An Efficient Architecture Search for Scalable Beamforming Design in Cell-Free Systems

Guanghui Chen, *Graduate Student Member, IEEE*, Zheng Wang, *Senior Member, IEEE*,
Yi Jia, *Graduate Student Member, IEEE*, Yongming Huang, *Senior Member, IEEE*,
Luxi Yang, *Senior Member, IEEE*

***Abstract*—Scalable beamforming design (SBD) is an effective way to improve the computation efficiency of downlink communications. To this end, this paper proposes a scalable unsupervised network (SUNet) for SBD in cell-free systems. According to the analysis of beamforming characteristics, we first derive two architectural conditions for the SUNet. Then we confirm that the proposed SUNet achieves SBD within the improved computation efficiency as long as these two architectural conditions are met. With respect to the SUNet, an effective architecture search (EAS) algorithm is proposed. By applying a subpopulation selection strategy, the EAS algorithm automatically searches the desired architectures to fulfill the two architectural conditions of the SUNet, which avoids the labor-intensive task of manually setting the architectures. Furthermore, when the EAS algorithm calculates the individual fitness value, a parameter mapping strategy is proposed to map the parameters of the trained SUNet to the targeted SUNet as initial parameters, which reduces the number of iterations in the training process of the SUNet. Experimental results show that the SUNet reaches 95% of the upper bound of the scalable sum rate, where the sizes of areas as well as the number of base stations (BSs) and users are 5 times of these in training.**

***Index Terms*—Scalable beamforming design, architecture search, unsupervised deep learning, cell-free systems.**

I. Introduction

To provide ultra-reliable low-latency communications (URLLC) in the beyond 5G (B5G) wireless systems, the idea of cell-free networks has emerged [1]. For a cell-free network, multiple base stations (BSs) cooperate to simultaneously serve users within the network coverage area, which potentially resolves many of the interference issues that appear in current cellular networks [2]. Moreover, due to the high directivity of the beamforming design, it is also applied in cell-free systems to improve the performance of downlink communications [3]. Nevertheless, as the beamforming design is non-convex, effective optimization is a challenging problem [4], especially when the number of BSs and users increases. By applying a convex optimization method to approximate the solution, the weighted minimum mean square error (WMMSE) algorithm [5] obtained a stable solution for the beamforming design. Note that the WMMSE algorithm in cell-free systems tends to operate centrally rather than locally due to the limited computing power of users and the reduced signaling burden [6], [7]. The WMMSE algorithm has demonstrated excellent performance and was, thus, widely utilized as a benchmark for the beamforming design [8], [9]. Unfortunately, the WMMSE algorithm requires multiple matrix inversions, which seriously increases the computational complexity of the beamforming design.

Deep learning can improve the computation efficiency compared to traditional optimization algorithms [10], and it has been widely applied in communication systems for the beamforming design [11]. In particular, [12] applied the deep neural networks (DNNs) to design the downlink beamforming, which reduces the computational complexity compared with the WMMSE algorithm. Similarly, the convolutional neural networks (CNNs) were utilized in [13] to further reduce the computational complexity of the downlink beamforming design by leveraging the parameter sharing of the convolution kernel. Unfortunately, the above works take a supervised approach to train, which requires the label information. For this reason, based on the loss function characterized by the negative value of the sum rate, [14] proposed two unsupervised DNNs with lower computational complexity for the hybrid analog-digital beamforming design in cell-free systems. Following this unsupervised training approach, [15] used the deep reinforcement learning (DRL) to realize the beamforming design for cell-free systems, and it also effectively improves the computational efficiency with respect to the WMMSE algorithm. However, as the sizes of wireless networks rapidly increase, the number of users in cell-free systems increases significantly, where the computational burden of training a deep learning algorithm to implement the beamforming design becomes extremely large and unbearable. This indeed imposes a pressing challenge to the deep learning.

Recently, the scalability provides a new idea for solving the above problems efficiently. Specifically, the scalability means that the algorithm is able to scale to an enlarged wireless network without re-optimizing the parameters that are trained with respect to the small-sized wireless networks [16], thus leading to an improved computation efficiency of

This work was supported by the National Natural Science Foundation of China under Grants 62225107 and 62371124, the Natural Science Foundation on Frontier Leading Technology Basic Research Project of Jiangsu under Grant BK20222001, the Fundamental Research Funds for the Central Universities under Grant 2242022k60002, the Major Key Project of PCL, and the Postgraduate Research & Practice Innovation Program of Jiangsu Province under Grant KYCX23_0259. (*Corresponding authors: Yongming Huang and Zheng Wang*.)

G. Chen, Z. Wang, Y. Jia, Y. Huang, and L. Yang are with the School of Information Science and Engineering, and the National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China. Y. Huang and L. Yang are also with the Pervasive Communications Center, Purple Mountain Laboratories, Nanjing 211111, China. (e-mail: cgh@seu.edu.cn, z.wang@ieee.org, jiay@seu.edu.cn, huangym@seu.edu.cn, lxyang@seu.edu.cn).

the beamforming design. Note that the scalability defined in this paper is different from that defined in [17]. The scalability in [17] means that each access point (AP) in cell-free systems can realize the channel estimation, precoding, power control, fronthaul signaling with finite computational complexity and resource requirements, when the number of users tends to be infinite. This is outside the scope of this paper. The scalability in this paper refers to the fact that the deep learning models obtained by training on the small-sized wireless networks do not need to re-optimize the network parameters to directly scale to the large-sized wireless networks [16]. It also means that this scalable approach only needs to be trained on the small-sized wireless networks instead of the large-sized wireless networks, thus effectively improving the computational efficiency of the beamforming design in cell-free systems.[1] During this scalability process, the number of users and BSs usually change dynamically, so the dimension of the output beamforming should also change accordingly. This results in a necessary requirement for scalable beamforming design (SBD), i.e., the dimension of the output beamforming changes with that of the input channel.[2]

Some existing deep learning algorithms are hard to flexibly meet the above condition. Particularly, the deep learning methods in [12], [13], [14], [15] usually contain multiple fully connected (FC) layers [22], in which the number of neurons in these FC layers is fixed and cannot be changed easily, if the dimension of the input channel changes. To improve the scalability, the graph neural networks (GNNs) [23] have been applied to communication systems. The GNNs usually model the channel of communication systems according to the node and edge features of the graph [24]. For instance, [25] fulfilled SBD in device to device (D2D) communication systems by designing the message passing graph neural networks (MPGNNs). Analogously, to achieve SBD in heterogeneous D2D communication systems, [16] proposed a heterogeneous interference graph neural network (HIGNN). However, these GNNs algorithms are difficult to apply in cell-free systems, because they model one node according to one BS rather than multiple BSs in cell-free systems. This leads to an increase in the feature dimension of the node from one-dimensional (1D) to two-dimensional (2D). To this end, it is necessary to develop SBD in cell-free systems. It is interesting to observe that the output dimension of the convolution operation is determined by the input dimension and the convolutional architectural parameters, i.e., the sizes of the convolutional kernel, sliding step and zero padding, the number of the convolutional kernels. More specifically, for an output three-dimensional (3D) feature of the convolution operation, the width and height of the 3D feature are decided by the input dimension as well as the sizes of the convolutional kernel, sliding step and zero padding, where the last dimension of the 3D feature is decided by the input dimension and the number of the convolutional kernels. Accordingly, it is encouraged to derive some architectural conditions for implementing SBD in cell-free systems by combining these convolutional architectural parameters with the beamforming characteristics.

On the other hand, the performance of the deep learning mainly depends on its architectures and the corresponding parameters [26]. Unfortunately, the above works manually set these architectures, which is labor-intensive as such a manual design requires many trial-and-error processes [27]. To solve this problem, the neural architecture search (NAS) [28] algorithms are proposed to automatically search the architectures of deep learning algorithms. Current popular NAS algorithms are the evolutionary computation-based NAS algorithms, which effectively integrates the advantages of the evolutionary computation and gradient descent algorithms [29]. Typically, a genetic algorithm is used in [30] to search for the best architectures of CNNs, and experimental results have demonstrated its effectiveness. Since then, numerous evolutionary computation-based NAS algorithms have been proposed. These NAS algorithms achieve the automatic search of the architectures, which avoids the labor-intensive task of setting the architectures manually. However, in the processes of searching for the architectures, the changing relationship between output and input is not taken into account by these NAS algorithms, making them difficult to meet the requirement of SBD in cell-free systems. Thus, it is necessary to investigate an automatic architecture search algorithm that is suitable for SBD in cell-free systems.

Based on the above considerations, this paper has the following two objectives. One objective is to design a deep learning model that enables SBD in cell-free systems, i.e., the deep learning model is able to scale to the large-sized wireless networks by only training on the small-sized wireless networks without training on the large-sized wireless networks. Obviously, this would effectively improve the computational efficiency of the beamforming design in cell-free systems. Another objective is to design an automatic architecture search algorithm suitable for SBD in cell-free systems without the labor-intensive task of manually setting the architectures. To achieve these two objectives, this paper proposes a scalable unsupervised network (SUNet) [3] with an effective architecture search (EAS) algorithm. More specifically, the major contributions are summarized as follows:

1) A SUNet incorporating the residual CNNs and the attention module is proposed to achieve SBD in cell-free systems, which improves the computational efficiency of

[1] Note that the few-shot learning [18], meta-learning [19], transfer learning [20] and large-scale pre-trained models [21] first train a model on numerous different source task datasets to learn a priori knowledge (e.g., model initialization parameters), and then retrain the model on the target task dataset with the learned prior knowledge. Based on the above training process, if these four methods are applied to implement SBD in cell-free systems, they require training the model on both the small- and large-sized wireless networks. This incurs a huge computational burden and thus hardly meets the scalability defined in this paper.

[2] When scaling from the large- to small-sized wireless networks, it is easy to train the DNNs with padding zeros for inactive parameters. Yet, when scaling from the small- to large-sized wireless networks, it is hard to train the DNNs with padding zeros for inactive parameters. Thus, this paper uses SBD to solve the scaling problem of the small- to large-sized wireless networks.

[3] In the practical scenarios of hosting large events such as concerts or football matches, the number of users is huge, where the computational burden of training a deep learning model for the beamforming design in such practical scenarios is correspondingly huge and unacceptable. Apparently, the SUNet is well suited for such practical scenarios with higher computational efficiency, i.e., the SUNet also has good practicality in practical scenarios.

