Mixture of Gradient: A Unified Enhancing Approach for Deep-Learning-Based Wireless Network Optimization

Nan Cheng, *Senior Member, IEEE*, Longfei Ma, *Student Member, IEEE*, Yanpeng Dai, *Member, IEEE*, Xiucheng Wang, *Graduate Student Member, IEEE*, Qihao Li, *Member, IEEE*, Wei Quan, *Senior Member, IEEE*, Hui Liang, *Member, IEEE*, and Xuemin Shen, *Fellow, IEEE*

***Abstract*—Deep learning plays increasingly important role in future wireless network management and optimization. Existing training methods such as label-based supervised learning and label-free learning have inherent limitations. The performance of supervised learning is limited by labels, while label-free training methods require extensive exploration. To address these limitations, this article proposes a novel mixture of gradients (MoG) method, which integrates gradients from different sources within the training process in order to improve the convergence performance of neural networks (NNs). Particularly, MoG is a modular, plug-and-play solution requiring no structural modifications to existing NNs. Its implementation necessitates only minor modifications to the loss function, where the label-based supervised loss is combined with a label-free loss through weighted summation. The label-free loss can be either unsupervised loss or reinforcement learning loss. This flexibility allows seamless integration into nearly all NN-based methods, making it applicable to a wide range of wireless optimization problems with minimal implementation cost. Extensive simulations across multiple classic wireless scenarios demonstrate that MoG can significantly enhance the performance of NN decision-making, leading to higher transmission rates.**

***Index Terms*—Deep learning (DL), mixture of gradient (MoG), training paradigm, wireless network optimization.**

Received 6 March 2025; revised 31 March 2025; accepted 2 April 2025. Date of publication 9 April 2025; date of current version 27 June 2025. This work was supported in part by the National Key Research and Development Program of China under Grant 2022YFB2901900, and in part by the Dongguan Strategic Scientist Teams Project under Grant 20231900700022. This article was presented in part at the 2023 IEEE 98th Vehicular Technology Conference: VTC2023-Fall, Hong Kong, China , October 10–13, 2023 [DOI: 10.1109/VTC2023-Fall60731.2023.10333602]. *(Corresponding author: Nan Cheng.)*

Nan Cheng is with the School of Electrical Engineering and Intelligentization, Dongguan University of Technology, Dongguan 523808, China, and also with the State Key Laboratory of ISN and School of Telecommunications Engineering, Xidian University, Xi'an 710071, China (e-mail: dr.nan.cheng@ieee.org).

Longfei Ma and Xiucheng Wang are with the State Key Laboratory of ISN and School of Telecommunications Engineering, Xidian University, Xi'an 710071, China (e-mail: lfma@stu.xidian.edu.cn; xcwang_1@stu.xidian.edu.cn).

Yanpeng Dai is with the School of Information Science and Technology, Dalian Maritime University, Dalian 116026, China (e-mail: yanpengdai@dlmu.edu.cn).

Qihao Li is with the College of Communication Engineering, Jilin University, Changchun 130021, Jilin, China (e-mail: qihaol@jlu.edu.cn).

Wei Quan is with the School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China (e-mail: weiquan@bjtu.edu.cn).

Hui Liang is with the School of Electrical Engineering and Intelligentization, Dongguan University of Technology, Dongguan 523000, China, and also with the Frontier Academic Center, Peng Cheng Laboratory, Shenzhen 518000, China (e-mail: huiliang@dgut.edu.cn).

Xuemin Shen is with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: sshen@uwaterloo.ca).

Digital Object Identifier 10.1109/JIOT.2025.3559063

I. Introduction

Wireless network optimization focuses on rationally allocating network resources to meet service demands and maximize system performance [2], [3]. With the rapid development of 6G networks, emerging services demonstrate unprecedented diversity and personalization, including millisecond-level delay for Industrial Internet of Things (IoT), ultrahigh throughput in extended reality (XR), and million-device connectivity for digital twin systems [4], [5]. Confronted with diverse service requirements, traditional optimization methods encounter exponential growth in computational complexity within complex wireless environments, struggling to satisfy the real-time response requirements. Deep learning (DL) has recently emerged as a promising paradigm in signal detection, user access control, and radio resource management, leveraging its capacity for end-to-end nonlinear mapping [6], [7], [8]. For instance, convolutional neural networks (CNNs) excel at extracting local features from data, enabling them to effectively identify key frequency information from spectrum maps [9]. Recurrent neural networks (RNNs), adept at processing time-series data, enhance channel estimation accuracy by capturing temporal dependencies within signals [10]. In ultradense network scenarios, graph neural network (GNN)-based frameworks have demonstrated improved throughput and spectral efficiency through modeling the correlations between channel state information and network topology [11]. Crucially, network performance is not only determined by the inherent strengths of the neural network (NN) architecture but is also significantly influenced by the training methodology employed. Specifically, supervised learning (SUP) often suffers from overfitting due to its reliance on labeled datasets [12], whereas reinforcement learning (RL) faces inherent challenges in balancing exploration efficiency with convergence stability within complex wireless environments [13]. These training-level performance bottlenecks ultimately limit the applicability of DL solutions in various scenarios in 6G networks.

Unlike classical convex optimization algorithms that maintain broad applicability across communication network scenarios by virtue of their mathematical generality [14],

current advancements in training methods for learning-based wireless networks are typically realized through problem-specific designs to achieve enhanced performance in particular scenarios. This inherent specialization constrains their generalization capabilities across diverse network environments. Although recent studies have sought to improve model robustness by introducing penalty terms in loss functions or regularization based on physical models [15], [16], such improvements remain constrained by their prior assumptions regarding wireless environment statistics. Consequently, existing training paradigms face generalization bottlenecks under varying network conditions. When fundamental network characteristics evolve, previously effective training approaches may suffer significant performance degradation or even complete failure [17], [18]. This scenario dependency primarily originates from excessive sensitivity to network features [19]. While emerging meta-learning frameworks have demonstrated potential for adaptive optimization strategy adjustment in wireless communication systems [20], [21], their practical implementation faces challenges due to deployment complexity caused by dual-level optimization structures. Although problem-specific training techniques work well in 4G/5G networks, the explosive growth of diversified and personalized scenarios in 6G networks poses efficiency challenges for such scenario-specific designs [22]. This evolution necessitates the development of a NN enhancement technique applicable across various communication scenarios, essential for fulfilling 6G networks' stringent service performance requirements.

Therefore, it is of great interest in developing a training framework that is decoupled from model architectures and problem-agnostic. Fortunately, advances in artificial intelligence (AI) have demonstrated promising innovations in training mechanisms. For instance, knowledge distillation (KD) techniques have achieved generalized model performance enhancement through teacher-student collaborative learning frameworks that implement knowledge transfer at the loss level [23]. The core value of such methods lies in their ability to integrate different training information through well-designed loss functions, while preserving the original network architecture, thereby providing insights for overcoming existing training paradigm constraints. However, there are still challenges in applying these methods to wireless networks, as they typically require pretrained teacher models during knowledge transfer. Furthermore, current training methods have not comprehensively addressed gradient coordination between supervised and unsupervised information, which limits the improvement of NNs' cognitive and decision-making capacities in complex wireless environments. Notably, the effectiveness of addressing this issue has already been validated in the AI field [24].

In this article, we propose a novel generalized training paradigm based on current NN architectures and training methods in wireless networks, called mixture of gradient (MoG), which is widely applicable to various problems in wireless communication systems. The focus of this article does not aim to enhance the performance of a specific scenario and problem, but rather strives to provide a comprehensive, universally applicable framework designed to accommodate a wide range of communication scenarios. Specifically, for various network optimization problems, we first employ existing optimization algorithms to generate a collection of high-quality solutions, which serves as the labeled training dataset. A supervised loss term is constructed based on the discrepancy between these labeled data and the decision solutions output by the NN model. Subsequently, we develop label-free unsupervised loss or RL loss according to the optimization objectives and model outputs. These loss components are linearly combined with the supervised loss through weighting coefficients, forming a novel loss function for optimizing NN parameters. This training paradigm naturally achieves gradient mixing by simultaneous backpropagation of multisource loss. The main contributions of this article are as follows.

1) For future learning-based wireless network, we propose an innovative MoG training paradigm that achieves scenario-agnostic performance enhancement for NN-based algorithms by integrating existing training methods. This method serves as a general and simple technique, which is completely unconstrained by NN architecture and specific problem characteristics.
2) The proposed MoG method, as a plug-and-play solution, can be implemented by simply modifying the loss function. Only the gradient source needs to be changed during training, which does not cause any additional computation for forward inference and does not require additional training data.
3) Extensive experiments conducted in several classic wireless network optimization scenarios show that compared with traditional training methods, the proposed MoG method improves the decision-making performance of mainstream NN models, thereby significantly enhancing the sum rate of wireless systems.

The preliminary version of this work was published in [1], which only focused on continuous optimization in a single scenario. This paper considers different optimization problems in a broader range of wireless network scenarios and involves more existing training methods, fully demonstrating the effectiveness and generality of the proposed scheme.

