_t^z will be uploaded to the cloud server and engage in the global model updating. In the next global episode, the updated global model $\boldsymbol{\theta}(i+1)$ will be distributed to these idle clients for further training. The operations of model aggregation will be conducted one global episode after another until the global model reaches a converged state or the maximum number of global episodes.

In order to accelerate the global model convergence in an optimal iterative manner, the accuracy-aware weighted strategy is proposed to adequately utilize available high-accuracy models. With $\boldsymbol{\theta}(i)$ and $\xi(1<\xi\le K)$ local models $\theta_t^1,\ldots,\theta_t^\xi$ received from RANs, the $(i+1)$-th global model $\boldsymbol{\theta}(i+1)$ updated at the cloud server can be represented as

$$\boldsymbol{\theta}(i+1)=\sum_{z=1}^{\xi}\beta_i^z\cdot\theta_t^z+(1-\sum_{z=1}^{\xi}\beta_i^z)\cdot\boldsymbol{\theta}(i), \quad (7)$$

where the high-precision local model θ_t^z will be assigned the higher weight $\beta_i^z=(p_t^z\cdot\beta)/(\sum_{z=1}^{\xi}p_t^z)$ to aggregate global model. The parameter $p_t^z=1-\|(\mathcal{D}^z-\widehat{\mathcal{D}}^z)\|_F^2/\|\mathcal{D}^z\|_F^2$ represents tensor factorization accuracy of the local model θ_t^z, and

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on October 28,2024 at 03:10:43 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2024.3480136

β is a federated learning-rate factor designed to control the magnitude of global model updates in each iteration.

Notice that each agent will take different times to learn one local model with various rewards, the latency-based depreciation function $\nu(\cdot)$ is introduced to control the weight factor of global model aggregation built on the outdated degree of local model θ_t^z, defined as

$$\nu(i-i_z)=\begin{cases}1, & i-i_z\leq 1,\\ \frac{b}{i-i_z}, & i-i_z>1,\end{cases} \tag{8}$$

where i_z represents the global episode corresponding to the last model received from the cloud server, and b is a constant controlling the intensity of the depreciation. Built on this, the global model $\boldsymbol{\theta}(i+1)$ defined in Eq. (7) can be further updated as

$$\begin{aligned}\boldsymbol{\theta}(i+1)=&\sum\nolimits_{z=1}^{\xi}\beta_i^z\cdot\nu(i-i_z)\cdot\theta_t^z\\&+[1-\sum\nolimits_{z=1}^{\xi}\beta_i^z\cdot\nu(i-i_z)]\cdot\boldsymbol{\theta}(i).\end{aligned} \tag{9}$$

To enhance the training gain of the global model, all hyper parameters of local and global training like local learning rate η and federated learning-rate factor β in the i-th global episode will be fine-tuned and introduced into $\boldsymbol{\theta}(i+1)$.

4.2 Feature-based Resource Tensor Establishment

The resource tensors in all RANs consist of three sub-matrices corresponding to the computing, storage and bandwidth resources of diverse services under different rate measurements over a period of time, respectively. Through monitoring of resource request traffic of each RAN conducted by network service providers, the amount of required resource-related data of M types of services will be firstly collected. Then, the consumed resources in the z-th$(1\leq z\leq K)$ RAN can be divided into N^z time slots depending on its measurement rate, meaning that N^i may not be equal to $N^j(1\leq i,j\leq K)$ due to different measurement rates. In this way, resource tensors of different RANs are established as $\boldsymbol{\mathcal{X}}^z$ with the size of $M\times N^z\times 3$ based on the features extracted from temporal stability and service correlation, which are crucial prerequisites for accurately predicting resource requirements [16], [30].

Due to the resource requirements of each RAN measured under different rate measurements, there are different numbers of time slot columns included in resource tensors $\boldsymbol{\mathcal{X}}^1$, $\boldsymbol{\mathcal{X}}^2,\ldots,\boldsymbol{\mathcal{X}}^K$, as shown in Fig. 5. This means that each tensor is characterized by different sizes of time, and thus cannot be jointly factorized with high accuracy. Essentially, each sub-matrix completion of tensor $\boldsymbol{\mathcal{X}}^z$ can be formulated as

$$\begin{aligned}&\min_{\boldsymbol{U}_k^z,\boldsymbol{\Sigma}_k^z,\boldsymbol{V}_k^z}L(\boldsymbol{U}_k^z,\boldsymbol{\Sigma}_k^z,\boldsymbol{V}_k^z)\\&=\|(\boldsymbol{X}_k^z-\boldsymbol{U}_k^z\boldsymbol{\Sigma}_k^z(\boldsymbol{V}_k^z)^T)\|_F^2,\end{aligned} \tag{10}$$

where $\boldsymbol{X}_k^z(1\leq k\leq 3)$ is the k-th sub-matrix of tensor $\boldsymbol{\mathcal{X}}^z$, while $\boldsymbol{U}_k^z$, $\boldsymbol{V}_k^z$ and $\boldsymbol{\Sigma}_k^z$ represent factor and diagonal matrices in the Singular Value Decomposition (SVD) of $\boldsymbol{X}_k^z$, respectively. The sub-matrix completion defined in Eq. (10) can be addressed with higher accuracy if all such sub-matrices in RANs can be reshaped into the matrices with the same size by updating their time slot columns and then integrating them into a tensor in each BS. To achieve this goal, the CP-ALS [18] is firstly introduced to fill the unknown elements of the sub-matrices, which can eliminate their negative effect on rank and SVD calculation.

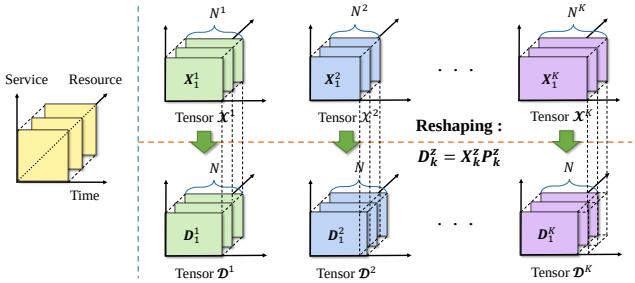


Fig. 5. Overview process of third-order resource tensor reshaping.

Notice that $\boldsymbol{X}_1^z,\boldsymbol{X}_2^z$ and $\boldsymbol{X}_3^z$ of each tensor exist similar features in function consumption, then we introduce $\boldsymbol{M}^z{=}\boldsymbol{U}_k^z(\boldsymbol{M}^z{\in}\mathbb{R}^{M\times N})$ to update matrix $\boldsymbol{U}_k^z$ defined in Eq. (10). Here, N is the average number of time slots of tensors in all RANs, denoted by $N{=}\lceil\frac{1}{K}\sum_{z=1}^{K}N^z\rceil$. In order to capture the common features in the time domain, we introduce an invariance constraint into factor matrix $\boldsymbol{V}_k^z$, namely $(\boldsymbol{V}_k^z)^T\boldsymbol{V}_k^z$ is constant over diverse types of resources. Therefore, $\boldsymbol{V}_k^z$ can be represented as $\boldsymbol{V}_k^z{=}\boldsymbol{P}_k^z\boldsymbol{N}^z$, where $\boldsymbol{N}^z{\in}\mathbb{R}^{N\times N}$ is constant in different sub-matrices of $\boldsymbol{\mathcal{D}}^z$, and $\boldsymbol{P}_k^z{\in}\mathbb{R}^{N^z\times N}$ is a row-wise orthogonal matrix when N is greater than N^z, and otherwise, it is a column-wise orthogonal matrix with $(\boldsymbol{P}_k^z)^T\boldsymbol{P}_k^z{=}\boldsymbol{I}$. Based on these, we introduce $\boldsymbol{P}_k^z$ into sub-matrix completion defined in Eq. (10) to reshape $\boldsymbol{X}_1^z,\boldsymbol{X}_2^z$ and $\boldsymbol{X}_3^z$ by $\boldsymbol{D}_k^z{=}\boldsymbol{X}_k^z\boldsymbol{P}_k^z$. Thus, Eq. (10) can be further described as

$$\begin{aligned}&\min_{\boldsymbol{M}^z,\boldsymbol{\Sigma}_k^z,\boldsymbol{N}^z}L(\boldsymbol{M}^z,\boldsymbol{\Sigma}_k^z,\boldsymbol{N}^z)\\&=\|(\boldsymbol{D}_k^z-\boldsymbol{M}^z\boldsymbol{\Sigma}_k^z(\boldsymbol{N}^z)^T)\|_F^2.\end{aligned} \tag{11}$$