TABLE I: List of high frequency notations

Notation	Description	Notation	Description
$\mathcal{Q}$, $\mathcal{I}$	The sets of BSs and users	Q, I	The number of BSs and users
M, N	The number of BS and user antennas	i, j, q	The user indexes and the BS index
$\mathbf{H}_i$	The channel of the $\mathcal{Q}$ to the i^{th} user	$\mathbf{H}$	The channel of the $\mathcal{Q}$ to the $\mathcal{I}$
$\mathbf{H}_{\text{mod}}^{\text{3D}}$	The 3D real channel of the $\mathcal{Q}$ to the $\mathcal{I}$	$\boldsymbol{v}_i^q$	The beamforming of the q^{th} BS to the i^{th} user
$\boldsymbol{v}_i$	The beamforming of the $\mathcal{Q}$ to the i^{th} user	$\mathbf{V}$	The beamforming of the $\mathcal{Q}$ to the $\mathcal{I}$
$(\bullet)^{\text{H}}$	The conjugate transpose of the matrix	$\mathbb{C}$, $\mathbb{R}$	The sets of complex and real numbers
n	The index of the individual in a population	l	The index of the SUNet's layer
t	The index of the evolutionary computation iteration	h	The index of the SUNet training iteration
N_{pop}	The population size	$N_{\text{pop}}^{\text{sub}}$	The subpopulation size
p_{m}	The mutation probability	p_{c}	The crossover probability
$G_{\max}$	The maximum number of iterations	P_t	The population of the t^{th} iteration
P_t^{sub}	The subpopulation of the t^{th} iteration	Q_t^{sub}	The new offspring subpopulation of the t^{th} iteration
$\mathbf{V}_{\text{SUNet}}^{l-1}$	The input of the l^{th} layer of the SUNet	$\mathbf{V}_{\text{SUNet}}^{l}$	The output of the l^{th} layer of the SUNet
$\mathcal{A}_{\Theta}^{l}(\bullet)$	The attention module in the l^{th} layer of the SUNet	$\mathbf{A}_{\text{SUNet}}^{l}$	The output of $\mathcal{A}_{\Theta}^{l}(\bullet)$
$\mathcal{R}_{\Theta}^{l}(\bullet)$	The residual block in the l^{th} layer of the SUNet	$\mathbf{R}_{\text{SUNet}}^{l}$	The output of $\mathcal{R}_{\Theta}^{l}(\bullet)$

the beamforming design. Specifically, the residual CNNs are able to achieve the dimension of the output beamforming varying with that of the input CSI, i.e., it realizes SBD in cell-free systems. The attention module mitigates the drawback of cell-free systems, i.e., longer-range BSs serving users consume valuable power and bandwidth resources while contributing little useful power due to high path losses.

2) An EAS algorithm is proposed. By utilizing a subpopulation selection strategy, the EAS algorithm automatically searches for the desired architectures of the SUNet, which avoids the labor-intensive task of manually setting the architectures.
3) A parameter mapping strategy is proposed. When the EAS algorithm calculates the individual fitness value, the targeted SUNet of each individual is trained from the initialization, which needs many iterations to be converged. For this reason, the parameter mapping strategy maps the parameters of the trained SUNet to the targeted SUNet of these individuals as initial parameters, which reduces the number of iterations in the training process of the SUNet.

The rest of this paper is organized as follows: In Section II, the system model is introduced. In Section III, the principle of the SUNet is introduced. In Section IV, the principle of the EAS algorithm is introduced. In Section V, some numerical results are given, which include the performance evaluation of the EAS algorithm, and the scalable performance evaluation of the SUNet. In Section VI, some conclusions are provided. The main notations are summarized in Table I.

II. System Model

Consider a cell-free downlink system with Q BSs and I users, which are equipped with M and N antennas, respectively. All BSs are connected to a central processing unit (CPU) through backhaul links to exchange information, where the CPU is able to obtain global channel information to collaboratively design the beamforming for improving the system sum rate [9]. That is, SBD operates centrally. Let $\mathcal{Q} = \{1, \cdots, Q\}$ and $\mathcal{I} = \{1, \cdots, I\}$ be the sets of BSs and users, respectively. To simplify the notation, let i and j represent the indexes of users, and q represent the index of BS. The received signal of the i^{th} user is defined as

$$\boldsymbol{y}_i = \mathbf{H}_i \boldsymbol{v}_i s_i + \sum_{j \neq i} \mathbf{H}_i \boldsymbol{v}_j s_j + \boldsymbol{z}_i \in \mathbb{C}^{N \times 1}, \tag{1}$$

where $\mathbf{H}_i \in \mathbb{C}^{N \times QM}$ represents the channel matrix of the BS set $\mathcal{Q}$ to the i^{th} user. $\boldsymbol{v}_i \in \mathbb{C}^{QM \times 1}$ represents the beamforming vector of the BS set $\mathcal{Q}$ to the i^{th} user. s_i represents the data sent to the i^{th} user. $\boldsymbol{z}_i$ represents the additive noise following the complex Gaussian distribution $\mathcal{CN}\left(\mathbf{0}, \sigma_i^2 \mathbf{I}\right)$, in which $\mathbf{I}$ is the identity matrix. Based on $\mathbf{H}_i$ and $\boldsymbol{v}_i$, the channel and beamforming matrices of the BS set $\mathcal{Q}$ to the user set $\mathcal{I}$ are defined as $\mathbf{H} = \left[\begin{array}{cccc} \mathbf{H}_1^{\text{H}}, & \cdots & ,\mathbf{H}_i^{\text{H}}, & \cdots & ,\mathbf{H}_I^{\text{H}} \end{array}\right]^{\text{H}} \in \mathbb{C}^{IN \times QM}$ and $\mathbf{V} = \left[\begin{array}{cccc} \boldsymbol{v}_1, & \cdots & ,\boldsymbol{v}_i, & \cdots & ,\boldsymbol{v}_I \end{array}\right] \in \mathbb{C}^{QM \times I}$, respectively. The achievable rate of the i^{th} user is defined as

$$R_i = \log \left| \boldsymbol{I} + \mathbf{H}_i \boldsymbol{v}_i \boldsymbol{v}_i^{\text{H}} \mathbf{H}_i^{\text{H}} \mathbf{R}_{\tilde{v}_i \tilde{v}_i}^{-1} \right|, \tag{2}$$

where $(\bullet)^{\text{H}}$, $(\bullet)^{-1}$ and $|\bullet|$ represent the conjugate transpose, inverse and determinant of the matrix, respectively. $\mathbf{R}_{\tilde{v}_i \tilde{v}_i}$ represents the covariance matrix at the i^{th} user, which is defined as

$$\mathbf{R}_{\tilde{v}_i \tilde{v}_i} = \sum_{j \neq i} \mathbf{H}_i \boldsymbol{v}_j \boldsymbol{v}_j^{\text{H}} \mathbf{H}_i^{\text{H}} + \sigma_i^2 \boldsymbol{I}. \tag{3}$$

The beamforming design can be transformed into the maximization sum rate problems under the power constraint, which is defined as

$$\begin{aligned} &\max_{\boldsymbol{v}_i} \sum_{i \in \mathcal{I}} R_i, \\ &\text{s.t.} \sum_{i \in \mathcal{I}} \left(\boldsymbol{v}_i^q\right)^{\text{H}} \boldsymbol{v}_i^q \leq \text{P}_{\max}, \forall q \in \mathcal{Q}, \end{aligned} \tag{4}$$

where $\boldsymbol{v}_i^q$ represents the beamforming vector of the q^{th} BS to the i^{th} user. The relationship between $\boldsymbol{v}_i^q$ and $\boldsymbol{v}_i$ is described as $\boldsymbol{v}_i = \left[(\boldsymbol{v}_i^1)^{\text{H}}, \quad \cdots \quad ,(\boldsymbol{v}_i^q)^{\text{H}}, \quad \cdots \quad ,(\boldsymbol{v}_i^Q)^{\text{H}} \right]^{\text{H}}$. $\text{P}_{\max}$ represents the BS maximum power.

Since the optimization problem (4) is non-convex, traditional optimization algorithms usually approximate to it by a convex optimization problem. For example, the WMMSE algorithm [5] first introduces a weighting matrix to transform

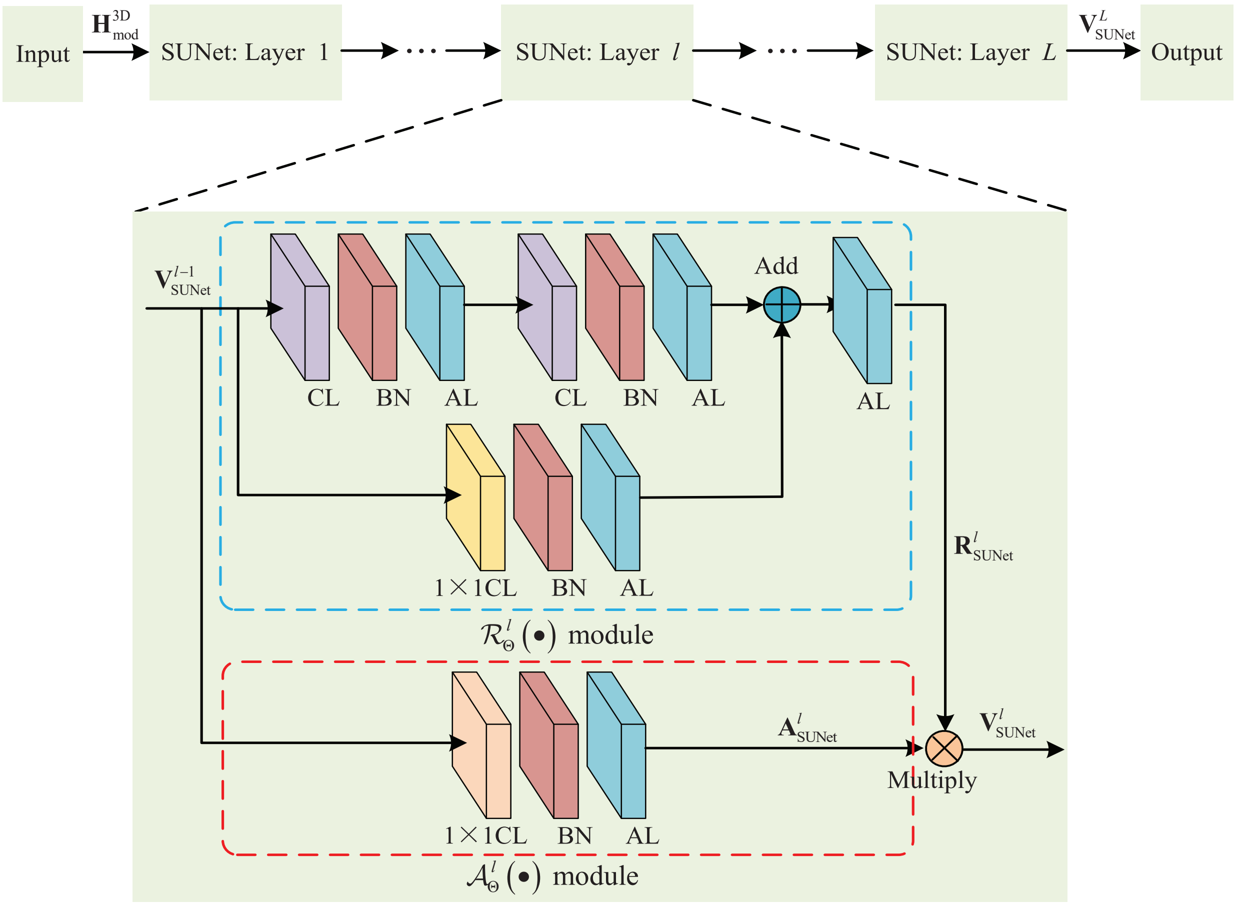
Input
$\mathbf{H}_{\text{mod}}^{\text{3D}}$
SUNet: Layer 1
SUNet: Layer l
SUNet: Layer L
$\mathbf{V}_{\text{SUNet}}^{L}$
Output
$\mathbf{V}_{\text{SUNet}}^{l-1}$
CL
BN
AL
CL
BN
AL
Add
AL
$\mathbf{R}_{\text{SUNet}}^{l}$
1×1CL
BN
AL
$\mathcal{R}_{\Theta}^{l}(\bullet)$ module
$\mathbf{A}_{\text{SUNet}}^{l}$
$\mathbf{V}_{\text{SUNet}}^{l}$
Multiply
1×1CL
BN
AL
$\mathcal{A}_{\Theta}^{l}(\bullet)$ module

