Learning-Based Resource Allocation in Heterogeneous Ultra Dense Network

Xiangyu Zhang, *Graduate Student Member, IEEE,* Zhengming Zhang, *Graduate Student Member, IEEE,* and Luxi Yang , *Senior Member, IEEE*

***Abstract*—Learning-based Resource Allocation (LRA) is envisioned as an integral element of 6G. This paper proposes a novel learning-based paradigm to address resource allocation problems in Heterogeneous Ultra-Dense Networks (HUDNs). Our paradigm is a highly efficient realization for utilizing the inherence properties in HUDN, which comprise the local validity and correlation attenuation. Concretely, we formulate the HUDNs as a heterogeneous bipartite graph model and propose the corresponding Heterogeneous Bipartite Graph Neural Network (HBGNN). The local sampling characteristic of HBGNN matches the inherence properties. Meanwhile, our paradigm combines data-driven and model-driven learning and employs online and offline training. Hence, two of LRA's obstacles, the over-reliance on the perfect data set and the low calculation efficiency, are mitigated, and the realizability of our paradigm are improved. Besides, entropy regularization is utilized to guarantee the effectiveness of exploration in the configuration space. We apply our approach to a representative resource allocation problem, the Jointly User Association and Power Allocation (JUAPA) problem. We formulate JUAPA as a combination classification and regression problem and adopt a dynamical hyperparameter output layer to address the discrete variable of user association. Simulation results demonstrate that the proposed method has better performance and higher computational efficiency than traditional optimization algorithms.**

***Index Terms*—Resource allocation, heterogeneous ultra dense network, graph neural network, Deep Learning.**

I. Introduction

With the progress of Internet of Everything, the next-generation mobile communication systems are expected to provide higher transmitting rates, higher connectivity density, and ultra-low latency communication. To meet these needs, extensive flexible deployment of different kinds of small base stations (BSs) to enable more connections, namely, Heterogeneous Ultra-Dense networks (HUDNs), has been envisioned as one of the key technologies in 5G and 6G [1] [2]. Generally, HUDN is an evolving architecture from the legacy of dense networks [3], but it faces more challenges in radio resource management. The wireless radio resource in the time, frequency, and spatial domain is more flexible and complex because it enables some novel technologies of HUDN, such as mmWave [4], the massive MIMO [5], cell-less, user-centric [6]. Some prior works [7]–[11] demonstrated that effectively allocating the resources could significantly suppress the interference and improve performance in terms of throughput capacity, energy efficiency, load balancing, etc.

The researchers investigated resource allocation problems, such as user association and power control, using theory-driven methods that resort to mathematical models and techniques, such as convex theory, graph theory, and game theory, to obtain exact solutions [12]–[15]. However, these methods face three critical challenges in the HUDN scenario. The first challenge arises from the complexity. The resource allocation problems always contain the multi-correlative variables, which corresponds to a complex combinatorial optimization problem [16]. A second factor is that the high density of wireless networks significantly expands the problem's dimension and poses computational challenges. As a final point, these methods require precise and analytically tractable models, such as the channel model, antenna model, and local environment model, and closed-form expressions between the manageable resource and the optimization criteria, which is difficult to implement. [17].

Inspired by the recent successes of Deep Learning (DL), some works have attempted to apply deep learning algorithms in resource allocation, which is named "Learning to Resource Allocate" (LRA) [18]–[21]. The basis of LRA is to train a neural network to approximate the relationship between the ensemble of all network parameters and optimal system parameters [22], which is an "end-to-end" approach without the need for problem modeling, problem simplification, or algorithm design. Thus LRA avoid the above mentioned problem, but continues to face several challenges, which is the reliance on a high-quality data set and are computationally intensive.

Solving the aforementioned challenges is indeed the motivation of this work. Specifically, we propose a novel learning-based paradigm to address resource allocation problems in HUDN to achieve better performance in terms of communication quality, robustness, generalization, and calculation efficiency. We apply our approach in a representative resource allocation problem, which is the Jointly User Association and Power Allocation (JUAPA) problem to maximize the global transmitting rate. The JUAPA is viewed as two typical machine learning problems, one for classification problem (User Association) and another for prediction problem(Power Allocation). Following the LRA, we train a neural network to solve these two tasks, which maps the relationship between the

This work was supported by the National Natural Science Foundation of China under Grants 61971128 and U1936201, and the National Key Research and Development Program of China under Grant 2020YFB1804901.

X. Zhang, Z. Zhang, and L. Yang are with the National Mobile Communications Research Laboratory, School of Information Science and Engineering, Southeast University, Nanjing 210096, China, and also with the Pervasive Communication Research Center of Purple Mountain Laboratories, Nanjing 211111, China (e-mail: xyzhang@seu.edu.cn; zmzhang@seu.edu.cn; lxyang@seu.edu.cn).

channel and optimal configurations(user association matrix, the power of BS).

To reduce the reliance of the LRA on the label, we train the neural network using a combination of data-driven and model-driven methods, referred to as supervised and unsupervised learning. Supervised learning train the neural network with some optimization solutions. Thus, the performance of supervised algorithms is definitely limited by the label. Therefore, we leverage a label-independent algorithm, i.e., the unsupervised learning algorithm. The unsupervised learning utilizes the precise communication model and optimizes the JUAPA by gradient descent [22], which is a model-driven algorithm. We also introduce HUDN's character to alleviate the difficulty that LRA faces. We first demonstrate the local validity and correlation attenuation property that HUDN inheres and then build the Neural Network's architecture following these properties. We model the HUDN as a Heterogeneous Bipartite Graph Model (HBGM) and propose a corresponding neural network , the Heterogeneous Bipartite Graph Neural Network (HBGNN). The HBGNN is designed to extract the features of neighboring nodes, which implies that the architecture of neural network removes the irrelevant information from the beginning, thereby reducing the workload and data requirement. Besides, we define two learning stages, the Upstream Learning Stage (ULS) and the Downstream Learning Stage (DLS) for improving the algorithm's calculation efficiency and performance in the real-time scenario.

A. Related Works

As for the theory-driven methods, the convex theory, graph theory, and game theory are leveraged to solve this problem. The paper [12] jointly considered the user association, power control and beamforming in Heterogeneous Networks and solved the problem with the distributed pricing update strategy. In [13], user association and power allocation in mm-wave-based UDNs were considered as a mixed-integer programming problem. Through relaxing the integer variable into continuous variables, the problem was solved by Lagrangian Dual Decomposition. A joint power allocation and user association strategy in HUDNs using non-cooperative game theory were developed in [14]. The proposed game was divided into two subgames, the Backhaul Game, and the Access Game. The Backhaul Game was implemented between BS and relay nodes (RNs) in the backhaul links, and the Access Game was implemented between the BS/RNs and UEs in the access links.

The existing data-driven method can be broadly summarized into two categories, the deep reinforcement learning-based method, and the unsupervised learning-based method. The difference between them is the way that updates the parameter of neural network . The reinforcement learning model the user association and power allocation problem as a Markov Decision Problem and finds the desired solution by repeatedly attempting the desired solution without environment information requirements [21], [23]–[25]. However, the practical application of reinforcement-learning-based methods is limited by the sample inefficient [26]. The unsupervised learning-based method was firstly proposed by [27]. This paper identifies a class of optimization algorithms that can be accurately approximated by a fully connected neural network and demonstrate it by utilizing the fully connected neural network to approximate a popular interference management algorithm, namely, the WMMSE algorithm [28]. Following the same line, the paper [29] utilized a convolutional neural network to allocate power. Besides, the same problem was investigated by paper [19] with multi constraints. The paper [30]–[32] proposed different graph neural networks to approach this problem.

Although plenty of works concentrates on the work of LRA, this field still has many challenges. The first thing is that most of the works still concentrated on the optimization problem with a single continuous variable, i.e., power. For the discrete variable, i.e., user association matrix, it could fail to converge due to the uncontinuity of the objective function. Meanwhile, the objective function with the multi-correlative variable exists abundant local optimal points. Finding the designed solution in local optimal points would be a challenge, not only in LRA but also in DL. Besides, the efficient learning algorithm and generalization promotion algorithm is urgently needed to cooperate with the complicated and changeable wireless communication system.

B. Contributions

The main contributions of this paper are as follows:

- We propose a novel learning-based paradigm to address resource allocation problems in HUDN. This paradigm addresses the LRA's bottlenecks, which are highly dependent on a high-quality data set and computationally demanding.
- We model HUDN as a Heterogeneous Bipartite Graph Model and theoretically demonstrate the graph's intrinsic properties, namely local validity and correlation attenuation. Following these characteristics, we construct a Heterogeneous Bipartite Graph Neural Network. The proposed neural network architecture enjoys the scalable character and locally computed features suited for the dynamic user and large-scale user scenarios in HUDN.
- We solve the resource allocation problem in the HUDN by combining two DL learning technologies, supervised and unsupervised learning. We demonstrate that the combination of supervised and unsupervised learning outperforms the supervised label in terms of performance and data efficiency. Besides, we add the exploration item, the entropy of user association output, in the loss function to guarantee the effectiveness of exploration in the learning stage.
- To improve the performance in real-time scenario, we separate the training process into two stages, the ULS and DLS. The ULS learns generalized mapping with existing data, an offline training. The DLS is an online training, that fine-turn the neural network. The DLS aims to further improve the global transmitting rate in a real-time scenario. Combining online training and off-line training reduces the amount of time required online.
- We apply our algorithm to the Jointly User Association and Power Allocation problem. To cooperate with

the discrete variable of user association, we adopt the dynamical hyper-parameter output layer and derive the optimal parameter for stable convergence and minimum error brought by the relaxed user association indicator. Numerical results are provided to show the effectiveness of our algorithm.

C. Organization of This Paper

The rest of the paper is structured as follows. Section II introduces the system model of HUDN, the channel model, and formulates the Jointly User Association and Power Allocation as an optimization problem. In Section III, we establish a graph model for heterogeneous UDN and use a graph neural network to solve the joint optimization problem. Section V analyzes the performance of our proposed algorithm through simulation. Finally, we conclude that our methods and outlook the future research direction at the end of the paper.

II. System Model And Problem Formulation

This section presents the basic information of the scenario, the graph model, and the communication model. We then formulate the joint user association and power control as an optimization problem.