The problem defined in Eq. (11) is devoted to the SVD-based matrix factorization for resource prediction in 5G. Experientially, it can be tackled with higher accuracy through tensor factorization if all matrices with the same size can be integrated into a tensor to recover together. Built on this observation, we introduce $\boldsymbol{T}^z$, a factor matrix obtained by $\boldsymbol{\Sigma}_k^z{=}diag(T_{k:}^z)$, where $T_{k:}^z$ is the k-th row of the matrix $\boldsymbol{T}^z$, to ensure all diagonal matrices included in all RANs have the same size. Therefore, built on the factorization relationship between tensor and resource type slice [17], the problem described in Eq. (11) can be formulated as the following tensor factorization:

$$\begin{aligned}&\min_{\boldsymbol{M}^z,\boldsymbol{N}^z,\boldsymbol{T}^z}L(\boldsymbol{M}^z,\boldsymbol{N}^z,\boldsymbol{T}^z)\\&=\|(\boldsymbol{\mathcal{D}}^z-[\![\boldsymbol{M}^z,\boldsymbol{N}^z,\boldsymbol{T}^z]\!])\|_F^2,\end{aligned} \tag{12}$$

where $\boldsymbol{\mathcal{D}}^z$ denotes a tensor combined by matrices $\boldsymbol{D}_k^z$. In this way, the feature-based reshaped tensors $\boldsymbol{\mathcal{D}}^1$, $\boldsymbol{\mathcal{D}}^2$, $\ldots$, $\boldsymbol{\mathcal{D}}^K$ with the same sizes in K RANs are established built on the original tensors $\boldsymbol{\mathcal{X}}^1$, $\boldsymbol{\mathcal{X}}^2,\ldots,\boldsymbol{\mathcal{X}}^K$, which have different time slots as shown in Fig. 5. Benefiting from the multiplication of a tensor by an orthogonal matrix not altering the low-rank feature of the tensor, the prediction of required resources still can be treated as a low-rank tensor completion problem with the goal of minimizing the objective function discrepancy between $\widehat{\boldsymbol{\mathcal{D}}}^z$ and $\boldsymbol{\mathcal{D}}^z$. After $\boldsymbol{M}^z,\boldsymbol{N}^z$ and $\boldsymbol{T}^z$ are obtained, the reshaped sub-matrices

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on October 28,2024 at 03:10:43 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2024.3480136

can be recovered with the reverse procedure of reshaping, i.e., $\widehat{\boldsymbol{X}}_k^z = \boldsymbol{D}_k^z(\boldsymbol{P}_k^z)^T = \boldsymbol{M}^z\boldsymbol{\Sigma}_k^z(\boldsymbol{N}^z)^T(\boldsymbol{P}_k^z)^T$.

Algorithm 1: Tensor Reshaping and Recovery

Input: The original $\boldsymbol{\mathcal{X}}^z(1 \le z \le K)$
Output: The recovered $\boldsymbol{\mathcal{X}}^z$

1 Initialize $\boldsymbol{M}^z$, $\boldsymbol{N}^z$, $\boldsymbol{\Sigma}_k^z$, $\boldsymbol{T}^z$
2 **for** $z \in [1, K]$ *in a parallel manner* **do**
3 Pre-fill $\boldsymbol{\mathcal{X}}^z$ by CP-ALS
4 **repeat**
5 Compute $\boldsymbol{N}^z\boldsymbol{\Sigma}_k^z(\boldsymbol{M}^z)^T\boldsymbol{X}_k^z = \boldsymbol{U}_k^z\boldsymbol{\Delta}_k^z(\boldsymbol{V}_k^z)^T$ by Truncated SVD (TSVD)
6 **if** $N < N^z$ **then**
7 $\boldsymbol{P}_k^z = \boldsymbol{V}_k^z(\boldsymbol{U}_k^z)^T$
8 **else**
9 $\boldsymbol{P}_k^z = \boldsymbol{U}_k^z(\boldsymbol{V}_k^z)^T$
10 Establish $\boldsymbol{\mathcal{D}}^z$ by $\boldsymbol{D}_k^z = \boldsymbol{X}_k^z\boldsymbol{P}_k^z$
11 Update $\boldsymbol{M}^z, \boldsymbol{N}^z, \boldsymbol{T}^z$ by intelligently selecting factorization rules according to Algroithm 2
12 $\boldsymbol{\Sigma}_k^z \leftarrow diag(T_{k:}^z)$
13 Update sub-matrices of $\boldsymbol{\mathcal{X}}^z$ by $\boldsymbol{X}_k^z \leftarrow \boldsymbol{W}_k^z \odot \boldsymbol{X}_k^z + (\boldsymbol{E} - \boldsymbol{W}_k^z) \odot \boldsymbol{M}^z\boldsymbol{\Sigma}_k^z(\boldsymbol{N}^z)^T(\boldsymbol{P}_k^z)^T$
14 **until** $\boldsymbol{\mathcal{X}}^z$ *converges*
15 **return** $\boldsymbol{\mathcal{X}}^z$

The pseudocode of pre-filling-based tensor reshaping and recovery is summarized in Algorithm 1. First of all, the initialization is conducted as shown in Line 1. Next, Line 2 illustrates the whole operations which are executed in a parallel manner. After $\boldsymbol{\mathcal{X}}^z$ is pre-filled in Line 3, the calculation of $\boldsymbol{P}_k^z$ and the process of tensor reshaping are described in Lines 6-9 and 10, respectively. Following that, the problem of tensor factorization is addressed in Line 11, and the procedure of tensor recovery is exhibited in Lines 12-13.

4.3 3-Tuple Design

The local training of DRL-based required resource prediction is carried out by an agent in BS of each RAN to achieve the specified objective as described in Eq. (12). Essentially, it is a classical MDP-based iteration process to implement iterative updates of a three-tuple $\mathscr{M} = <\mathcal{S}, \mathscr{A}, \mathscr{R}>$, which involves the state space $\mathcal{S}$, the taken action space $\mathscr{A}$, and the subsequent reward space $\mathscr{R}$ after taking actions. The details of the MDP-based three-tuple design are introduced as follows.

(1) State space: $\mathcal{S}$ contains the state of reshaped tensor $\boldsymbol{\mathcal{D}}$, predicted tensor $\widehat{\boldsymbol{\mathcal{D}}}$, factor matrices $\boldsymbol{M}, \boldsymbol{N}$ and $\boldsymbol{T}$ with the sizes of $M \times N \times 3$, $M \times N \times 3$, $M \times R$, $N \times R$ and $3 \times R$, respectively. The state values of such factor matrices will be stretched into vectors and concatenated to form the inputs for the local models. Built on this, the state S_t of the environment in each RAN at time t can be denoted as

$$S_t = < \boldsymbol{\mathcal{D}}, \widehat{\boldsymbol{\mathcal{D}}}_t, \boldsymbol{M}_t, \boldsymbol{N}_t, \boldsymbol{T}_t > . \tag{13}$$

The state S_t will transit to the next state S_{t+1} if action A_t is successfully executed by the agent.