The remainder of this article is organized as follows. In Section II, the system model and problem formulation are presented. We propose a MoG-based training paradigm for the network optimization problem in Section III, followed by the extensive simulation results presented in Section IV. Finally, Section V concludes this article.

II. System Model and Problem Formulation

In this section, we investigate two landmark problems: 1) power control in device-to-device (D2D) networks and 2) resource block (RB) allocation in cellular networks, which are described in detail as follows.

A. Power Control in Device-to-Device Networks

We consider a D2D network with K single-antenna transceiver pairs randomly distributed on a 2-D plane, as shown in Fig. 1. The network comprises K direct-link channels, each established between a distinct transceiver pair for communication. However, there are $K \times (K-1)$ cross-link channels due to shared spectrum resources, which impairs

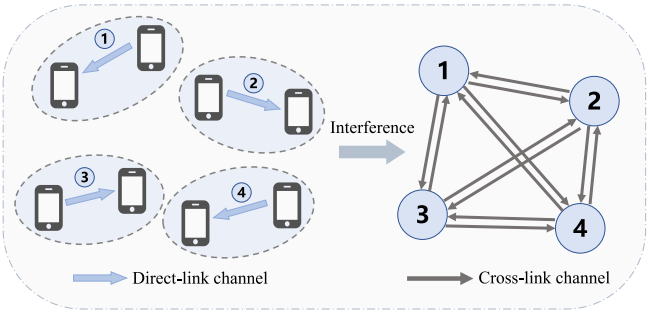

Fig. 1. System model of D2D network.

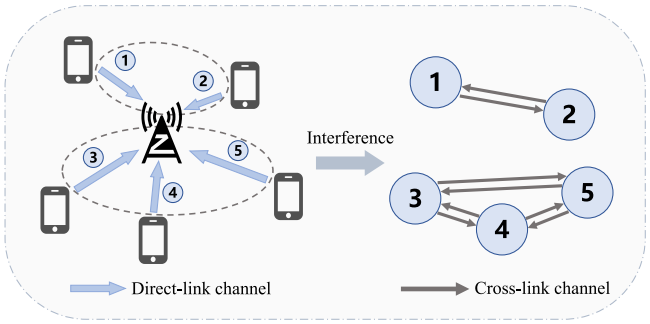

Fig. 2. System model of cellular network.

the expected data transmission. This network feature requires proper allocation of the transmission power to ensure efficient communication between the designated communication pairs.

The optimization objective of power control is to maximize the sum rate across all pairs in the network, which is defined as follows:

Problem 1:

$$\max_{p_1,\ldots,p_K} \sum_{i=1}^{K} \log_2\left(1+\frac{|h_{i,i}|^2 p_i}{\sum_{j\neq i}|h_{j,i}|^2 p_j+\sigma^2}\right) \tag{1}$$

$$\text{s.t. } 0 \le p_i \le 1 \quad \forall i \tag{1a}$$

where p_i represents the transmission power allocated to the ith link's sender, and $h_{i,j}$ denotes the channel state between the sender of the ith link and the receiver of the jth link. σ^2 corresponds to the noise power at the receiver. The logarithmic function in the objective represents the Shannon capacity of each link, reflecting the theoretical maximum transmission rate achievable under this SINR.

B. Resource Blocks Allocation in Cellular Networks

This scenario examines the uplink transmission in a cellular network, where a base station (BS) serves K randomly distributed user equipments (UEs) within its coverage area. It is assumed that the BS has N orthogonal RBs that will be allocated to active UEs. The orthogonality of the RBs means that parallel transmissions using different RBs will not interfere with each other. However, the number of UEs in a cell is often larger than the number of RBs to improve spectral efficiency. This imbalance makes it necessary to allocate a single RB to multiple transmission links at the same time, which inevitably leads to channel interference between links sharing the same RB. In the example of Fig. 2, direct-link channels 1 and 2 share the same RB, and the other direct-link channels share the same RB.

Similarly, the optimization objective of RB allocation is to optimize the sum rate over all UEs in the cell, which is defined as follows:

Problem 2:

$$\max_{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_K} \sum_{i=1}^{K} \log_2\left(1+\frac{|h_{i,i}|^2 p_i}{\sum_{i\neq j} x_{i,j}|h_{i,j}|^2 p_j+\sigma^2}\right) \tag{2}$$

$$\text{s.t. } x_{i,j} \in \{0,1\} \quad \forall i,j \tag{2a}$$

$$p_i = f(\cdot) \quad \forall i \tag{2b}$$

where $\boldsymbol{x}_i = [x_{i,1},\ldots,x_{i,K}]$, $x_{i,j}$ denotes the RB selection of UE i and UE j. Specifically, $x_{i,j}=1$ indicates that the two UEs use the same RB for data transmission, while $x_{i,j}=0$ indicates that different RBs are used. It should be noted that for UEs sharing the same RBs, power control is reformulated as an optimization problem, consistent with that discussed in the previous section. Given that the main focus of this part is the allocation of RBs, a predetermined optimization algorithm $f(\cdot)$ [25] is used to allocate the power to the UEs after the selection of RBs.

III. Mixture of Gradients for Wireless Network

The proposed MoG is a simple integration of existing methods, so it is necessary to first provide a comprehensive overview of the NN training techniques currently used in the field of wireless networks.

A. Different Training Methods for Wireless Network

The rapid development of DL has given rise to several NN training methods, including SUP based on data labels, unsupervised learning (UNSUP) without data labels, and RL utilizing reward mechanisms, and this section will present how different training methods optimize the parameters of NNs and discuss their features.

1) *SUP*, as a training method that requires the high-quality solution $\boldsymbol{y}^*$ of optimization objective as the label, has been widely used in early NN-based wireless network optimization algorithms. During SUP training of an NN, the model's output $\boldsymbol{y}=\mathcal{F}(\boldsymbol{x};\boldsymbol{\theta})$ must closely approximate the target label $\boldsymbol{y}^*$, so the model parameters $\boldsymbol{\theta}$ in the mapping function $\mathcal{F}$ are updated according to the following equation:

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{y}} \|\boldsymbol{y}^* - \boldsymbol{y}\|_2^2 \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{x};\boldsymbol{\theta}) \tag{3}$$

where α is learning rate and $\boldsymbol{x}$ is the input of NN. Despite its simplicity and effectiveness, the practical application of SUP faces two persistent challenges: 1) obtaining high-quality solutions as *labels* and 2) *overfitting*. Unlike the field of computer vision, which benefits from abundant, high-quality datasets, wireless communication systems are frequently confronted with nonconvex optimization problems that pose substantial difficulties in acquiring labeled training data. Despite this challenge, the solutions generated by conventional algorithms can be leveraged to serve as labels [26].

However, this approach inherently constrains the potential of NNs. Exclusive reliance on SUP fundamentally restricts NNs from exceeding the performance of their training labels, as the learning process is bounded by the inherent limitations of algorithmic frameworks. Moreover, since the training goal of SUP is to minimize the distance between the output and the label, the performance of the NN drops dramatically if the distribution of y^* corresponding to the test environment is different from that of the labels, which is known as overfitting. However, it can also be shown that SUP can make the NN learn the features and distributions of the solutions corresponding to the labels, which is one of the inspirations for the proposed method.

2) *UNSUP* offers a robust method for training NNs by recasting the optimization problem into a differentiable loss function. This approach directly updates the NN parameters by applying the chain rule, which significantly simplifies the training process and has the potential to discover complex patterns in the data that are not easily detected through SUP [27]. By adopting the UNSUP approach, the NN is more inclined to identify and utilize the underlying structure of the dataset during the training process, resulting in a more robust and generalized model. Take the optimization goal (1) as an example, the parameters of the NN can be updated through the following chain derivation:

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{p}} \mathcal{H}(\boldsymbol{p}|\boldsymbol{h}) \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{h}; \boldsymbol{\theta}) \tag{4}$$

where $\mathcal{H}(\boldsymbol{p}|\boldsymbol{h}) = \sum_{i=1}^{K} \log_2 [1 + ([p_i|h_{i,i}|]/[\sum_{j \neq i} p_j |h_{j,i}| + \sigma^2])]$, $\boldsymbol{p} = \mathcal{F}(\boldsymbol{h}; \boldsymbol{\theta})$. Compared to other methods, UNSUP not only eliminates the reliance on labeled data but also mitigates the influence of the randomness of exploration on the training efficiency of NNs. However, given the nonconvex nature of (1), optimization of the power vector $\boldsymbol{p}$ via gradient descent may converge to a local optimum without assurance of global convergence. Therefore, a feasible strategy to improve the efficiency of UNSUP training can involve the refinement of the directional information between the current power vector $\boldsymbol{p}$ and the optimal solution $\boldsymbol{p}^*$.