A. Scenario

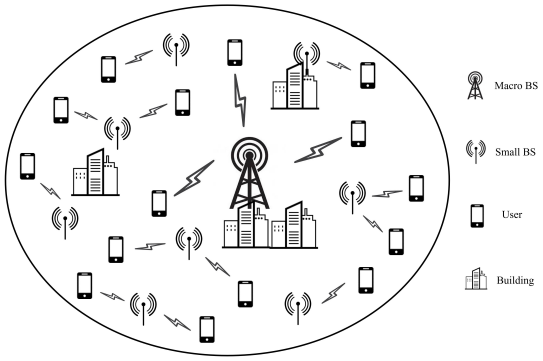


Fig. 1: The architecture of a heterogeneous ultra-dense network.

As shown in Fig.1, we investigate resource allocation problems in the downlink of the HUDNs scenario, where ultra-dense small BSs and Marco BSs cooperatively service UEs in the converging area. Let $\mathcal{J} = \{1, \ldots, J_m, \cdots, J\}$ represent the set of BS stations, where $J = J_m + J_n$, the first J_m items denote the Marco BSs, and the last of J_n items are the small BS. We assume that the small BSs and macro BSs interfere with each other for sharing the same frequency band, and the UEs connected to the same BS can avoid interference by precoding. We denote the single-antenna UEs as $\mathcal{I} = \{1, \ldots, I\}$.

We use the binary integer variable $x_{i,j}$ to indicate the association variable between UE i and BS j. If UE i is associated with BS j, $x_{i,j} = 1$, otherwise $x_{i,j} = 0$. Let p_j be the transmit power from BS j and $g_{i,j}$ be the power gain from BS j to UE i. The SINR of UE i receiving from BS j can be written as

$$\mu_{i,j} = \frac{p_j g_{i,j}}{\sum_{n \in \mathcal{J}} p_n g_{i,n} + \sigma^2}, \tag{1}$$

where σ^2 is the variance of Additive White Gaussian Noise (AWGN). Then the global transmitting rate for UE i from BS j can be calculated by $\gamma_{i,j} = (B/N_j) \log_2 \left(1 + \mu_{i,j}\right)$, where B is the system bandwidth, N_j is the connected user of BS j.

B. Channel Model

Traditionally, the channel gain $g_{i,j}$ can be written as $g_{i,j} = \beta_m(w) G_m(w) \tilde{h}_m(t)$, where $\beta_m(w)$ denote the large-scale channel gain, which generally depends on the distance between UE and BS. And $G_m(w)$ and $\tilde{h}_m(t)$ are the BS antenna gain and the small-scale fading, respectively. We only focus on the configurable parameter, whose adjustment period is often counted by minutes or hours. Thus, the statistics channel information is more meaningful than instantaneous information. Our simulation utilizes the large-scale channel gain to represent the statistics channel information. The detail is shown in Section V.

C. Graph Model

The graph is a ubiquitous non-Euclidean data structure that extensively exists in pharmacy, chemistry, and related fields. A graph $G = (V, E)$ comprises a set of nodes V and a set of edges E that connects two nodes and describes the relationship between different nodes. Here, we model the HUDNs as a heterogeneous bipartite graph $G_h = \{V, E\}$. As shown in Fig.2(a), graph G contains two different kinds of nodes representing the BS and UE, respectively. The edges only exist between the BS and UE, representing the UE whether connected to or interfered by the BS. In other words, the received power from BS to UE with edge connected is higher than the noise. Thus, equation (1) can be rewritten as

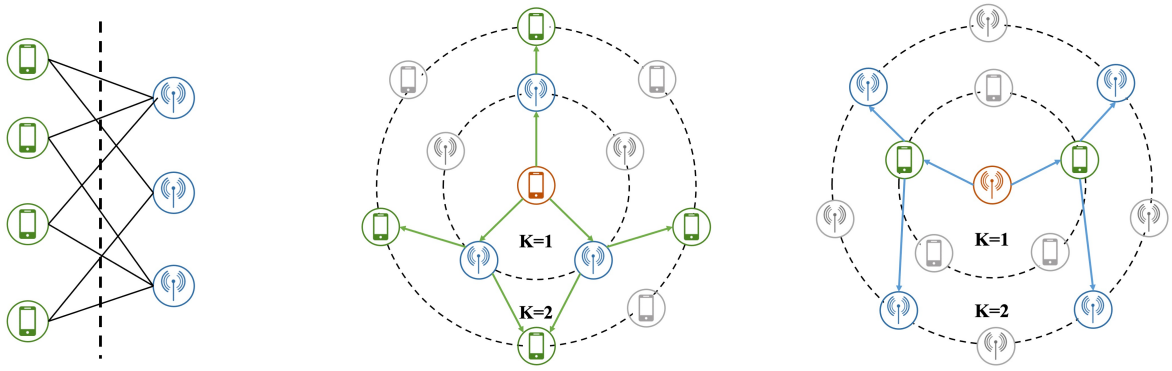
$$\mu_{i,j} = \frac{p_j g_{i,j}}{\sum_{n \in \mathcal{N}_2(j)} p_n g_{i,n} + \sigma^2}, \tag{2}$$

Property 1: In the heterogeneous bipartite graph, the same order neighbors of any node are isomorphic.

It is clear that the 1st-order neighbors are BS nodes shown in Fig.2(b) for $K = 1$, the 2nd-order neighborhood of UE is the union set of UE that connect to the one order BS, shown in Fig.2(b) for $K = 2$. The corresponding neighborhood of BS is shown in Fig.2(c). We denotes the $\mathcal{N}_k(i)$ as the kth-order neighbor set for node i.

D. Problem Formulation

In this section, we formulate the JUAPA as an optimization problem. Before modeling the problem, we give the following constraints: (1) User scheduling constraint: A user can only be associated with one BS at a time. (2) Maximum power constraint: The maximum transmit power of each BS is p_{max}. Our objective is to maximize the global transmitting rate



(a) The bipartite graph model (b) The incidence relation of UE (c) The incidence relation of BS

Fig. 2: Heterogeneous graph and heterogeneous graph neural network of HUDNs.

through the joint optimization of power allocation and user association. The optimization objective function is

$$\begin{aligned} \arg\max_{\mathbf{x},\mathbf{p}} R &= \mathrm{Tr}(\mathbf{x}^{\mathrm{H}}\mathbf{R}) \\ \text{s.t. C1} &: \sum_{j\in\mathcal{J}} x_{i,j} = 1, \quad \forall i \in \mathcal{I} \\ \text{C2} &: 0 \le p_j \le p_{max}, \quad \forall j \in \mathcal{J}, \end{aligned} \tag{3}$$

where the $\mathbf{x} = [x_{i,j}] \in \mathbb{Z}^{I\times J}$ is the association matrix and $\mathbf{p} = [p_j] \in \mathbb{R}^J$ is the power vector. The $\mathbf{R} = [\gamma_{i,j}] \in \mathbb{R}^{I\times J}$ is the transmitting rate matrix and the R is the global transmitting rate.

When embedding the JUAPA as a learning problem on the graph, the UA problem is to select the appropriate AP from UE's first-order neighbor, which can be treated as an edge classification problem. In contrast, the PA problem is to predict the power of BSs, which is a node feature classification problem. The two problems are typically solved by supervised learning or semi-supervised learning. In the LRA, we can leverage the communication model to learn more useful information, which can exceed the performance of labels. Because the two problems are highly coupled with each other, we build a single HBGNN to approximate the optimal representation function, which jointly solves JUAPA once a time. We will introduce the learning algorithm in Section III.

III. Propose Method

In this section, we present the LRA, which is the basic theory of our algorithm, with the example of JUAPA. After that, we emphatically discuss the key motivation of modeling the HUDN as HNG and the incentive of our formulated HBGNN for our problem (3). Besides, the effective implementation of our algorithm that considers the real communication scenario will also be discussed.

A. Optimizing Resource Allocation Problem in Machine Learning View

The resource allocation problem has a similar form. In this section, we take the JUAPA as an example to explain the resource allocation problem in LRA. The normal solution for the problem (3) is to find a mapping

$$\mathbf{x}^*, \mathbf{p}^* = f^*(\mathbf{G}), \tag{4}$$

where the $\mathbf{G} = [g_{i,j}]$ is the channel matrix, the $\mathbf{x}^*, \mathbf{p}^*$ represents the optimal solution for problem (3), respectively, and the f represents the corresponding optimal mapping.

In LRA, the optimal mapping for f^* is approximated by a neural network, which is

$$\mathbf{x}^*, \mathbf{p}^* = \pi(\mathbf{G}, \theta^*), \tag{5}$$

where π represents the function between the input and output of neural network. The θ^* is the optimal parameter of the neural network. The parameters of neural network are updated to maximum the following metric,

$$R^* = \mathrm{F}(\mathbf{x}^*, \mathbf{p}^*, \mathbf{G}) = \mathrm{F}(\pi(\mathbf{G}, \theta^*), \mathbf{G}), \tag{6}$$

where F represents the Lagrange function of resource allocation problem.

Next, we introduce the supervised and unsupervised learning in LRA using the JUAPA as an example. The problem (3) contains two tasks. The User Association (UA) problem can be viewed as a typical classification task, and the Power Allocation (PA) problem is a regression task. The supervised and unsupervised methods are two traditional ways to obtain the π^*.

The supervised method utilizes some example (a data set obtained from past configure data) constituted by G (the input data) and $\mathbf{x}^*, \mathbf{p}^*$ (the optimal label) to let the Graph Neural Network fit the optimal mapping. The loss for the supervised method is

$$L_s = -\sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}} y_{ij} \ln x_{ij} + MSE(\hat{\mathbf{P}}, \mathbf{P}), \tag{7}$$

where $y_{i,j}$ is the label for $x_{i,j}$, which is the solution of optimization-based method. $y_{i,j} = 1$ when the UE j connected to BS i, otherwise, $y_{i,j} = 0$. MSE is the operation of mean squad error for power $\mathbf{P}$.