(2) Action space: $\mathscr{A}$ represents the set of specialized iterative factorization actions to factorize the predicted tensor $\widehat{\boldsymbol{\mathcal{D}}}$ at each step, and can be expressed as

$$\mathscr{A} = \{a_1, a_2, \ldots, a_\kappa\}. \tag{14}$$

Where, action $A_t = \{a_i | a_i \in \mathscr{A}(1 \le i \le \kappa)\}$ denotes the chosen action at time t to implement tensor factorization following specific rules. In order to facilitate the understanding, three iteration rules associated with actions a_1, a_2 and a_3 specified by CP-ALS [18], AdaGrad-based GCP and Adam-based GCP [20], respectively, are described as follows.

CP-ALS-based rule. The iteration rule of factor matrix $\boldsymbol{M}_t$, $\boldsymbol{N}_t$ and $\boldsymbol{T}_t$ can be expressed as

$$\begin{cases} \boldsymbol{M}_{t+1} = \mathbb{D}_t^{M \times NT}((\boldsymbol{N}_t \odot \boldsymbol{T}_t)^T)^\dagger, \\ \boldsymbol{N}_{t+1} = \mathbb{D}_t^{N \times MT}((\boldsymbol{T}_t \odot \boldsymbol{M}_t)^T)^\dagger, \\ \boldsymbol{T}_{t+1} = \mathbb{D}_t^{T \times MN}((\boldsymbol{M}_t \odot \boldsymbol{N}_t)^T)^\dagger, \end{cases} \tag{15}$$

where $\mathbb{D}_t^{M \times NT}$, $\mathbb{D}_t^{N \times MT}$ and $\mathbb{D}_t^{T \times MN}$ are $M \times NT$, $N \times MT$ and $T \times MN$ unfolding matrix of $\boldsymbol{\mathcal{D}}$ at time t, respectively. The symbols $\odot$ and $\dagger$ indicate the column-wise $Khatri\text{–}Rao$ product and the Moore-Penrose generalized inverse of matrices, respectively.

AdaGrad-based GCP rule. The iteration rule of factor matrix $\boldsymbol{M}_t$, $\boldsymbol{N}_t$ and $\boldsymbol{T}_t$ can be given by

$$\begin{cases} \boldsymbol{M}_{t+1} = \boldsymbol{M}_t - \frac{\alpha}{\sqrt{s_t + \epsilon_m}} \cdot \frac{\partial f(\boldsymbol{M}_t, \boldsymbol{N}_t, \boldsymbol{T}_t)}{\partial \boldsymbol{M}_t}, \\ \boldsymbol{N}_{t+1} = \boldsymbol{N}_t - \frac{\alpha}{\sqrt{s_t + \epsilon_m}} \cdot \frac{\partial f(\boldsymbol{M}_t, \boldsymbol{N}_t, \boldsymbol{T}_t)}{\partial \boldsymbol{N}_t}, \\ \boldsymbol{T}_{t+1} = \boldsymbol{T}_t - \frac{\alpha}{\sqrt{s_t + \epsilon_m}} \cdot \frac{\partial f(\boldsymbol{M}_t, \boldsymbol{N}_t, \boldsymbol{T}_t)}{\partial \boldsymbol{T}_t}. \end{cases} \tag{16}$$

Where, $f(\boldsymbol{M}_t, \boldsymbol{N}_t, \boldsymbol{T}_t)$ represents the weighted objective function of arguments $\boldsymbol{M}_t$, $\boldsymbol{N}_t$ and $\boldsymbol{T}_t$, while s_t and α denote the cumulative sum of the squared gradients and the learning rate, respectively. Moreover, the symbol ϵ_m is a smooth item designed to prevent the denominator from being 0.

Adam-based GCP rule. The iteration rule of factor matrix $\boldsymbol{M}_t$, $\boldsymbol{N}_t$ and $\boldsymbol{T}_t$ can be represented as

$$\begin{cases} \boldsymbol{M}_{t+1} = \boldsymbol{M}_t - \alpha \frac{\widehat{S}_M^t}{\sqrt{\widehat{V}_M^t + \epsilon_m}}, \\ \boldsymbol{N}_{t+1} = \boldsymbol{N}_t - \alpha \frac{\widehat{S}_N^t}{\sqrt{\widehat{V}_N^t + \epsilon_m}}, \\ \boldsymbol{T}_{t+1} = \boldsymbol{T}_t - \alpha \frac{\widehat{S}_T^t}{\sqrt{\widehat{V}_T^t + \epsilon_m}}. \end{cases} \tag{17}$$

Here, $\widehat{S}_M^t$ and $\widehat{V}_M^t$ are first and second moment estimation vectors of $\boldsymbol{M}_t$ after bias correction, satisfying $\widehat{S}_M^t = S_M^t/(1 - \lambda_1^t)$ and $\widehat{V}_M^t = V_M^t/(1 - \lambda_2^t)$, respectively. Wherein, λ_1^t and λ_2^t are decay factors in Adam optimization algorithm.

(3) Reward space: $\mathscr{R}$ is the set of immediate rewards R_t obtained after the agent executes action A_t in the current state S_t, designed to motivate the agents to optimize the global model with the goal of minimizing the prediction errors between $\boldsymbol{\mathcal{D}}$ and $\widehat{\boldsymbol{\mathcal{D}}}$. The R_t at time t can be defined as

$$R_t = -\|\boldsymbol{\mathcal{D}}_t - [\![\boldsymbol{M}_t, \boldsymbol{N}_t, \boldsymbol{T}_t]\!]\|_F^2, \tag{18}$$

which can be considered as the weighted accumulated residual between $\boldsymbol{\mathcal{D}}_t$ and $[\![\boldsymbol{M}_t, \boldsymbol{N}_t, \boldsymbol{T}_t]\!]$.

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on October 28,2024 at 03:10:43 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2024.3480136

4.4 DRL-based Tensor Factorization in Each RAN

The local training in each RAN is conducted through combining DRL with κ available rules, specified by typical tensor factorization algorithms such as CP-ALS, AdaGrad-based GCP and Adam-based GCP, to minimize loss function between $\mathscr{D}$ and $[\![\boldsymbol{M}, \boldsymbol{N}, \boldsymbol{T}]\!]$ defined in Eq. (12). The agent installed DQNs will flexibly select one of the available rules to factorize tensor, relying on the action-value function $Q(S_t, A_t)$, which represents the expected return from state S_t at time t, denoted as $Q(S_t, A_t)$=$\mathbb{E}(G_t|S_t, A_t)$. Here, G_t represents the accumulative return from the current state S_t to the final state. Essentially, the agent aims to find the optimal action-value function $Q^*(S_t, A_t)$, which stands for an optimal policy π^* for action exploration, and can be given by

$$Q^*(S_t, A_t) = \mathbb{E}[R_t + \gamma \max_{A \in \mathscr{A}} Q^*(S_{t+1}, A)]. \tag{19}$$

Here, γ denotes the discounting factor designed to guide the agent to find the optimal policy and avoid the sequence becoming excessively long in continuous tasks.

