A Deep Learning Framework for Optimization of MISO Downlink Beamforming

Wenchao Xia, *Student Member, IEEE,* Gan Zheng, *Senior Member, IEEE*, Yongxu Zhu, Jun Zhang, *Member, IEEE,* Jiangzhou Wang, *Fellow, IEEE*, and Athina P. Petropulu, *Fellow, IEEE*

Abstract

Beamforming is an effective means to improve the quality of the received signals in multiuser multiple-input-single-output (MISO) systems. This paper studies fast optimal downlink beamforming strategies by leveraging powerful deep learning techniques. Traditionally, finding the optimal beamforming solution relies on iterative algorithms which leads to high computational delay and is thus not suitable for real-time implementation. In this paper, we propose a deep learning framework for the optimization of downlink beamforming. In particular, the solution is obtained based on convolutional neural networks and exploitation of expert knowledge, such as the uplink-downlink duality and the known structure of optimal solutions. Using this framework, we construct three beamforming neural networks (BNNs) for three typical optimization problems, i.e., the signal-to-interference-plus-noise ratio (SINR) balancing problem, the power minimization problem and the sum rate maximization problem. The BNNs for the former two problems adopt the supervised learning approach, while the BNN for the sum rate maximization problem employs a hybrid method of supervised and unsupervised learning to improve the performance. Simulation results show that with much reduced computational complexity, the BNNs can achieve near-optimal solutions to the SINR balancing and power minimization problems, and can achieve a performance close to that of the weighted minimum mean squared error algorithm for the sum rate maximization problem. In summary, this work paves the way for fast realization of optimal beamforming in multiuser MISO systems.

Index Terms

Deep learning, beamforming, MISO, beamforming neural network.

W. Xia and J. Zhang are with the Jiangsu Key Laboratory of Wireless Communications, Nanjing University of Posts and Telecommunications, Nanjing 210003, China (e-mail: 2015010203@njupt.edu.cn, zhangjun@njupt.edu.cn).

G. Zheng and Y. Zhu are with the Wolfson School of Mechanical, Electrical and Manufacturing Engineering, Loughborough University, Leicestershire, LE11 3TU, UK (e-mail: g.zheng@lboro.ac.uk, y.zhu4@lboro.ac.uk).

J. Wang is with the School of Engineering and Digital Arts at the University of Kent, Kent, CT2 7NT, UK (e-mail: j.z.wang@kent.ac.uk).

A. P. Petropulu is with the Department of Electrical & Computer Engineering Rutgers, The State University of New Jersey, Piscataway, NJ 08854 (e-mail: athinap@rutgers.edu).

Part of this work was accepted to IEEE Int. Conf. Commun. (ICC) [1], Shanghai, China, May, 2019.

I. INTRODUCTION

Downlink beamforming techniques have attracted much attention in the past decades for its ability to realize the performance gain of the multiple antennas. Beamforming has been formulated in various ways, i.e., as a signal-to-interference-plus-noise ratio (SINR) balancing problem (also known as interference balancing problem) under a total power constraint [2–4], as a power minimization problem under quality of service (QoS) constraints [5–8], or as a sum rate maximization problem under a total power constraint [2, 9–11]. Existing approaches to finding the optimal beamforming solutions heavily rely on tailor-made iterative algorithms and convex optimization, which is in turn solved by general iterative algorithms such as the interior point method. For instance, the SINR balancing problem can be solved by the iterative algorithm of [12]. The power minimization problem can be reformulated as a second-order cone programming (SOCP) [7, 8] or semidefinite programming (SDP) problem [13, 14], which can be solved directly by an optimization software package such as CVX [15]. Its optimal solution can also be obtained using iterative algorithms such as Algorithm A of [16] and the dual algorithm of [5, 12]. However, the optimal solution to the sum rate maximization problem is usually hard to obtain because the problem is nonconvex. Locally optimal solutions are obtained via iterative algorithms, such as the weighted minimum mean squared error (WMMSE) algorithm [9, 10], and asymptotically optimal solutions are obtained using the water filling algorithm combined with zero-forcing (ZF) beamforming [11].

The main drawbacks of existing iterative algorithms are the high computational complexity and the resulting latency. As a result, the beamforming technique is unable to meet the demands of real-time applications in the fifth-generation (5G) system and beyond, such as autonomous vehicles and mission critical communications. Even in non-real-time applications, where the small-scale fading varies in the order of milliseconds, the latency introduced by the iterative process renders the beamforming solution outdated. To address this challenge, researchers have proposed some simple heuristic beamforming solutions which admit closed-form solutions, such as the maximum-ratio transmission beamforming, the ZF beamforming, and the regularized ZF (RZF) beamforming. These heuristic beamforming solutions are directly computed based on the channel state information (CSI) without iteration, and thus involve low computational delay. However, the reduction of delay is achieved at the cost of performance loss. The tradeoff

between delay and performance seems to restrict the potential of the beamforming techniques and its applications in practice.

Thanks to the recent advances in deep learning (DL) techniques, it becomes possible to find the optimal beamforming in real time by taking into account both the performance and the computational delay simultaneously. This is because the DL technique trains neural networks offline and then deploys the trained neural networks for online optimization. The computational complexity is transferred from the online optimization to the offline training, and only simple linear and nonlinear operations are needed when the trained neural network is used to find the optimal beamforming solution, thus greatly reducing the computational complexity and delay.

Benefiting from the development of specialized hardware, such as graphic processing units and field programmable gate arrays, DL can be implemented using these hardware resources conveniently. Accordingly, DL techniques have been widely used in many applications including wireless communications. A lot of research has attempted to use DL to deal with some issues in the physical layer, including channel decoding [17, 18], detection [19–21], channel estimation [22–24], and resource management [25–32]. Among these efforts, the autoencoder based on unsupervised DL, investigated in [33, 34], is an ambitious attempt to learn an end-to-end communications system [35]. DL can also facilitate resource management [25, 26], e.g. power allocation [27–31]. Finally, [36, 37] provide an overview on the recent advances in DL-based physical layer communications and [38] suggests potential applications of DL to the physical layer.

However, with the exception of [39–42], there are no works focusing on the beamforming design in multi-antenna communications based on DL. A common method used in the already published papers is codebook-based beam selection. For example, [39] designed a decentralized robust precoding scheme based on DNN in a network MIMO configuration. The projection over a finite dimensional subspace in [39] reduced the difficulty, but also limited the performance. [40] used a DL model to predict the beamforming matrix directly from the signals received at the distributed BSs based on omni or quasi-omni beam patterns in millimeter wave systems, whose sum rate performance was restricted by the quantized codebook constraint. [39, 40] predicted the beamforming matrix in the finite solution space at the cost of performance loss. Different from [39, 40], [41, 42] directly estimated the beamforming matrix without exploiting the problem structure in which the number of variables to predict increases significantly as the numbers of

transmit antennas and users increase. This will lead to high training complexity of the neural networks when the numbers of transmit antennas and users are large. Furthermore, we notice that none of them addressed the SINR balancing problem under a total power constraint and power minimization problem under SINR constraints.