However, the supervised learning method may not perform well in LRA. The main reason broadly has three aspects. The

first is the label, i.e. $\mathbf{x}^*, \mathbf{p}^*$ could be sub-optimal. Thus, supervised learning may never converge to the global optimal. The second is the overfitting or underfitting caused by improper hyperparameter make the neural network unable to converge in θ^*. The last thing is that the examples may be sampled from a small number of specific situations, which is biased performing well in other situations. These limitations mainly result from the imperfect data set, which is hard to correct by the supervised learning method itself. To eliminate the negative effects, we identify the update direction of θ by utilizing the gradient descent, which is the unsupervised manner. Prior to implementing unsupervised learning, we relax the $x_{i,j}$ into continuous variables in the first place because the objective function R is indifferentiable with respect to $\mathbf{x}$ for the $\mathbf{x} \in \mathbb{Z}$ and the constrain C1 changes to $0 \leq x_{i,j} \leq 1, \sum_{j \in \mathcal{J}} x_{i,j} \leq 1, \quad \forall i \in \mathcal{I}$.

Theorem 1: The relaxed problem is equivalent to (3).

Proof: The derivative of R to $x_{i,j}$ is

$$\frac{\partial R}{\partial x_{i,j}} = \gamma_{i,j} + \sum_{n=1}^{J} x_{i,n} B \frac{1}{(1+\mu_{i,n}) \ln 2} \frac{\partial \mu_{i,n}}{\partial x_{i,j}} > 0. \quad (8)$$

The first part $\gamma_{i,j} \geq 0$ and the second part is

$$\frac{\partial \mu_{i,n}}{\partial x_{i,j}} = \frac{x_{i,n} p_n g_{i,n} p_j g_{i,j}}{\left(\sum_{n \in \mathcal{J}} (1 - x_{i,n}) p_n g_{i,n} + \sigma^2\right)^2}. \quad (9)$$

The equation indicates that the objective is monotone increasing in the feasible area of $x_{i,j}$. Thus, the optimal solution is either in $x_{i,j} = 1$ or $x_{i,j} = 0$, which completes the proof.

Combining with the equation (6), we can apply the gradient ascend to obtain the θ^*, which can be represented by: $\theta \leftarrow \theta + \partial R / \partial \theta$. Then, we can apply the unsupervised learning by utilizing the loss function $L_u = -R$. The total loss for LRA is $L = L_s + L_u$.

Theorem 2: (The feasibility of HBGNN) The solution of the combination of supervised and unsupervised learning is at least a feasible solution.

Proof: In our method, the constrain condition is stratified by the output layer of the neural network, which is shown in Section III.B. Thus, the optimal solution only exists when $\nabla_{x_{i,j}} R = 0$ and $\nabla_p R = 0$. In the same way, the training processing is converged when $\nabla_\theta R \to 0$. Thus the solution of HBGNN achieves the saddle point and satisfies the KKT condition, which completes the proof.

B. Key Properties of Optimal Mapping Function

Most of the existing works of LRA rely on the powerful approximating ability of neural networks. However, this way sets higher demands on the data set and computing devices and shows low tolerance of the computer efficiency, robustness, and generalization. For further improving the performance, a valid method is to model a neural network architecture following the intrinsic property of f^*. In this section, we investigate the relationship between $g_{i,j}$ and $x_{i,j}$, p_m based on the optimal mapping function's. We first define the Correlation Coefficient as the indicator to measure the importance of $g_{i,j}$ with respect to $x_{i,j}$, p_m.

Definition 1 : (Correlation Coefficient) The correlation coefficient ξ is denoted by the partial derivative of these two variables in the function f^*. For example, the correlation coefficient $\xi(p_m, g_{i,j}) = |\frac{\partial p_m}{\partial g_{i,j}}|$ represent the relation between the channel information $g_{i,j}$ and the power of the mth BS p_m in equation (4). Any change of $g_{i,j}$ will have no impact on p_m, when the $\xi(p_j, g_{i,j}) \to 0$.

Theorem 3: (Correlation Attenuation Property) The correlation coefficient decreases monotonically to the shortest path between the two nodes in HBG.

Theorem 4: (Local Validity Property) In our model problem (3), the $\xi(\mathcal{X}, g_{i,j}) \to 0$, when the node that $\mathcal{X}$ located is the 5th or higher-order neighbors. In another word, solution of the mapping function f^* only needs no more than the 4th-order neighbors' channel information.

The proof of Theorem 3 and 4 is shown in Appendix A. Theorems 3 and 4 indicate that the inherence properties of the mapping function f^* have. The GNN is the only neural network respecting these properties [33], which is the primary motivation for us to adopt GNNs to solve radio resource management problems.

C. Heterogeneous Bipartite Graph Neural Network

In this subsection, we formulate the HBGNN for the HUDN. As discussed above, the function f^* embraces the *local validity property* and *correlation coefficient attenuation property*. Following these, we propose a specific HBGNN for HUDN.

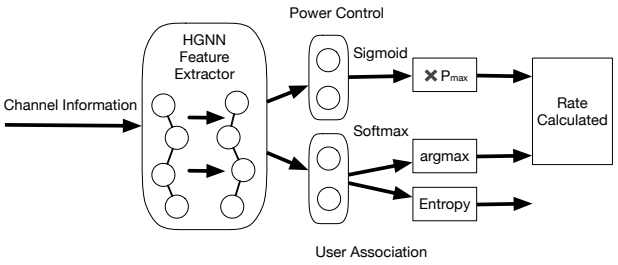


Fig. 3: The HBGNN architecture of representation functions.

As shown in Fig.3, the central part of HBGNN contains two components, the feature extractor and the output layer. The feature extractor receives the channel information and extracts the valuable information for the output layer. The output layer converts the feature to the corresponding resource. As for the JUAPA problem, we have two output layers, which correspond to the power $\mathbf{p}$ and association matrix $\mathbf{x}$. The two resources, UA and PA, share one feature extractor because the UA and PA problems are coupled, and the two problems receive the same input feature G.

1) Heterogeneous Bipartite Graph Neural Network Layer: Typically, the GNN utilizes the message passing architecture [34]. The message passing architecture in each layer contains two operations, the aggregating and the combining. The aggregating operation collects the feature from the neighbor nodes and encapsulates them. After aggregating, the combining operation applies a nonlinear transformation to the concatenation of encapsulating and the nodes' own features to get the next layer feature. The whole process can be represented by:

- **Aggregating Operation**:
  $a_v^{(k)} = \text{AGGREGATE}^{(k)}\left(\left\{h_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)$
- **Combing Operation**:
  $h_v^{(k)} = \text{COMBINE}^{(k)}\left(h_v^{(k-1)}, a_v^{(k)}\right)$

where k indicates the kth layer of GNN. The AGGREGATE and the COMBINE in this equation represent the operations of the GNN, which are realized by two multi-layer fully connected neural networks, respectively [35].

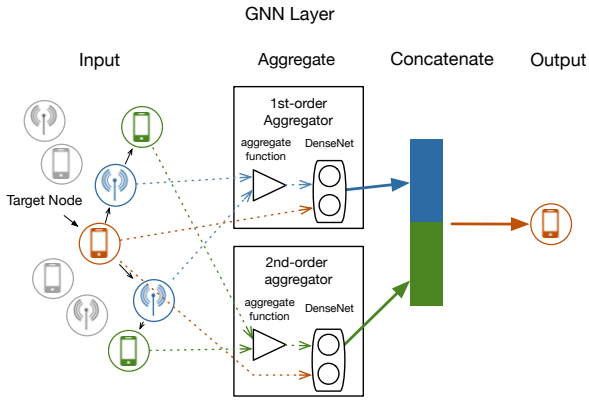


Fig. 4: The heterogeneous bipartite graph neural networks for HUDNs.

Directly applying the GNN into the HBGM is infeasible because the features for heterogeneous nodes are different. However, the bipartite graph embraces Property 1. By utilizing this characteristic, we employ two different order aggregating kernels: one-order and second-order aggregator, to extract the features from one-order and second-order neighborhoods, respectively. We call it Heterogeneous Bipartite Graph Neural Networks (HBGNN).

Fig.4 shows our HBGNN layer on the heterogeneous bipartite graph G. In the first step, the two message-passing kernels respectively sample features of the 1st and 2nd-order targets' neighbors and aggregate these features by the aggregating function. To accommodate the dynamic user number situation, we utilize the sum function as the aggregating function. Next, the aggregated features and the target node's own feature generate the following layers' features with a combining function. We utilize a fully connected neural network as the combining function. The output of the HBGNN layer is the concatenation of both order features.

The summarization of the HBGNN layer is shown in Algorithm 1. In algorithm 1, step 5 is the message passing process for 1st-order neighbors and 2nd-order neighbors. The two message-passing kernels aggregate the corresponding feature and encode the feature with the two multi-layer fully connected neural networks, which is represented by $\mathfrak{L}^{(1)}$ and $\mathfrak{L}^{(2)}$. The $[\mathbf{x}, \mathbf{y}]$ denotes the concatenation of two vector $\mathbf{x}, \mathbf{y}$.

2) The HBGNN and Feature Extractor: Applying HBGNN to radio resource allocation problems still has some obstruction. The HBGNN can only handle the graph information on the node. However, the channel information G is located at the edge of the graph and the association matrix $\mathbf{x}$, the power

Algorithm 1 HBGNN

1: **Input**: Heterogeneous Graph G; input feature $\mathbf{x}_v$; layer num K; one order neighborhood set of node $v_1 \in \mathcal{N}_1(v)$; second order neighborhood set of node $v_2 \in \mathcal{N}_2(v)$; aggregate function $\mathcal{F}$; activation function σ.
2: Randomly initialize the weight of one order aggregator and second order aggregator $\mathbf{W_1}$ $\mathbf{W_2}$ and sample a set of nodes $\mathbf{v}$ from the graph; $\mathbf{h}_v^0 \leftarrow \mathbf{x}_v$
3: **For** k = 1 to K **do**:
4: **For** v = $1 \in V$ **do**:
5: Calculation the feature for i = $1, 2$ ord neighbor
 $\boldsymbol{f_i}_{\mathcal{N}_k(v)}^k \leftarrow \sum_{u \in \mathcal{N}_1(v)} \mathbf{h}_u^{k-1}$
 $\boldsymbol{f_i}_v^k \leftarrow \mathfrak{L}^{(i)}\left(\left[\mathbf{h}_v^{k-1}, \boldsymbol{f_i}_{\mathcal{N}_k(v)}^k\right]\right)$
 $\mathbf{h}_v^k \leftarrow \left[\boldsymbol{f_1}_v^k, \boldsymbol{f_2}_v^k\right]$
6: **End For**
7: $\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \left\|\mathbf{h}_v^k\right\|_2, \forall v \in \mathcal{V}$
8: **End For**

$\mathbf{p}$ is the output of nodes. To coordinate with this, we define the input feature of the node as the collection of channel information with edge connect. A single layer HBGNN can only aggregate 1st-order and 2nd-order neighbors' information. Following Theorem 2, the feature extractor stacks two HBGNN layers to aggregate up to the 4th order neighbor information.