The structure of DQN-based tensor factorization includes experience replay buffer $\mathbb{B}$, experimental environment, predictive Q-network and target Q-network. The environment is responsible for interacting with the agent, executing the actions chosen and then transitioning to a new state. The buffer $\mathbb{B}$ is designed to store transition samples in the form of (S_t, A_t, R_t, S_{t+1}), and provides data for Q-network to execute local training. The predictive Q-network with built-in model θ is introduced to approximate $Q^*(S_t, A_t)$, while the target Q-network with the built-in model $\theta^{'}$ is utilized to approximate the optimal target action-value as a reference for the predictive Q-network to be updated.

With $\epsilon-greedy$ policy in mind, each agent will randomly choose an action at time t with a probability of $\epsilon_{rd}(0 \leq \epsilon \leq \epsilon_{rd})$ from action space $\mathscr{A}$, or greedily take an action corresponding to maximum action-value with a probability of $1-\epsilon_{rd}(\epsilon_{rd} < \epsilon \leq 1)$, which can be described as

$$A_t = \begin{cases} \text{argmax}_{A \in \mathscr{A}} \ Q(S_t, A; \theta_t^z), & \epsilon_{rd} < \epsilon \leq 1, \\ \text{random}_{A \in \mathscr{A}} \quad A, & 0 \leq \epsilon \leq \epsilon_{rd}. \end{cases} \tag{20}$$

Here, ϵ_{rd} denotes the exploration factor decreasing until 0 with the increasing epochs, designed to make the policy gradually become greedy and approach a deterministic policy over time. The agent will execute action A_t with the reward R_t defined in Eq. (18), and then transition to the next state S_{t+1}. In this way, an experimental transition sample (S_t, A_t, R_t, S_{t+1}) is created and stored in $\mathbb{B}$, whose samples will be replaced by newly generated samples. In order to eliminate the correlation among transition samples and enhance the data utilization rate, a mini-batch of transitions is selected for model training based on random-priority from $\mathbb{B}$.

On this basis, the update of predictive Q-network θ_t^z and target Q-network $\theta_t^{'z}$ are conducted to approximate the optimal action-value function $Q_p^*(S_t, A_t; \theta_t^z)$ and $Q_t^*(S_t, A_t; \theta_t^{'z})$. Wherein, the predictive Q-network is updated at each epoch and the target Q-network synchronizes with the predictive Q-network each C iteration epochs. The predictive and target Q-networks are updated at each step of iterations, built on the loss function,

$$\begin{aligned} L(\theta_t^z) = \mathbb{E}[\frac{1}{2}(R_t + \gamma \max_{A \in \mathscr{A}} Q_t(S_{t+1}, A; \theta_t^{'z}) \\ - Q_p(S_t, A_t; \theta_t^z))^2]. \end{aligned} \tag{21}$$

Here, $Q_p(S_t, A_t; \theta_t^z)$ is the action-value calculated by predictive Q-network, and $R_t + \gamma \max_{A \in \mathscr{A}} Q_t(S_{t+1}, A; \theta_t^{'z})$ is the target action-value as the reference to update θ_t^z. To accurately learn $Q_p^*(S_t, A_t; \theta_t^z)$, Mini-Batch Stochastic Gradient Descent (MBSGD) is exploited to optimize the loss function $L(\theta_t^z)$, defined in Eq. (21), so as to update predictive Q-network as

$$\begin{aligned} \theta_{t+1}^z = \theta_t^z - \eta \cdot \mathbb{E}[(R_t + \gamma \max_{A \in \mathscr{A}} Q_t(S_{t+1}, A; \theta_t^{'z}) \\ - Q_p(S_t, A_t; \theta_t^z)) \nabla Q_p(S_t, A_t; \theta_t^z)], \end{aligned} \tag{22}$$

where η is the local learning rate.

The pseudocode of the DRL-based tensor factorization is illustrated in Algorithm 2. The initialization settings are conducted as listed in Line 1. The local model θ_t^z and $\theta_t^{'z}$ are copied from the global model as shown in Line 2. Lines 4-5 describe the process of action selection depending on the $\epsilon-greedy$ policy. In Lines 6-7, an experimental transition sample is generated and stored in the experience replay buffer $\mathbb{B}$. In Line 9, a mini-batch of transition samples is chosen based on random-priority for local training. The construction of the loss function related to predictive and target Q-network is illustrated in Lines 10-11. For each epoch τ, θ_t^z is updated based on MBSGD defined in Line 12. It is noteworthy that $\theta_t^{'z}$ synchronizes with θ_t^z every C epochs as described in Line 13. Finally, the model parameters of the prediction network will be uploaded to the cloud server for model aggregation.

Algorithm 2: DRL-based tensor factorization

Input: initial $\boldsymbol{\theta}(i)$
Output: the updated θ_t^z
1 Initialize $\mathbb{B}$, Q_p, Q_t, $<\mathcal{S}, \mathscr{A}, \mathscr{R}>$, ϵ_{rd} and η
2 $\theta_t^z = \boldsymbol{\theta}(i)$, $\theta_t^{'z} = \boldsymbol{\theta}(i)$
3 **for** *each local epoch* τ **do**
4 Generate a random value for ϵ
5 Select and execute A_t built on Eq. (20)
6 Obtain R_t and S_{t+1}
7 Store (S_t, A_t, R_t, S_{t+1}) into $\mathbb{B}$
8 **if** $\mathbb{B}$ *is full* **then**
9 Randomly choose (S_i, A_i, E_i, S_{i+1}) from $\mathbb{B}$
10 $G_t \leftarrow R_t + \gamma \max_{A \in \mathscr{A}} Q_t(S_{t+1}, A; \theta_t^{'z})$
11 $L(\theta_t^z) \leftarrow \mathbb{E}[\frac{1}{2}(G_t - Q_p(S_t, A_t; \theta_t^z))^2]$
12 $\theta_{t+1}^z \leftarrow \theta_t^z - \eta \cdot \nabla L(\theta_t^z)$
13 Every C epochs, $\theta_t^{'z} \leftarrow \theta_t^z$
14 **return** θ_t^z to the cloud server

4.5 Tensor Recovery

After the training of FDRL-based tensor factorization in the z-th RAN has been implemented, the optimal factor

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on October 28,2024 at 03:10:43 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2024.3480136

matrices $\boldsymbol{M}^z$, $\boldsymbol{N}^z$ and $\boldsymbol{T}^z$ defined in Eq. (12) are obtained. In order to obtain the predicted resource tensor, we need to conduct the reverse procedure of pre-filling-based tensor reshaping with $\boldsymbol{M}^z$, $\boldsymbol{N}^z$ and $\boldsymbol{T}^z$. Specifically, we first extract $\boldsymbol{\Sigma}_k^z$ from $\boldsymbol{T}^z$, namely $\boldsymbol{\Sigma}_k^z{=}diag(T_{k:}^z)$, where $T_{k:}^z$ is the k-th row of $\boldsymbol{T}^z$. Then, sub-matrices of tensor $\boldsymbol{\mathcal{X}}^z$ can be recovered by

$$\boldsymbol{X}_k^z = \boldsymbol{W}_k^z \odot \boldsymbol{X}_k^z + (\boldsymbol{E} - \boldsymbol{W}_k^z) \odot \boldsymbol{M}^z \boldsymbol{\Sigma}_k^z (\boldsymbol{N}^z)^T (\boldsymbol{P}_k^z)^T, \quad (23)$$

where $\boldsymbol{W}_k^z$ is a sub-matrix of weight tensor $\boldsymbol{\mathcal{W}}^z$ with element $\omega_{i,j,k}^z$ as a binary variable to represent whether item of the corresponding location is unknown, and $\boldsymbol{E}$ denotes an all ones matrix which has the same size as $\boldsymbol{W}_k^z$. In this way, we can finally obtain the prediction tensor $\widehat{\boldsymbol{\mathcal{X}}}^z$ by integrating updated sub-matrices $\boldsymbol{X}_k^z$, and further acquire the unknown required resources.