3) *RL* updates the NN parameters based on the sum rate that can be reached at the current state by letting the NN interact with the wireless environment continuously, so that the action of the NN can obtain higher returns. The two optimization problems in the wireless networks we focus on are characterized by continuous and high-dimensional action spaces, respectively, which pose significant challenges for value-based RL algorithms. Such algorithms require explicit discretization of all actions and extensive exploration in the action space to evaluate every possible choice. This leads to substantial training and storage costs, and often results in difficulty converging. Therefore, in this article, we discuss two commonly used and classical policy-based RL algorithms: 1) policy gradient (PG) [28] and 2) deterministic PG (DPG) [29], which explore directly in the policy space and naturally handle continuous actions. Despite the absence of a necessity for labeled data in both RL algorithms and UNSUP techniques, a unique advantage of RL over UNSUP is its independence from the differentiability of the optimization function. RL algorithms estimate agent performance directly through reward signals and utilize this assessment to iteratively update policy parameters. Briefly, policy-based RL methods leverage gradient estimates of expected returns to guide the optimization process without focusing on the specific form of the optimization objective. This is in contrast to UNSUP that require direct analytical gradients of the objective function for optimization, a requirement that is circumvented in the RL paradigm.

As a classic policy-based RL algorithm, PG updates the $\boldsymbol{\theta}$ as follows:

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha R \nabla_{\boldsymbol{\theta}} \log \mathcal{F}(\boldsymbol{x}; \boldsymbol{\theta}) \tag{5}$$

where R is the value of reward. For the above two concerned problems, R is the sum rate of the system. A distinctive feature of PG can be seen in (5), i.e., PG does not directly output a deterministic $\boldsymbol{y}$, but rather outputs a distribution of $\boldsymbol{y}$, which is sampled to obtain a specific $\boldsymbol{y}$.

Unlike PG, DPG directly learns a deterministic strategy that outputs a single optimal action in a given state rather than a probability distribution of the action. In the DPG, the $\boldsymbol{\theta}$ are updated as follows:

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{y}} \mathcal{V}(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{\theta}_v) \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{x}; \boldsymbol{\theta}) \tag{6}$$

where $\mathcal{V}$ is the value NN used to evaluate the reward of the $\boldsymbol{y}$ output by $\mathcal{F}$ under $\boldsymbol{x}$. A distinctive feature of DPG can be seen in (6), in addition to the actor NN $\mathcal{F}$ a value NN $\mathcal{V}$ also needs to be trained, and according to [29] the parameters $\boldsymbol{\theta}_v$ of $\mathcal{V}$ are updated as $\boldsymbol{\theta}_v = \boldsymbol{\theta}_v - \alpha \| \mathcal{V}(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{\theta}_v) - R \|$.

Although policy-based RL algorithms show higher exploration efficiency compared to value-based methods, the intrinsic complexity of exploration remains a critical challenge. Specifically, these algorithms guide the exploration process by adjusting the policy parameters in the direction of expected reward increase. PG methods introduce stochasticity by sampling from a probability distribution of actions, while DPG methods adopt a deterministic approach, choosing actions that are predicted to maximize expected returns. Despite these well-designed optimization methods, the lack of explicit labels for guidance means that the NN still needs to undergo a large number of iterations to refine its policy and thus accumulate higher cumulative rewards. Although the iterative approach of DPG is more directional than simple random sampling, it may not always lead to optimal exploration. The NN is still required to delicately balance the exploration of novel actions against the exploitation of previously identified rewarding actions. Overall, the policy-based RL algorithm retains a certain amount of randomness in exploration, which can help NNs find optimal policies in some cases, but has the potential to reduce efficiency.

As a result, despite the advancements in exploration techniques, policy-based RL algorithms continue to face the challenge of rationalizing the exploration-exploitation tradeoff. Leading NNs toward an action space where high-performance solutions are more likely to be available is a promising option

that can significantly improve the performance of RL-based methods.

Based on the above analysis, we summarize the comparison of existing training methods in Table I.

B. Implementation of MoG Based on Existing DL Methods

As mentioned in previous discussions, KD has been proposed, which improves the training effect by integrating various supervised information. Specifically, KD enhances the performance of the smaller, less complex student model $\mathcal{F}$ by enabling it to mimic the behavior of a larger, high-performance teacher model $\mathcal{G}$. This is achieved by jointly minimizing the divergence in output distributions between the teacher and student models, alongside the loss function of the student model. As a result, the parameter set $\boldsymbol{\theta}$ of the student model $\mathcal{F}$ undergoes iterative refinement, assimilating the knowledge distilled from the teacher model. This process facilitates the student model's approximation of the teacher's performance as follows:

$$\boldsymbol{\ell}_{\boldsymbol{y}=\mathcal{F}(\cdot)} = \boldsymbol{\ell}_{\text{hard}}(\boldsymbol{y}, \boldsymbol{t}) + \beta \boldsymbol{\ell}_{\text{soft}}(\boldsymbol{y}, \boldsymbol{g}) \tag{7}$$

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{y}} \boldsymbol{\ell} \nabla_{\boldsymbol{\theta}} \mathcal{F}(\cdot; \boldsymbol{\theta}) \tag{8}$$

where $\boldsymbol{y}$ and $\boldsymbol{g}$ are the outputs of the student model $\mathcal{F}$ and the teacher model $\mathcal{G}$, respectively. $\boldsymbol{\ell}$ is the loss function, which consists of $\boldsymbol{\ell}_{\text{hard}}(\boldsymbol{y}, \boldsymbol{t})$ that measures the distance between the student's output and the true label $\boldsymbol{t}$, and $\boldsymbol{\ell}_{\text{soft}}(\boldsymbol{y}, \boldsymbol{g})$ that measures the distance between the student's output and the teacher's output, weighted by the factor β. $\boldsymbol{\theta}$ represents the parameters of the student model $\mathcal{F}$.

The extensive practical applications of KD demonstrate that combining two distinct loss functions can significantly enhance NN performance, with the efficacy of this methodology being thoroughly validated through empirical evidence. It is important to note that the differences between these loss functions primarily come from the origin of the labels, which essentially provide diverse gradient information for parameter optimization of the NNs. Based on the above analysis, the integration of existing training methods may be effective in solving problems in the field of wireless networks. Therefore, we integrate SUP loss based on algorithm labels into a framework usually associated with UNSUP or RL that does not rely on labels, becoming a new training paradigm called MoG. The details of MoG are described below.

As described in Section III-B, in SUP, labels are generated by a predefined algorithmic method aimed at minimizing the discrepancy between the NN's output y and the target output y^*. In UNSUP, the NN is optimized to maximize the objective function, while in RL, it is optimized to increase the expected reward. Implementing MoG based on these methods is very simple, i.e., weighted combination of the loss term of SUP with UNSUP and RL, thus achieving a mixture of different gradients during parameter update.

Adding a supervised term to the loss function of UNSUP, the parameters are updated as follows:

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{y}} \left(\mathcal{H}(\boldsymbol{y}|\boldsymbol{x}) + \beta \left\| \boldsymbol{y} - \boldsymbol{y}^* \right\|_2^2 \right) \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{x}; \boldsymbol{\theta}) \tag{9}$$

where $\mathcal{H}(\boldsymbol{y}|\boldsymbol{x})$ is the objective function to be maximized, β is a coefficient balancing the unsupervised objective with the supervised term.

The parameter update process of the PG algorithm can be reformulated as follows:

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \left(R \nabla_{\boldsymbol{\theta}} \log \mathcal{F}(\boldsymbol{x}; \boldsymbol{\theta}) + \nabla_{\boldsymbol{y}} \beta \left\| \boldsymbol{y} - \boldsymbol{y}^* \right\|_2^2 \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{x}; \boldsymbol{\theta}) \right). \tag{10}$$

Similarly, after the addition, the parameter update equation of the DPG algorithm can be represented as follows:

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{y}} \left(\mathcal{V}(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{\theta}_v) + \beta \left\| \boldsymbol{y} - \boldsymbol{y}^* \right\|_2^2 \right) \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{x}; \boldsymbol{\theta}) \tag{11}$$

with $\mathcal{V}$ representing the value function parameterized by $\boldsymbol{\theta}_v$.

A careful analysis of (9)–(11) reveals that the direction of the gradient update for the NN is related to the output vector $\boldsymbol{y}$. The main difference between the various NN architectures is the intrinsic mechanism used to extract features from the data, rather than the form of the output. Consequently, the proposed MoG training paradigm is generalizable, being compatible with a range of NN architectures including multilayer perceptrons (MLPs), CNNs, GNNs.