Fig. 1: The proposed SUNet.

the optimization problem (4) into an equivalent minimization MSE optimization problem with three convex optimization variables $\boldsymbol{v}_i$, $\boldsymbol{u}_i$, and $\boldsymbol{w}_i$. This MSE optimization problem is then solved by using the block coordinate descent (BCD) [31] algorithm to iteratively fix two optimization variables and update the third optimization variable. Specifically, the optimization variables $\boldsymbol{u}_i$ and $\boldsymbol{w}_i$ have closed expressions, i.e.,

$$\boldsymbol{u}_i = \left(\sum_{j \in \mathcal{I}} \mathbf{H}_i \boldsymbol{v}_j \boldsymbol{v}_j^{\mathrm{H}} \mathbf{H}_i^{\mathrm{H}} + \sigma_i^2 \mathbf{I} \right)^{-1} \mathbf{H}_i \boldsymbol{v}_i, \tag{5}$$

$$\boldsymbol{w}_i = \left(1 - \boldsymbol{v}_i^{\mathrm{H}} \mathbf{H}_i^{\mathrm{H}} \boldsymbol{u}_i\right)^{-1}. \tag{6}$$

When the optimization variables $\boldsymbol{u}_i$ and $\boldsymbol{w}_i$ are fixed after updating, the optimization variable $\boldsymbol{v}_i$ is convex, which can be solved by standard convex optimization algorithms. Through the above iterative solution process, the stable solution of the optimization problem (4) is obtained. However, due to the high computational complexity of multiple matrix inversions, the WMMSE algorithm becomes unaffordable when the system size goes large.

To improve the calculation efficiency of the beamforming design, deep learning becomes a good alternative. However, many deep learning-based beamforming design algorithms like the DNNs [12] and CNNs [13] can not easily realize SBD. By contrast, the GNNs-based MPGNNs [25] and HIGNN [16] can implement SBD, which cannot be applied to cell-free systems. Additionally, these deep learning algorithms set their architectures manually, which is labor-intensive because this manual setting requires many trial-and-error processes. For this purpose, it is necessary to carry out the research of the EAS for SBD in cell-free systems.

III. Design of SUNet

In this section, we pay our attention on SBD for cell-free systems. Specifically, the SUNet is first proposed to map the beamforming from the channel. Meanwhile, based on the beamforming characteristics, the two architectural conditions of the SUNet are also given. Finally, the SBD of the SUNet is derived on the fulfillment of the two architectural conditions.

A. Proposed SUNet

Fig.1 illustrates the proposed SUNet. It includes the input layer, SUNet and output layer. The input layer converts the 2D complex channel matrix $\mathbf{H} \in \mathbb{C}^{IN \times QM}$ into the 3D real channel tensor. The SUNet maps the 3D real beamforming tensor from the 3D real channel tensor. The output layer transforms the 3D real beamforming tensor into the 3D complex beamforming tensor, which is satisfied with the power constraint.[4]

1) Input Layer: $\mathbf{H} \in \mathbb{C}^{IN \times QM}$ and $\mathbf{V} \in \mathbb{C}^{QM \times I}$ are complex matrices, while deep learning algorithms are performed in real numbers. For this reason, the input layer first solves the modulus value of $\mathbf{H} \in \mathbb{C}^{IN \times QM}$ to get the 2D real channel matrix $\mathbf{H}_{\text{mod}}^{\text{2D}} \in \mathbb{R}^{IN \times QM}$. Then $\mathbf{H}_{\text{mod}}^{\text{2D}} \in \mathbb{R}^{IN \times QM}$ is converted into the 3D real channel tensor $\mathbf{H}_{\text{mod}}^{\text{3D}} \in \mathbb{R}^{Q \times I \times MN}$.

2) SUNet: In order to achieve SBD in cell-free systems, the dimension of the output beamforming should flexibly change with that of the input channel. The transformer [32], long short-term memory (LSTM) [33] and gated recurrent unit (GRU) [34] are actually some state-of-the-art deep learning algorithms. However, the multi-layer perceptron (MLP) in the feedforward network of the transformer, and the FC in the LSTM and GRU fail to achieve SBD in cell-free systems, since the number of neurons in the MLP and FC is fixed without changing when the input dimension varies. On the contrary, the output dimension of the convolution operation is determined by the input dimension and the convolutional architectural parameters. Therefore, it is possible to derive some architectural conditions for SBD implementation in cell-free systems by combining these convolutional architectural parameters with the beamforming characteristics. For this reason, the SUNet applies the CNNs based on the convolutional operation instead of the transformer, LSTM and GRU. Furthermore, since the residual block (RB) structure of the CNNs effectively avoids the gradient disappearance problem, following [35], the SUNet applied the CNNs to construct the RB structure, i.e., $\mathcal{R}_{\Theta}^{l}(\bullet)$ module in Fig.1. It contains the convolution layer (CL), 1×1 CL, batch normalization (BN) layer, activation layer (AL). More specifically, the CL in $\mathcal{R}_{\Theta}^{l}(\bullet)$ mainly realizes the feature extraction [36]. The 1×1 CL in $\mathcal{R}_{\Theta}^{l}(\bullet)$ adjusts its architectures to ensure that the outputs of the CL and 1×1 CL in $\mathcal{R}_{\Theta}^{l}(\bullet)$ have the same dimension, which forms an identity mapping [35] to effectively avoid the gradient disappearance problem. The BN layer reduces the overfitting probability [37], and the AL achieves the non-linearization of the feature [38].

In addition, cell-free systems exist with the unfavourable fact that longer-range BSs serving users take up precious power and bandwidth resources yet contribute little useful power at the users due to high path losses [2]. In other words, for a user in cell-free systems, the modulus of the channel information for the longer-range BSs will usually be smaller than that of the shorter-range BSs due to the larger path losses of the longer-range BSs. Accordingly, to reduce the above-mentioned unfavourable problem, when dealing with the modulus of the channel information corresponding to the longer-range BSs, their corresponding weights can be set to smaller values for suppressing, and vice versa. For this purpose, the SUNet also applies an attention mechanism [39], as illustrated in the $\mathcal{A}_{\Theta}^{l}(\bullet)$ of Fig.1. It includes the 1×1 CL, AL and BN layer, which are similar to $\mathcal{R}_{\Theta}^{l}(\bullet)$, except that the 1×1 CL in the attention module $\mathcal{A}_{\Theta}^{l}(\bullet)$ has different architectural parameters. To be more specific, when the elements corresponding to the longer- and shorter-range BSs in $\mathbf{H}_{\text{mod}}^{\text{3D}} \in \mathbb{R}^{Q \times I \times MN}$ are operated through the attention module $\mathcal{A}_{\Theta}^{l}(\bullet)$, the 1×1 CL of the attention module $\mathcal{A}_{\Theta}^{l}(\bullet)$ will compute the attention weights for the longer- and shorter-range BSs, respectively. Owing to the parameter-sharing mechanism of the 1×1 CL, i.e., the parameters of the 1×1 CL are the same for the elements corresponding to the longer- and shorter-range BSs in $\mathbf{H}_{\text{mod}}^{\text{3D}} \in \mathbb{R}^{Q \times I \times MN}$, the corresponding weights of the longer-range BSs are smaller than those of the shorter-range BSs. That is, the elements corresponding to the shorter-range BSs in $\mathbf{H}_{\text{mod}}^{\text{3D}} \in \mathbb{R}^{Q \times I \times MN}$ are weighted with larger values for highlighting, while the elements corresponding to the longer-range BSs in $\mathbf{H}_{\text{mod}}^{\text{3D}} \in \mathbb{R}^{Q \times I \times MN}$ are weighted with smaller values for suppressing. This effectively reduces the above-mentioned unfavourable problem in cell-free systems.

Finally, it is well known that the shallow neural networks have the limited feature extraction ability, while the deep neural networks have the better feature extraction ability [40]. Consequently, this paper employs the deep structure that takes the output features of the previous layer as the inputs of the next layer, i.e., the $l-1^{th}$ layer's output features $\mathbf{V}_{\text{SUNet}}^{l-1}$ are used as the inputs of the l^{th} layer. In summary, the SUNet applies the deep structure that combines the residual CNNs and the attention mechanism.

3) Output Layer: Since the beamforming required by the optimization problem (4) is complex numbers, the 3D real beamforming tensor outputted by the SUNet is converted to the 3D complex beamforming tensor by using the following equation.

$$\mathbf{V}_{\text{com}} = \mathbf{V}_{\text{SUNet}}^{L}[:,:,0:M] + j\mathbf{V}_{\text{SUNet}}^{L}[:,:,M:2M], \tag{7}$$

where $\mathbf{V}_{\text{SUNet}}^{L}$ represents the output 3D real beamforming tensor of the SUNet, and $\mathbf{V}_{\text{com}}$ represents the converted 3D complex beamforming tensor. Besides, as the power constraint is a decoupled convex constraint, it can be satisfied by a projection function [41]. For this reason, $\mathbf{V}_{\text{com}}$ is inputted into a projection function to satisfy the power constraint, i.e.,

$$\boldsymbol{v}_i^q = \begin{cases} \boldsymbol{v}_i^q & \text{if } \sum\limits_{i \in \mathcal{I}} \left(\boldsymbol{v}_i^q\right)^{\text{H}} \boldsymbol{v}_i^q \leq \text{P}_{\text{max}}, \\ \dfrac{\boldsymbol{v}_i^q}{\sum\limits_{i \in \mathcal{I}} \left(\boldsymbol{v}_i^q\right)^{\text{H}} \boldsymbol{v}_i^q} \text{P}_{\text{max}} & \text{otherwise.} \end{cases} \tag{8}$$

B. SBD of SUNet

In this subsection, we deduce the SBD of the SUNet in detail. Specifically, for the l^{th} layer of the SUNet, its calculation formulas are defined as

$$\mathbf{R}_{\text{SUNet}}^{l} = \mathcal{R}_{\Theta}^{l}(\mathbf{V}_{\text{SUNet}}^{l-1}), \tag{9}$$

$$\mathbf{A}_{\text{SUNet}}^{l} = \mathcal{A}_{\Theta}^{l}(\mathbf{V}_{\text{SUNet}}^{l-1}), \tag{10}$$

$$\mathbf{V}_{\text{SUNet}}^{l} = \mathbf{R}_{\text{SUNet}}^{l} \otimes \mathbf{A}_{\text{SUNet}}^{l}, \tag{11}$$

where $\mathbf{V}_{\text{SUNet}}^{l-1}$ represents the input of the l^{th} layer of the SUNet. Note that $\mathbf{V}_{\text{SUNet}}^{0} = \mathbf{H}_{\text{mod}}^{\text{3D}} \in \mathbb{R}^{Q \times I \times MN}$. Θ represents the parameters of the SUNet. $\mathcal{R}_{\Theta}^{l}(\bullet)$ represents the RB operation in the l^{th} layer of the SUNet. $\mathbf{R}_{\text{SUNet}}^{l}$ represents the output of $\mathcal{R}_{\Theta}^{l}(\bullet)$. $\mathcal{A}_{\Theta}^{l}(\bullet)$ represents the operation of the attention module in the l^{th} layer of the SUNet. $\mathbf{A}_{\text{SUNet}}^{l}$ represents the

[4] In this paper, the first, second and third dimensions of the 3D channel and beamforming tensors are represented as the width, height and third dimension, respectively.

output of $\mathcal{A}_{\Theta}^{l}(\bullet)$. $\otimes$ represents the Hadamard product of the 2D matrix and 3D tensor, in which the dimensions of the 2D matrix are equal to the width and height of the 3D tensor.[5] $\mathbf{V}_{\text{SUNet}}^{l}$ represents the output of the l^{th} layer of the SUNet.