5 Performance Evaluation

In this section, we conduct comprehensive simulation experiments on a universal server, equipped with an AMD Ryzen-7-5800H CPU and an NVIDIA GeForce RTX 3070 GPU, to evaluate the performance of APMR in 5G. Firstly, the experimental simulation setup is introduced, followed by the presentation of evaluation metrics. Then, we compare the performance results of APMR with six well-known approaches, i.e., CP-ALS, GCP, GCP*, LTC [21], L-PAW [10] and DATC [29]. The first three are conventional tensor factorization approaches, among which, GCP and GCP* adopt AdaGrad-based and Adam-based GCP rules, respectively. The last two L-PAW and DATC independently propose a novel DL-based tensor completion model.

5.1 Simulation Setup

The experiments have been conducted in Python-3.9.10 with Pytorch-based DL framework and Tensor ToolBox [31] of Matlab to construct the predictive and target Q-networks for tensor factorization. The target Q-network owns the identical structure as predictive Q-network, and is updated by copying its weights every 5 epochs. Each of them is comprised of six fully connected layers, i.e., one input layer, four hidden layers and one output layer with $(M{\times}R{+}N{\times}R{+}3{\times}R)$, 4096, 1024, 128, 16 and 3 neurons, respectively, using the ReLU function as the activation function. The sizes of the input and output layers are equal to the number of state items and optional actions, respectively. Three iteration rules ($\kappa{=}3$) associated with taken actions, i.e., CP-ALS, AdaGrad-based GCP and Adam-based GCP, are introduced into agents for tensor factorization. The most hyper parameters for our experiments, including replay buffer capacity $\mathbb{B}_s$, discount factor γ, exploring factor ϵ_{rd}, etc., are listed in Table 2. Moreover, the tensor rank R of reshaped tensors, the local learning rate η and the capacity of mini-batch size B_s are set in the range of $[10, 25]$, $[0.0001, 0.001]$ and $[16, 64]$, respectively.

There are 3 RANs, i.e., $K{=}3$, in the simulation scenarios. The datasets used for the performance evaluation are derived from [32], where resource consumption data of one-month time span in each RAN is chosen and expanded into two tensors by adding random noise. Among them, one is used for training, and another is used to evaluate the performance of diverse approaches. In order to simulate the failure that may occur in the process of data sampling, 30% of elements in each resource tensor are randomly zeroed. Then, we adopt the Box-Cox data transformation to bring data closer to a normal distribution, which can make the gradient descent more stable. Due to the different rate measurements, the original tensors created by all RANs are reshaped into a series of standard third-order tensors with 464 time columns by the tensor reshaping algorithm. On this foundation, the resource prediction problem of 248 services at 464 time slots is reformulated into a tensor factorization problem of a series of $248{\times}464{\times}3$ tensors. Then, the local training is conducted in each BS for resource prediction during each global episode, while the model distribution and aggregation are executed at the start and end of the global episode, respectively.

TABLE 2
Parameters Setting in Simulations

Parameters	Descriptions	Value
M	The number of services in each RAN	248
K	The number of RANs	3
T	The global training episodes	40
t	The local training epochs in each episode	40
$\mathbb{B}_s$	The capacity of replay buffer	400
ϵ_{rd}	The exploring factor	0.9
γ	The discount factor	0.9
β	The federated learnin rate factor	0.5
ϵ_m	The smooth item of GCP rule	$1{\times}10^{-6}$
$-$	The decay coefficient of learning rate	0.95
$-$	The decay coefficient of exploring factor	0.9

There are four metrics adopted to evaluate the performance of APMR, presented as follows.

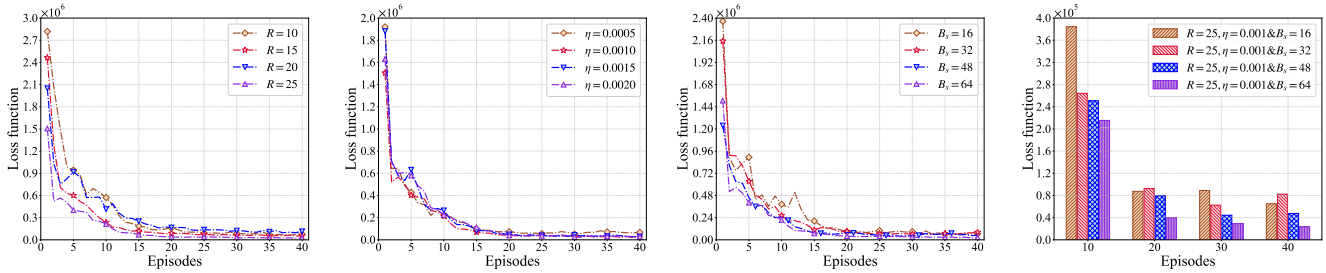
- **Loss Function.** The loss function is denoted as the sum of $L(\theta)$ defined in Eq. (21) in one episode, designed to indicate the convergence performance of APMR. The more stable the loss function is, the better the convergence of APMR will be.
- **Prediction Error Ratio.** The prediction error ratio e_s can be represented as $e_s{=}\sqrt{\|(\boldsymbol{\mathcal{X}} - \widehat{\boldsymbol{\mathcal{X}}})\|_F^2 / \|\boldsymbol{\mathcal{X}}\|_F^2}$. This metric is exploited to measure the fitness between resource tensor $\widehat{\boldsymbol{\mathcal{X}}}$ and $\boldsymbol{\mathcal{X}}$. The smaller e_s is, the closer $\widehat{\boldsymbol{\mathcal{X}}}$ is to the completed tensor.
- **Relative Error (RE).** The relative error e_r is defined as $e_r{=}\sqrt{\|x(i,j,k) - \widehat{x}(i,j,k)\| / x(i,j,k)}$, to measure the relative errors between required resource entries included in $\widehat{\boldsymbol{\mathcal{X}}}$ and $\boldsymbol{\mathcal{X}}$, where $x(i,j,k)$ and $\widehat{x}(i,j,k)$ are raw data and the recovered data, respectively.
- **Computing Cost Efficiency.** This metric is defined as the quotient of the prediction error to the computing costs, designed to measure the efficiency of the prediction in tensor factorization. Since the computing resources are proportional to the number of episodes, the lower prediction errors and the smaller iteration times are, the higher efficiency is.

5.2 Simulation Results

(1) Convergence of APMR:

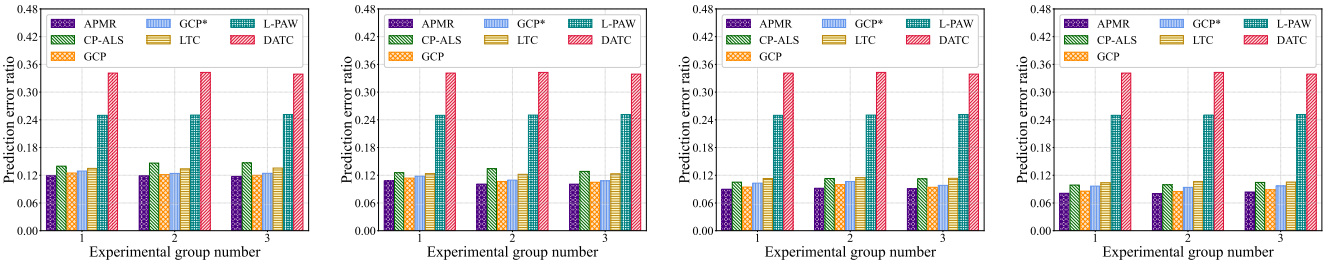
Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on October 28,2024 at 03:10:43 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2024.3480136



(a) The varied rank R (b) The varied learning rate η (c) The varied batch size B_s (d) The optimal parameters

Fig. 6. Convergence of APMR in terms of loss function with different parameters of rank R, learning rate η and batch size B_s.