Remark 1: The two-layer HBGNN model naturally respects the correlation coefficient attenuation property through the natural importance resample.

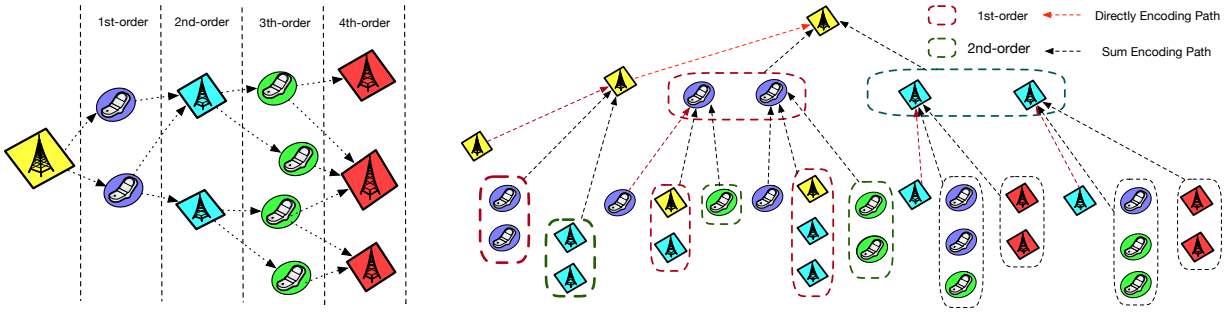
The HBGNN model repetitively samples the nodes' information and passes this information to the outputs in various forms. The closer neighbor gets more chances to be sampled and the more forms to be delivered. Fig.5 shows an illustration of the feature extractor. The upper purple node's information is sampled three times and passed to the output feature in 3 different ways. However, the first red node's information is sampled twice and has only one form to deliver.

Remark 2: The HBGNN allows for adjusting the trade-off between approximate accuracy and convergence speed. The neural network's feature flows are determined by the adjacent matrix of the bipartite graph. The more edges, the more information is provided for output decision-making, but at the expense of extra calculation.

3) The output layer of HBGNN: The output layers of PA utilize the sigmoid output function to generate the normalized power $\mathbf{p}$ of BSs. Different from the PA problem, the output of UA are discrete variables. The normal way to deal with this is to generate the user association configuration with softmax output function, which can be shown as:

$$\hat{x}_{i,j} = \frac{\exp\left(z_{i,j}\right)}{\sum_{n=1}^{J} \exp\left(z_{i,n}\right)} \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \tag{10}$$

where the $z_{i,j}$ is the jth output of ith UE node and the $\hat{x}_{i,j}$ is the output of softmax layer. The softmax operation treats the variable $x_{i,j}$ as a stochastic variable, whose value indicates the probability of associating with the corresponding BS. Normally, the UE associate with the BS with the highest



(a) The corresponding HUDN model with 4th-order neighbor

(b) The tree map of the node feature's composition

Fig. 5: An illustration of the node feature's composition

probability. When calculated using the formula $\hat{x}_{i,j}$, UEs' rates can be defined as the expectation rates associated with different BS. It will have a great distance to the actual rate, which leads to estimated error. This error will be eliminated when $\hat{x}_{i,j}$ satisfy the binary constraint, but it will trap in the saturation region of the softmax function. The unsupervised learning loss will not backpropagate at saturation region. To cope with this situation, we employ the softmax output function with the low-temperature parameter [36], which can be shown as:

$$\hat{x_{i,j}} = \frac{\exp\left(z_{i,j}/T\right)}{\sum_{n=1}^{J}\exp\left(z_{i,n}/T\right)} \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}. \tag{11}$$

We dynamic adjust the temperature to keep the $\hat{x}_{i,j}$ out of the saturation region and maintain the error in the tolerance interval.

Theorem 5: We denotes the $\hat{x}_{\max}^{(t)}$ as the the maximum value of $x_{i,j}(t), \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}$ at the train step t. Then, the best parameter for this problem is $T = \frac{\hat{x}_{\max}^{(t-1)}(\hat{x}_{\max}^{(t-1)}-1)}{2\hat{x}_{\max}^{(t-1)}-1}$.

The proof of Theorem 5 is shown in Appendix B.

Then the loss function for our proposed method comprises two items: the supervised part, the unsupervised part, which can be shown as $L = w_s L_s - w_r R$, where w_s and w_r are weights for the supervised part and the unsupervised part.

D. Generalization-Representation Learning and Specialization-Representation Learning

As a concept in machine learning, generalization ability refers to a model's ability to adapt to new, previously unknown data that is drawn from the same distribution as the one used to train the model [37]. In our problem, the HBGNN needs the generalization ability for obtain the desired solution for new channel. We define specialization ability as the performance achieved by a trained HBGNN in a single scene. HBGNNs with high generalization abilities may overlook the unique properties of the single scenario, limiting the effectiveness of HBGNNs in single scenarios. In order to take advantage of both the good generalization ability and the good specialization ability, our proposed method divides the learning process into two stages. ULS improves generalization ability, called Generalization Representation Learning (GRL). The DLS is called as Specialization Representation Learning (SRL), for improving the specialization ability.

Algorithm 2 Generalization Representation Learning Algorithm

1: **Input**: The Data set D; Training times T; Batch size B; learning rate η; the decay factor of learning rate τ; The window length l;
2: Generate the Heterogeneous Graph G; Create an empty memory set M and the average best rate $r_b = 0$; Build the HBGNN neural network and initialize the parameter of HBGNN.
3: **For** t = 1 to T **do**:
4: Sample a batch of data D_B from the data set D;
5: Calculate the $\mathbf{x}, \mathbf{p} = \text{HBGNN}(\mathbf{G}_{D_B})$, the batch loss function L in section III.A and the global transmitting rate R_B and add the R_B to the memory set M;
6: Calculate the average rate R_l of recent l elements in memory set M.
7: **If** $R_l > R_B$: Training the HBGNN neural network with L_B.
8: **Else**: $\eta \leftarrow \eta \times \tau$
9: **End If**
10: **End For**

In the ULS, we train the neural network with different training samples to learn the generalization representation. The upstream task is an offline training process that only needs historical data sets. This part is summarized in Algorithm 2. To eliminate the variance of loss, we update the neural network parameter with the batch sample, which is following steps 4 to 8. Additionally, we dynamically adjust the learning rate in accordance with the variation tendency of the rate, as in Step 7 to Step 8 of Algorithm 2.

The SRL part ensures that the neural network perfectly matches the current scenarios. As the downstream stage of the training process, trained HBGNN models are finely turned with the current scenario data. It saves time compared to training from begin. Also, the HBGNN can avoid trapping into a local optimum. To further eliminate the local-optimization effect that the overfitting brings, we add the entropy of the UA outputs to avoid the outcome overconfidence [38], [39], which is $L_{entropy} = -\sum_{j\in\mathcal{K}_s}\sum_{i\in\mathcal{N}} e_{ij}\ln e_{ij}$, where $e_{nk} = \exp\left(z_{nk}\right)/\sum_{i\in\mathcal{N}}\exp\left(z_{jk}\right)$. Besides, we set $w_s = 0$

Algorithm 3 Generalization Representation Learning Algorithm

1: **Input**: The learning rate η; convergence bound e; the memory set M
2: Load the HBGNN neural network and the parameters of HBGNN.
3: **while True**:
4: Calculate the $\mathbf{x}, \mathbf{p} = \text{HBGNN}(\mathbf{G_s})$, the loss L_{SRL} in Section III.D and the global transmitting rate R_{SRL}.
5: Add the rate R_{SRL} into set $\mathcal{M}$ and alculate the average rate r_n of recent l elements in memory set M.
6: **If** $|r_n - r_b| < e$: **End while**

for the label of previous data because the supervised learning may constrain the neural network to explore the feasible area of the space. Then, the loss function for SRL is: $L_{\text{SRL}} = -w_r R - w_e L_{entropy}$, where w_e is weighted for the loss of entropy. The SRL part is summarized in algorithm 3.

To sum up, the GRL part aims to learn the general representation. The learning process is time-consuming. Fortunately, we train it off-line. The SRL part aims to suit the real-time scenario. The SRL is trained based on the GRL and is conducted online. By combining the two parts, the strategy is flexible in actual implementation. HUDNs can directly apply the outcomes of general representation learning without the SRL part in situations where the scenario changes rapidly. If HUDNs wish to achieve better performance, retraining will enable the neural network to match the scenario perfectly.

IV. Numerical Results

In this section, we demonstrate the efficiency of the proposed algorithm with simulations. We consider a square area with a side length of $L = 200$ in an urban environment, wherein $J_n = 100$ small BSs are randomly distributed in $J_m = 5$ macro BSs covering the area. In this area, there are 20 buildings with 20m length, 20m width, and 30m high for the demonstration of non-line-of-sight transmission. The buildings' location is randomly generated. The BSs are deployed on the roof of the building if it is located inside the building. We assume $I = 120$ UEs are spread following a mutually independent fixed stochastic probability in every adjusted time frame.

The channel model is grouped into line-of-sight (LoS) and non-line-of-sight (NLoS) transmissions. Because our objective is considered in the long term, we only consider the large-scale channel gain. Then the channel between BS and UE can be obtained by the following equation,

$$\zeta(w) = \begin{cases} D^{\text{L}} w^{-\theta^{\text{L}}}, & \text{LoS} \\ D^{\text{NL}} w^{-\theta^{\text{NL}}}, & \text{NLoS} \end{cases}. \tag{12}$$

Where D^{L} and D^{NL} represent the LoS and NLoS path loss at the unit reference distance, respectively. θ^{L} and θ^{NL} represent LoS and NLoS path loss indexes, respectively.

As mentioned in Section IV.B, the neural network in our simulation composes two parts, the feature extractor, and the output layer. The feature extractor is formed by two HBGNN layers. Both layers use 128 neurons for each node. The output layer for power control uses one neuron for each node. As for user association, the output layer uses K neutrons to represent the connected index for every BS. The weight of supervised learning, unsupervised learning entropy is 10, 1, 1e-2. The other simulation parameters are listed in Table I.