Motivated by the aforementioned facts and the universal approximation theorem [43, 44], we propose a general DL framework to achieve not only near-optimal beamforming matrix, but also reduce complexity and latency as compared to the iterative methods. Based on the proposed framework, we develop beamforming neural networks (BNNs) to solve the three aforementioned optimization problems. Learning the optimal beamforming solution is highly nontrivial, and there are still challenges that need to be addressed in designing the BNNs. Firstly, the popular neural network software packages such as Keras and Tensorflow currently (March 2019) do not support complex numbers as input or output [35]. Both channel and beamforming vectors are inherently complex. Naive transformation of complex beamforming vectors to real vectors by concatenating the real and imaginary parts and predicting the real beamforming vectors directly not only lead to high complexity of prediction, but also may lose the specific structures of the problems of interest. Secondly, the power minimization problem has strict QoS constraints and guaranteeing a feasible solution using neural networks is a challenge. In addition, different from the SINR balancing problem and power minimization problem, there is no practically useful algorithm that can achieve the optimal solution to the sum rate maximization problem (and other nonconvex beamforming problems), and thus the supervised learning method based on locally optimal solution cannot achieve good performance. In this paper, we will tackle these challenges, and our main contributions are summarized as follows:

- We provide a DL-based framework for the beamforming optimization in the multiple-input-single-output (MISO) downlink, where the BS has multiple antennas while each user terminal has a single antenna. The proposed framework is designed based on the CNN structure. Different from existing works where the CNN was applied to power control [29, 30], resource allocation [45], and wireless scheduling [46], the proposed framework combines the signal processing module with the neural network module and exploits expert knowledge such as the uplink-downlink duality and the known structure of optimal solutions, so as to improve learning efficiency by specifying the best parameters to be learned; those parameters are typically not the direct beamforming matrix. This framework can deal with

three types of beamforming optimization problems: 1) problems whose optimal solutions are easy to find and the constraints are easy to meet; 2) problems whose optimal solutions are easy to find but the constraints are hard to meet; and 3) problems which have no practically useful algorithm that can achieve optimal solutions efficiently. Under this framework, we propose three BNNs for solving three typical optimization problems in MISO systems, i.e., the SINR balancing problem under a total power constraint, the power minimization problem under QoS constraints, and the sum rate maximization problem under a total power constraint.

- In the proposed supervised BNNs for the SINR balancing problem and the power minimization problem, instead of estimating the beamforming matrix with NK elements, where N is the number of the transmit antennas at the BS and K is the number of users, we exploit the uplink-downlink duality of solutions [5, 6, 12] and predict the virtual uplink power allocation vector with only K elements. Thus, the demand on the prediction capability of the BNNs in terms of network neurons and layers is significantly reduced. Also, the training and prediction complexity and cost are reduced. In the proposed BNN for the sum rate maximization problem, we exploit the known structure of optimal solutions and predict two power allocation vectors with totally $2K$ elements. This approach still has advantages compared to predicting the beamforming matrix directly.
- We propose a hybrid two-stage BNN with both supervised and unsupervised learning to find the beamforming solution to the sum rate maximization problem [29], since no practically useful algorithm can find the global optimum. In the first stage, we use the supervised learning method with the mean squared error (MSE)-based loss function to make the predictions as close as possible to the WMMSE algorithm, which is known to achieve the locally optimal solution. In the second stage, we modify the metric in the loss function to be the sum rate, and update the network parameters according to the unsupervised learning method, which achieves a performance close to that of the WMMSE algorithm.

The remainder of this paper is organized as follows. Section II introduces the system model and formulates three beamforming optimization problems in the MISO downlink. Section III provides the framework for the beamforming optimization and then Sections IV, V and VI propose the BNNs under the framework for the SINR balancing problem, the power minimization problem,

and the sum rate maximization problem, respectively. Numerical results are presented in Section VII. Finally, conclusion is drawn in Section VIII.

Notations: The notations are given as follows. Matrices and vectors are denoted by bold capital and lowercase symbols, respectively. $(\mathbf{A})^T$ and $(\mathbf{A})^H$ stand for transpose and conjugate transpose of $\mathbf{A}$, respectively. The notations $||\bullet||_1$ and $||\bullet||_2$ are l_1 and l_2 norm operators, respectively. The operator diag$(\mathbf{a})$ denotes the operation to diagonalize the vector $\mathbf{a}$ into a matrix whose main diagonal elements are from $\mathbf{a}$. Finally, $\mathbf{a} \sim \mathcal{CN}(\mathbf{0}, \mathbf{\Sigma})$ represents a complex Gaussian vector with zero-mean and covariance matrix $\mathbf{\Sigma}$.

II. System Model

We consider a downlink transmission scenario where a BS equipped with N antennas serves K single-antenna users. The channel between user k and the BS is denoted as $\mathbf{h}_k \in \mathbb{C}^{N\times 1}$. The received signal at user k is given by

$$y_k = \mathbf{h}_k^H \sum_{k'=1}^{K} \mathbf{w}_{k'} x_{k'} + n_k, \tag{1}$$

where $\mathbf{w}_k$ represents the beamforming vector for user k, $x_k \sim \mathcal{CN}(0,1)$ is the transmitted symbol from the BS to user k, and $n_k \sim \mathcal{CN}(0,\sigma^2)$ denotes the additive Gaussian white noise (AWGN) with zero mean and variance σ^2. The received SINR of user k equals

$$\gamma_k^{dl} = \frac{|\mathbf{h}_k^H \mathbf{w}_k|^2}{\sum_{k'=1,k'\neq k}^{K} |\mathbf{h}_k^H \mathbf{w}_{k'}|^2 + \sigma^2}. \tag{2}$$

One conventional optimization problem seeks to maximize $\min_k \gamma_k^{dl}/\rho_k$ subject to a transmit power constraint, where ρ_k's are constant weights denoting the importance of the sub-streams. Such an optimization problem is referred to as interference or SINR balancing, and has been investigated in many works [2–4]. The SINR balancing problem is formulated as:

$$\textbf{P1:} \max_{\mathbf{W}} \min_{1\leq k\leq K} \quad \frac{\gamma_k^{dl}}{\rho_k}, \quad \text{s.t.} \sum_{k=1}^{K} ||\mathbf{w}_k||^2 \leq P_{max}, \tag{3}$$

where $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K]$ is a set of beamforming vectors and P_{max} is the power budget.

Another important problem is the power minimization problem under a set of SINR constraints [6, 7]. A network operator may be more interested in how to minimize the transmit power while fulfilling the demands for QoS, i.e.,

$$\textbf{P2:} \min_{\mathbf{W}} \quad \sum_{k=1}^{K} ||\mathbf{w}_k||^2, \quad \text{s.t. } \gamma_k^{dl} \geq \Gamma_k, \forall k, \tag{4}$$

where Γ_k is the SINR constraint of user k. For ease of reference, we define $\mathbf{\Gamma} = [\Gamma_1, \cdots, \Gamma_K]^T$ as the SINR constraint vector.

Finally, the weighted sum rate maximization problem under the power constraint is also an important issue that has attracted lots of attention [2, 9, 10], which can be formulated as:

$$\textbf{P3:} \max_{\mathbf{W}} \quad \sum_{k=1}^{K} \alpha_k \log_2(1+\gamma_k^{dl}), \quad \text{s.t. } \sum_{k=1}^{K} ||\mathbf{w}_k||^2 \leq P_{max}, \tag{5}$$

where α_k is a constant weight of user k.