(a) R=10 and η=0.0015 (b) R=15 and η=0.0015 (c) R=20 and η=0.0010 (d) R=25 and η=0.0010

Fig. 7. Prediction error ratios of APMR, CP-ALS, GCP, GCP*, LTC, L-PAW and DATC with the varied parameter of rank R and learning rate η.

Fig. 6 evaluates the convergence of APMR in the form of loss function, with the varied parameters during the global training process over 40 episodes. In Fig. 6(a), the tensor rank R of reshaped tensors is set to 10, 15, 20 and 25, respectively. We can obviously see that the loss function of APMR gradually converges as the number of global training increases, and global model converges the fastest when rank R is 25. Such a tendency greatly validates the effectiveness of APMR for the resource prediction and indicates the built-in DQN models of APMR have learned the latent rules to adopt the optimal policy for tensor factorization. To find the optimal parameters of APMR that yield the fastest convergence, further experiments with varied parameters of learning rate η and batch size B_s have been conducted and illustrated in Figs. 6(b) and 6(c), respectively. In Fig. 6(b), the rank R is set to 25 for better performance while the learning rate η is set to 0.0005, 0.001, 0.0015 and 0.002, respectively. It is clear to see that the loss function gradually decreases in the first 20 iteration episodes and maintains stability in subsequent iteration episodes. Among them, the loss of APMR performs best in convergence and stability when the learning rate η is set to 0.001.

Built on the observations from Fig. 6(b), the learning rate η is set to 0.001 while the batch size B_s changes in the range of $[16, 64]$ in Fig. 6(c). The results in Figs. 6(c)-6(d) elaborate that the global models with batch size B_s=64 have quicker convergence and more stability than others. Built on the results of Figs. 6(b)-6(d), we set the learning rate η=0.001 and batch size B_s=64 when the rank R=25, so as to acquire better performance in our subsequent simulations. In the same way, we obtained the optimal parameters of η and B_s for R=10, R=15 and R=20, respectively, and set them in subsequent simulations. All the convergence tendencies in term of loss function demonstrate that APMR can train an efficient DQN model via FDRL within the specified iteration episodes, which can be further adopted for required resource prediction in 5G.

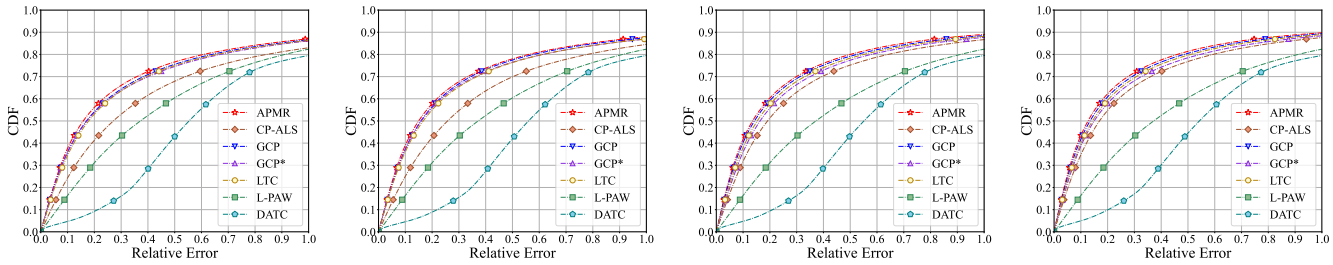
(2) Comparisons in Prediction Accuracy: In order to validate the superiority of APMR, we compare the prediction error ratios of APMR with CP-ALS, GCP, GCP*, LTC, L-PAW and DATC with varied parameters of rank R and learning rate η. Built on the analysis of optimal parameters under different R, the settings of parameter pairs for (R, η) are set to $(10, 0.0015)$, $(15, 0.0015)$, $(20, 0.0010)$ and $(25, 0.0010)$ in Figs. 7(a)-7(d), respectively. Due to no coefficient of tensor rank R, both L-PAW and DATC are repeatedly conducted in the same experiment conditions in each group of simulations with different R, respectively. Thus, the error ratios and running time of both L-PAW and DATC are nearly constant across various experimental groups. To ensure a fair comparison, we employ identical stopping condition for APMR and all comparison approaches. For each given (R, η), three experimental groups are conducted with the same parameters and tensors from 3 RANs to be predicted.

It can be obviously seen from Figs. 7(a)-7(d) that APMR still holds the smallest prediction error ratios compared to CP-ALS, GCP, GCP*, LTC, L-PAW and DATC, no matter how the (R, η) are set. Specifically, our APMR achieves the average reductions of 17.70%, 2.76%, 5.64%, 11.96%, 52.63% and 65.19% in Fig. 7(a), 20.23%, 4.92%, 7.76%, 16.18%, 58.85% and 69.77% in Fig. 7(b), 17.54%, 5.44%, 11.39%, 20.13%, 63.78% and 73.39% in Fig. 7(c) as well as 19.01%, 5.21%, 14.65%, 22.24%, 67.39% and 76.03% in Fig. 7(d), respectively. It is noticed that in Fig. 7(d) all of them can achieve the minimum prediction error ratio when R=25 and η=0.0010, compared to other parameter settings, which can be better used for accurate resource prediction via FDRL-based tensor factorization in 5G.

Meanwhile, the running time of APMR, CP-ALS, GCP, GCP*, LTC, L-PAW and DATC are recorded and shown in Table 3, which lists the running time of them with varied (R, η). It is shown from Fig. 7 and Table 3 that APMR can

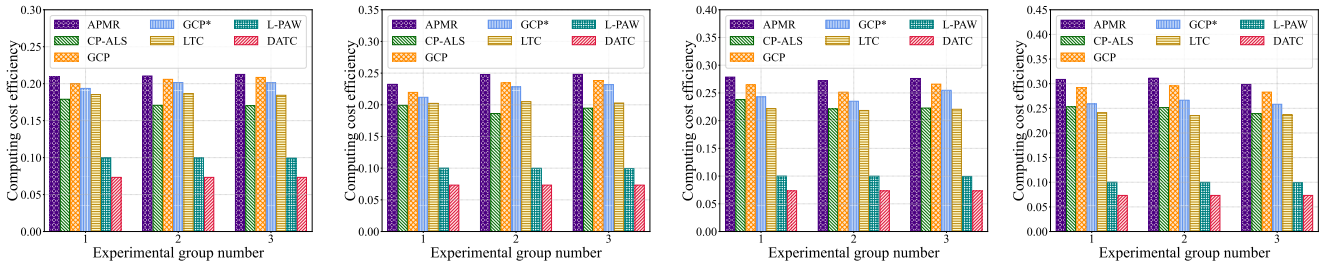
Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on October 28,2024 at 03:10:43 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2024.3480136



(a) R=10 and η=0.0015 (b) R=15 and η=0.0015 (c) R=20 and η=0.0010 (d) R=25 and η=0.0010

Fig. 8. Relative error of APMR, CP-ALS, GCP, GCP*, LTC, L-PAW and DATC on CDF with different tensor rank R and learning rate η.



(a) R=10 and η=0.0015 (b) R=15 and η=0.0015 (c) R=20 and η=0.0010 (d) R=25 and η=0.0010

Fig. 9. Computing cost efficiency of APMR, CP-ALS, GCP, GCP*, LTC, L-PAW and DATC with different tensor rank R and learning rate η.