When referring to the power control problem described in Section II-A, the parameter update of NN can be expressed as follows:

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{p}} \left(\mathcal{H}(\boldsymbol{p}|\boldsymbol{h}) + \beta \left\| \boldsymbol{p} - \boldsymbol{p}^* \right\|_2^2 \right) \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{h}; \boldsymbol{\theta}) \tag{12}$$

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \left(R \nabla_{\boldsymbol{\theta}} \log \mathcal{F}(\boldsymbol{h}; \boldsymbol{\theta}) + \nabla_{\boldsymbol{p}} \beta \left\| \boldsymbol{p} - \boldsymbol{p}^* \right\|_2^2 \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{h}; \boldsymbol{\theta}) \right) \tag{13}$$

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{p}} \left(\mathcal{V}(\boldsymbol{p}|\boldsymbol{h}; \boldsymbol{\theta}_v) + \beta \left\| \boldsymbol{p} - \boldsymbol{p}^* \right\|_2^2 \right) \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{h}; \boldsymbol{\theta}) \tag{14}$$

where $\mathcal{H}(\boldsymbol{p}|\boldsymbol{h}) = R = \sum_{i=1}^{K} \log_2 [1 + ([|h_{i,i}|^2 p_i]/[\sum_{j \neq i} |h_{j,i}|^2 p_j + \sigma^2])]$, $\boldsymbol{p}^*$ is the high-quality solution obtained by existing optimization algorithms, the input of NN is the channel state matrix $\boldsymbol{h}$, and the output is the power control vector $\boldsymbol{p}$.

Similarly, for the RB allocation problem, the parameter update process can be expressed as follows:

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \left(R \nabla_{\boldsymbol{\theta}} \log \mathcal{F}(\boldsymbol{h}; \boldsymbol{\theta}) + \nabla_{\mathbf{X}} \beta \left\| \mathbf{X} - \mathbf{X}^* \right\|_2^2 \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{h}; \boldsymbol{\theta}) \right) \tag{15}$$

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \nabla_{\mathbf{X}} \left(\mathcal{V}(\mathbf{X}|\boldsymbol{h}; \boldsymbol{\theta}_v) + \beta \left\| \mathbf{X} - \mathbf{X}^* \right\|_2^2 \right) \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{h}; \boldsymbol{\theta}) \tag{16}$$

where $R = \sum_{i=1}^{K} \log_2 [1 + ([|h_{i,i}|^2 p_i]/[\sum_{i \neq j} x_{i,j} |h_{i,j}|^2 p_j + \sigma^2])]$, $\mathbf{X}^*$ is the high-quality solution obtained by existing optimization algorithms, the input of NN is the channel state matrix $\boldsymbol{h}$, and the output is the RB allocation vector $\mathbf{X}$. It is worth noting that the nondifferentiability of the optimization objective makes UNSUP unusable, so we did not provide the expression corresponding to (9).

The above description shows that when applying MoG to specific problems, only minor modifications to existing methods are needed, without involving complex targeted designs. For easier understanding, we have drawn a flowchart of MoG, as shown in Fig. 3. The benefits of such approach have been substantiated in Section IV, where performance enhancements are comprehensively demonstrated. This novel

TABLE I
COMPARISON OF EXISTING NN TRAINING METHODS FOR WIRELESS NETWORKS

Existing Method	No Need for Labels	High Convergence Stability	Convergence Optimality Guarantee	Low Overfitting Risk
Supervised Learning	✗	✓	Close to Label	✗
Unsupervised Learning	✓	✗	✗	✓
Reinforcement Learning	✓	✗	✗	✓

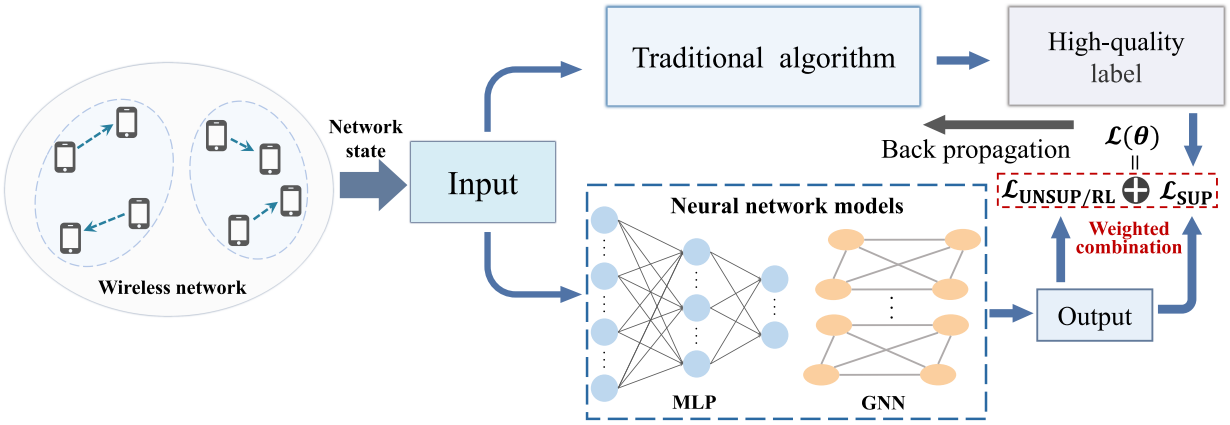

Fig. 3. Proposed MoGs framework, where $\ell_{\text{UNSUP/RL}}$ denotes the loss function for traditional UNSUP and RL, and ℓ_{SUP} denotes the supervised loss.

strategy leverages the strengths of different learning mechanisms, potentially leading to significant improvements in the domain of wireless communication systems.

C. Discussion of MoG

In the proposed MoG training paradigm, high-quality labeled datasets can be regarded as the teacher distribution. By supplementing the supervised term constructed from the labels to UNSUP and RL, the model performance can be improved. The training process of MoG can be seen as introducing a mechanism similar to KD in UNSUP and RL. The effectiveness of this scheme can be analyzed from the following aspects. First, the labels used in SUP are usually the optimal or high-performing suboptimal solutions. During the parameter update process, the NN continuously narrows the distance between the output and the label, thus being able to approach the performance of the label solution [30]. Although in the classic KD process, the direction of knowledge transfer is from a large-scale teacher model to a small student model, this strategy of optimizing NN parameters by imitating high-performance distributions has been proven to have wide applicability [31]. In the MoG training method, the label data used is generated by existing algorithms, rather than from large-scale teacher NNs. Similar to the output of the teacher model, the high-quality solutions provided by algorithms also contain complex optimization information, which is difficult to obtain through autonomous exploration in UNSUP or RL. Moreover, the optimal solution in the label explicitly provides a gradient direction close to the global optimum for training. The introduction of algorithm labels is similar to supplementing prior knowledge in the model training process, thus reducing ineffective exploration. From another perspective, compared to the traditional single

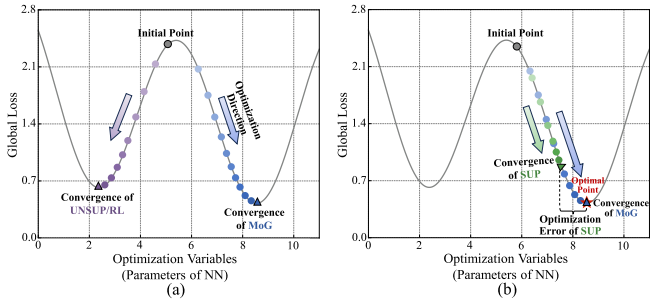

Fig. 4. Example of NN parameter optimization. (a) Shows the reason why MoG outperforms UNSUP or RL, and (b) shows the reason why MoG outperforms SUP.

training pattern, using both SUP and UNSUP/RL simultaneously is equivalent to providing a more diverse information source for parameter updating [32]. SUP provides clear label information, helping NNs find the best gradient direction for the current training samples. At the same time, UNSUP or RL drives NNs to learn the inherent structure of data. By setting the weights of supervised terms properly to balance the proportions of different loss terms, the role of label information and autonomous exploration can be fully utilized.

Furthermore, during the training process, the NN aims not only to minimize the difference with the label distribution but also to maximize the reward of RL or minimize the loss of UNSUP. This joint optimization strategy is designed to ensure that while the NN assimilates the advantages of the optimization algorithm's solutions, it also satisfies the optimization objective of UNSUP/RL. This approach is equivalent to introducing an additional regularization term during training, which can prevent over-reliance on the optimization algorithm's solutions and maintain the model's generalizability [33].

In order to intuitively demonstrate the reason for the performance improvement brought by MoG, we draw a simple example as shown in Fig. 4. The black dots in the figure indicate the initial point of NN training, the dots with different shades of colors indicate the optimized trajectories of NN training using different methods. The optimization direction is indicated by the darkening of the color, and the triangles indicate the convergence positions. As can be seen in Fig. 4(a), for a given initial position, using the UNSUP or RL methods may converge to the local optimum along the gradient descent, whereas MoG is more likely to converge to the global optimum since it utilizes the positional information of the optimum embedded in the labels. However, due to the limitation of the number of labels used for training, the NN trained by SUP inevitably has some optimization error between its convergence position and the global optimum, as shown in Fig. 4(b). In other words, there will always be a certain inherent distance between the convergence position provided by the labels and the global optimum in the case of limited data amount [30], [34]. In this case, the UNSUP or RL component in the MoG can help the NN to better converge to the optimal point. It should be noted that the given example is only a simple qualitative illustration, and a rigorous theoretical justification of the proposed method is still challenging.