TABLE I: SIMULATION PARAMETERS

Parameter	Value/Status
Cell area	200m × 200km
Bandwidth	20MHz
The power of Macro BS	50dBm
The power of small BS	20dBm
Batch size	64
Learning Rate	1e-4
Decay Factor	0.99
Window length	100
θ^{L},θ^{NL}	2.09,3.75
D^{L},D^{NL}	10.38,14.54

We compare our proposed method with several schedules, which are Maximal Achievable Rate Association with Maximum Power (MARAMP), Maximal Sum-Utility association with Maximum Power(MSUAMP), that proposed in paper [40], Maximal Sum-Utility association with Power Control (MSUAPC), User Association with Maximizing the Weighted Sum of Effective Rates(UAMWSER) and the Joint User Association and Power Control with Maximizing the Weighted Sum of Effective Rates (JUAPCMWSER) in the paper [41].

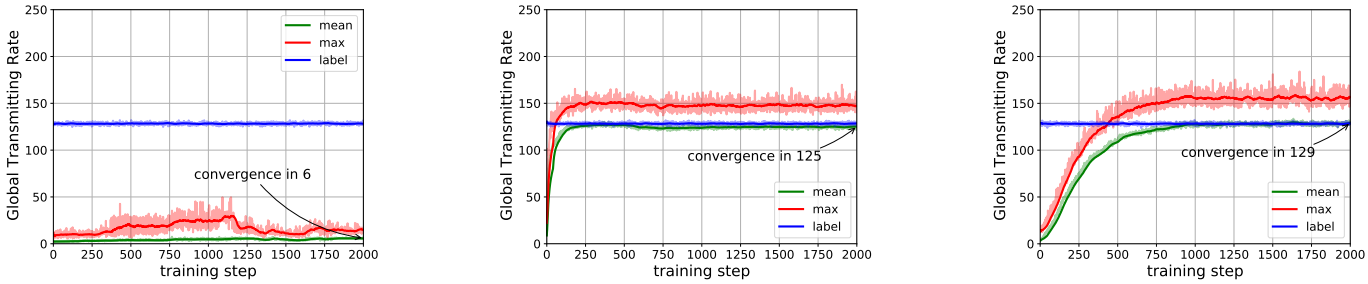
A. The Convergence Performance of different neural network

We first show the convergence performance of generalization representation learning in solving the UA, PA, and JUAPA, respectively. Furthermore, we compare the convergence performance by utilizing different neural networks, Fully-Connected Neural Networks (FCNN), CNN, and our HBGNN. The graph structure is ignored when utilizing the FCNNs and CNN as the neural network. The channels are simply arranged as a vector for FCNNs and a matrix for CNN. The FCNN contains four hidden layers, and each layer has 2048 neural cells. The CNN contains four layers, and each layer utilizes 64 3×3 kernels in each hidden layer. The kernels produce three channels. All kernels' output will be concatenated for the next layer. We present the global transmitting rate at each step of the training process. The global transmitting rate is measured by bits/sec/Hz. The 'label' in the legend means the performance of the label used in supervised learning. The 'max' and the 'mean' correspond to the max value and mean value of the global transmitting rate in each training batch.

TABLE II: The Models' Papameter Amounts

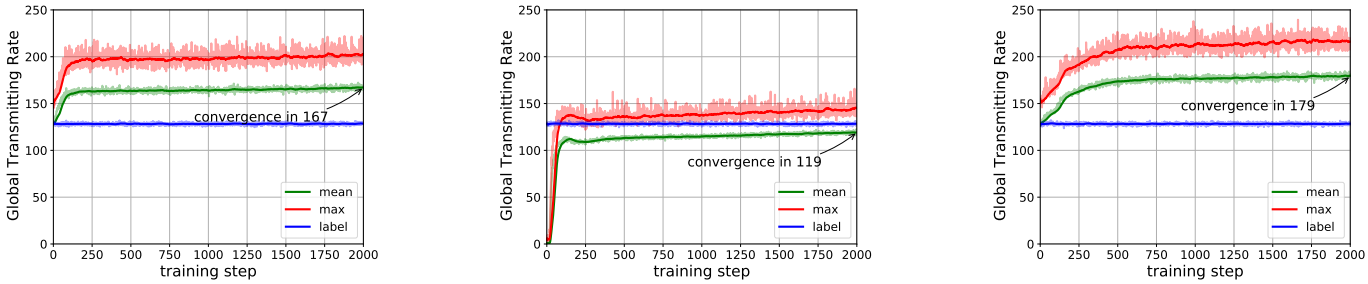
fully-connected	CNN	**HBGNN**
$6.4e^7$	$5.0e^5$	$4.7e^5$

We show the parameter amounts in Table II. The FCNN has more parameter for inferring such relationship, while the CNN and HBGNN have fewer parameters because they can utilize



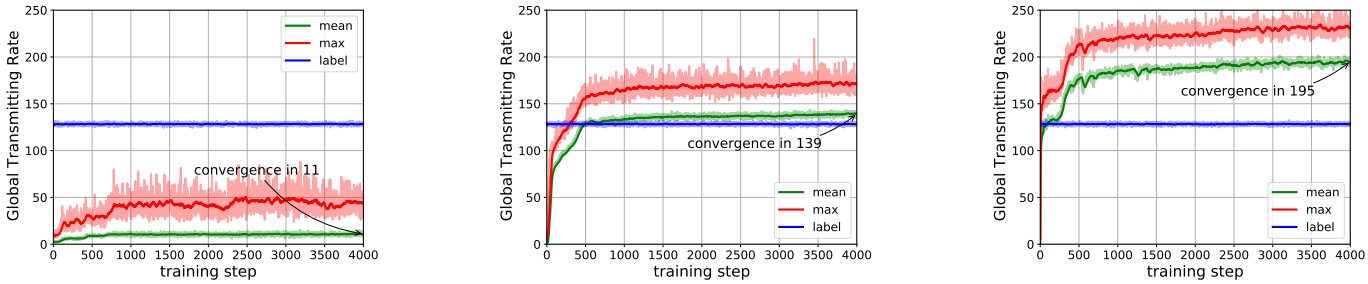
(a) Fully-Connected Neural Network (b) CNN (c) HBGNN

Fig. 6: The convergence curve for UA with different neural networks



(a) Fully-Connected Neural Network (b) CNN (c) HBGNN

Fig. 7: The convergence curve for PA with different neural networks



(a) Fully-Connected Neural Network (b) CNN (c) HBGNN

Fig. 8: The convergence curve for JUAPA with different neural networks

the transmission and inference relationship. This table shows that our network has smaller parameters, thus implying that our neural network computing efficiency and storage efficiency are slightly higher than the other two methods.

Fig 6,7,8 reveal several conclusions, as follows:

- The fully connected neural network failed to converge to a better lever than the label. However, it performs well in power control problems. The main reason is that the discrete variable of UA is hard to learn without structure information.
- The CNN convergence is faster than the HBGNN, however, the performance is not as good as the HBGNN's due to CNN's partial but not complete capture of the relationship.
- The unsupervised learning mainly affects the results because most results show a higher rate than the label.
- The joint adjustment of UA and PA will be better than the single adjustment shown in Fig.6(c), Fig.7(c), and Fig.8(c), which justify that there are gains in jointly solving the PA and UA with one neural network.
- The HBGNN shows the best performance in the three networks because it rationally utilizes communication structure information.

B. The Performance Analysis

In this section, we show the performance of the mentioned algorithms from the global transmitting rate, the cumulative distribution function (CDF) of the global transmitting rate, which is shown in Fig.9, Fig.10, respectively. Also, we compare our algorithm in different to UE density and BS density. In each figure, the experience are performed with 200 monte carlo experiment. We keep the location of BS and randomly generate the location of UE in monte carlo experiment.

As illustrated in Fig.9, the MARAMP has the most low-rate users among all strategies since all of UEs are con-

TABLE III: The global transmitting rate of each algorithms

Algorithm	MARAMP	MSUAMP	MSUAPC	UAMWSER	JUAPCMWSER	**GRL**	**SRL**
Global transmitting rate (bit/sec/Hz)	57.259	108.081	136.219	111.266	153.227	**195.451**	**221.000**

nected to the Macro BS according to the signal strength. The MSUAMP and UAMWSER show better performance because the users are more uniformly connect to the APs, but the UEs suffers from strong interference from Macro BS. MSUAMP and UAMWSER achieve 136.219 bits/sec/Hz and 153.227 bits/sec/Hz, respectively, with the active power configuration structure. Our algorithms show overwhelming superiority over other algorithms. The GRL and SRL achieve 195.451 bits/sec/Hz and 221.000 bits/sec/Hz. The result is also concluded in Table III.

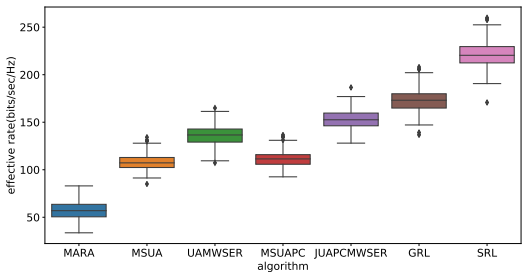


Fig. 9: The average rates of different association strategies..

In Fig.10, the same conclusion can be obtained. There are some UE reaching the highest rate in the MARAMP, but there are also more UEs with lower rates. MSUAMP, UAMWSER, and MSUAPC perform similarly. In comparison, the JUAPCMWSER shows better balance and achieves a higher maximum effective rate. The GRL displays excellent load balancing capabilities. The result of SRL shows the SRL improve the maximum UE rate in all scenario as well as the number of UE with low transmitting rate.