TABLE 3
Running Time with Different Tensor Rank

Methods	Average Running Time (Seconds) with varied (R, η)			
	(10,0.0015)	(15,0.0015)	(20,0.0010)	(25,0.0010)
APMR	619.8700	763.0574	846.0420	1006.6342
CP-ALS	440.2211	573.0201	605.3078	626.6998
GCP	703.4007	918.2314	969.6331	1154.5060
GCP*	705.8419	822.3610	969.8556	1123.6336
LTC	681.7496	802.0051	915.2187	1001.9406
L-PAW	1141.5547	1138.4152	1140.8790	1136.9720
DATC	1480.8406	1491.1192	1503.4350	1509.5421

accurately predict the required resources with the fastest speed in almost all scenarios compared to other approaches except for CP-ALS. This is because CP-ALS only involves simple tensor iteration calculations rather than neural networks, so the computational complexity is much lower than other methods but incurs coarse-grained prediction accuracy compared to APMR, GCP, GCP* and LTC.

(3) Cumulative distribution function (CDF) of RE: In this subsection, we evaluate the CDF of APMR compared to CP-ALS, GCP, GCP*, LTC, L-PAW and DATC under varied (R, η) as RE changes in the range of $[0, 1.0]$. Due to no coefficient of tensor rank R, L-PAW and DATC are repeatedly conducted in the same experiment conditions in each group of simulations with different R, respectively. Figs. 8(a)-8(d) show that APMR, GCP, GCP* and LTC have similar accuracy of predicted required resources, which holds $70\%-80\%$ prediction relative errors less than 0.4, while CP-ALS, L-PAW and DATC hold around $63\%-73\%$, $52\%-54\%$ and $28\%-30\%$ prediction result error less than 0.4, respectively. Due to the resource requirements of UEs being abrupt in 5G, resource tensor is hard to attain a strictly low-rank feature in real-world datasets. Hence, DL-based L-PAW and DATC are unable to effectively extract the features inherent in the resource tensor for learning. Different from L-PAW and DATC, APMR can flexibly capture the correlations across different feature dimensions in real-world datasets based on feature extraction of neural networks and the feedback after executing tensor factorization rules. Therefore, APMR can achieve moderated advancement in CDF, with the less relative errors, compared to CP-ALS, GCP, GCP*, LTC, L-PAW and DATC as demonstrated in Figs. 8(a)-8(d).

(4) Comparisons in Computing Cost Efficiency: In this subsection, Fig. 9 evaluates the computing cost efficiency of APMR, CP-ALS, GCP, GCP*, LTC, L-PAW and DATC in three experimental groups under varied rank R and learning rate η, respectively. In order to ensure fairness in the computing cost efficiency of these solutions, we set the running times of epochs to be 40. The batch size B_s is set to be 64 constantly, while the parameter pairs of (R, η) are set to $(10, 0.0015)$, $(15, 0.0015)$, $(20, 0.0010)$ and $(25, 0.0010)$ in four scenarios, respectively. It is clear to see that APMR outperforms in computing cost efficiency, with the achievements of 21.61%, 2.85%, 5.99%, 13.59%, 111.11% and 187.27% on average in Fig. 9(a), 25.74%, 5.18%, 8.42%, 19.39%, 143.29% and 231.05% in Fig. 9(b), 21.35%, 5.78%, 12.95%, 25.2%, 176.14% and 275.78% in Fig. 9(c) and 23.48%, 5.50%, 17.19%, 28.65%, 206.73% and 317.47% in Fig. 9(d), compared to CP-ALS, GCP, GCP*, LTC, L-PAW and DATC, respectively. Thus, APMR is more appropriate for 5G resource prediction than other solutions due to its higher computing cost efficiency.

6 Conclusion

This paper has investigated the accurate prediction of the required multi-dimensional resources in 5G via FDRL to satisfy the ever-increasing requirements of diversified network services. APMR has three remarkable characteristics distinguishing from the previous work: (1) An efficient pre-filling-based tensor reshaping strategy has been designed to

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on October 28,2024 at 03:10:43 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2024.3480136

establish a series of standard third-order tensors in RANs; (2) The FDRL-based framework, which introduces multiple specific iteration rules to factorize resource tensor in an intelligent and parallel manner, has been proposed to train a scalable optimal model for accurate prediction of the required resources in 5G; (3) The accuracy-aware and latency-based depreciation strategies have been designed to accelerate the global model convergence with more accurate rewards, by assigning the higher weights to accurate and newly-learned local models. Furthermore, extensive experimental simulations are conducted to validate that APMR outperforms in prediction accuracy and computing cost efficiency compared to the state-of-the-art approaches.

As a part of our future work, we will be devoted to the implementation of APMR in large-scale networks and real-world 5G scenarios to investigate its scalability and feasibility. Notice that the full 5G networks have not been established in the world, we intend to extend our work into hybrid 5G, in which the traditional 4G and the emerging 6G will coexist for a long while. The combination of them offers new opportunities to fulfill the QoS of 5G applications with on-demand resource allocation. In order to achieve this, the potential directions include FDRL acceleration, user mobility model across different RANs and user trajectory tracking built on our current investigations.

ACKNOWLEDGMENT

This work is partially supported by the Natural Science Foundation of China (No.62372192).

REFERENCES

[1] N. M. Laboni, S. J. Safa, S. Sharmin, M. A. Razzaque, M. M. Rahman, and M. M. Hassan, "A hyper heuristic algorithm for efficient resource allocation in 5G mobile edge clouds," *IEEE Trans. Mob. Comput.*, vol. 23, no. 1, pp. 29–41, 2024.

[2] N. Yarkina, A. Gaydamaka, D. Moltchanov, and Y. Koucheryavy, "Performance assessment of an ITU-T compliant machine learning enhancements for 5G RAN network slicing," *IEEE Trans. Mob. Comput.*, vol. 23, no. 1, pp. 719–736, 2024.

[3] D. Saxena, J. Kumar, A. K. Singh, and S. Schmid, "Performance analysis of machine learning centered workload prediction models for cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 4, pp. 1313–1330, 2023.

[4] J. Gao, H. Wang, and H. Shen, "Task failure prediction in cloud data centers using deep learning," *IEEE Trans. Serv Comput.*, vol. 15, no. 3, pp. 1411–1422, 2022.

[5] Y. Kwon, S. Lee, H. Yi, D. Kwon, S. Yang, B.-g. Chun, L. Huang, P. Maniatis, M. Naik, and Y. Paek, "Mantis: Efficient predictions of execution time, energy usage, memory usage and network usage on smart mobile devices," *IEEE Trans. Mob. Comput.*, vol. 14, no. 10, pp. 2059–2072, 2014.

[6] M. Bagaa, T. Taleb, J. B. Bernabe, and A. Skarmeta, "QoS and resource-aware security orchestration and life cycle management," *IEEE Trans. Mob. Comput.*, vol. 21, no. 8, pp. 2978–2993, 2022.

[7] J. Zhu, P. He, Z. Zheng, and M. R. Lyu, "Online QoS prediction for runtime service adaptation via adaptive matrix factorization," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 10, pp. 2911–2924, 2017.

[8] Z. Liu, Q. Z. Sheng, X. Xu, D. Chu, and W. E. Zhang, "Context-aware and adaptive qos prediction for mobile edge computing services," *IEEE Trans. Serv Comput.*, vol. 15, no. 1, pp. 400–413, 2022.

[9] S. Kianpisheh, M. Kargahi, and N. M. Charkari, "Resource availability prediction in distributed systems: An approach for modeling non-stationary transition probabilities," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 8, pp. 2357–2372, 2017.

[10] Z. Chen, J. Hu, G. Min, A. Y. Zomaya, and T. El-Ghazawi, "Towards accurate prediction for high-dimensional and highly-variable cloud workloads with deep learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 4, pp. 923–934, 2019.