We choose the above problems as representative examples to demonstrate the effectiveness of our proposed DL beamforming framework. The practical algorithms to find optimal solutions are available for **P1** [8, 12, 47] and **P2** [5, 7, 8, 12, 13], so supervised learning can be adopted. In this work, for simplicity, we assume the optimal solution to problem **P2** always exists and do not consider the infeasibility of QoS constraints. Under this assumption, **P2** still has the additional challenge of satisfying the strict QoS constraints. **P3** is a difficult nonconvex problem and is usually solved using the iterative WMMSE approach [9, 10], therefore supervised learning is insufficient and further improvement is needed. In the rest of the paper, we will show how the solutions to these three types of problems can be efficiently learned by the proposed DL-based beamforming framework.

III. A DL-based Framework for Beamforming Optimization

DL-based neural networks were initially designed for solving classification problems, but they can also achieve satisfactory performance in regression problems. For example, the DNN was used to predict transmit power [27, 28]. Existing works mainly take real data, such as channel gains and transmit power, as input and output, but channel and beamforming matrices are both complex. In addition, predicting the beamforming matrix with NK elements directly may lead to inaccurate and even under-fitting results. Obviously we can use wider or deeper neural networks with more neurons to improve the learning ability, but such a huge network will lead to high training and implementation complexities and cannot guarantee the learning performance. For example, too deep or wide neural networks can cause over-fitting.

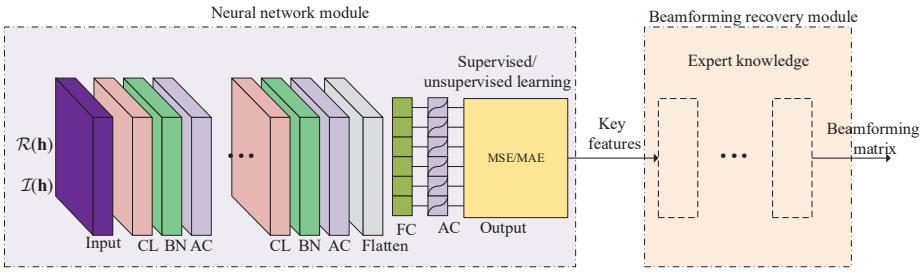


Fig. 1. A DL-based framework for the beamforming optimization in MISO downlink, which includes two main modules: the neural network module and the beamforming recovery module. The neural network module is composed of an input layer, convolutional (CL) layers, batch normalization (BN) layers, activation (AC) layers, a flatten layer, a fully-connected (FC) layer, and an output layer, whereas the key features and the functional layers in the beamforming recovery module are specified by the expert knowledge.

The proposed DL-based framework for the beamforming optimization in MISO downlink is shown in Fig. 1. We choose the CNN architecture as the base of the framework, because the CNN has strong ability of extracting features. In addition, the CNN can reduce the number of learned parameters by sharing weights and biases [30]. The CNN has a lot of applications in wireless networks, such as power control [29, 30], resource allocation [45], and wireless scheduling [46]. To overcome the challenge of predicting the beamforming matrix directly, we take the expert knowledge of the beamforming matrix into account. The proposed framework, instead of estimating the beamforming matrix directly, only predicts the key features extracted from the beamforming matrix according to the expert knowledge specific to the problem of interest. Therefore the demand for the prediction capability of the BNNs in terms of network neurons and layers, as well as the complexity, is significantly reduced.

A. *Structure of the Proposed Framework*

The proposed framework includes two main modules: the neural network module and beamforming recovery module. The neural network module is composed of an input layer, convolutional layers, batch normalization layers, activation layers, a flatten layer, a fully-connected layer, and an output layer, whereas key features and the functional layers in the beamforming recovery module are specified by the expert knowledge. For ease of clarification, we assume that, besides the input, output, flatten, and fully-connected layers, there are $L = |\mathcal{L}|$ groups of functional layers in the neural network module and each group includes a convolutional layer, a batch normalization layer, and an activation layer. Below we give a brief introduction to these

layers.

1) Input Layer: The complex channel coefficients are fed into the neural network module to predict the key features, which are not supported by the current neural network software. To deal with this issue, two data transformations are available. One is to separate the complex channel vector, for example $\mathbf{h} = [\mathbf{h}_1^T, \cdots, \mathbf{h}_K^T]^T \in \mathbb{C}^{NK\times 1}$, into the in-phase component $\Re(\mathbf{h})$ and quadrature component $\Im(\mathbf{h})$, where $\Re(\mathbf{h})$ and $\Im(\mathbf{h})$ contain the real and imaginary parts of each element in $\mathbf{h}$, respectively. We call this transformation **I/Q transformation**. Another transformation, suggested by [48], is to map the complex channel vector $\mathbf{h}$ into two real vectors $\mathfrak{P}(\mathbf{h}_k)$ and $\mathfrak{M}(\mathbf{h}_k)$, where the former contains the phase information and the latter includes the magnitude information of $\mathbf{h}$. This transformation is referred to as **P/M transformation**. As far as we know, there is no evidence to show which transformation is better. In this work, we adopt **I/Q transformation** of complex channels and formulate the input of the first convolutional layer as $[\Re(\mathbf{h}), \Im(\mathbf{h})]^T \in \mathbb{R}^{2\times NK}$. Note that the samples are fed into the neural network module in batches during the training process.

2) Convolutional Layer: Each convolutional layer $l \in \mathcal{L}$ creates c_l convolution kernels of size $a_l \times a_l$ that are convolved with the layer input $\boldsymbol{I}_{\text{conv},l} \in \mathbb{R}^{b_{l-1}^{(1)}\times b_{l-1}^{(2)}\times c_{l-1}}$, where $b_{l-1}^{(1)}$ and $b_{l-1}^{(2)}$ are the height and width of the output of the convolutional layer $l-1$, respectively. Note that $c_0 = 1$ $b_0^{(1)} = 2$, and $b_0^{(2)} = NK$. The parameters of the convolution kernels, including the weights $\boldsymbol{\Xi}_l \in \mathbb{R}^{a_l\times a_l\times c_l}$ and a bias vector $\boldsymbol{\xi}_l \in \mathbb{R}^{c_l\times 1}$, are shared among different elements in $\boldsymbol{I}_{\text{conv},l}$ to extract features. More specifically, the output $\boldsymbol{O}_{\text{conv},l} \in \mathbb{R}^{b_l^{(1)}\times b_l^{(2)}\times c_l}$ of the convolutional layer l is

$$\boldsymbol{O}_{\text{conv},l} = \mathsf{Conv}\left(\boldsymbol{I}_{\text{conv},l}, \boldsymbol{\Xi}_l, \boldsymbol{\xi}_l\right), l \in \mathcal{L}, \tag{6}$$

where the operator $\mathsf{Conv}(\cdot,\cdot,\cdot)$ denotes the convolution operation.