IV. Simulation Results

In this section, we present a comprehensive evaluation to substantiate the effectiveness and broad applicability of the proposed method. The performance of MoG is systematically examined through extensive experiments encompassing diverse optimization challenges and multiple NN architectures, as outlined in the previous sections. For clarity and precision of the exposition, the training patterns are categorized into two different types based on the data sources: 1) dataset pattern and 2) RL pattern. The dataset pattern trains the model on static, predetermined datasets, whereas the RL pattern trains through the interaction of the model with wireless network environment. The optimization goals of these two patterns are completely identical. We deploy two NN architectures, MLP and GNN, based on related studies [11], [26], and implemented them using Pytorch 2.0.0 and DGL 0.6.0, respectively. For the power control problem, we use the state-of-the-art optimization algorithm FPLinQ [25] to obtain the label, and for the RB allocation problem, we use a search algorithm to obtain the optimal solution as the label. Both use the channel state information as the input to the NN and the output of as the respective optimization variables. Throughout the training phase, the stochastic gradient descent (SGD) optimizer is utilized for the adjustment of model parameters.

A. Performance Evaluation of Power Control

First, in order to visually show the training trajectories of the various methods and to illustrate the advantages of the proposed method in guiding the NN toward the global optimum, Fig. 5 is plotted. This figure builds the MoG framework based on UNSUP and SUP, with training conducted on MLP. The number of user pairs K is set to 20, and the amount of training data is 5000. Fig. 5(a) presents the normalized mean square error (NMSE) for the outputs of the model compared to the labels across both the training and test datasets. Notably, the UNSUP method, which is trained without the use of labeled data, has the smallest difference in NMSE between the training and test sets. This result is readily comprehensible because in the absence of label data, the model is less likely to overfit the training set, which often helps the model generalize to a global optimum. Despite the absence of labeled data, the model's outputs still exhibit a convergence toward the labels as training progresses, reflecting the capability of UNSUP to edge closer to the optimal solution through autonomous exploration of the data's inherent structure. In sharp contrast, the SUP method, which relies exclusively on labeled data, exhibits the largest difference in NMSE between the training and test sets, a result that is consistent with expectations. The proposed MoG achieves the lowest NMSE on the test set, even though its NMSE on the training set is slightly higher compared to SUP. This indicates that MoG possesses a superior capacity to approximate the global optimum, which is attributed to its effective integration of the strengths inherent in both UNSUP and SUP approaches.

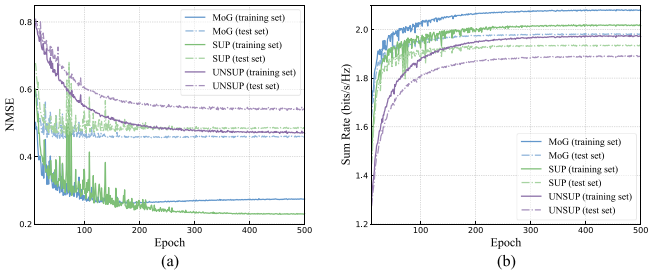

Fig. 5. Comparative convergence analysis. (a) Shows the variation of NMSE during the training process for the different methods, and (b) shows the variation of the sum rate during the training process for the different methods.

Fig. 5(b) depicts the ultimate performance metric of the model on both the training and test sets, i.e., the sum rate of systems. It is obvious that MoG achieves the highest sum rate on the testing set. Remarkably, MoG concurrently obtains the highest sum rate on the training set as well, a result that is different from the one observed in Fig. 5(a). This outcome proves that the straightforward SUP does not represent the most effective way for a model to learn knowledge from labeled data in addressing the classic issues of wireless networks. While the performance of UNSUP is not outstanding in this case, significant performance enhancements can be obtained by merging UNSUP and SUP in the MoG paradigm.

In the following contents, to comprehensively illustrate the performance of various methods, we present simulation results from diverse NN architectures and training patterns to show the viability of the proposed MoG framework. In particular, for the power control problem, we give three strategies for setting the distillation weight β, which will be employed in the subsequent simulations: fixed weights (MoG w/ fixed β), increasing weights with training (MoG w/ increasing β), and decreasing weights with training (MoG w/ decreasing β). Specifically, when training the MLP in dataset pattern, for fixed weights, we set β to 10; for increasing weights, we set the initial value of β to 1 and multiply it by 1.2 for every 2 epochs, with an upper limit of 10; for decreasing weights, we

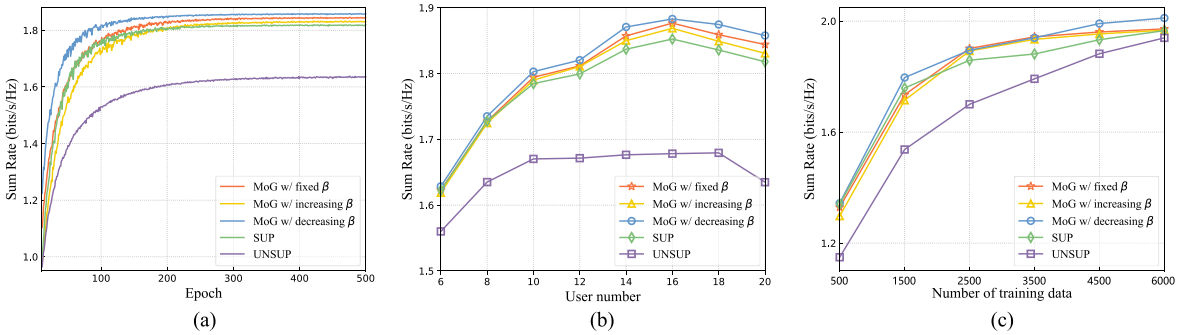

Fig. 6. Performance comparison of MLP training in dataset pattern. (a)–(c) Show the convergence of different methods for training MLP in dataset pattern, the sum rate variation with the number of users, and the sum rate variation with the number of training data, respectively.

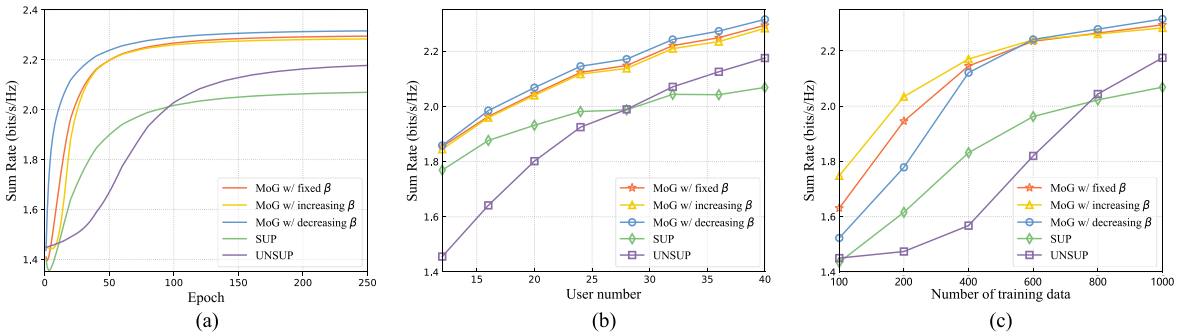

Fig. 7. Performance comparison of GNN training in dataset pattern. (a)–(c) show the convergence of different methods for training GNN in dataset pattern, the sum rate variation with the number of users, and the sum rate variation with the number of training data, respectively.

set the initial value of β to 20 and multiply it by 0.8 for every 20 epochs. When training the GNN, for fixed weights, we set β to 10; for increasing weights, we set the initial value of β to 1 and multiply it by 1.2 for each epoch, with an upper limit of 20; for decreasing weights, we set the initial value of β to 100 and multiply it by 0.75 for each epoch.

Fig. 6 shows the performance of various training methods applied to MLP within the dataset pattern. Unless otherwise specified, we assume that $K = 20$, and the training and test sets contain 2000 and 1000 samples, respectively. Fig. 6(a) demonstrates the convergence performance of the MLP when trained under the dataset pattern, showing that MoGs outperform both SUP and UNSUP. Notably, MoG with decreasing β exhibits optimal performance and convergence rate. Fig. 6(b) examines the adaptability of MLP to varying user counts, also showing that MoGs ensure optimal performance. It can be observed that as the number of users increases, the sum rate first rises and then shows a decline. The rise in the sum rate is attributed to the fact that the theoretical maximum the sum rate increases with the number of users, which corresponds to an increase in the upper bound of the sum rate achievable by the model. However, as the number of users continues to increase, the complexity of power allocation increases. In the face of increasing complexity, the representation capabilities of MLPs are insufficient to effectively solve the problem. This inadequacy leads to a widening of the gap between the model output and the optimal solution, and hence to a degradation of performance. Fig. 6(c) explores the sum rate under different training dataset sizes, where MoGs still show a performance advantage. SUP outperforms UNSUP for smaller datasets. However, as the number of data increases, the performance gap between UNSUP and SUP narrows.