In what follows, we combine the formulas (9)-(11) and the beamforming characteristics to derive the two architectural conditions of the SUNet. More specifically, for the optimization problem (4), the beamforming has the following characteristics: when the dimension of the input 3D real channel tensor $\mathbf{H}_{\text{mod}}^{\text{3D}}$ is $Q \times I \times MN$, the output beamforming should be a 3D complex tensor with dimension $Q \times I \times M$, which can be converted into a 3D real tensor with dimension $Q \times I \times 2M$. That is, the width and height of the input 3D real channel tensor are equal to those of the output 3D real beamforming tensor, where the size of the third dimension is changed from MN to $2M$. On the other hand, the output dimension of the SUNet is determined by the input dimension and its architectures, which should satisfy some conditions for the above beamforming characteristics, as follows.

To convenience the subsequent description, let k_l^w, k_l^h, p_l^w, p_l^h, s_l^w, s_l^h represent the width and height for the convolution kernel, zero padding and sliding step in $\mathcal{R}_{\Theta}^{l}(\bullet)$, respectively. Also, let c_l represent the number of the convolution kernels in the CL of $\mathcal{R}_{\Theta}^{l}(\bullet)$. Note that k_l^w, k_l^h, c_l, s_l^w, s_l^h, p_l^w, p_l^h $l = 1, \cdots, L$ are also the architectures of the SUNet.

Remark 1: When $s_l^w = 1$, $s_l^h = 1$, if $p_l^w = \frac{1}{2}(k_l^w - 1)$, $p_l^h = \frac{1}{2}(k_l^h - 1)$, both p_l^w, p_l^h and k_l^w, k_l^h are positive integers, then $w_{\mathbf{V}_{SUNet}^{l}} = w_{\mathbf{V}_{SUNet}^{l-1}}$, $h_{\mathbf{V}_{SUNet}^{l}} = h_{\mathbf{V}_{SUNet}^{l-1}}$ due to the same convolution [42], [43]. In which, $w_{\mathbf{V}_{SUNet}^{l}}$, $h_{\mathbf{V}_{SUNet}^{l}}$,$w_{\mathbf{V}_{SUNet}^{l-1}}$ and $h_{\mathbf{V}_{SUNet}^{l-1}}$ represent the width and height of $\mathbf{V}_{SUNet}^{l}$ and $\mathbf{V}_{SUNet}^{l-1}$, respectively.

Remark 2: When $s_l^w > 1$, $s_l^h > 1$, if $p_l^w = \frac{1}{2}(w_{l-1}s_l^w - w_{l-1} - s_l^w + k_l^w)$, $p_l^h = \frac{1}{2}(h_{l-1}s_l^h - h_{l-1} - s_l^h + k_l^h)$, and p_l^w, p_l^h, s_l^w, s_l^h, k_l^w, k_l^h are positive integers, then $w_{\mathbf{V}_{SUNet}^{l}} = w_{\mathbf{V}_{SUNet}^{l-1}}$, $h_{\mathbf{V}_{SUNet}^{l}} = h_{\mathbf{V}_{SUNet}^{l-1}}$ due to the same convolution [42], [43].

Obviously, as long as the architectures k_l^w, k_l^h, s_l^w, s_l^h, p_l^w, p_l^h in each layer of the SUNet satisfy *Remark 1* or *Remark 2*, the width and height of the output beamforming $\mathbf{V}_{\text{SUNet}}^{L}$ of the SUNet are $Q \times I$, which are equal to the width and height of the input 3D real channel tensor $\mathbf{H}_{\text{mod}}^{\text{3D}}$. In addition, if the architecture c_L in the last layer of the SUNet is set to $2M$, then the output beamforming $\mathbf{V}_{\text{SUNet}}^{L}$ of the SUNet is a 3D real tensor with dimension $Q \times I \times 2M$. Consequently, the two architectural conditions of the SUNet are summarized in *Remark 3*.

Remark 3: The two architectural conditions of the SUNet can be expressed as:

1) *The k_l^w, k_l^h, s_l^w, s_l^h, p_l^w, p_l^h in each layer of the SUNet satisfy Remark 1 or Remark 2.*
2) *The c_L in the last layer of the SUNet is equal to $2M$.*

For *Remark 3*, the architectural condition 1 determines the width and height of the output 3D real beamforming tensor of the SUNet, where the architectural condition 2 determines the third dimension of the output 3D real beamforming tensor of the SUNet. It is obvious that the SUNet will output the 3D real beamforming tensor of dimension $Q \times I \times 2M$ when the dimension of the input 3D real channel tensor $\mathbf{H}_{\text{mod}}^{\text{3D}}$ is $Q \times I \times MN$ and the two architectural conditions in *Remark 3* are satisfied.

In the following, we study SBD for the architectures of the SUNet satisfying the two architectural conditions in *Remark 3*.

Remark 4: When the sizes of wireless networks gradually increase, the number of BSs and users increase, i.e., $Q \rightarrow Q + \Delta Q$ and $I \rightarrow I + \Delta I$, and the dimension of the input 3D real channel tensor $\boldsymbol{H}_{mod}^{3D}$ is changed from $Q \times I \times MN$ to $(Q + \Delta Q) \times (I + \Delta I) \times MN$, where ΔI and ΔQ are integers. Obviously, as long as the two conditions in Remark 3 are satisfied, the dimension of the output beamforming $\mathbf{V}_{SUNet}^{L}$ of the SUNet is changed from $Q \times I \times 2M$ to $(Q + \Delta Q) \times (I + \Delta I) \times 2M$ without re-optimizing the parameters, which satisfies the necessary condition of SBD in the introduction.

In summary, based on *Remark 1*, *Remark 2*, *Remark 3* and *Remark 4*, as long as the architectures of the SUNet satisfy the two conditions in *Remark 3*, the SUNet achieves SBD for cell-free systems.

IV. Design of EAS Algorithm

According to Section III.B, as long as the two architectural conditions in *Remark 3* are met, the SUNet realizes SBD for cell-free systems. Since manually setting the architectures of the SUNet is labor-intensive, this section proposes the EAS algorithm to automatically search the desired architectures of the SUNet. In particular, the EAS algorithm applies the subpopulation selection strategy to make the searched architectures satisfy the two architectural conditions in *Remark 3*. Meanwhile, the EAS algorithm also uses the parameter mapping strategy to reduce the number of iterations in the training process of the SUNet. For clarity, we first introduce the subpopulation selection strategy and parameter mapping strategy, then introduce the EAS algorithm, and finally discuss the generality of the EAS algorithm.[6]

A. Subpopulation Selection Strategy

For each layer of the SUNet, the convolution kernel size $k_l^w \times k_l^h$, the sliding step size $s_l^w \times s_l^h$, and the zero padding size $p_l^w \times p_l^h$ should all be lower than the width and height $Q \times I$ of the input 3D channel tensor channel $\mathbf{H}_{\text{mod}}^{\text{3D}}$, otherwise it will cause logical errors. Besides, the number of the convolution kernels c_L in the last layer of the SUNet should be equal to $2M$. To realize these, the population initialization is first introduced to code these architectures, which is shown in Function 1. Firstly, the bit of the individual is defined as

$$b = L\left(3\left(\lfloor \log_2(Q) \rfloor + \lfloor \log_2(I) \rfloor\right) + \lceil \log_2(2M) \rceil\right), \quad (12)$$

where $\lfloor \bullet \rfloor$ and $\lceil \bullet \rceil$ represent the rounded down and up integers, respectively. Then, an individual $P_{0,n}$ with a binary number of

[5] For example, for the 2D matrix $A \in \mathbb{R}^{a \times b}$ and the 3D tensor $B \in \mathbb{R}^{a \times b \times c}$, $A \otimes B$ is calculated as follows: $A \in \mathbb{R}^{a \times b}$ is first copied c times to become $C \in \mathbb{R}^{a \times b \times c}$, and then the Hadamard product is performed on $C \in \mathbb{R}^{a \times b \times c}$ and $B \in \mathbb{R}^{a \times b \times c}$.

[6] To simplify the notation, let n, l, t and h represent the index of the individual in a population, the index of the SUNet's layer, the index of the evolutionary computation iteration, the index of the SUNet training iteration, respectively.