3) Batch Normalization Layer: The batch normalization layers are introduced in the neural network module, which can be put before or after the activation layers [49] according to practical experience. In the proposed framework, we adopt the former where the batch normalization layers normalize the output of the convolutional layers through subtracting the batch mean and dividing by the batch standard deviation, i.e.,

$$\boldsymbol{Z}_{\text{bn},l,c}[i,j] = \frac{\boldsymbol{O}_{\text{conv},l,c}[i,j] - \mu_{l,c}}{\sqrt{\mathsf{Var}_{l,c} + \epsilon_{l,c}}}, l \in \mathcal{L}, c = 1, \cdots, c_l, i = 1, \cdots, b_l^{(1)}, j = 1, \cdots, b_l^{(2)} \tag{7}$$

where $\boldsymbol{X}[i,j]$ denotes (i,j)-th element of matrix $\boldsymbol{X}$, $\boldsymbol{O}_{\text{conv},l,c} \in \mathbb{R}^{b_l^{(1)} \times b_l^{(2)}}$ is the c-th slice of $\boldsymbol{O}_{\text{conv},l}$, $\mu_{l,c} = \frac{\sum_{f=1}^{F}\sum_{i=1}^{b_l^{(1)}}\sum_{j=1}^{b_l^{(2)}} \boldsymbol{O}_{\text{conv},l,c}^{(f)}[i,j]}{F b_l^{(1)} b_l^{(2)}}$ and $\text{Var}_{l,c} = \frac{\sum_{f=1}^{F}\sum_{i=1}^{b_l^{(1)}}\sum_{j=1}^{b_l^{(2)}} \left(\boldsymbol{O}_{\text{conv},l,c}^{(f)}[i,j] - \mu_{l,c}\right)^2}{F b_l^{(1)} b_l^{(2)}}$ are the batch mean and variance of the c-th slice, respectively, $\epsilon_{l,c}$ is a small float added to the variance to avoid dividing by zero, and F is the batch size. Note that such a simple normalization process may change what the layer can represent. To address this issue, two trainable parameters $\theta_{l,c}$ and $\beta_{l,c}$ are introduced to scale and shift the normalized value $\boldsymbol{Z}_{\text{bn},l,c}[i,j]$ as $\hat{\boldsymbol{Z}}_{\text{bn},l,c}[i,j] = \beta_{l,c}\boldsymbol{Z}_{\text{bn},l,c}[i,j] + \theta_{l,c}$. This "denormalization" process is allowed by changing only these two parameters, instead of changing all parameters which may lead to the instability of the neural network module. Besides, the work in [49] claimed that the batch normalization layer can reduce the probability of over-fitting, enable a higher learning rate, and make the neural network less sensitive to the initialization of weights. Note that the batch normalization layers are element-wise functions, such that they do not change their respective input shapes.

4) Activation Layer: Since the predicted variables are continuous and positive real numbers, it is suggested that the activation functions that can generate negative values, such as tanh and linear functions, should not be used in the last activation layer. The rectified linear unit (ReLU) and sigmoid functions are good choices for the last activation layer, which are given as

$$\mathsf{ReLU}(z) = \max(0, z) \text{ and } \mathsf{sigmoid}(z) = \frac{1}{1+e^{-z}}, \tag{8}$$

respectively. The most common choice for the intermediate activation layers is the ReLU function. Note that the functions performed in the activation layers are element-wise functions, such that their outputs have the same shapes of their inputs, respectively.

5) Flatten Layer, Fully-connected Layer, and Output Layer: The flatten layer is only used to change the shape of its input into a vector, for the fully-connected layer to interpret. The output $\mathbf{o}_{\text{fc}} \in \mathbb{R}^{m\times 1}$ of the fully-connected layer is

$$\mathbf{o}_{\text{fc}} = \boldsymbol{\Pi}\boldsymbol{i}_{\text{fc}} + \boldsymbol{\pi}, \tag{9}$$

where $\boldsymbol{i}_{\text{fc}} \in \mathbb{R}^{2NKc_L\times 1}$ is the input vector, $\boldsymbol{\Pi} \in \mathbb{R}^{m\times 2NKc_L}$ and $\boldsymbol{\pi} \in \mathbb{R}^{m\times 1}$ account for the weight matrix and bias vector, respectively, and m is the number of the neurons in the fully-connected layer. The main function of the output layer is to generate the predicted results after the neural network finishes training.

Note that apart from these functional layers, the loss function also plays an important role in the proposed framework, which is marked on the output layer in Fig. 1. The loss function

together with the learning rate guides the learning process of the neural network. In other words, the loss function "tells" the neural network how to update its parameters. Since the output values are continuous, it is suggested to utilize the mean absolute error (MAE) or the MSE as a metric. Given the predicted results of the f-th sample in the neural network module is $\hat{\mathbf{q}}^{(f)}$ and the target result is $\mathbf{q}^{(f)}$, the MAE and MSE are defined as

$$\text{MAE} = \frac{1}{FK}\sum_{f=1}^{F}||\mathbf{q}^{(f)} - \hat{\mathbf{q}}^{(f)}||_1 \text{ and } \text{MSE} = \frac{1}{FK}\sum_{f=1}^{F}||\mathbf{q}^{(f)} - \hat{\mathbf{q}}^{(f)}||_2^2, \tag{10}$$

respectively. Generally speaking, the MAE function is more robust and is not affected by outliers. On the contrary, the MSE loss function is highly sensitive to outliers in the dataset because the MSE function tries to adjust the model according to these outlier values, at the expense of other samples [50]. In this work, the training dataset is generated by simulations and outliers are not an issue. Then we choose the MSE as the loss metric because its gradient is easier to calculate than that of the MAE.

6) Beamforming Recovery Module: The beamforming recovery module is an important component whose aim is to recover the beamforming matrix from the predicted key features at the output layer. The functional layers in the beamforming recovery module are designed according to the expert knowledge of the beamforming optimization which maps/converts the key features to the beamforming matrix. The expert knowledge is problem-dependent and has no unified form, but what is in common is that the expert knowledge can significantly reduce the number of variables to be predicted compared to the beamforming matrix. For example, the uplink-downlink duality and specific solution structures are the typical expert knowledge for beamforming optimization.

The key features should be chosen carefully to meet some constraints required by applying the universal approximation theorem [27, 43], so that a feedforward network exists which can approximate the continuous mapping from the channel coefficients to the key features. More specifically, assume that $\boldsymbol{\tau}$ is a vector containing the chosen key features, the mapping function $f(\bullet)$ from $\mathbf{h}$ to $\boldsymbol{\tau}$, i.e., $\boldsymbol{\tau} = f(\mathbf{h})$, should be a real-valued continuous function over a compact set. The compact set requirement holds whenever the possible values of the input $\mathbf{h}$ are bounded. However, the continuity of the mapping function depends on the choice of the key features.

In next three sections we will propose three BNNs under the proposed framework for problems **P1**, **P2**, and **P3**, respectively, and provide implementation details to show how to make use of

the expert knowledge and choose the key features.

B. Computational Complexity

The computational complexity of the proposed framework involves two main tasks: the online prediction and the offline training. To the best of our knowledge, complexity analysis of the offline training is still an open issue mainly because of the complex implementation of the backpropagation process. However, since the training is performed offline, and updated at a much longer time-scale compared to the online prediction, we assume its complexity can be afforded [51]. Thus, we focus on the complexity of the online prediction. In addition, the functional layers are problem-dependent in the beamforming recovery module, so only the complexity of the neural network module is analyzed below.