Fig. 7 presents a comparative analysis of the performance when training a GNN using diverse methods within the dataset pattern. Unless otherwise specified, we assume that $K = 40$, and the training and test sets contain 1000 and 500 samples, respectively. Fig. 7(a) presents the convergence performance of the GNN, which shows that MoGs outperform the benchmark methods and the MoG with decreasing β achieves optimal performance and convergence speed. UNSUP converges to a better final performance compared to SUP, but the convergence speed is slower. Fig. 6(b) explores the robustness of GNN to varying user quantities, and MoGs still have performance advantages. In addition, it can be observed that SUP is better with fewer users, while UNSUP outperforms SUP as the number of users increases. Unlike the phenomenon observed in Fig. 6(b), the GNN does not show a performance decline with an increase in user numbers, which can be attributed to its enhanced representational capabilities in comparison to the MLP. Fig. 7(c) shows the GNN's performance under different training dataset sizes. Consistent with the MLP results, MoGs provide optimal performance in most dataset sizes. SUP demonstrates an advantage with smaller datasets, while UNSUP outperforms SUP as the dataset expands. Within different MoGs, the MoG with a fixed β and the MoG with an increasing β are more suitable for smaller datasets, whereas the MoG with a decreasing β is better suited for larger datasets.

Within RL pattern, we employ DPG algorithm as a benchmark and set up MoG based on it. When training the MLP,

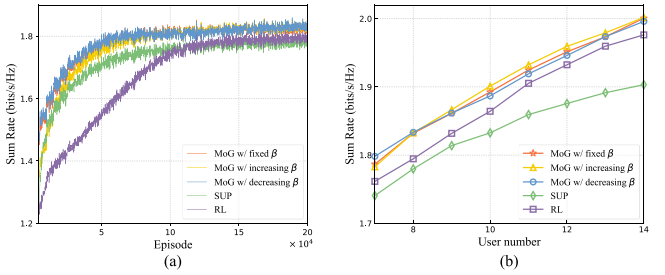

Fig. 8. Performance comparison of MLP in RL pattern. (a) and (b) show the convergence of different methods for training MLP and the variation of the sum rate with the number of users in RL pattern, respectively.

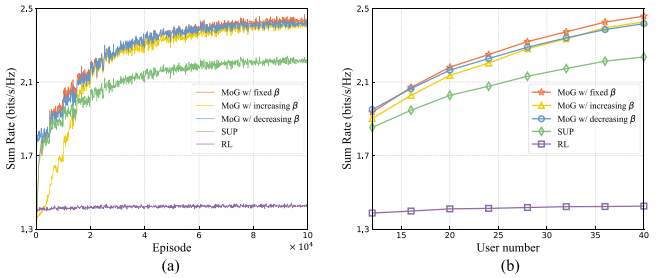

Fig. 9. Performance comparison of GNN in RL pattern. (a) and (b) show the convergence of different methods for training GNN and the variation of the sum rate with the number of users in RL pattern, respectively.

for fixed weights, we set β to 1; for increasing weights, we set the initial value of β to 1 and multiply it by 1.2 for every 5000 episodes, with an upper limit of 2; for decreasing weights, we set the initial value of β to 20 and multiply it by 0.75 for every 5000 episodes. When training the GNN, for fixed weights, we set β to 10; for increasing weights, we set the initial value of β to 1 and multiply it by 1.2 for every 2000 episodes, with an upper limit of 20; for decreasing weights, we set the initial value of β to 100 and multiply it by 0.75 for every 2000 episodes. Fig. 8(a) shows the convergence performance of MLP in RL pattern where $K = 10$. The results indicate that MoGs outperform SUP in terms of final performance. While the final performance of RL method is close to that of MoGs, its convergence rate is significantly slower. In Fig. 8(b), the performance of MLP is evaluated across varying user counts within the RL pattern. MoGs consistently exhibit optimal performance, while SUP is the least effective. Moreover, within the narrower user range represented in the figure, no performance degradation with increasing number of users is observed.

Fig. 9(a) presents the convergence trajectory of GNN in RL pattern where $K = 40$. It is observed that the convergence speeds of MoGs and SUP are comparable. However, the final performance achieved by MoGs is obviously superior to that of SUP. Notably, the RL method fails to converge, indicating its incompatibility with GNN structures for this particular problem setting. In Fig. 9(b), the performance of GNN is evaluated across varying user counts within the RL pattern. Similar to the training results of MLP, the performance metrics of both MoGs and SUP improve with the increase in users, and MoGs consistently outperforms the SUP. Given the RL method's failure to converge in the concerned scenario, it consistently performs poorly. This consistently poor performance

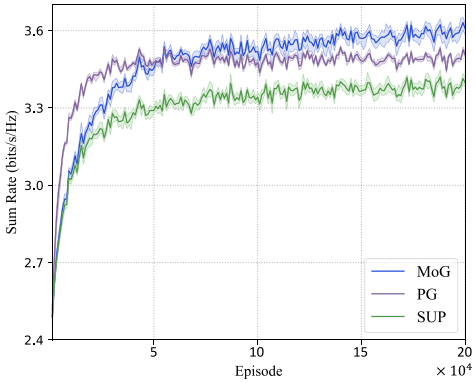

Fig. 10. Convergence comparison based on PG.

reflects the limitations of the RL method when applied to the GNN architecture in this issue.

B. Performance Evaluation of Resource Block Allocation

In the subsequent contents, we conduct a comprehensive evaluation of the proposed MoG with respect to the RB allocation problem. Different from the power control problem discussed before, the optimization target for RB assignment is inherently nondifferentiable, and thus cannot use UNSUP for training. Consequently, we adopt the RL pattern for evaluation. In order to fully validate the effectiveness of the proposed MoG, we employ PG and DPG algorithms as benchmarks, respectively, on which MoG is constructed. Unlike the content presented earlier, this section does not show the outcomes of training for a range of weighting strategies. Instead, it presents the results obtained using the optimal strategy identified. This adjustment is predicated on the observation that, within the context of RB allocation tasks, it is difficult for different weighting strategies to consistently demonstrate a performance advantage under different conditions. Nonetheless, it is important to note that the most effective weighting strategy identified through the comparative analysis significantly outperforms the performance of the benchmark algorithms. The specific settings of the three weight strategies are as follows: 1) for fixed weights, we set β to 1; 2) for increasing weights, we set the initial value of β to 1 and multiply it by 1.5 for every 10^4 episodes, with an upper limit of 10; and 3) for decreasing weights, we set the initial value of β to 10 and multiply it by 0.4 for every 10^4 episodes. Moreover, since the application of GNN architectures in solving the RB allocation problem has not been widely explored and has received limited attention in the research community so far. As a result, we only utilize MLP architecture for addressing this problem, which facilitates a more concentrated investigation into the performance of the proposed MoG scheme in this context.

Figs. 10 and 11 show the training results within PG framework. Specifically, Fig. 10 provides a comparative analysis of the convergence performance of different methods, where the user count is set at 6 and the number of RBs is fixed at 3. The solid line depicted in the figure represents the mean performance achieved through multiple trainings under 8 different random seeds, while the shaded area depicts the range of performance variability. Notably, the MoG demonstrates superior performance compared to the benchmark algorithms.

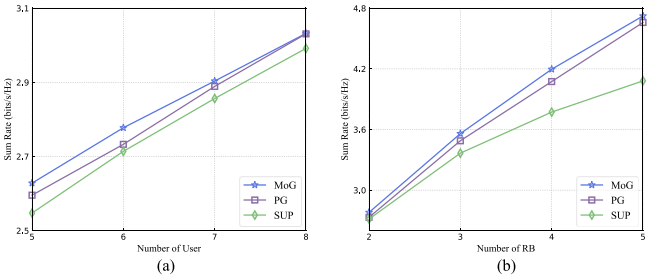

Fig. 11. Performance comparison based on PG. (a) and (b) show the sum rate of MoG based on PG and other methods under different user and RB numbers.

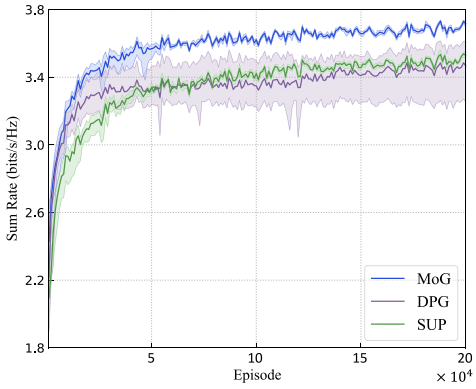

Fig. 12. Convergence comparison based on DPG.

Furthermore, PG method outperforms SUP and exhibits the most rapid convergence speed among the methods evaluated.

Fig. 11 gives a comparative evaluation of the performance across various methods, accounting for the difference in the number of users and RBs. Specifically, Fig. 11(a) sets the RB count at two, and Fig. 11(b) fixes the user number at six. The results indicate that MoG outperforms other methods, with SUP showing the poorest outcomes. Notably, the performance gap between MoG and PG is marginal. This may be attributed to the operational mechanics of PG framework, where the model updates its parameters using a single training sample at each iteration, constraining the batch size to 1. Such a constraint is suboptimal for SUP, which typically requires the selection of an appropriate batch size—often ranging from several dozens to hundreds—customized to the specific scenario in order to optimize the training results.