Function 1: Population Initialization

```
Input: $N_{\text{pop}}$, $Q$, $I$, $M$, $L$.
Output: $P_0$.
$b \leftarrow$ Calculate the individual's bit by using (12);
$P_0 \leftarrow \emptyset$;
for $n \leftarrow 1$ to $N_{\text{pop}}$ do
    $P_{0,n} \leftarrow$ Randomly generate a individual with $b$ bits;
    $P_0 \leftarrow P_0 \cup P_{0,n}$;
end
Return: $P_0$.
```

Function 2: Subpopulation Selection Strategy

```
Input: $P_t$.
Output: $P_t^{\text{sub}}$.
$P_t^{\text{sub}} \leftarrow \emptyset$;
for $n \leftarrow 1$ to $N_{\text{pop}}$ do
    $P_{t,n} \leftarrow$ Calculate the $n^{th}$ individual;
    for $l \leftarrow 1$ to $L$ do
        $(k_l^w, k_l^h, p_l^w, p_l^h, s_l^w, s_l^h, c_l) \leftarrow$ Decode the architectures of each layer from $P_{t,n}$ using (13);
    end
    if (L layer architectures $(k_l^w$, $k_l^h$, $p_l^w$, $p_l^h$, $s_l^w$, $s_l^h)$ all satisfy Remarks 1 or 2) and $(c_L = 2M)$ then
        $P_t^{\text{sub}} \leftarrow P_t^{\text{sub}} \cup P_{t,n}$;
    end
end
Return: $P_t^{\text{sub}}$.
```

b bits is randomly generated, in which the encoding between the individual $P_{0,n}$ and the architectures of the l^{th} layer of the SUNet is represented as

$$\begin{cases} k_l^w = \text{Con}_{\text{bin}}^{\text{dec}}(P_{0,n}[l_0 : l_1]), \\ k_l^h = \text{Con}_{\text{bin}}^{\text{dec}}(P_{0,n}[l_1 : l_2]), \\ p_l^w = \text{Con}_{\text{bin}}^{\text{dec}}(P_{0,n}[l_2 : l_3]), \\ p_l^h = \text{Con}_{\text{bin}}^{\text{dec}}(P_{0,n}[l_3 : l_4]), \\ s_l^w = \text{Con}_{\text{bin}}^{\text{dec}}(P_{0,n}[l_4 : l_5]), \\ s_l^h = \text{Con}_{\text{bin}}^{\text{dec}}(P_{0,n}[l_5 : l_6]), \\ c_l = \text{Con}_{\text{bin}}^{\text{dec}}(P_{0,n}[l_6 : l_7]), \end{cases} \tag{13}$$

where $\text{Con}_{\text{bin}}^{\text{dec}}(\bullet)$ represents converting the binary to the decimal. $l_0 = (l-1)\,(3\,(\lfloor \log_2(Q) \rfloor + \lfloor \log_2(I) \rfloor) + \lceil \log_2(2M) \rceil)$, $l_1 = l_0 + \lfloor \log_2(Q) \rfloor$, $l_2 = l_1 + \lfloor \log_2(I) \rfloor$, $l_3 = l_2 + \lfloor \log_2(Q) \rfloor$, $l_4 = l_3 + \lfloor \log_2(I) \rfloor$, $l_5 = l_4 + \lfloor \log_2(Q) \rfloor$, $l_6 = l_5 + \lfloor \log_2(I) \rfloor$, $l_7 = l_6 + \lceil \log_2(2M) \rceil$, and $(l_7 - l_0) \times L = b$. Evidently, the encoding of the formula (13) ensures that $k_l^w \times k_l^h$, $s_l^w \times s_l^h$, $p_l^w \times p_l^h$ are all lower than $Q \times I$, and c_L can be taken to $2M$. Finally, the initialized population P_0 is generated by looping N_{pop} times.

Based on SectionIII.B, the SBD of cell-free systems can be implemented only if the searched architectures satisfy the two architectural conditions of *Remark 3*. However, traditional NAS algorithms do not take the changing relationship between output and input dimensions into account, and thus struggle to ensure that the searched architectures satisfy the two architectural conditions in *Remark 3*. For this reason, the subpopulation selection strategy is proposed, as shown in Function 2. Firstly, an empty subpopulation set P_t^{sub} is defined. Then, the first individual $P_{t,1}$ of the population P_t is decoded into the architectures of each layer of the SUNet by using the formula (13). If these architectures satisfy *Remark 1* or *Remark 2*, and $c_L = 2M$, the individual is stored in P_t^{sub}, otherwise discarded. Finally, the above operations are repeated N_{pop} times to obtain the subpopulation P_t^{sub}.

Function 3: Parameter Mapping Strategy

```
Input: $D_{\text{train}}$, $D_{\text{val}}$, SUNet$_{\text{tea}}$, $\mathbf{W}_{\text{tea}}$, $H$, $R$, ind$_n$.
Output: $f_n$.
SUNet$_n \leftarrow$ Decode the $n^{th}$ individual ind$_n$;
$\mathbf{W}_n \leftarrow$ Represent the parameters required by SUNet$_n$;
$\mathbf{W}_{\text{tea}} \leftarrow$ Extract the parameters of SUNet$_{\text{tea}}$;
Map $\mathbf{W}_{\text{tea}}$ as the initial parameters of $\mathbf{W}_n$;
$h \leftarrow 1$;
while $h < H$ do
    for each batch data in $D_{train}$ do
        $\ell \leftarrow$ Calculate the loss function by using (14);
        $\nabla \mathbf{W}_n \leftarrow$ Calculate gradients using $\partial \ell / \partial \mathbf{W}_n$;
        $\mathbf{W}_n \leftarrow$ Update parameters using $\mathbf{W}_n - R \times \nabla \mathbf{W}_n$;
    end
    $h \leftarrow h + 1$;
end
$f_n \leftarrow$ Calculate the fitness value of SUNet$_n$ on $D_{\text{val}}$ by using $\sum_{i \in \mathcal{I}} R_i$
Return: $f_n$.
```

B. Parameter Mapping Strategy

In computing the fitness value, traditional NAS algorithms usually first decode each individual in P_t^{sub} into the corresponding architectures, then initialize and train the model on the training data, and finally calculate the fitness value of each individual on the validation data. This means that traditional NAS algorithms need to train the model from initialization for each individual in P_t^{sub}, which usually takes many iterations to converge [44]. On the other hand, the crossover and mutation operations in the evolutionary computation will not completely break the individual structure, i.e., the partial structures of newly generated individuals are similar to their parents. If the parameters corresponding to these similar parts can be mapped, the models of the newly generated individuals do not need to be trained from the initialization. Evidently, this will reduce the number of iterations in the training process. For this purpose, the parameter mapping strategy is proposed, as shown in Function 3.

The inputs of Function 3 are the training data D_{train}, the validation data D_{val}, the trained teacher model SUNet$_{\text{tea}}$ and its parameters $\mathbf{W}_{\text{tea}}$, the n^{th} individual ind$_n$, the number of iterations H, the learning rate R, respectively. The output of Function 3 is the fitness value f_n of the n^{th}

individual ind_n. Firstly, the n^{th} individual ind_n in the subpopulation P_t^{sub} is decoded into the corresponding SUNet_n. Secondly, the required parameters of SUNet_n can be represented as $\mathbf{W}_n = \left[\ \mathbf{W}_n^1, \ \cdots \ , \mathbf{W}_n^l, \ \cdots \ , \mathbf{W}_n^L\ \right]$, in which $\mathbf{W}_n^l \in \mathbb{R}^{w_{\mathbf{W}_n^l} \times h_{\mathbf{W}_n^l} \times c_{\mathbf{W}_n^l}}$ is the parameters of the l^{th} layer of SUNet_n. Next, the parameters of the trained teacher model $\text{SUNet}_{\text{tea}}$ are extracted and represented as $\mathbf{W}_{\text{tea}} = \left[\ \mathbf{W}_{\text{tea}}^1, \ \cdots \ , \mathbf{W}_{\text{tea}}^l, \ \cdots \ , \mathbf{W}_{\text{tea}}^L\ \right]$, where $\mathbf{W}_{\text{tea}}^l \in \mathbb{R}^{w_{\mathbf{W}_{\text{tea}}^l} \times h_{\mathbf{W}_{\text{tea}}^l} \times c_{\mathbf{W}_{\text{tea}}^l}}$ is the parameters of the l^{th} layer of $\text{SUNet}_{\text{tea}}$. Then, the parameters $\mathbf{W}_{\text{tea}}^l$ of $\text{SUNet}_{\text{tea}}$ are mapped into the parameters $\mathbf{W}_n^l$ of SUNet_n as initial parameters. Specifically, if $w_{\mathbf{W}_{\text{tea}}^l} \geq w_{\mathbf{W}_n^l}$ or $h_{\mathbf{W}_{\text{tea}}^l} \geq h_{\mathbf{W}_n^l}$, then the central $w_{\mathbf{W}_n^l} \times h_{\mathbf{W}_n^l} \times c_{\mathbf{W}_{\text{tea}}^l}$ dimension data of $\mathbf{W}_{\text{tea}}^l$ are mapped into $\mathbf{W}_n^l$. Otherwise, the all data of $\mathbf{W}_{\text{tea}}^l$ are mapped into the central part of $\mathbf{W}_n^l$, where the values of other parts of $\mathbf{W}_n^l$ are assigned with 1. At this point, the dimension of the initial parameters of $\mathbf{W}_n^l$ is $w_{\mathbf{W}_n^l} \times h_{\mathbf{W}_n^l} \times c_{\mathbf{W}_{\text{tea}}^l}$. If $c_{\mathbf{W}_{\text{tea}}^l} \geq c_{\mathbf{W}_n^l}$, then the first $w_{\mathbf{W}_n^l} \times h_{\mathbf{W}_n^l} \times c_{\mathbf{W}_n^l}$ dimension data of $\mathbf{W}_n^l$ are used as the initial parameters. Otherwise, the all data of $\mathbf{W}_n^l$ are used as the initial parameters, in which the values of other parts of $\mathbf{W}_n^l$ are assigned with 1. Subsequently, the gradient descent algorithm is used to update the parameters of the mapped SUNet_n through an unsupervised training way (see lines 6-13), where the unsupervised loss function is defined as

$$\ell = -\sum_{i \in \mathcal{I}} R_i. \tag{14}$$

Finally, the fitness value f_n of SUNet_n on the validation data D_{val} is calculated by using $\sum\limits_{i \in \mathcal{I}} R_i$.

C. EAS Algorithm

Based on the above two strategies, the EAS algorithm is proposed to automatically search for the desired architectures that satisfy the two architectural conditions of the SUNet, which is shown in Algorithm 1. The inputs of Algorithm 1 are the population size N_{pop}, the subpopulation size $N_{\text{pop}}^{\text{sub}} < N_{\text{pop}}$, the maximum number of iterations $G_{\max}$, the crossover probability p_{c}, and the mutation probability p_{m}, respectively. The output of Algorithm 1 is the best SUNet. Firstly, an initial population P_0 with the size of N_{pop} is generated by using the encoding strategy in Function 1, where an initial subpopulation P_0^{sub} is selected by using Function 2. Then, the fitness value of each individual in the subpopulation P_0^{sub} is calculated by using the parameter mapping strategy of Function 3. Subsequently, the new offspring subpopulation Q_t^{sub} is generated by the selection, crossover, and mutation operations (see lines 10-18). Afterwards, the subpopulation P_{t+1}^{sub} that enters the next iteration is selected by the environmental selection (see lines 19-20). Finally, the processes of the new offspring subpopulation Q_t^{sub} and subpopulation P_{t+1}^{sub} are iterated to $G_{\max}$ to output the best individual, which is decoded into the corresponding SUNet.