Given there are c_l kernels of size $a_l \times a_l$ in the l-th convolutional layer, then the numbers of multiplication and addition operations of convolutional layer l are the same and equal to $a_l^2 b_l^{(1)} b_l^{(2)} c_{l-1} c_l$. Thus, the total time complexity of all convolutional layers measured by the number of multiplications is $\mathcal{O}\left(\sum_{l\in\mathcal{L}} a_l^2 b_l^{(1)} b_l^{(2)} c_{l-1} c_l\right)$ [52]. It is known that the batch normalization layers and activation layers are element-wise functions, thus the computational complexity of total batch normalization layers and total activation layers in L groups is $\mathcal{O}\left(\sum_{l\in\mathcal{L}} b_l^{(1)} b_l^{(2)} c_l\right)$. The numbers of multiplication and addition operations of the fully-connected layer are also the same and equal to $b_L^{(1)} b_L^{(2)} c_L m$, respectively. Then the time complexity of the fully-connected layer is given as $\mathcal{O}\left(b_L^{(1)} b_L^{(2)} c_L m\right)$. Besides, the complexity of the input, output, and flatten layers are ignored due to the simplicity of their functions. If all convolutional layers use the kernels of size 3×3 and apply stride 1 and zero padding 1, then $b_l^{(1)} = 2$ and $b_l^{(2)} = NK, \forall l \in \mathcal{L}$. Based on the above analysis and assuming the parameters of the neural network module are fixed, predicting the output of the neural network module needs $2NK \sum_{l\in\mathcal{L}} (9 c_l c_{l-1} + c_l) + 2NK c_L m + 2m$ arithmetic operations including multiplications, divisions, and exponentiations, and has an approximate complexity $\mathcal{O}(NK)$.

IV. BNN for SINR Balancing Problem

As mentioned above, estimating the beamforming matrix directly leads to the higher complexity of prediction due to the large amount of variables. In order to reduce the prediction complexity, we introduce a scheme which first predicts the power allocation vector as the key

feature and then achieves the corresponding beamforming matrix based on the predicted results. Such a scheme is based on the expert knowledge named the uplink-downlink duality.

A. Uplink-Downlink Duality

Before we present the BNN for the SINR balancing problem **P1**, we first introduce the following lemma to describe the uplink-downlink duality of problem **P1** [12].

Lemma 1. *Given* $\tilde{\mathbf{W}} = [\tilde{\mathbf{w}}_1, \tilde{\mathbf{w}}_2, \ldots, \tilde{\mathbf{w}}_K]$ *and* P_{max}*, we have*

$$C^{dl}(\tilde{\mathbf{W}}, P_{max}) = C^{ul}(\tilde{\mathbf{W}}, P_{max}), \tag{11}$$

where $C^{dl}(\tilde{\mathbf{W}}, P_{max})$ *and* $C^{ul}(\tilde{\mathbf{W}}, P_{max})$ *are given as*

$$\begin{aligned} C^{dl}(\tilde{\mathbf{W}}, P_{max}) = &\max_{\mathbf{p}} \min_{1\le k\le K} \frac{\gamma_k^{dl}(\tilde{\mathbf{W}}, \mathbf{p})}{\rho_k} \\ &\textit{s.t.}\ ||\mathbf{p}||_1 \le P_{max}, \\ &||\tilde{\mathbf{w}}_k||_2 = 1, \forall k, \end{aligned} \tag{12}$$

and

$$\begin{aligned} C^{ul}(\tilde{\mathbf{W}}, P_{max}) = &\max_{\mathbf{q}} \min_{1\le k\le K} \frac{\gamma_k^{ul}(\tilde{\mathbf{W}}, \mathbf{q})}{\rho_k} \\ &\textit{s.t.}\ ||\mathbf{q}||_1 \le P_{max}, \\ &||\tilde{\mathbf{w}}_k||_2 = 1, \forall k, \end{aligned} \tag{13}$$

respectively, with

$$\gamma_k^{dl}(\tilde{\mathbf{W}}, \mathbf{p}) = \frac{p_k |\mathbf{h}_k^H \tilde{\mathbf{w}}_k|^2}{\sum_{k'=1, k'\neq k}^{K} p_{k'} |\mathbf{h}_k^H \tilde{\mathbf{w}}_{k'}|^2 + \sigma^2}, \tag{14}$$

and

$$\gamma_k^{ul}(\tilde{\mathbf{W}}, \mathbf{q}) = \frac{q_k |\mathbf{h}_k^H \tilde{\mathbf{w}}_k|^2}{\sum_{k'=1, k'\neq k}^{K} q_{k'} |\mathbf{h}_{k'}^H \tilde{\mathbf{w}}_k|^2 + \sigma^2}. \tag{15}$$

Note that $\mathbf{p} = [p_1, \ldots, p_K]^T$ *and* $\mathbf{q} = [q_1, \ldots, q_K]^T$ *are downlink and uplink power vectors, respectively*[1].

Note that problem (12) is an equivalent virtual problem of problem **P1** whose optimal solutions are connected by $\mathbf{W}^* = \tilde{\mathbf{W}}^* \mathbf{P}^*$ where $\mathbf{P}^* = \text{diag}(\mathbf{p}^*)$, $\mathbf{W}^*$ is the optimal solution to problem

[1]Lemma 1 can be easily extended to the case with non-identical noise power levels. More details can refer to [12].

P1, and $\tilde{\mathbf{W}}^*$ and $\mathbf{p}^*$ are the optimal solutions to problem (12). Based on **Lemma 1**, we find that the uplink and downlink scenarios have the same achievable SINR region and the normalized beamforming designed for the uplink reception immediately carries over to the downlink transmission [12]. Thus we first obtain the optimal power allocation $\mathbf{q}^*$ and beamforming matrix $\tilde{\mathbf{W}}^*$ for the easier-to-solve uplink problem (13) instead of the downlink problem (12). Then given the optimal beamforming $\tilde{\mathbf{W}}^*$, the optimal $\mathbf{p}^*$ is obtained as the first K components of the dominant eigenvector of the following matrix [53]

$$\boldsymbol{\Upsilon}(\tilde{\mathbf{W}}^*, P_{max}) = \begin{bmatrix} \mathbf{DU} & \mathbf{D}\boldsymbol{\sigma} \\ \frac{1}{P_{max}}\mathbf{1}^T\mathbf{DU} & \frac{1}{P_{max}}\mathbf{1}^T\mathbf{D}\boldsymbol{\sigma} \end{bmatrix}, \tag{16}$$

where $\boldsymbol{\sigma} = \sigma^2\mathbf{1}$, $\mathbf{1} = [1, 1, \ldots, 1]^T \in \mathbb{R}^{K\times 1}$, $\mathbf{D} = \text{diag}\{\rho_1/|(\tilde{\mathbf{w}}_1^*)^H\mathbf{h}_1|^2, \ldots, \rho_K/|(\tilde{\mathbf{w}}_K^*)^H\mathbf{h}_K|^2\}$, and

$$[\mathbf{U}]_{kk'} = \begin{cases} |(\tilde{\mathbf{w}}_{k'}^*)^H\mathbf{h}_k|^2, & \text{if } k' \neq k, \\ 0, & \text{else.} \end{cases} \tag{17}$$

Finally, the downlink beamforming matrix is derived as $\mathbf{W}^* = \tilde{\mathbf{W}}^*\mathbf{P}^*$. Thus, instead of predicting $\mathbf{W}$ directly, we can predict the uplink power allocation vector $\mathbf{q}$. In the supervised learning method, the prediction performance of the BNN depends on the quality of training samples. To generate the training samples, the optimal $\mathbf{q}^*$ and $\tilde{\mathbf{W}}^*$ can be found by an iterative optimization algorithm in [12, Table 1].