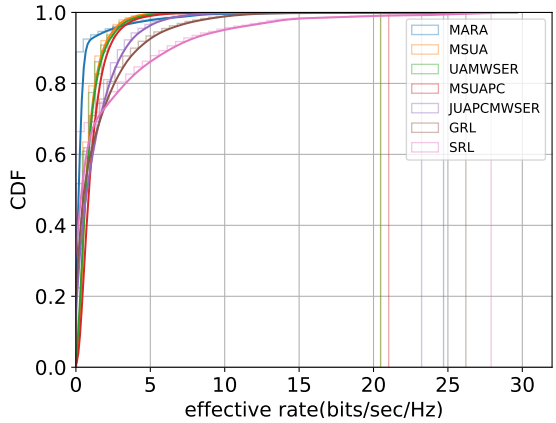
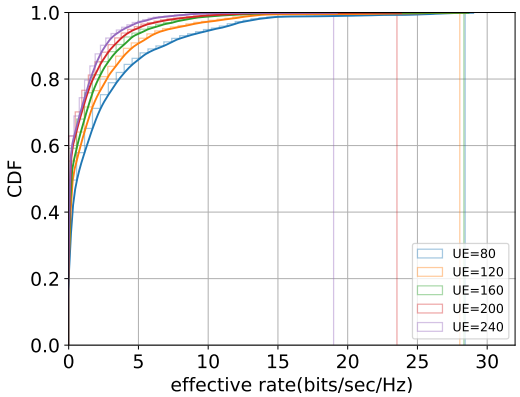
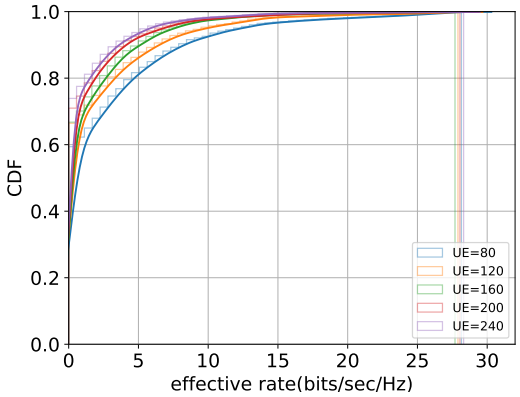


Fig. 10: The CDFs of effective rates of associated users under different association strategies.

Fig.11 shows the CDF in different UE density when the number of small BS is 100. This chart demonstrates that our



(a) GRL

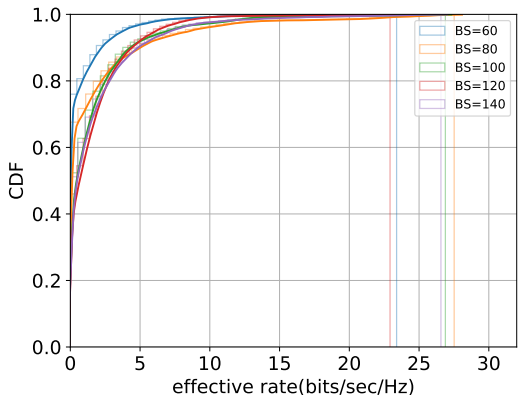


(b) SRL

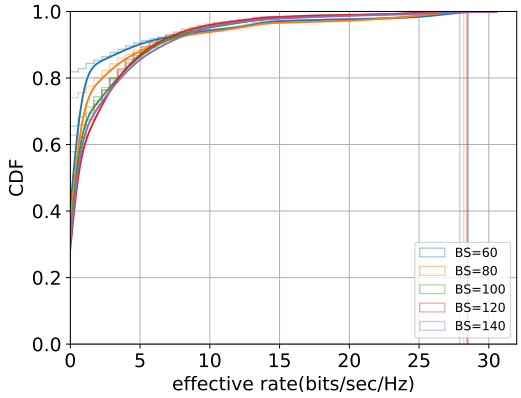
Fig. 11: The performance of our method in different UE density

algorithms work better when the UE density is low. With a rise in the number of UEs, the BS must maintain a low power level to minimize interference. As a consequence, the global data transmission rate is decreased. SRL's findings demonstrates that it is capable of resolving performance degradation. SRL's performance in the scenario with 120 UE is almost identical to that of GRL with 80 UE.

Fig.12 illustrates the CDF at various BS densities for 120 UE. The more BS, the closer the UE and the connected BS are, and the more degree of freedom the algorithm has to adjust. As a result, the performance of the algorithm can be increased by increasing the amount of BS. The lines corresponding to the 60 and 120 BSs demonstrate this. However, as indicated by the lines corresponding to the 120 and 140 BSs, a higher boundary exists. When the number of BS exceeds 120, the performance reaches saturation, indicating that the algorithm keeps the surplus BS's power to zero. SRL's figure is comparable to Fig.11. Every aspect of performance has been enhanced.



(a) GRL



(b) SRL

Fig. 12: The performance of our method in different BS density

However, the performance prompts are more in line with the 60 BS scenario. And the majority of them are saturated.

C. The Comparison of Computational Efficiency

To compare the computational efficiency of all 7 algorithms, we count the calculation time of each algorithm. We test in 200 events with python on the workstation with Intel Core i7-9700K. The calculation time of GRL only includes the inference time, the calculation time of SRL will contain the online training time.

The result is concluded in Table IV. One can conclude categorically that the algorithm involving iteration consumes more computation time. The MARAMP and GRL exhibit exceptional performance, which enables the communication system to handle multiple messages in a short period of time. After learning, the GRL requires only a few matrix multiplication operations to solve the problem. In comparison to the GRL, the optimization base problems require more than ten times the amount of computation. JUAPCMWSER's computation time is exponentially increased by the multilayer iteration process of two optimal problems. In the case of SRL, the learning process can also be viewed as an iterative one. Additionally, it requires time to converge. However, because the SRL has only one layer iteration, the majority of the retaining time is spent on back-propagation and parameter adjustment.

V. Conclusion

Our article proposed a learning-based paradigm for resolving HUDN resource allocation problems. This paradigm utilized a Heterogeneous Bipartite Graph Neural Network as the model, which was constructed using the HUDN's inherent properties, local validity, and correlation attenuation. This paradigm employs a loss function that combines supervised and unsupervised learning in order to train this model. Meanwhile, online and offline training are used to comply with HUDN. We modified our algorithm for the JUAPA problem and created a dynamic output layer to correspond to it. The results of our method demonstrated that our proposed method outperformed both the optimization and LRA methods. In future work, the more efficient model-based learning algorithm can be investigated in order to further improve efficiency and reduce data relay.

Appendix A
Proof for Theorem 3 and 4

The Theorem 3 and 4 mainly focuse on the relationship between $g_{i,j}$ and the $\mathcal{X}$ in function f^*, where $\mathcal{X}$ denotes the outputs. For easy expression, we utilize the power of BS as example and we denote m, n as two different indices for BS. We aim at proofing the p_m dose not need the information $g_{i,j}$, because the $\xi(p_m, g_{i,j}) \to 0$.

Let us start from the derivation of $\mathbf{p}^*$. The optimal solution for problem (10) must satisfy the Karush–Kuhn–Tucker (KKT) conditions, for which $\frac{\partial \mathcal{L}}{\partial \mathbf{p}^*} = 0$, where $\mathcal{L}$ is the Lagrangian function for problem (10). Then we have

$$\mathbf{\Phi}(\mathbf{p}^*, \mathbf{G}) = \frac{\partial \mathcal{L}}{\partial \mathbf{p}^*} = \mathbf{J} - \mathbf{c} = \mathbf{0}, \tag{14}$$

where $\mathbf{J} = \frac{\partial R}{\partial \mathbf{p}^*}$ is Jacobian matrix. The j_ath elements $J_{j_a} = \sum_{i \in \mathcal{I}, j \in \mathcal{J}} (\frac{\partial R_{i,j}}{\partial p_{j_a}})$. Taking the derivative of J_{j_a} with respect to $g_{i,j}$, we can get the following equation set

$$\begin{cases} f_1(\frac{\partial p_1}{\partial g_{i,j}}, \cdots, \frac{\partial p_{|j|_0}}{\partial g_{i,j}}) = 0 \\ \qquad\qquad \vdots \qquad\qquad . \\ f_{|j|_0}(\frac{\partial p_1}{\partial g_{i,j}}, \cdots, \frac{\partial p_{|j|_0}}{\partial g_{i,j}}) = 0 \end{cases} \tag{15}$$

Further, the correlation coefficient of $\frac{\partial p_1}{\partial g_{i,j}}, \cdots, \frac{\partial p_{|j|_0}}{\partial g_{i,j}}$ is the solution of the following linear equation set

$$\mathbf{W}\boldsymbol{\xi}(g_{i,j}) = \mathbf{B}(g_{i,j}) \tag{16}$$

. The elements of coefficient matrix $\mathbf{W}$ is shown in the bottom of this page. In Equation (13), $\omega_i^{(c)}$ and $\omega_i^{(i)}$ represent the connected and interfered items of ith UE, respectively. The $\Sigma_{i,j}^{(r)}$, $\Sigma_{i,j}^{(i)}$ is the total receive signal and the interference signal respectively. The ϕ_m is the UE set that connecting to the BS m and ψ_m is the UE set that interfered by the BS m. This

TABLE IV: Computational Efficiency

Algorithm	MARA	MSUA	MSUAPC	UAMWSER	JUAPCMWSER	**GRL**	**SRL**
Mean time(s)	0.037	3.334	285.503	0.204	1019.489	**0.039**	**97.697**

equation shows the diagonal elements of $\mathbf{W}$ contain all of connected and interfered items of the corresponding BS's 1st-order neighbors. Besides, the off-diagonal elements are the sum of connected and interfered items that are included in the intersection of two corresponding BSs' 1st-order neighbors.

The bias term B can be be divided into two conditions,

- when $j = x[i]$

$$B_n = \begin{cases} \Sigma_{i,j}^{(i)}\omega_i^{(c)} & n = j \\ p_{x[i]}g_{i,x[i]}\frac{\Sigma_{i,x[i]}^{(i)}\Sigma_{i,x[i]}^{(r)} - g_{i,n}p_{x[i]}\left(\Sigma_{i,x[i]}^{(i)}\right)}{\left(\Sigma_{i,x[i]}^{(r)}\right)^2\left(\Sigma_{i,x[i]}^{(i)}\right)^2} & n = \mathcal{N}_1 \end{cases} \tag{17}$$

- when $j \in \mathcal{N}_1(i)/(x[i])$

$$B_n = \begin{cases} -g_{i,n}p_j\omega_i^{(c)} & n = x[i] \\ B_j & n = j \\ -p_j g_{i,n}\omega_i^{(c)}\omega_i^{(i)} & n \in \mathcal{N}_1(i)/(j, x[i]) \\ 0 & else \end{cases} \tag{18}$$

where $B_j = \frac{p_{x[i]}g_{i,x[i]}}{\left(\Sigma_{i,x[i]}^{(r)}\right)\left(\Sigma_{i,x[i]}^{(i)}\right)} - p_n g_{i,n}\omega_i^{(c)}\omega_i^{(i)}$. This equation shows the $\mathbf{B} = \mathbf{0}$ When jth BS is the higher than 1 order neighbor of i_β.