[11] A. K. Singh, D. Saxena, J. Kumar, and V. Gupta, "A quantum approach towards the adaptive prediction of cloud workloads," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 12, pp. 2893–2905, 2021.

[12] L. Nie, X. Wang, S. Wang, Z. Ning, M. S. Obaidat, B. Sadoun, and S. Li, "Network traffic prediction in industrial Internet of Things backbone networks: A multitask learning mechanism," *IEEE Trans. Industr. Inform.*, vol. 17, no. 10, pp. 7123–7132, 2021.

[13] J. Li, J. Yao, D. Xiao, D. Yang, and W. Wu, "Evogwp: Predicting long-term changes in cloud workloads using deep graph-evolution learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 35, no. 3, pp. 499–516, 2024.

[14] B. Feng, Z. Ding, and C. Jiang, "Fast: A forecasting model with adaptive sliding window and time locality integration for dynamic cloud workloads," *IEEE Trans. Serv Comput.*, vol. 16, no. 2, pp. 1184–1197, 2023.

[15] I. K. Kim, W. Wang, Y. Qi, and M. Humphrey, "Forecasting cloud application workloads with cloudinsight for predictive resource management," *IEEE Trans. Cloud Comput.*, vol. 10, no. 3, pp. 1848–1863, 2022.

[16] H. Huang, Z. Li, J. Tian, G. Min, W. Miao, and D. O. Wu, "Accurate prediction of required virtual resources via deep reinforcement learning," *IEEE ACM Trans. Netw.*, vol. 31, no. 2, pp. 920–933, 2022.

[17] K. Xie, C. Peng, X. Wang, G. Xie, J. Wen, J. Cao, D. Zhang, and Z. Qin, "Accurate recovery of internet traffic data under variable rate measurements," *IEEE ACM Trans. Netw.*, vol. 26, no. 3, pp. 1137–1150, 2018.

[18] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[19] E. Acar, D. M. Dunlavy, and T. G. Kolda, "A scalable optimization approach for fitting canonical tensor decompositions," *J. Chemom.*, vol. 25, no. 2, pp. 67–86, 2011.

[20] D. Hong, T. G. Kolda, and J. A. Duersch, "Generalized canonical polyadic tensor decomposition," *SIAM review*, vol. 62, no. 1, pp. 133–163, 2020.

[21] K. Xie, X. Wang, X. Wang, Y. Chen, G. Xie, Y. Ouyang, J. Wen, J. Cao, and D. Zhang, "Accurate recovery of missing network measurement data with localized tensor completion," *IEEE ACM Trans. Netw.*, vol. 27, no. 6, pp. 2222–2235, 2019.

[22] C.-L. Wu, T.-C. Chiu, C.-Y. Wang, and A.-C. Pang, "Mobility-aware deep reinforcement learning with seq2seq mobility prediction for offloading and allocation in edge computing," *IEEE Trans. Mob. Comput.*, vol. 23, no. 6, pp. 6803–6819, 2024.

[23] S. Lee and D.-H. Choi, "Federated reinforcement learning for energy management of multiple smart homes with distributed energy resources," *IEEE Trans. Industr. Inform.*, vol. 18, no. 1, pp. 488–497, 2020.

[24] H. Huang, C. Zeng, Y. Zhao, G. Min, Y. Zhu, W. Miao, and J. Hu, "Scalable orchestration of service function chains in NFV-enabled networks: A federated reinforcement learning approach," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 8, pp. 2558–2571, 2021.

[25] M. Diamanti, P. Charatsaris, E. E. Tsiropoulou, and S. Papavassiliou, "Incentive mechanism and resource allocation for edge-fog networks driven by multi-dimensional contract and game theories," *IEEE Open J. Commun. Soc.*, vol. 3, pp. 435–452, 2022.

[26] D. Minovski, N. Ögren, K. Mitra, and C. Åhlund, "Throughput prediction using machine learning in LTE and 5G networks," *IEEE Trans. Mob. Comput.*, vol. 22, no. 3, pp. 1825–1840, 2023.

[27] A. Bentaleb, A. C. Begen, S. Harous, and R. Zimmermann, "Data-driven bandwidth prediction models and automated model selection for low latency," *IEEE Trans. Multimedia.*, vol. 23, pp. 2588–2601, 2021.

[28] K. Xie, H. Lu, X. Wang, G. Xie, Y. Ding, D. Xie, J. Wen, and D. Zhang, "Neural tensor completion for accurate network monitoring," in *IEEE Conf. Comput. Commun.*, 2020, pp. 1688–1697.

[29] K. Xie, Y. Ouyang, X. Wang, G. Xie, K. Li, W. Liang, J. Cao, and J. Wen, "Deep adversarial tensor completion for accurate network traffic measurement," *IEEE ACM Trans. Netw.*, vol. 31, no. 5, pp. 2101–2116, 2023.

[30] H. Tan, G. Feng, J. Feng, W. Wang, Y.-J. Zhang, and F. Li, "A tensor-based method for missing traffic data completion," *Transp Res. Part C. Emerg. Technol.*, vol. 28, pp. 15–27, 2013.

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on October 28,2024 at 03:10:43 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2024.3480136

[31] T. G. K. Brett W. Bader, "MATLAB tensor toolbox, Version 3.6." [Online]. Available: https://www.tensortoolbox.org
[32] A. Shanny, D. Catalin, E. Dick, I. Alexandru, J. Mathieu, L. Hui, and W. Lex, "Gwa-t-12 bitbrains." [Online]. Available: http://gwa.ewi.tudelft.nl/datasets/gwa-t-12-bitbrains

Haojun Huang is an Associate Professor in the School of Electronic Information and Communications at Huazhong University of Science and Technology, China. He received his PhD degree in Communication and Information Engineering from the University of Electronic Science and Technology of China in 2012, and the BS degree in Computer Science from Wuhan University of Technology in 2005. His current research interests include Artificial Intelligence, Wireless Communications, Computer Networks, and Internet of Things.

Qifan Wang received the BS degree in Electronic Information Engineering from Zhengzhou University, China, in 2022. He is currently pursuing the Master's degree in Information and Communication Engineering with Huazhong University of Science and Technology, China. His research interests include Computer Networks and Reinforcement Learning.

Weimin Wu received the BS degree in computer software from Xidian University, Xi'an, China, in 1992, the M.E. degree in computer application from Sichuan University, Chengdu, China, in 1995, and the PhD degree in communications and information systems from Huazhong University of Science and Technology, Wuhan, China, in 2007. He is currently an Associate Professor with the School of Electronic Information and Communications, Huazhong University of Science and Technology. His current research interests include the areas of Internet Streaming, Broadband Wireless Communications, and Networking.

Wang Miao is currently a lecturer in the Department of Computer Science at the University of Exeter, United Kingdom. He received his PhD degree in Computer Science from the University of Exeter, United Kingdom in 2017. His research interests focus on Vehicle Edge Computing, Unmanned Aerial Networks, Wireless Communication Networks, Software Defined Networking, Network Function Virtualisation, Applied Machine Learning, and Stochastic Performance Modelling and Analysis.

Geyong Min is a Professor of High Performance Computing and Networking in the Department of Computer Science at the University of Exeter, United Kingdom. He received the PhD degree in Computing Science from the University of Glasgow, United Kingdom, in 2003, and the BS degree in Computer Science from Huazhong University of Science and Technology, China, in 1995. His research interests include Computer Networks, Wireless Communications, Parallel and Distributed Computing, Ubiquitous Computing, Multimedia Systems, Modelling and Performance Engineering.

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on October 28,2024 at 03:10:43 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.