Note that $\boldsymbol{\Upsilon}(\tilde{\mathbf{W}}^*, P_{max})$ is a non-negative matrix and the optimal objective value of problem **P1** is the reciprocal of the largest eigenvalue of $\boldsymbol{\Upsilon}(\tilde{\mathbf{W}}^*, P_{max})$ [53]. According to the Perron-Frobenius theory, for any nonnegative real matrix $\boldsymbol{\Omega}$ with spectral radius $\chi(\boldsymbol{\Omega})$, there exist a vector $\boldsymbol{\delta} \geq 0$ such that $\boldsymbol{\Omega}\boldsymbol{\delta} = \chi(\boldsymbol{\Omega})\boldsymbol{\delta}$ [54]. Based on [12, Theorem 3], the sequence of the target value of problem **P1** provided by the iterative algorithm in [12, Table 1] is strictly monotonically increasing and the largest eigenvalue of $\boldsymbol{\Upsilon}(\tilde{\mathbf{W}}^*, P_{max})$ is unique. Then the corresponding eigenvector containing $\mathbf{q}$ is a continuous and bounded function of $\mathbf{h}$ according to [55, Chapter 3]. Thus, we can use a neural network to approximate the mapping function from $\mathbf{h}$ to $\mathbf{q}$ [43].

B. BNN Structure

The proposed BNN for problem **P1**, shown in Fig. 2, is based on the proposed BNN framework in Fig. 1. The functions and operations of the basic layers such as the input, convolutional, batch normalization, and output layers, are the same as those in the proposed framework. Therefore,

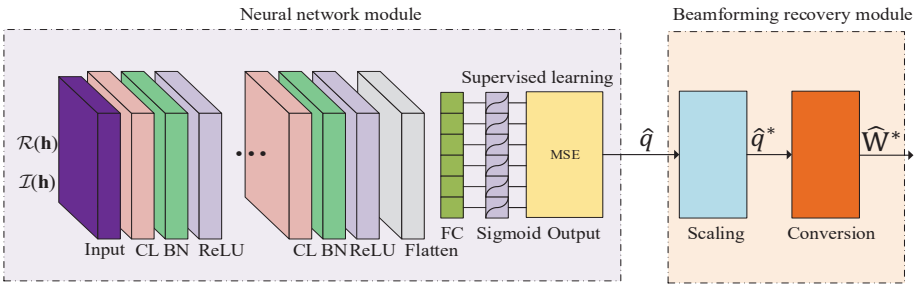


Fig. 2. BNN for the SINR balancing problem.

we do not explain these layers here and readers can refer to Section III for detail. Note that in the proposed BNN for problem **P1**, the intermediate activation layers are fulfilled with the ReLU function whereas the last activation layer is implemented using the sigmoid function. Besides the existing layers in the framework, a scaling layer and a conversion layer are also introduced in the BNN for problem **P1**, which belong to the beamforming recovery module. In the following, we give the details of the scaling layer and the conversion layer.

1) Scaling Layer: Due to the existence of prediction error, it is almost impossible to guarantee that the output of the output layer always meets the power constraint in problem **P1**. According to [56], the optimal solution is achieved when the equality of the constraint in problem **P1** holds. Therefore, we scale the results of the output layer $\hat{\mathbf{q}}$ to meet the power constraint by the following transformation,

$$\hat{\mathbf{q}}^* = \frac{P_{max}}{||\hat{\mathbf{q}}||_1}\hat{\mathbf{q}}. \tag{18}$$

2) Conversion Layer: After receiving the scaled power allocation vector $\hat{\mathbf{q}}^*$, we can achieve the downlink beamforming matrix $\hat{\mathbf{W}}^*$ as the final output of the BNN based on $\hat{\mathbf{q}}^*$ by the conversion layer. The beamforming recovery implemented by the conversion layer includes the following process:

1) Calculate $\mathbf{T}^* = \sigma^2\mathbf{I}_N + \sum_{k=1}^{K}\hat{q}_k^*\mathbf{h}_k\mathbf{h}_k^H$.
2) Calculate $\tilde{\mathbf{w}}_k^* = \tilde{\mathbf{w}}_k^*/||\tilde{\mathbf{w}}_k^*||_2, \forall k$, where $\tilde{\mathbf{w}}_k^* = (\mathbf{T}^*)^{-1}\mathbf{h}_k$.
3) Find the maximal eigenvalue ψ_{max}^* of $\boldsymbol{\Upsilon}(\tilde{\mathbf{W}}^*, P_{max})$ and the associated eigenvector with respect to ψ_{max}^*, i.e., $\boldsymbol{\Upsilon}(\tilde{\mathbf{W}}^*, P_{max})\left[\begin{smallmatrix}\hat{\mathbf{p}}^*\\ 1\end{smallmatrix}\right] = \psi_{max}^{(i)}\left[\begin{smallmatrix}\hat{\mathbf{p}}^*\\ 1\end{smallmatrix}\right]$.
4) Output $\hat{\mathbf{W}}^* = \tilde{\mathbf{W}}^*\hat{\mathbf{P}}^*$ as the final result where $\hat{\mathbf{P}}^* = \text{diag}(\hat{\mathbf{p}}^*)$.

Note that the time complexity of the beamforming recovery module is $\mathcal{O}(KN^2 + N^3 + K^3)$. In the proposed BNN for the SINR balancing problem **P1**, the supervised learning with the loss

function based on the MSE metric is adopted.

V. BNN FOR POWER MINIMIZATION PROBLEM

Similar to the BNN for the SINR balancing problem **P1**, the BNN for the power minimization problem **P2** obtains the downlink beamforming matrix according to the uplink-downlink duality, i.e., the expert knowledge. Specifically, we first predict the uplink power allocation vector as the key features using the trained neural network, then obtain the normalized beamforming matrix based on the predicted results. Finally, the downlink beamforming matrix is recovered from the normalized beamforming matrix by the uplink-downlink conversion method.

A. Uplink-Downlink Duality

Note that the conversion method adopted in the BNN for problem **P1** can not be used again, because the power budget P_{max} is unknown in the power minimization problem **P2**. Instead, we employ the conversion method in the following lemma [47].