We need to investigate the exact solution of equation (16). However, to obtain the (16) is hard, but some characters can still be investigated. Without loss of generality, we introduce three assumptions, which is

- For single BS, the interfering UE number of is greater than the connected UE number, which is $|\psi_m|_0 > |\phi_m|_0$.
- For any UE, the SINR is greater than 1 under the optimal condition, which is $\mu_{i,j} > 1$.
- $\mu_{i,j}$

We rearrangement the matrix $\mathbf{W}$ and split it in to blocks by following the way

$$\left[\begin{array}{c|c} W_{m,m} & \mathbf{W}_v^{\mathrm{H}} \\ \hline \mathbf{W}_v & \overline{\mathbf{W}} \end{array}\right] \tag{19}$$

where $\mathbf{W}_v^{\mathrm{H}} = \begin{bmatrix}\mathbf{W}_{m,\mathcal{N}_2(m)} & \mathbf{0} & \mathbf{0}\end{bmatrix}$, $\overline{\mathbf{W}}$ is the complementary minor of $\mathbf{W}$. Combining with Equation (13), (17), (18) and (19), we can have the following conclusion.

- Equation (16) always has solutions. For the rank of augmented matrix $[\mathbf{W}|\mathbf{B}]$ always equal with $\mathbf{W}$. And the BS without UE connected can be ignore for the optimal solution p^* for these BS is definitely 0. Thus, we only consider the situation when $|\mathcal{I}|_0 \leq |\mathcal{J}|_0$, where the W is a symmetric non-singular matrix.
- The coefficient matrix $\mathbf{W}$ keep same for each $g_{i,j}$ and the bias item is related with $g_{i,j}$.
- Equation (19) implicates that the property of $\mathbf{W^{-1}}$ that The elements in the $\mathbf{W^{-1}}$ decreases monotonically with the distance to the diagonal. And the elements of $\mathbf{W}_{m,else} \to 0$

After that, the solution for ξ_m can be obtained from equation (19) following the matrix inversion lemma, which is

$$\begin{aligned} \xi(p_m, g_{i,j}) &= |\frac{1}{\nu}(B_m + \mathbf{W}_v^{\mathrm{H}}\overline{\mathbf{W}}^{-1}\mathbf{B})| \\ &= |\frac{1}{\nu}(B_m + \sum_{j\in\mathcal{N}_1(i)}\sum_{l\in\mathcal{N}_2(m)} W_{l,m}B_j\overline{W}_{m,j})| \end{aligned} \tag{20}$$

where the $\overline{W}_{m,j}$ is the m, j element of $\overline{\mathbf{W}}^{-1}$. The solution have two conditions, which is the $W_{l,m} \geq 0$ and $B_j \geq 0$ when the $j = x[i]$, and the $W_{l,m} \geq 0$ and $B_j \leq 0$ when $j \in \mathcal{N}_1(i)/(x[i])$. For both conditions, we only need to consider the $\overline{W}_{m,j}$, whose value in different order can be concluded as follow:

- **Condition 1**: when $j \in \mathcal{N}_2(m)$, the $\overline{W}_{m,j}$ is from the position of $\mathcal{N}_2(m), \mathcal{N}_2(m)$ in $\overline{\mathbf{W}}$.
- **Condition 2**: when $j \in \mathcal{N}_4(m)$, the $\overline{W}_{m,j}$ is from the position of $\mathcal{N}_2(m), \mathcal{N}_4(m)$
- **Condition 3**: when the path between m and j is greater then 4, the $\overline{W}_{m,j}$ is from the position of $\mathcal{N}_4(m), else$ and the $\overline{W}_{m,j} \to 0$

From Condition 3, the neighbor greater than 4th order can be ignore, which proofs Theorem 3. Combine three conditions, the farther the relationship between the two nodes is, the

$$W_{m,n} = \begin{cases} -\sum_{i\in\phi_m} g_{i,m}^2\omega_i^{(c)} + \sum_{i\in\psi_m} g_{i,m}^2\omega_i^{(c)}\omega_i^{(i)} & m = n \\ -\sum_{i\in(\psi_m\cap\phi_n)\cup(\phi_m\cap\psi_n)} g_{i,m}g_{i,n}\omega_i^{(c)} + \sum_{i\in\psi_m\cap\psi_n} g_{i,m}g_{i,n}\omega_i^{(c)}\omega_i^{(i)} & n \in \mathcal{N}_2(m) \\ 0 & else \end{cases} \tag{13}$$

$$\text{where}: \omega_i^{(c)} = \frac{1}{\left(\Sigma_{i,x[i]}^{(r)}\right)^2}, \quad \omega_i^{(i)} = \frac{p_{x[i]}g_{i,x[i]}\left(\Sigma_{i,x[i]}^{(r)} + \Sigma_{i,x[i]}^{(i)}\right)}{\left(\Sigma_{i,x[i]}^{(i)}\right)^2} = \mu_{i,j}^2 + 2\mu_{i,j}$$

$$\Sigma_{i,x[i]}^{(r)} = P_{x[i]}g_{i,x[i]} + \sum_{n\in\mathcal{N}_1(i)} p_n g_{i,n} + \sigma^2, \quad \Sigma_{i,x[i]}^{(i)} = \sum_{n\in\mathcal{N}_1(i)} p_n g_{i,n} + \sigma^2$$

correlation coefficient gradually decreases, which completes the proof of the Theorem 4.

APPENDIX B
PROOF FOR THEOREM 5

First, we inference the saturation region of the softmax function. The gradient of loss to weight cross the softmax funtion is

$$\frac{\partial L_u}{\partial \omega} = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\left(\frac{\partial L_u}{\partial \gamma_{i,j}}\frac{\partial \gamma_{i,j}}{\partial \omega}x_{i,j} + \frac{\partial L_u}{\partial x_{i,j}}\frac{\partial x_{i,j}}{\partial \omega}\gamma_{i,j}\right). \quad (21)$$

Then we focus on the softmax part

$$\frac{\partial x_{i,j}}{\partial \omega} = \frac{\partial x_{i,j}}{\partial z_{n,j}}\frac{\partial z_{n,j}}{\partial \omega}, \forall n \in \mathcal{N}. \quad (22)$$

The $\frac{\partial x_{i,j}}{\partial z_{n,j}}$ is derivated from [42], which is given by:

$$\begin{aligned}\frac{\partial x_{i,j}}{\partial z_{i,j}} &= \frac{1}{T}\frac{\exp(z_{i,j}/T)\cdot(\Sigma - \exp(z_{i,j}/T))}{\Sigma^2} = \frac{1}{T}\hat{x}_{i,j}(1-\hat{x}_{i,j})\\ \frac{\partial x_{i,j}}{\partial z_{i,n}} &= -\frac{1}{T}\frac{\exp(z_{i,j}/T)\cdot\exp(z_{i,n}/T)}{\Sigma^2} = -\frac{1}{T}\hat{x}_{i,j}\hat{x}_{i,n}.\end{aligned} \quad (23)$$

From the equation above, the softmax function is in the functional area the when $x_{i,n} = 0.5, \forall n \in J$. Then we define E_B as the saturability, where the E_B is sum of $E^B_{i,j} = \frac{1}{T}\hat{x}_{i,j}(1-\hat{x}_{i,j})$.

On the other side, we define the $E_R = |R - \hat{R}|$ as rate error. To minimize the E_R, the $x_{i,j}$ need to approaching the one hot vector. We simplify the rate error as,

$$E^R = \sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}} B|\log_2(\frac{1+\mu_{i,j}}{1+\hat{\mu}_{i,j}})| \quad (24)$$

When $x_{i,j} = 0$, the $E^R_{i,j} = B\log_2(1+\hat{\mu}_{i,j})$. For $x_{i,j} = 1$, the equation can be approximated as $E^R_{i,j} \approx B|\log_2(\frac{\mu_{i,j}}{\hat{\mu}_{i,j}})|$, due to $\mu_{i,j} \gg 1$ in most time. It is worth to mention that the corresponding $\hat{x}_{i,j}$ is the maximum of $\hat{x}_{i,n}$ and we denote it with $\hat{x}_{\max}$. The $\frac{\mu_{i,j}}{\hat{\mu}_{i,j}}$ can be further simplified as:

$$\frac{\mu_{i,j}}{\hat{\mu}_{i,j}} = \frac{(1-\hat{x}_{\max})p_j g_{i,j}}{\hat{x}_{\max}} \cdot \frac{\sum_{n\in\mathcal{J}\setminus j}(1-\hat{x}_{i,n})p_n g_{i,n} + \sigma^2}{\sum_{n\in\mathcal{J}\setminus j}(1-x_{i,n})p_n g_{i,n} + \sigma^2} \quad (25)$$

Because the $\mu_{i,j} \gg 1$, we can get the $\sum_{n\in\mathcal{J}\setminus j}(1-\hat{x}_{i,n})p_n g_{i,n} \approx \sum_{n\in\mathcal{J}\setminus j}(1-x_{i,n})p_n g_{i,n}$. Then, we only need to focus on to minimize

$$\begin{aligned}E^R_{i,j} &\approx B * |\log_2(\frac{(1-\hat{x}_{\max})p_j g_{i,j}}{\hat{x}_{i,j}})|\\ &\approx B * |-\frac{3}{2}(1-2\hat{x}_{\max}) + \log_2(p_j g_{i,j})|\\ &\text{when:} x_{i,j} = 1.\end{aligned} \quad (26)$$

The last line uses the taylor expansion to log function. For $\frac{1}{2} < \hat{x}_{\max} < 1$, we only use the first two orders of Taylor series.