Algorithm 1: EAS

```
   Input: $N_{\text{pop}}$, $N_{\text{pop}}^{\text{sub}}$, $G_{\max}$, $p_{\text{c}}$, $p_{\text{m}}$.
   Output: The best SUNet.
1  $P_0^{\text{sub}} \leftarrow \emptyset$;
2  while Size$\big(P_0^{\text{sub}}\big) < N_{\text{pop}}^{\text{sub}}$ do
3      $P_0 \leftarrow$ Initialize the population by Function 1;
4      $P_0^{\text{mid}} \leftarrow$ Select the subset of $P_0$ by Function 2;
5      $P_0^{\text{sub}} \leftarrow P_0^{\text{sub}} \cup P_0^{\text{mid}}$;
6  end
7  Evaluate the fitness value by Function 3;
8  $t \leftarrow 0$;
9  while $t < G_{\max}$ do
10     $Q_t^{\text{sub}} \leftarrow \emptyset$
11     while Size$\big(Q_t^{\text{sub}}\big) < N_{\text{pop}}^{\text{sub}}$ do
12         $p_1, p_2 \leftarrow$ Select two parents from $P_t^{\text{sub}}$;
13         $q_1, q_2 \leftarrow$ Generate two offspring by $p_{\text{c}}$ and $p_{\text{m}}$;
14         $q_1^{\text{sub}}, q_2^{\text{sub}} \leftarrow$ Select the subset by Function 2;
15         if $q_i^{\text{sub}} \notin \emptyset$ then (where $i = 1$ and $i = 2$)
16             $Q_t^{\text{sub}} \leftarrow Q_t^{\text{sub}} \cup q_i$ ;
17         end
18     end
19     Evaluate the fitness value by Function 3;
20     $P_{t+1}^{\text{sub}} \leftarrow$ Select $N_{\text{pop}}^{\text{sub}}$ individual from $P_t^{\text{sub}} \cup Q_t^{\text{sub}}$;
21     $t \leftarrow t + 1$;
22 end;
23 Select the best individual and decode it to the corresponding SUNet.
```

For the convenience of the reader, a toy example of the scalability of the single layer SUNet is shown in Fig.2. Firstly, when training on the small-sized wireless networks, the dimension of the input 3D real channel tensor can be denoted as $Q \times I \times MN$. By applying the EAS algorithm to search the single layer SUNet architectures, the searched architectures are $k_1^w \times k_1^h = 3 \times 5$, $s_1^w \times s_1^h = 1 \times 1$, $p_1^w \times p_1^h = 1 \times 2$, $c_1 = 2M$, respectively. Accordingly, the width and height of the output 3D real beamforming tensor are $\left(\frac{Q+2\times1-3}{1}+1\right) \times \left(\frac{I+2\times2-5}{1}+1\right) = Q \times I$. In addition, due to $c_1 = 2M$, the third dimension of the output 3D real beamforming tensor is $2M$. That is, the dimension of the output 3D real beamforming tensor is $Q \times I \times 2M$. Subsequently, when scaling to the large-sized wireless networks, the number of users and BSs usually increases, i.e., the dimension of the input 3D real channel tensor is changed from $Q \times I \times MN$ to $(Q + \Delta Q) \times (I + \Delta I) \times MN$. Without re-optimizing the parameters, the architectures are still $k_1^w \times k_1^h = 3 \times 5$, $s_1^w \times s_1^h = 1 \times 1$, $p_1^w \times p_1^h = 1 \times 2$, $c_1 = 2M$, respectively. Consequently, the width and height of the output 3D real beamforming tensor are changed into $\left(\frac{Q+\Delta Q+2\times1-3}{1}+1\right) \times \left(\frac{I+\Delta I+2\times2-5}{1}+1\right) = (Q + \Delta Q) \times (I + \Delta I)$. Similarly, the dimension of the output 3D real beamforming tensor is changed from $Q \times I \times 2M$ to $(Q+\Delta Q) \times (I+\Delta I) \times 2M$. Note that the multi-layer SUNet still has this scalability, since the judgement condition for the subpopulation selection strategy of Function 2 in the EAS algorithm is that the architectures of all layers of the SUNet satisfy *Remarks* 1 or 2 and $c_L = 2M$.

D. Generality of EAS Algorithm

The core elements of the EAS algorithm are the subpopulation selection strategy (Function 2) and parameter mapping

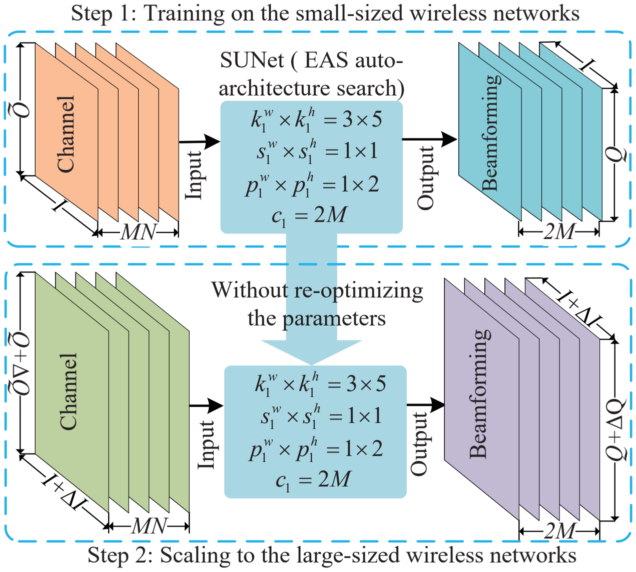
Step 1: Training on the small-sized wireless networks
Channel
Q
I
MN
Input
SUNet (EAS auto-architecture search)
$k_1^w \times k_1^h = 3\times5$
$s_1^w \times s_1^h = 1\times1$
$p_1^w \times p_1^h = 1\times2$
$c_1 = 2M$
Output
Beamforming
I
Q
$2M$
Without re-optimizing the parameters
Channel
$Q+\Delta Q$
$I+\Delta I$
MN
Input
$k_1^w \times k_1^h = 3\times5$
$s_1^w \times s_1^h = 1\times1$
$p_1^w \times p_1^h = 1\times2$
$c_1 = 2M$
Output
Beamforming
$I+\Delta I$
$Q+\Delta Q$
$2M$
Step 2: Scaling to the large-sized wireless networks

Fig. 2: A toy example of the single layer SUNet's scalability.

strategy (Function 3). To be more specific, the subpopulation selection strategy is designed to implement SBD in cell-free systems, which is difficult to apply to search the architectures of various other deep learning frameworks. However, note that the subpopulation selection strategy is represented as Function 2. To apply the EAS algorithm to search the architectures of various other deep learning frameworks, it is sufficient that Function 2 is not called in Algorithm 1 (i.e., for Algorithm 1, the 4th line becomes $P_0^{\text{mid}} \leftarrow P_0$, and the 14th line becomes $q_1^{\text{sub}}, q_2^{\text{sub}} \leftarrow q_1, q_2$). On the other hand, the parameter mapping strategy is designed based on the crossover and mutation operations of the evolutionary computation. Obviously, in addition to searching the architectures of the proposed SUNet for SBD in cell-free systems, the proposed parameter mapping strategy can be applied easily to search the architectures of various other deep learning frameworks. In summary, the EAS algorithm has good generality.

V. Numerical Results

In this section, numerical results are presented for the EAS algorithm and SUNet in cell-free systems. We first introduce the experimental environment and system parameters. Next, the performance of the EAS algorithm is evaluated. Finally, the scalable performance of the SUNet is evaluated.

A. Experimental Setup

The channel model is selected as the geographic location channel model [5], which is widely utilized in the beamforming design. Unless otherwise specified, the number of BSs and users are set to 16, and the number of BS and user antennas are set to 4 and 2, respectively. The SUNet and EAS algorithm are implemented by the PyTorch. For the SUNet, the optimizer selects the Adam optimizer, where the batch size, the learning rate and the number of layers are set to 64, 0.1, and 5, respectively. For the EAS algorithm, the population size N_{pop}, the subpopulation size $N_{\text{pop}}^{\text{sub}}$, the maximum number of iterations $G_{\max}$, the crossover probability p_{c} and the mutation probability p_{m} are set to 50, 25, 100, 0.8, and 0.005, respectively. Besides, the training data D_{train} and validation data D_{val} are set to 6400 and 640, respectively. [7]

B. Performance Evaluation of EAS algorithm

The performance of the parameter mapping strategy in Function 3 is first evaluated. The range of the convolutional kernel size $k_l^w \times k_l^h$ is from 1×1 to $Q \times I = 16 \times 16$, and the range of the number of the convolutional kernels c_l is from 1 to $2M = 8$. In order not to lose generality, the architectures of the trained teacher model SUNet$_{\text{tea}}$ are selected as the middle value, i.e., $k_l^w \times k_l^h = 7 \times 7$, $c_l = 4$, $s_l^w \times s_l^h = 1 \times 1$, $p_l^w \times p_l^h = 3 \times 3$. When the architectures and its corresponding parameters of SUNet$_{\text{tea}}$ are mapped to others, this usually includes four cases: increase the number of convolution kernels c_l, reduce the number of convolution kernels c_l, increase the size of convolution kernel $k_l^w \times k_l^h$, reduce the size of convolution kernel $k_l^w \times k_l^h$. To verify the performance of the parameter mapping strategy in these four cases, the comparison results of the SUNet based on the Xavier initialization and the parameter mapping strategy initialization are shown in Table II.

The Mapping in Table II represents using the parameter mapping strategy as initial parameters. The Xavier in Table II represents using the Xavier initializer [47] to generate initial parameters, where the Xavier initializer is a common initialization method for deep learning algorithms. By comparing the results of the SUNet in the Xavier initialization and the parameter mapping strategy initialization, the number of iterations H of the parameter mapping strategy initialization is much smaller than that of the Xavier initialization, when the fitness value reaches an approximate value. Thus, the parameter mapping strategy effectively reduces the number of iterations H in the training process of the SUNet. In addition, based on Table II, the number of iterations H in Function 3 is selected to 40.

The evolutionary trajectories of the EAS algorithm are shown in Fig.3. With the increase of iteration numbers, the average fitness value of the EAS algorithm gradually increases from 230 to about 270. When the number of iterations is about 50, the EAS algorithm converges.

The result of the SUNet being searched for the architectures by the traditional NAS [44] is shown in Fig.4. Note that if the dimension of the input 3D real channel tensor $\mathbf{H}_{\text{mod}}^{\text{3D}}$ is $Q \times I \times MN$, then the output beamforming should be a 3D real tensor with dimension $Q \times I \times 2M$ for SBD in cell-free systems. However, by applying the traditional NAS [44] to search for the architectures of the SUNet, the dimension of the output 3D real beamforming tensor is $Q' \times I' \times M'$ when the 3D real channel tensor $\mathbf{H}_{\text{mod}}^{\text{3D}}$ with dimension $Q \times$

[7] It is well known that a few samples and complex models with many parameters usually increase the overfitting probability of neural networks [45], [46]. However, the SUNet has a simple network that consists of only the CL, BN and AL, where the number of parameters is also smaller due to the weight sharing mechanism of the CL. Moreover, this simple network SUNet is also sufficiently trained with 6400 samples. On the other hand, the SUNet adds a BN layer after each CL to reduce the overfitting probability.

TABLE II: Comparison results of the Xavier initialization and parameter mapping strategy initialization.

	Increase the number of convolution kernels c_l		Reduce the number of convolution kernels c_l		Increase convolution kernel size $k_l^w \times k_l^h$		Reduce convolution kernel size $k_l^w \times k_l^h$	
	Mapping	Xavier	Mapping	Xavier	Mapping	Xavier	Mapping	Xavier
The number of iterations H	35	63	37	66	32	59	38	70
Fitness value	235.21	235.42	232.26	232.38	235.78	235.94	230.35	230.49

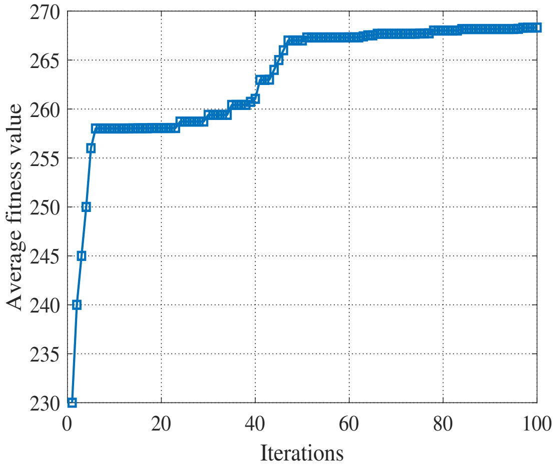

Fig. 3: Evolutionary trajectories of the EAS algorithm.