Lemma 2. *Given the optimal beamforming matrix* $\tilde{\mathbf{W}}^* = [\tilde{\mathbf{w}}_1^*, \ldots, \tilde{\mathbf{w}}_K^*]$ *for the uplink problem*[2]*, i.e.,*

$$\begin{aligned} &\min_{\mathbf{q}, \tilde{\mathbf{W}}} \sum_{k=1}^{K} q_k \\ \textit{s.t. } &\gamma_k^{ul}(\tilde{\mathbf{W}}, \mathbf{q}) \geq \Gamma_k, \\ &||\tilde{\mathbf{w}}_k||_2 = 1, \forall k, \end{aligned} \tag{19}$$

where $\gamma_k^{ul}(\tilde{\mathbf{W}}, \mathbf{q})$ *is given as in* (15)*.*

The optimal beamforming vectors $\mathbf{w}_k^*, \forall k$*, for the downlink problem* ***P2****, can be obtained by multiplying the optimal normalized beamforming vector* $\tilde{\mathbf{w}}_k^*$ *by a scaling factor, i.e.,* $\mathbf{w}_k^* = p_k^* \tilde{\mathbf{w}}_k^*, \forall k$*, where* p_k^* *is the* k*-th element of vector* $\mathbf{p}^* = [p_1^*, \ldots, p_K^*]^T \in \mathbb{R}^{K \times 1}$ *and*

$$\mathbf{p}^* = \sigma^2 \mathbf{\Psi}^{-1} \mathbf{1}, \tag{20}$$

[2]In this work, for simplicity, we assume the solution to problem **P2** always exists. However, it can happen that the wireless network only satisfies some of the users and thus the user selection is needed. To address this issue, a possible solution is to train another neural network for user selection, and then optimize the beamforming matrix among the selected users.

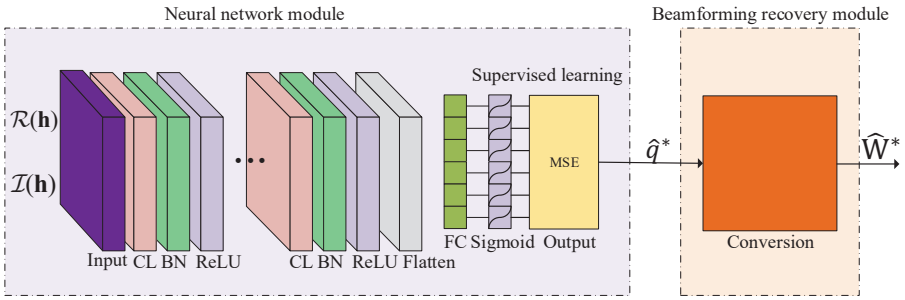


Fig. 3. BNN for the power minimization problem.

where

$$[\mathbf{\Psi}]_{kk'} = \begin{cases} \frac{1}{\Gamma_k}|\mathbf{h}_k^H \tilde{\mathbf{w}}_k^*|^2, \ \textit{if } k = k', \\ -|\mathbf{h}_k^H \tilde{\mathbf{w}}_{k'}^*|^2, \ \textit{else}. \end{cases} \tag{21}$$

The vector $\mathbf{p}^*$ of the scaling factors is the optimal downlink power allocation vector. Given the optimal normalized beamforming matrix $\tilde{\mathbf{W}}^*$, **Lemma 2** allows us to achieve the optimal downlink power vector $\mathbf{p}^*$ by (20), then $\mathbf{W}^* = \tilde{\mathbf{W}}^* \mathbf{P}^*$. Actually, if we know the uplink power allocation vector $\mathbf{q}$, the normalized beamforming matrix $\tilde{\mathbf{W}}$ can be inferred as

$$\tilde{\mathbf{w}}_k = \frac{\mathbf{T}^{-1}\mathbf{h}_k}{||\mathbf{T}^{-1}\mathbf{h}_k||_2}, \forall k, \tag{22}$$

where $\mathbf{T} = \sigma^2 \mathbf{I}_N + \sum_{k=1}^K q_k \mathbf{h}_k \mathbf{h}_k^H$. Therefore, the only results that need to be predicted by the BNN is the uplink power allocation vector $\mathbf{q}$, which reduces significantly the computational complexity compared to the strategy that attempts to predict the beamforming matrix directly. The iterative algorithm in [5] provides a way to achieve the optimal $\mathbf{q}^*$ as the training samples in the supervised learning method. Besides, such an iterative algorithm suggests the mapping function from $\mathbf{h}$ to $\mathbf{q}$ is continuous [27, Theorem 1], so it can be approximated by a neural network.

B. BNN Structure

The BNN for problem **P2** in Fig. 3 is also based on the proposed BNN framework. However, the operations of the conversion layer in Fig. 3 are different from those in the BNN for problem **P1**. After receiving the uplink power allocation vector $\hat{\mathbf{q}}^*$ from the output layer, the beamforming recovery in the conversion layer performs the following operations:

1) Calculate $\mathbf{T}^* = \sigma^2 \mathbf{I}_N + \sum_{k=1}^K \hat{q}_k^* \mathbf{h}_k \mathbf{h}_k^H$.

2) Calculate $\tilde{\mathbf{w}}_k^* = \tilde{\mathbf{w}}_k^*/||\tilde{\mathbf{w}}_k^*||_2, \forall k$, where $\tilde{\mathbf{w}}_k^* = (\mathbf{T}^*)^{-1}\mathbf{h}_k$.

3) Calculate the downlink power allocation vector $\hat{\mathbf{p}}^* = \sigma^2(\mathbf{\Psi}^*(\tilde{\mathbf{W}}^*, \mathbf{\Gamma}))^{-1}\mathbf{1}$.

4) Output the downlink beamforming vectors $\hat{\mathbf{w}}_k^* = \hat{p}_k^*\tilde{\mathbf{w}}_k^*, \forall k$, as the final results.

Here, the time complexity of the beamforming recovery module is $\mathcal{O}(KN^2 + N^3 + K^3)$. Note that the predicted power vector $\hat{\mathbf{q}}^*$ by the BNN is, in general, not exact. The prediction error will lead to the inaccuracy of power allocation vector $\hat{\mathbf{p}}^*$ as well as the downlink beamforming $\hat{\mathbf{W}}^*$. More specifically, if the predicted power vector $\hat{\mathbf{q}}^*$ has an acceptable accuracy with respect to the target power vector $\mathbf{q}^*$, i.e., $||\mathbf{q}^* - \hat{\mathbf{q}}^*||_2^2 < \varepsilon$ where ε is a small constant, then we can obtain a suboptimal solution whose objective value is larger than that of the optimal solution, i.e., $\sum_{k=1}^K ||\hat{\mathbf{w}}_k^*||_2^2 > \sum_{k=1}^K ||\mathbf{w}_k^*||_2^2$. Intuitively, the extra power consumption $q_{extra} = \sum_{k=1}^K ||\hat{\mathbf{w}}_k^*||_2^2 - \sum_{k=1}^K ||\mathbf{w}_k^*||_2^2$ can be regarded as the cost of the prediction error. However, if the predicted vector $\hat{\mathbf{q}}^*$ has a significant error, i.e., $||\mathbf{q}^* - \hat{\mathbf{q}}^*|_2^2 \gg \varepsilon$, the downlink beamforming $\hat{\mathbf{W}}^*$ inferred from the prediction $\hat{\mathbf{q}}^*$ may become infeasible since some elements of the vector $\hat{\mathbf{p}}^*$ have negative values. This suggests that different from problem **P1**, there is a certain probability of infeasibility of the BNN prediction for problem **P2**. However, our experiments show that the failure probability of the proposed BNN for problem **P2** is lower than 1% in most settings. More details will be given in Section VII. Moreover, the supervised learning with the loss function based on the MSE metric is adopted in the proposed BNN for problem **P2**.