Figs. 12 and 13 present the training results within DPG framework. Specifically, Fig. 12 provides a comparative analysis of the convergence performance of various methods, where the user count is set at 6 and the number of RBs is fixed at 3. Similarly, the solid line depicted in the figure represents the mean performance achieved through multiple trainings under 8 different random seeds, while the shaded area depicts the range of performance variability. The results demonstrate that MoG exhibits superior performance compared to the benchmark algorithms. Furthermore, the final performance of DPG and SUP are observed to be nearly identical in this scenario. However, it is noteworthy that DPG achieves a more rapid convergence speed. This acceleration, though, is accompanied by an increased amplitude of oscillation, indicating greater variability in the convergence path when compared to the more consistent approach of SUP.

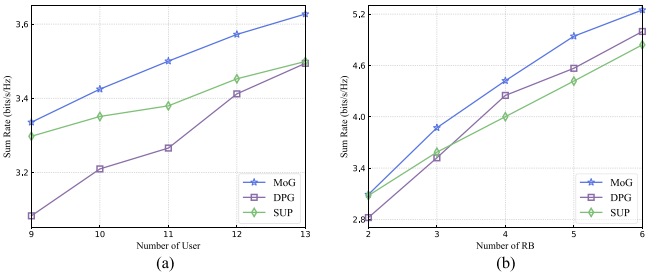

Fig. 13. Performance comparison based on DPG. (a) and (b) show the sum rate of MoG based on DPG and other methods under different user and RB numbers..

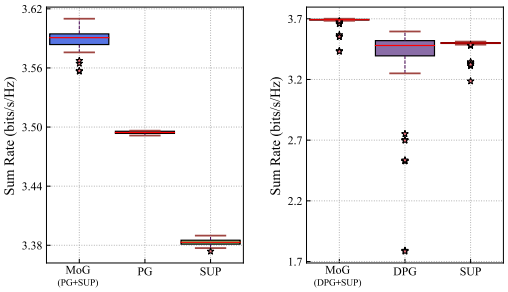

Fig. 14. Comparison of training stability.

Fig. 13 presents a comparison of the performance across various methods, accounting for the difference in the number of users and RBs. Specifically, Fig. 13(a) sets the RB count at 2, while Fig. 13(b) fixes the user number at 7. The results consistently indicate that MoG outperforms the benchmark algorithms. Notably, within DPG framework, MoG's performance superiority is more significant. This performance improvement can reasonably attributed to the training approach employed by DPG framework, which involves the selection of a data batch from the replay buffer for each iteration of parameter updates. This approach effectively exploits the potential performance capabilities of SUP, thereby facilitating the amplification of training efficacy through the application of weighted combinations within MoG paradigm.

To enable a comprehensive evaluation of the proposed method's performance characteristics, we present the following analytical visualizations. Fig. 14 demonstrates experimental outcomes specifically designed to assess the method's robustness and operational consistency across different conditions. To this end, the model is trained independently using one hundred distinct random seeds, and the performance distributions for various methods are recorded. Within PG framework, the performance of MoG constructed by combining PG and SUP is the best, while PG demonstrates superior performance than SUP. Notably, although the performance variance of MoG (as indicated by the amplitude of the oscillations) is slightly larger than other methods, it is still within an acceptable range of no more than 2%. Under the DPG framework, MoG also maintains its position as the most effective among the evaluated methods. The performance gap between DPG and SUP is relatively narrow, while DPG exhibits a notably higher degree of performance fluctuation compared to the other methods. This means that while DPG may achieve performance comparable to SUP, its results are much less stable. In the case of the

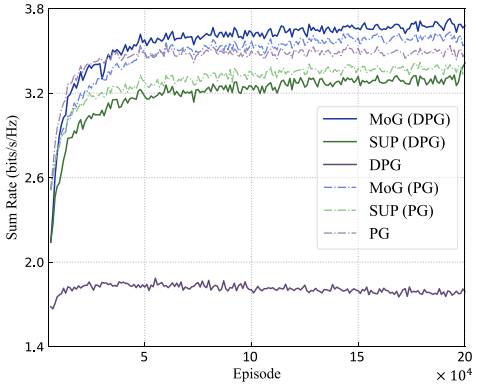

Fig. 15. Convergence performance in extreme cases.

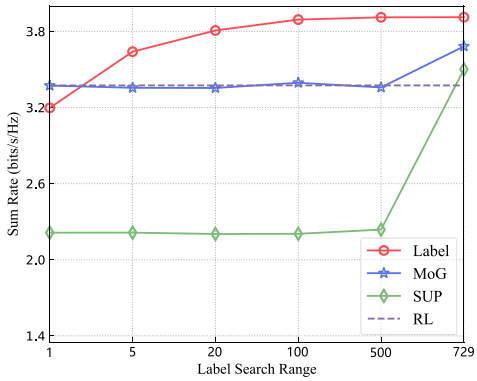

Fig. 16. Influence of label quality on performance.

largest deviation, oscillations of more than 50% are observed, which seriously affects reliability in practical application environments. In addition, it can be seen that the performance of SUP varies greatly under different RL frameworks.

Fig. 15 shows the convergence curves of different methods in an extreme case where the performance of DPG and PG deteriorates significantly. The performance degradation of the RL methods under a specific random seed indicates their instability, which corresponds to the performance lower limit shown in Fig. 14. In this figure, the most inferior performance within PG framework is ascribed to SUP, and within DPG framework, the nadir is represented by RL. It is observed that under the particular random seed, MoG still achieves commendable performance results despite the fact that the performance of the benchmark algorithm drops significantly below average. This is especially notable within the DPG framework, where MoG preserves a high level of performance, even as the performance of RL experiences a pronounced decline. This phenomenon could be attributed to the capacity of MoG to integrate multiple gradient sources, thereby avoiding the convergence to local optima in these extreme cases, which is a challenge faced by benchmark algorithms.

In the above results, we utilize the optimal solutions derived from search algorithm as labels for RB allocation task, which evidently provides a performance guarantee for SUP. A natural question arises: how much does label quality affect the effectiveness of the training? In this investigation, we employ optimal actions obtained from solution spaces of different sizes as labels for training (in the considered scenario involving 6 users and 3 RBs, the complete action space has a total of 729 actions). We graph the performance of different training methods alongside the performance of the label solutions, as depicted in Fig. 16. The value of the x-axis in the figure represents the range of actions for which labels are searched, e.g., an x-axis of 100 means that 100 different actions are randomly selected from the action space, and the optimal action among them is used as the label. It can be seen that in the absence of optimal solutions as labels, the performance of SUP deteriorates significantly, while the performance of MoG remains comparable to RL. Notably, even when the label performance falls below that achieved by RL, it does not substantially degrade the performance of MoG. It is only with the employment of the optimal solution as a label that MoG secures a significant performance advantage over the benchmark methods, and the performance of SUP is also notably enhanced.

This observation shows a limitation of MoG in addressing the RB allocation issue, i.e., the necessity of using the optimal solution as the training label. Conversely, for the previously mentioned power control task, despite the labels being generated by state-of-the-art optimization algorithms, they cannot be assumed to represent the optimal solution. Nevertheless, they still achieve satisfactory training outcomes. The differential impact of label quality on these two tasks deserves further exploration.

V. Conclusion

In this article, we have proposed a novel MoG training paradigm, aiming to improve the performance of learning-based methods in wireless communication systems. MoG integrates supervised and unsupervised gradients within the training process, which can be achieved simply by modifying the loss function. Simulation results in several classical wireless network scenarios confirm that MoG can significantly improve the performance of various NN models, without the expense of additional training data and inference overhead. The proposed MoG provides a flexible and efficient training framework that enhances NN-based methods across diverse wireless optimization problems. For future works, we will extend MoG to more complex wireless communication tasks and more advanced DL architectures.

References

[1] L. Ma, N. Cheng, X. Wang, Z. Yin, H. Zhou, and W. Quan, "Distilling knowledge from resource management algorithms to neural networks: A unified training assistance approach," in *Proc. IEEE 98th Veh. Technol. Conf. (VTC)*, 2023, pp. 1–5.

[2] N. Kato et al., "Optimizing space-air-ground integrated networks by artificial intelligence," *IEEE Wireless Commun.*, vol. 26, no. 4, pp. 140–147, Aug. 2019.

[3] H. Ju and R. Zhang, "Throughput maximization in wireless powered communication networks," *IEEE Trans. Wireless Commun.*, vol. 13, no. 1, pp. 418–428, Jan. 2014.

[4] G. Gui, M. Liu, F. Tang, N. Kato, and F. Adachi, "6G: Opening new horizons for integration of comfort, security, and intelligence," *IEEE Wireless Commun.*, vol. 27, no. 5, pp. 126–132, Oct. 2020.

[5] W. Saad, M. Bennis, and M. Chen, "A vision of 6G wireless systems: Applications, trends, technologies, and open research problems," *IEEE Netw.*, vol. 34, no. 3, pp. 134–142, May/Jun. 2020.