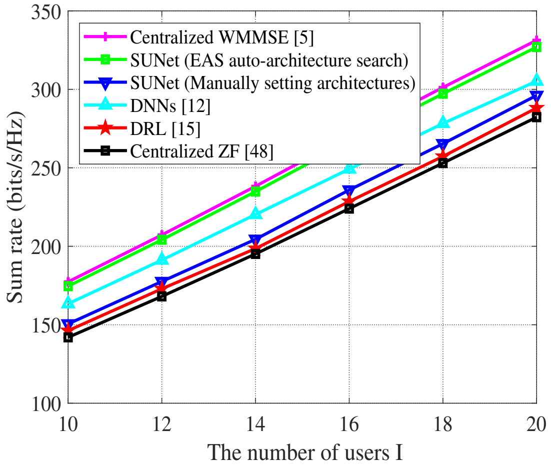

Fig. 5: Sum rate at different number of users.

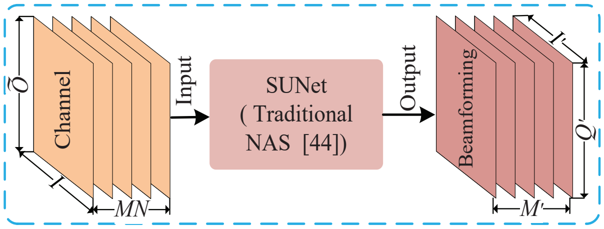

Fig. 4: Result of traditional NAS [44] for the SUNet.

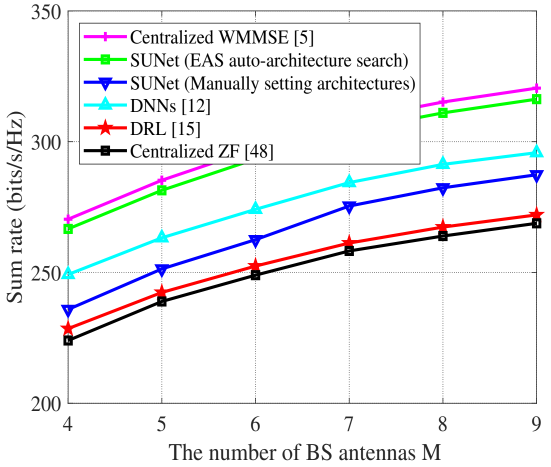

Fig. 6: Sum rate at different number of BS antennas.

$I \times MN$ is inputted, where Q', I' and M' are not equal to Q, I and $2M$, respectively. In other words, the architectures of the SUNet for applying the traditional NAS [44] to search do not satisfy the requirement of SBD in cell-free systems. This is because the traditional NAS [44] does not consider the dimension changing relationship between the input channel and the output beamforming. On the contrary, the proposed EAS algorithm uses the subpopulation selection strategy to take into account the dimension varying relationship between the input channel and output beamforming, which guarantees that the dimension of the output 3D real beamforming tensor must be $Q \times I \times 2M$ when the dimension of the input 3D real channel tensor $\mathbf{H}_{\text{mod}}^{\text{3D}}$ is $Q \times I \times MN$.

The sum rate performance at different number of users I and BS antennas M is shown in Fig.5 and Fig.6, respectively. Since the centralized WMMSE algorithm [5] is a stable solution for the beamforming design, its sum rate can be taken as an upper bound. Under the same conditions, the sum rate of the SUNet (EAS auto-architecture search) is higher than those of the SUNet (Manually setting architectures), DNNs [12], DRL [15] and centralized ZF [48], approaching this upper bound. This is because the SUNet (EAS auto-architecture search) is automatically searched for the desired architectures by the EAS algorithm. By contrast, the SUNet (Manually setting architectures), DNNs [12] and DRL [15] set up the architectures manually, which makes them difficult to find the desired architectures, resulting in lower sum rate performance. The centralized ZF [48] is a simple linear beamforming method that does not take noise into account, which leads to lower sum rate performance. On the other hand, although the sum rate of the centralized WMMSE algorithm [5] is slightly higher than that of the SUNet (EAS auto-architecture search), the centralized WMMSE algorithm [5] cannot be scalable because it re-optimizes the parameters when scaling from the small- to large-sized wireless networks. On the contrary, the SUNet (EAS auto-architecture search) is scalable from the small- to large-sized wireless networks without re-optimizing the parameters, which significantly improves the computational efficiency of the beamforming design.

C. Scalable Performance Evaluation of SUNet

In this subsection, the scalable performance of the SUNet is evaluated, where the architectures of the SUNet are automatically searched by the EAS algorithm.

1) Scaling to large area scenarios: When scaling to large area scenarios, the results of the SUNet in the scalable and traditional modes are shown in Table III. For the scalable mode, the SUNet is first trained on the scenario with 16×16 densities and 500×500 size areas, then the number of users and BSs are fixed, and finally the SUNet is not retrained and directly scaled to other scenarios with different size areas from 2 to 5 times. For the traditional mode, the SUNet is retrained when the size areas change. As can be seen from Table III, the time consumed by these four different size area scenarios is similar, because the number of users and BSs have not changed. In the scalable mode, when the size areas are 5 times those of the training, the scalable sum rate of the SUNet reaches about 96% of the centralized WMMSE algorithm, but the consumed time is about 0.00063s. In the traditional mode, the sum rate of the SUNet reaches about 98% of the centralized WMMSE algorithm in different size area scenarios, but the spent time is about 642s. Although the sum rate of the scalable mode is slightly lower than that of the traditional mode, the scalable mode takes much less time than the traditional mode. This also shows that the scalable mode of the SUNet effectively improve the computational efficiency of the beamforming design.

2) Scaling to both high density and large area scenarios: When scaling to both high density and large area scenarios, the results of the SUNet in the scalable and traditional modes are shown in Table IV. For the scalable mode, the SUNet is trained on the scenario with 16×16 densities and 500×500 size areas, then the SUNet is not retrained and directly scaled to other scenarios with different densities and size areas from 2 to 5 times. For the traditional mode, the SUNet is retrained, when the size areas and densities are changed. As can be seen from Table IV, in the high density and large size area scenarios, the sum rate of the traditional mode of the SUNet reaches about 98% of the centralized WMMSE algorithm, but the consumed time is higher and unaffordable. However, in the scalable mode, when the size areas and the number of BSs and users are 5 times those of the training, the scalable sum rate of the SUNet reaches about 95% of the centralized WMMSE algorithm, and the consumed time is merely 0.01617s. This effectively solves the disadvantage of higher consumption time in the traditional mode. In addition, by comparing the time results of the traditional and scalable modes, the scalable mode of the SUNet is more advantageous to improve the computation efficiency of the beamforming design when the number of BSs and users are larger. That is, the scalable mode of the SUNet is an effective way to deal with the scenarios with a larger number of BSs and users.

TABLE III: The SUNet's results in scalable and traditional modes when scaling to other large area scenarios.

Mode	(Q, I)	Size (m^2)	$\frac{\text{Scalable sum rate}}{\text{Sum rate of WMMSE}}$ (%)	Time (s)
Scalable	(16,16)	1000×1000	98.13	0.000631
		1500×1500	97.56	0.000632
		2000×2000	97.26	0.000631
		2500×2500	96.67	0.000633
Traditional	(16,16)	1000×1000	98.81	642.43
		1500×1500	98.68	642.54
		2000×2000	98.52	642.62
		2500×2500	98.45	642.51

TABLE IV: The SUNet's results in scalable and traditional modes when scaling to other high density and large area scenarios.

Mode	(Q, I)	Size (m^2)	$\frac{\text{Scalable sum rate}}{\text{Sum rate of WMMSE}}$ (%)	Time (s)
Scalable	$(32, 32)$	1000×1000	97.73	0.00227
	$(48, 48)$	1500×1500	97.35	0.00524
	$(64, 64)$	2000×2000	96.69	0.01035
	$(80, 80)$	2500×2500	95.39	0.01617
Traditional	$(32, 32)$	1000×1000	98.76	3317.83
	$(48, 48)$	1500×1500	98.59	7436.02
	$(64, 64)$	2000×2000	98.35	18240.48
	$(80, 80)$	2500×2500	98.13	89365.34

VI. Conclusion

In this paper, the two architectural conditions of the SUNet are given based on the beamforming characteristics. As long as these two architectural conditions are met, the SUNet achieves SBD for cell-free systems. Meanwhile, the EAS algorithm applies the subpopulation selection strategy to make the searched architectures satisfy these two architectural conditions, which avoids the labor-intensive task of manually setting the architectures. Additionally, the EAS algorithm uses the parameter mapping strategy to reduce the number of iterations in the training process of the SUNet. Finally, the experimental results show that the scalable sum rate of the SUNet reaches 95% of the WMMSE algorithm, when the size areas as well as the number of BSs and users are 5 times of these in training. The experimental results also show that the scalable mode of the SUNet is an effective way to deal with the scenarios with a larger number of BSs and users.

References

[1] A. A. Nasir, H. D. Tuan, H. Q. Ngo, T. Q. Duong, and H. V. Poor, "Cell-free massive mimo in the short blocklength regime for urllc," *IEEE Transactions on Wireless Communications*, vol. 20, no. 9, pp. 5861–5871, Sep. 2021.

[2] H. A. Ammar, R. Adve, S. Shahbazpanahi, G. Boudreau, and K. V. Srinivas, "User-centric cell-free massive mimo networks: A survey of opportunities, challenges and solutions," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 1, pp. 611–652, Firstquarter 2021.

[3] G. Interdonato, H. Q. Ngo, and E. G. Larsson, "Enhanced normalized conjugate beamforming for cell-free massive mimo," *IEEE Transactions on Communications*, vol. 69, no. 5, pp. 2863–2877, May 2021.

[4] F. Khalid, “Hybrid beamforming for millimeter wave massive multiuser mimo systems using regularized channel diagonalization,” *IEEE Wireless Communications Letters*, vol. 8, no. 3, pp. 705–708, Jun. 2018.

[5] Q. Shi, M. Razaviyayn, and Z.-Q. Luo, “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.

[6] D. Wang, M. Tao, X. Zeng, and J. Liang, “Federated learning for precoding design in cell-free massive mimo systems,” *IEEE Open Journal of the Communications Society*, vol. 4, pp. 1567–1582, Jul. 2023.

[7] Z. Wang, J. Zhang, H. Q. Ngo, B. Ai, and M. Debbah, “Uplink precoding design for cell-free massive mimo with iteratively weighted mmse,” *IEEE Transactions on Communications*, vol. 71, no. 3, pp. 1646–1664, Mar. 2023.