It is clear that E_R significantly lies on the situation, when $x_{i,j} = 1$. Hence, we only need to adjust T to minimize the equation (25), which is $\epsilon_B = 1 - 2\hat{x}_{\max}$ and to maximize the equation (22), which is $\epsilon_R = \frac{1}{T}\hat{x}_{\max}(1-\hat{x}_{\max})$. Then, the T can be solved by the problem,

$$T = \arg\max \alpha, \quad \text{where: } \alpha = \frac{\epsilon_R}{\epsilon_B} = \frac{\hat{x}_{\max}(1-\hat{x}_{\max})}{T(1-2\hat{x}_{\max})} \quad (27)$$

To solve this question, we first evaluate the $\frac{\partial \alpha}{\partial T}$, which is:

$$\frac{\partial \alpha}{\partial T} = -\frac{1}{T^2}\cdot\frac{\hat{x}_{\max}(1-\hat{x}_{\max})}{(1-2\hat{x}_{\max})} + \frac{1}{T}\cdot\frac{1-2\hat{x}_{\max}+2\hat{x}^2_{\max}}{(1-2\hat{x}_{\max})^2}\frac{\partial \hat{x}_{\max}}{\partial T} \quad (28)$$

Then let the $\frac{\partial \alpha}{\partial T} = 0$, the following equation can be get:

$$\frac{\partial \hat{x}_{\max}}{\partial T} = \frac{1}{T}\cdot\frac{\hat{x}_{\max}(1-2\hat{x}_{\max})(1-\hat{x}_{\max})}{1-2\hat{x}_{\max}+2\hat{x}^2_{\max}} \quad (29)$$

Then the solution of this differential equation is

$$T = e^{\int\frac{1-2\hat{x}_{\max}+2\hat{x}^2_{\max}}{\hat{x}_{\max}(1-2\hat{x}_{\max})(1-\hat{x}_{\max})}dx} = \frac{\hat{x}_{\max}(\hat{x}_{\max}-1)}{2\hat{x}_{\max}-1} \quad (30)$$

It is clear that we can not obtain the $\hat{x}^{(t)}_{\max}$ without T. However, we train the network iteratively, thus we can use the last time of $\hat{x}^{(t-1)}_{\max}$ to approximate it. The Theorem 2 is proofed.

REFERENCES

[1] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. K. Soong, and J. C. Zhang, "What Will 5G Be?" *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.

[2] W. Saad, M. Bennis, and M. Chen, "A Vision of 6G Wireless Systems: Applications, Trends, Technologies, and Open Research Problems," *arXiv:1902.10265 [cs, math]*, Feb. 2019.

[3] J. G. Andrews, X. Zhang, G. D. Durgin, and A. K. Gupta, "Are we approaching the fundamental limits of wireless network densification?" *IEEE Communications Magazine*, vol. 54, no. 10, pp. 184–190, Oct. 2016.

[4] M. Xiao, S. Mumtaz, Y. Huang, L. Dai, Y. Li, M. Matthaiou, G. K. Karagiannidis, E. Björnson, K. Yang, C. I, and A. Ghosh, "Millimeter Wave Communications for Future Mobile Networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 9, pp. 1909–1935, Sep. 2017.

[5] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta, "Massive MIMO for next generation wireless systems," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 186–195, Feb. 2014.

[6] M. Kamel, W. Hamouda, and A. Youssef, "Ultra-Dense Networks: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2522–2545, 2016.

[7] M. Kurras, M. Shehata, K. Hassan, and L. Thiele, "Spatial interference management with hierarchical precoding in ultra-dense heterogeneous networks," in *2015 IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, Oct. 2015, pp. 520–526.

[8] X. Liao, J. Shi, Z. Li, L. Zhang, and B. Xia, "A Model-Driven Deep Reinforcement Learning Heuristic Algorithm for Resource Allocation in Ultra-Dense Cellular Networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 983–997, Jan. 2020.

[9] J. Cao, T. Peng, X. Liu, W. Dong, R. Duan, Y. Yuan, W. Wang, and S. Cui, "Resource Allocation for Ultradense Networks With Machine-Learning-Based Interference Graph Construction," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 2137–2151, Mar. 2020.

[10] Y. Teng, M. Liu, F. R. Yu, V. C. M. Leung, M. Song, and Y. Zhang, "Resource Allocation for Ultra-Dense Networks: A Survey, Some Research Issues and Challenges," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2134–2168, 2019.

[11] Caidan Zhao, Xiaofei Xu, Zhibin Gao, and Lianfen Huang, "A coloring-based cluster resource allocation for ultra dense network," in *2016 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*. Hong Kong, China: IEEE, Aug. 2016, pp. 1–5.

[12] K. Shen and W. Yu, "Distributed Pricing-Based User Association for Downlink Heterogeneous Cellular Networks," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1100–1113, Jun. 2014.

[13] H. Zhang, S. Huang, C. Jiang, K. Long, V. C. M. Leung, and H. V. Poor, "Energy Efficient User Association and Power Allocation in Millimeter-Wave-Based Ultra Dense Networks With Energy Harvesting Base Stations," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 9, pp. 1936–1947, Sep. 2017.

[14] A. Khodmi, S. Ben Rejeb, N. Agoulmine, and Z. Choukair, "A Joint Power Allocation and User Association Based on Non-Cooperative Game Theory in an Heterogeneous Ultra-Dense Network," *IEEE Access*, vol. 7, pp. 111 790–111 800, 2019.

[15] J. Zheng, Y. Wu, N. Zhang, H. Zhou, Y. Cai, and X. Shen, "Optimal Power Control in Ultra-Dense Small Cell Networks: A Game-Theoretic Approach," *IEEE Transactions on Wireless Communications*, vol. 16, no. 7, pp. 4139–4150, Jul. 2017.

[16] X. You, C. Zhang, X. Tan, S. Jin, and H. Wu, *AI for 5G: Research Directions and Paradigms*, Std., Jul. 2018.

[17] Y. Zeng, X. Xu, S. Jin, and R. Zhang, "Simultaneous Navigation and Radio Mapping for Cellular-Connected UAV with Deep Reinforcement Learning," *arXiv:2003.07574 [cs, eess, math]*, Mar. 2020.

[18] M. Lee, G. Yu, and G. Y. Li, "Learning to Branch: Accelerating Resource Allocation in Wireless Networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 958–970, Jan. 2020.

[19] C. Sun, C. She, and C. Yang, "Unsupervised Deep Learning for Optimizing Wireless Systems with Instantaneous and Statistic Constraints," *arXiv:2006.01641 [cs, eess, math]*, Aug. 2020.

[20] Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, "LORM: Learning to Optimize for Resource Management in Wireless Networks With Few Training Samples," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 665–679, Jan. 2020.

[21] Z. Cheng, M. LiWangy, N. Chen, H. Lin, Z. Gao, and L. Huang, "Learning-Based Joint User-AP Association and Resource Allocation in Ultra Dense Network," *arXiv:2004.08194 [cs]*, Apr. 2020.

[22] A. Zappone, "Wireless Networks Design in the Era of Deep Learning: Model-Based, AI-Based, or Both?" *IEEE Transaction on Communications*, vol. 67, no. 10, p. 46, 2019.

[23] Y. Zhou, Z. M. Fadlullah, B. Mao, and N. Kato, "A Deep-Learning-Based Radio Resource Assignment Technique for 5G Ultra Dense Networks," *IEEE Network*, vol. 32, no. 6, pp. 28–34, Nov. 2018.

[24] D. Li, H. Zhang, K. Long, W. Huangfu, J. Dong, and A. Nallanathan, "User Association and Power Allocation Based on Q-Learning in Ultra Dense Heterogeneous Networks," in *2019 IEEE Global Communications Conference (GLOBECOM)*. Waikoloa, HI, USA: IEEE, Dec. 2019, pp. 1–5.

[25] H. Ding, F. Zhao, J. Tian, D. Li, and H. Zhang, "A deep reinforcement learning for user association and power control in heterogeneous networks," *Ad Hoc Networks*, vol. 102, p. 102069, May 2020.

[26] M. Botvinick, S. Ritter, J. X. Wang, Z. Kurth-Nelson, C. Blundell, and D. Hassabis, "Reinforcement learning, fast and slow," *Trends in Cognitive Sciences*, vol. 23, no. 5, pp. 408–422, 2019.

[27] 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 Transactions on Signal Processing*, vol. 66, no. 20, pp. 5438–5453, Oct. 2018.

[28] Q. Shi, M. Razaviyayn, Z.-Q. Luo, and C. He, "An Iteratively Weighted MMSE Approach to Distributed Sum-Utility Maximization for a MIMO Interfering Broadcast Channel," *IEEE Transactions on Signal Processing*, vol. 59, no. 9, pp. 4331–4340, Sep. 2011.

[29] W. Lee, M. Kim, and D.-H. Cho, "Deep Power Control: Transmit Power Control Scheme Based on Convolutional Neural Network," *IEEE Communications Letters*, vol. 22, no. 6, pp. 1276–1279, Jun. 2018.

[30] M. Eisen and A. Ribeiro, "Optimal Wireless Resource Allocation With Random Edge Graph Neural Networks," *IEEE Transactions on Signal Processing*, vol. 68, pp. 2977–2991, 2020.

[31] W. Cui, K. Shen, and W. Yu, "Spatial Deep Learning for Wireless Scheduling," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1248–1261, Jun. 2019.

[32] Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, "Graph Neural Networks for Scalable Radio Resource Management: Architecture Design and Theoretical Analysis," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 101–115, Jan. 2021.

[33] Z. Liu and J. Zhou, *Introduction to Graph Neural Networks*, ser. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2020.

[34] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A Comprehensive Survey on Graph Neural Networks," *arXiv:1901.00596 [cs, stat]*, Jan. 2019.

[35] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," *arXiv:1706.02216 [cs, stat]*, Sep. 2018.

[36] B. Gao and L. Pavel, "On the Properties of the Softmax Function with Application in Game Theory and Reinforcement Learning," p. 11.

[37] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[38] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," *arXiv:1707.06347 [cs]*, Aug. 2017, arXiv: 1707.06347.

[39] K. Sohn, D. Berthelot, C.-L. Li, Z. Zhang, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel, "FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence," *arXiv:2001.07685 [cs, stat]*, Jan. 2020.

[40] Q. Ye, B. Rong, Y. Chen, M. Al-Shalash, C. Caramanis, and J. G. Andrews, "User Association for Load Balancing in Heterogeneous Cellular Networks," *arXiv:1205.2833 [cs, math]*, Nov. 2012.

[41] T. Zhou, "Joint User Association and Power Control for Load Balancing in Downlink Heterogeneous Cellular Networks," *arXiv:1607.00853 [cs, math]*, Jul. 2016.

[42] G. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network," *arXiv:1503.02531 [cs, stat]*, Mar. 2015.