[6] K. B. Letaief, W. Chen, Y. Shi, J. Zhang, and Y.-J. A. Zhang, "The roadmap to 6G: AI empowered wireless networks," *IEEE Commun. Mag.*, vol. 57, no. 8, pp. 84–90, Aug. 2019.
[7] L. Ma et al., "Dynamic neural network-based resource management for mobile edge computing in 6G networks," *IEEE Trans. Cogn. Commun. Netw.*, vol. 10, no. 3, pp. 953–967, Jun. 2024.
[8] X. Wang et al., "RadioDiff: An effective generative diffusion model for sampling-free dynamic radio map construction," *IEEE Trans. Cogn. Commun. Netw.*, vol. 11, no. 2, pp. 738–750, Apr. 2025.
[9] T. Huynh-The, C.-H. Hua, Q.-V. Pham, and D.-S. Kim, "MCNet: An efficient CNN architecture for robust automatic modulation classification," *IEEE Commun. Lett.*, vol. 24, no. 4, pp. 811–815, Apr. 2020.
[10] A. S. M. M. Jameel, A. Malhotra, A. E. Gamal, and S. Hamidi-Rad, "Deep OFDM channel estimation: Capturing frequency recurrence," *IEEE Commun. Lett.*, vol. 28, no. 3, pp. 562–566, Mar. 2024.
[11] Y. Shen, J. Zhang, S. H. Song, and K. B. Letaief, "Graph neural networks for wireless communications: From theory to practice," *IEEE Trans. Wireless Commun.*, vol. 22, no. 5, pp. 3554–3569, May 2023.
[12] X. Ying, "An overview of overfitting and its solutions," *J. Phys., Conf. Ser.*, vol. 1168, no. 2, 2019, Art. no. 22022.
[13] A. Feriani and E. Hossain, "Single and multi-agent deep reinforcement learning for AI-enabled wireless networks: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 2, pp. 1226–1252, 2nd Quart., 2021.
[14] Z.-Q. Luo and W. Yu, "An introduction to convex optimization for communications and signal processing," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1426–1438, Aug. 2006.
[15] Y. Nie, J. Zhao, F. Gao, and F. R. Yu, "Semi-distributed resource management in UAV-aided MEC systems: A multi-agent federated reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 70, no. 12, pp. 13162–13173, Dec. 2021.
[16] Y. Zhang, D. Yu, X. Zhang, and Y. Liu, "An autoregressive model-based differential framework with learnable Regularization for CSI feedback in time-varying massive MIMO systems," *IEEE Commun. Lett.*, vol. 29, no. 1, pp. 230–234, Jan. 2025.
[17] Y. S. Nasir and D. Guo, "Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2239–2250, Oct. 2019.
[18] J. Liu and H. Zhang, "Power allocation in ultra-dense networks through deep deterministic policy gradient," *IEEE Wireless Commun. Lett.*, vol. 11, no. 12, pp. 2502–2506, Dec. 2022.
[19] K.-S. Shin, H.-H. Choi, and H. Lee, "Knowledge transfer-based multiagent Q-learning for medium access in dense cellular networks," *IEEE Wireless Commun. Lett.*, vol. 11, no. 12, pp. 2542–2545, Dec. 2022.
[20] Q. He, A. Moayyedi, G. Dán, G. P. Koudouridis, and P. Tengkvist, "A Meta-learning scheme for adaptive short-term network traffic prediction," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 10, pp. 2271–2283, Oct. 2020.
[21] Y. Wang et al., "Meta-reinforcement learning for reliable communication in THz/VLC wireless VR networks," *IEEE Trans. Wireless Commun.*, vol. 21, no. 9, pp. 7778–7793, Sep. 2022.
[22] Y. Lu and X. Zheng, "6G: A survey on technologies, scenarios, challenges, and the related issues," *J. Ind. Inf. Integr.*, vol. 19, Sep. 2020, Art. no. 100158.
[23] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.
[24] S. Fujimoto and S. S. Gu, "A minimalist approach to offline reinforcement learning," *Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 20132–20145, 2021.
[25] K. Shen and W. Yu, "FPLinQ: A cooperative spectrum sharing strategy for device-to-device communications," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2017, pp. 2323–2327.
[26] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for interference management," *IEEE Trans. Signal Process.*, vol. 66, no. 20, pp. 5438–5453, Oct. 2018.
[27] F. Liang, C. Shen, W. Yu, and F. Wu, "Towards optimal power control via ensembling deep neural networks," *IEEE Trans. Commun.*, vol. 68, no. 3, pp. 1760–1776, Mar. 2020.
[28] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 12, 1999, pp. 1057–1063.
[29] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 387–395.
[30] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[31] D. Lopez-Paz, L. Bottou, B. Schölkopf, and V. Vapnik, "Unifying distillation and privileged information," 2015, *arXiv:1511.03643*.
[32] D. Silver et al., "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.
[33] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2014.
[34] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, vol. 2. Berlin, Germany: Springer, 2009.

Nan Cheng (Senior Member, IEEE) received the B.E. and M.S. degrees from the Department of Electronics and Information Engineering, Tongji University, Shanghai, China, in 2009 and 2012, respectively, and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2016.

He was a Postdoctoral Fellow with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, Canada, from 2017 to 2019. He is currently a Professor with the School of Electrical Engineering and Intelligentization, Dongguan University of Technology, Dongguan, China, and also with the State Key Laboratory of ISN and School of Telecommunications Engineering, Xidian University, Xi'an, China. He has published over 90 journal papers in IEEE Transactions and other top journals. His current research focuses on B5G/6G, AI-driven future networks, and space–air–ground integrated network.

Prof. Cheng serves as an Associate Editor for IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE OPEN JOURNAL OF VEHICULAR TECHNOLOGY, and *Peer-to-Peer Networking and Applications*, and serves/served as the Guest Editors for several journals.

Longfei Ma (Student Member, IEEE) received the B.S. degree in telecommunication engineering from Xidian University, Xi'an, China, in 2022, where he is currently pursuing the M.S. degree.

His research interests include wireless communications and networking.

Yanpeng Dai (Member, IEEE) received the B.Eng. degree in telecommunication engineering from Shandong Normal University, Jinan, China, in 2014, and the Ph.D. degree in communication and information systems from Xidian University, Xi'an, China, in 2020.

He is currently an Associate Professor with the School of Information Science and Technology, Dalian Maritime University, Dalian, China. He was a visiting student with the University of Waterloo, Waterloo, ON, Canada. His research interests include resource management and interference coordination for heterogeneous wireless networks and maritime communication systems.

Xiucheng Wang (Graduate Student Member, IEEE) received the B.S. degree in telecommunication engineering from Xidian University, Xi'an, China, in 2021, where he is currently pursuing the Ph.D. degree.

His research area of interest is digital twin and graph neural networks of the wireless network.

Qihao Li (Member, IEEE) received the M.Sc. degree in information and communication technology from the University of Agder, Kristiansand, Norway, in 2014, and the Ph.D. degree from the University of Oslo, Oslo, Norway, in 2019.

After that, he was a Postdoctoral Fellow with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, from 2019 to 2021, and a Lecturer with the School of Electrical Engineering and Intelligentization, Dongguan University of Technology, Dongguan, China, from 2022 to 2023. Since 2023, he has been an Associate Professor with Jilin University, Changchun, China. His current research interests include Industrial IoT, intelligent communication and networking, optimal control and optimization, and green communications.

Dr. Li was the Workshop TPC Chair for IEEE/CIC ICCC' 23. He served as a member for Technical Program Committee for IEEE Globecom' 17-23, IEEE ICC' 18-23, and IEEE CIC ICCC' 17-23.

Wei Quan (Senior Member, IEEE) received the Ph.D. degree in communication and information system from Beijing University of Posts and Telecommunications, Beijing, China, in 2014.

He is currently a Full Professor with the School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing. He has co-authored more than 50 papers in prestigious international journals and conferences, including IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, *IEEE Communications Magazine*, IEEE WIRELESS COMMUNICATIONS, IEEE NETWORK, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, and IEEE COMMUNICATIONS LETTERS. His research interests focus on reliable transmission in mobile networks, vehicular networks, and Industrial IoT.

Prof. Quan is a Winner of the 2022 IEEE ComSoc Asia–Pacific Outstanding Young Researcher Award and the Principle Investigator of National Key Research and Development Program of China. He is an Associate Editor for the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, *Peer-to-Peer Networking and Applications*, *Journal of Internet Technology*, and IEEE ACCESS.

Hui Liang (Member, IEEE) received the Ph.D. degree from Jilin University, Changchun, China, in 2011.

He is currently an Associate Professor with the School of Electrical Engineering and Intelligentization, Dongguan University of Technology, Dongguan, Guangdong, China. His research interests include cloud native networks, cybertwin-enabled edge computing systems, and resource allocation in space–air–ground communications systems.

Xuemin (Sherman) Shen (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990.

He is a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His research focuses on network resource management, wireless network security, Internet of Things, AI for networks, and vehicular networks.

Dr. Shen is a Registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, a Chinese Academy of Engineering Foreign Member, an International Fellow of the Engineering Academy of Japan, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society.