[8] G. Chen, S. He, Z. An, Y. Huang, and L. Yang, “A deep learning method: Qos-aware joint ap clustering and beamforming design for cell-free networks,” *IEEE Transactions on Communications*, vol. 71, no. 12, pp. 7023–7038, 2023.

[9] J. Fu, P. Zhu, J. Li, Y. Wang, and X. You, “Beamforming design in short-packet transmission for urllc in cell-free massive mimo system,” *IEEE Systems Journal*, vol. 17, no. 3, pp. 4715–4724, Sep. 2023.

[10] L.-H. Shen, T.-W. Chang, K.-T. Feng, and P.-T. Huang, “Design and implementation for deep learning based adjustable beamforming training for millimeter wave communication systems,” *IEEE Transactions on Vehicular Technology*, vol. 70, no. 3, pp. 2413–2427, Mar. 2021.

[11] H. Huang, Y. Peng, J. Yang, W. Xia, and G. Gui, “Fast beamforming design via deep learning,” *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 1065–1069, Jan. 2020.

[12] 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.

[13] L.-H. Shen, T.-W. Chang, K.-T. Feng, and P.-T. Huang, “Design and implementation for deep learning based adjustable beamforming training for millimeter wave communication systems,” *IEEE Transactions on Vehicular Technology*, vol. 70, no. 3, pp. 2413–2427, Mar. 2021.

[14] H. Hojatian, J. Nadal, J.-F. Frigon, and F. Leduc-Primeau, “Decentralized beamforming for cell-free massive mimo with unsupervised learning,” *IEEE Communications Letters*, vol. 26, no. 5, pp. 1042–1046, May 2022.

[15] F. Fredj, Y. Al-Eryani, S. Maghsudi, M. Akrout, and E. Hossain, “Distributed beamforming techniques for cell-free wireless networks using deep reinforcement learning,” *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 2, pp. 1186–1201, Jun. 2022.

[16] X. Zhang, H. Zhao, J. Xiong, X. Liu, L. Zhou, and J. Wei, “Scalable power control/beamforming in heterogeneous wireless networks with graph neural networks,” in *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2021, pp. 01–06.

[17] E. Björnson and L. Sanguinetti, “Scalable cell-free massive mimo systems,” *IEEE Transactions on Communications*, vol. 68, no. 7, pp. 4247–4261, Jul. 2020.

[18] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, “Generalizing from a few examples: A survey on few-shot learning,” *ACM computing surveys (csur)*, vol. 53, no. 3, pp. 1–34, Jun. 2020.

[19] M. A. Jamal and G.-J. Qi, “Task agnostic meta-learning for few-shot learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 719–11 727.

[20] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, “A comprehensive survey on transfer learning,” *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021.

[21] X. Wang, G. Chen, G. Qian, P. Gao, X.-Y. Wei, Y. Wang, Y. Tian, and W. Gao, “Large-scale multi-modal pre-trained models: A comprehensive survey,” *Machine Intelligence Research*, vol. 20, pp. 447–482, Jun. 2023.

[22] S. Wan, H. Zhu, K. Kang, and H. Qian, “On the performance of fully-connected and sub-connected hybrid beamforming system,” *IEEE Transactions on Vehicular Technology*, vol. 70, no. 10, pp. 11 078–11 082, Oct. 2021.

[23] N. Jiang, F. Duan, H. Chen, W. Huang, and X. Liu, “Mafi: Gnn-based multiple aggregators and feature interactions network for fraud detection over heterogeneous graph,” *IEEE Transactions on Big Data*, vol. 8, no. 4, pp. 905–919, Aug. 2022.

[24] S. He, S. Xiong, Y. Ou, J. Zhang, J. Wang, Y. Huang, and Y. Zhang, “An overview on the application of graph neural networks in wireless networks,” *IEEE Open Journal of the Communications Society*, vol. 2, pp. 2547–2565, Nov. 2021.

[25] 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. 2020.

[26] Z.-H. Zhan, J.-Y. Li, and J. Zhang, “Evolutionary deep learning: A survey,” *Neurocomputing*, vol. 483, pp. 42–58, Apr. 2022.

[27] Y. Liu, Y. Sun, B. Xue, M. Zhang, G. G. Yen, and K. C. Tan, “A survey on evolutionary neural architecture search,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 2, pp. 550–570, Feb. 2023.

[28] B. Zoph and Q. V. Le, “Neural architecture search with reinforcement learning,” *arXiv preprint arXiv:1611.01578*, 2016.

[29] Y. Sun, B. Xue, M. Zhang, and G. G. Yen, “Evolving deep convolutional neural networks for image classification,” *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 2, pp. 394–407, Apr. 2019.

[30] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu, J. Tan, Q. V. Le, and A. Kurakin, “Large-scale evolution of image classifiers,” in *International Conference on Machine Learning*. PMLR, 2017, pp. 2902–2911.

[31] P. Gao, Z. Liu, P. Xiao, C. H. Foh, and J. Zhang, “Low-complexity block coordinate descend based multiuser detection for uplink grant-free noma,” *IEEE Transactions on Vehicular Technology*, vol. 71, no. 9, pp. 9532–9543, Sep. 2022.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, pp. 5998–6008, 2017.

[33] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, “Lstm: A search space odyssey,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

[34] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” *arXiv preprint arXiv:1412.3555*, 2014.

[35] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[36] Y. Zou and L. Cheng, “A transfer learning model for gesture recognition based on the deep features extracted by cnn,” *IEEE Transactions on Artificial Intelligence*, vol. 2, no. 5, pp. 447–458, Oct. 2021.

[37] M. Hasan, S. Das, and M. N. T. Akhand, “Estimating traffic density on roads using convolutional neural network with batch normalization,” in *2021 5th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT)*. IEEE, 2021, pp. 1–6.

[38] J. Zhang, C. Shen, H. Su, M. T. Arafin, and G. Qu, “Voltage over-scaling-based lightweight authentication for iot security,” *IEEE Transactions on Computers*, vol. 71, no. 2, pp. 323–336, Feb. 2021.

[39] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, “Cbam: Convolutional block attention module,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[40] H. Yi, S. Shiyu, D. Xiusheng, and C. Zhigang, “A study on deep neural networks framework,” in *2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, 2016, pp. 1519–1522.

[41] L. Pellaco, M. Bengtsson, and J. Jaldén, “Deep weighted mmse downlink beamforming,” in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 4915–4919.

[42] Stanford university-convolutional neural networks cheatsheet. [Online]. Available: https://stanford.edu/∼shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks.

[43] Medium - Convolutional Neural Networks Part 2: Padding and Strided Convolutions. [Online]. Available: https://medium.com/swlh/convolutional-neural-networks-part-2-padding-and-strided-convolutions-c63c25026eaa#:∼:text=Same%20convolution%20means%20when%20you,image%20size%20%3D%20output%20image%20size.

[44] Y. Sun, B. Xue, M. Zhang, and G. G. Yen, “Completely automated cnn architecture design based on blocks,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 4, pp. 1242–1254, Apr. 2020.

[45] X. Chen, L. Wu, M. He, L. Meng, and X. Meng, “Mlfont: Few-shot chinese font generation via deep meta-learning,” in *Proceedings of the 2021 International Conference on Multimedia Retrieval*, 2021, pp. 37–45.

[46] S. Huang, Y. Chen, D. Wu, G. Yu, and Y. Zhang, “Few-shot learning for human activity recognition based on csi,” in *2022 Asia Conference*

on Algorithms, Computing and Machine Learning (CACML). IEEE, 2022, pp. 403–409.

[47] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[48] E. Nayebi, A. Ashikhmin, T. L. Marzetta, H. Yang, and B. D. Rao, "Precoding and power optimization in cell-free massive mimo systems," *IEEE Transactions on Wireless Communications*, vol. 16, no. 7, pp. 4445–4459, Jul. 2017.

Guanghui Chen (Graduate Student Member, IEEE) received the M.S. degree in the information and communication engineering from Chongqing University, Chongqing, China, in 2021. He is currently pursuing the Ph.D. degree in the information and communication engineering with the School of Information Science and Engineering, Southeast University. His research interests focus on intelligent wireless communications.

Zheng Wang (Senior Member, IEEE) received the B.S. degree in electronic and information engineering from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2009, and the M.S. degree in communications from University of Manchester, Manchester, U.K., in 2010. He received the Ph.D degree in communication engineering from Imperial College London, UK, in 2015.

Since 2021, he has been an Associate Professor in the School of Information and Engineering, Southeast University (SEU), Nanjing, China. From 2015 to 2016 he served as a Research Associate at Imperial College London, UK. From 2016 to 2017 he was an senior engineer with Radio Access Network R&D division, Huawei Technologies Co.. From 2017 to 2020 he was an Associate Professor at the College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China. His current research interests include massive MIMO systems, machine learning and data analytics over wireless networks, and lattice theory for wireless communications.

Yi Jia (Graduate Student Member, IEEE) received the B.Eng. degree in information engineering from the School of Information Science and Engineering, Southeast University, Nanjing, China, in 2019, where she is currently pursuing the Ph.D. degree in information and communication engineering with the School of Information Science and Engineering. Her research interests mainly focus on intelligent wireless communications.

Yongming Huang (Senior Member, IEEE) received the B.S. and M.S. degrees from Nanjing University, Nanjing, China, in 2000 and 2003, respectively, and the Ph.D. degree in electrical engineering from Southeast University, Nanjing, in 2007.

Since March 2007 he has been a faculty in the School of Information Science and Engineering, Southeast University, China, where he is currently a full professor. He has also been the Director of the Pervasive Communication Research Center, Purple Mountain Laboratories, since 2019. During 2008-2009, Dr. Huang visited the Signal Processing Lab, Royal Institute of Technology (KTH), Stockholm, Sweden. His current research interests include intelligent 5G/6G mobile communications and millimeter wave wireless communications. He has published over 200 peer-reviewed papers, hold over 80 invention patents. He submitted around 20 technical contributions to IEEE standards, and was awarded a certificate of appreciation for outstanding contribution to the development of IEEE standard 802.11aj. He served as an Associate Editor for the IEEE Transactions on Signal Processing and a Guest Editor for the IEEE Journal Selected Areas in Communications. He is currently an Editor-at-Large for the IEEE Open Journal of the Communications Society and an Associate Editor for the IEEE Wireless Communications Letters.

Luxi Yang (Senior Member, IEEE) received the M.S. and Ph.D. degrees in the electrical engineering from Southeast University, Nanjing, China, in 1990 and 1993, respectively. Since 1993, he has been with the Department of Radio Engineering, Southeast University, where he is currently a Full Professor of information systems and communications, and the Director of the Digital Signal Processing Division. He has authored or co-authored of two published books and more than 200 journal papers, and holds 50 patents. His current research interests include signal processing for wireless communications, MIMO communications, intelligent wireless communications, and statistical signal processing. He received the first and second class prizes of science and technology progress awards of the State Education Ministry of China in 1998, 2002, and 2014. He is currently a member of Signal Processing Committee of the Chinese Institute of Electronics.