VI. BNN for Sum Rate Maximization Problem

Different from the SINR balancing problem **P1** and the power minimization problem **P2**, no practically useful algorithm is available to find the optimal solution to the sum rate maximization problem **P3** and we can not make use of uplink-downlink duality directly. However, we will exploit a connection between problems **P2** and **P3** to find some key features of the optimal solution to problem **P3**.

A. Solution Structure

A fact was mentioned in [57] that the optimal solution to problem **P2**, using the minimal amount of power to achieve the given SINR targets, must meet the power constraint in problem **P3** to achieve the maximal sum rate. More specifically, given the optimal transmit power $P^\star$ of problem **P2** and setting the total power constraint P_{max} in problem **P3** as $P^\star$, the SINR values

of each user in problem **P3** can be calculated. By setting the SINR targets in problem **P2** with these calculated SINR values, the solutions to problems **P2** and **P3** will be the same. According to the connection between problems **P2** and **P3**, it has been pointed out in [2] that the optimal downlink beamforming vectors for problem **P3** follows the structure as

$$\mathbf{w}_k^* = \sqrt{p_k}\frac{(\mathbf{I}_N + \sum_{k=1}^K \frac{\lambda_k}{\sigma^2}\mathbf{h}_k\mathbf{h}_k^H)^{-1}\mathbf{h}_k}{||(\mathbf{I}_N + \sum_{k=1}^K \frac{\lambda_k}{\sigma^2}\mathbf{h}_k\mathbf{h}_k^H)^{-1}\mathbf{h}_k||_2}, \forall k, \tag{23}$$

where λ_k is a positive parameter and $\sum_{k=1}^K \lambda_k = \sum_{k=1}^K p_k = P_{max}$ according to the strong duality of problem **P2**. This is because P_{max} is the optimal cost function in problem **P2** and $\sum_{k=1}^K \lambda_k$ is the dual function. Note that the parameter vector $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_K]^T$ can be considered as a virtual power allocation vector. The solution structure in (23) provides the required expert knowledge for the beamforming design in problem **P3** and $\boldsymbol{\lambda}$ and $\mathbf{p}$ are the key features. But to our best knowledge, there is no low-complexity algorithm in the literature that can find the optimal p_k^* and λ_k^* in (23). The WMMSE algorithm is a good choice to find the locally optimal solutions [9, 10], and such an iterative algorithm ensures the continuity of the mapping from the channel to the solution, and can be learned by a neural network [27, 30]. Therefore, we can obtain the power allocation vectors $\mathbf{p}$ and $\boldsymbol{\lambda}$ according to the WMMSE algorithm. The supervised learning with the loss function based on the MSE metric will be first used to achieve as close to the results of the WMMSE algorithm as possible, i.e.,

$$\text{Loss} = \frac{1}{2LK}\sum_{l=1}^{L}\left(||\underline{\mathbf{p}}^{(l)} - \hat{\mathbf{p}}^{(l)}||_2^2 + ||\underline{\boldsymbol{\lambda}}^{(l)} - \hat{\boldsymbol{\lambda}}^{(l)}||_2^2\right), \tag{24}$$

where $\underline{\mathbf{p}}^{(l)}$ and $\underline{\boldsymbol{\lambda}}^{(l)}$ are the power vectors obtained from the WMMSE algorithm, and $\hat{\mathbf{p}}^{(l)}$ and $\hat{\boldsymbol{\lambda}}^{(l)}$ are the predicted results of the BNN. It is worth pointing out that the results in the training samples of problems **P1** and **P2** are optimal, thus the MSE-based loss function is equivalent to the objective function and the supervised learning method updates network parameters towards the direction of the optimal solution. However, the WMMSE algorithm for problem **P3** is locally optimal and thus (24) is not equivalent to the real objective of problem **P3** which aims to maximize the weighted sum rate. To further improve the sum rate performance, we continue to train the BNN in an unsupervised learning way, whose loss function takes the objective function directly as a metric, i.e.,

$$\text{Loss} = -\frac{1}{2KL}\sum_{l=1}^{L}\sum_{k=1}^{K}\alpha_k^{(l)}\log_2\left(1+\gamma_k^{ul,(l)}\right). \tag{25}$$

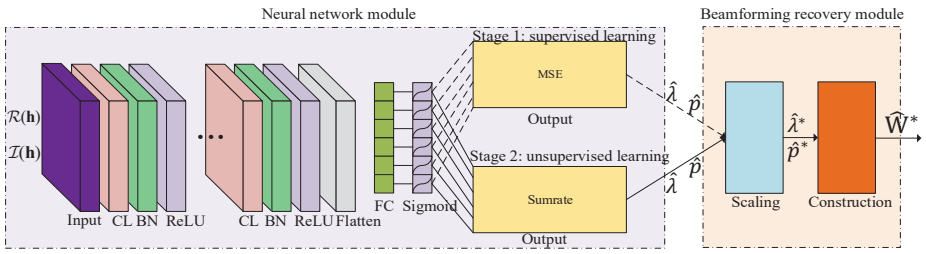


Fig. 4. BNN for the sum rate maximization problem.

B. Hybrid BNN Structure

The BNN for problem **P3** is presented in Fig. 4. The major difference from the BNNs in Figs. 2 and 3 is that the BNN in Fig. 4 has two stages of training. The first stage is responsible for pre-training using the supervised learning method with the loss function based on the MSE metric (24), while the second stage is responsible for enhanced training using the unsupervised learning method with the loss function whose metric is the objective function (25). Such a hybrid learning method of the supervised and unsupervised learning can significantly improve the learning performance and also accelerate convergence [29]. More specifically, the pre-training, as the approximation of WMMSE algorithm, starts with the random initialization of neural network parameters and the loss function (24). After the pre-training is finished, the neural network parameters are reserved and the loss function is replaced by (25), such that the second-stage training can achieve improved performance than the first-stage training.

Different from the BNNs in Figs. 2 and 3, the output layer in Fig. 4 generates $2K$ values including the power allocation vectors $\hat{\mathbf{p}}$ and $\hat{\boldsymbol{\lambda}}$. Then the scaling layer scales the results of the output layer $\hat{\mathbf{q}}$ and $\hat{\boldsymbol{\lambda}}$ to meet the power constraint by the following method:

$$\hat{\mathbf{p}}^* = \frac{P_{max}}{||\hat{\mathbf{p}}||_1}\hat{\mathbf{p}} \text{ and } \hat{\boldsymbol{\lambda}}^* = \frac{P_{max}}{||\hat{\boldsymbol{\lambda}}||_1}\hat{\boldsymbol{\lambda}}. \tag{26}$$

Finally, the construction layer constructs the downlink beamforming vectors according to (23):

$$\hat{\mathbf{w}}_k^* = \sqrt{\hat{p}_k^*}\frac{(\mathbf{I}_N + \sum_{k=1}^K \frac{\hat{\lambda}_k^*}{\sigma^2}\mathbf{h}_k\mathbf{h}_k^H)^{-1}\mathbf{h}_k}{||(\mathbf{I}_N + \sum_{k=1}^K \frac{\hat{\lambda}_k^*}{\sigma^2}\mathbf{h}_k\mathbf{h}_k^H)^{-1}\mathbf{h}_k||_2}, \forall k. \tag{27}$$

Thus, the time complexity of the beamforming recovery module for problem **P3** is $\mathcal{O}(KN^2+N^3)$.

VII. Simulation Results

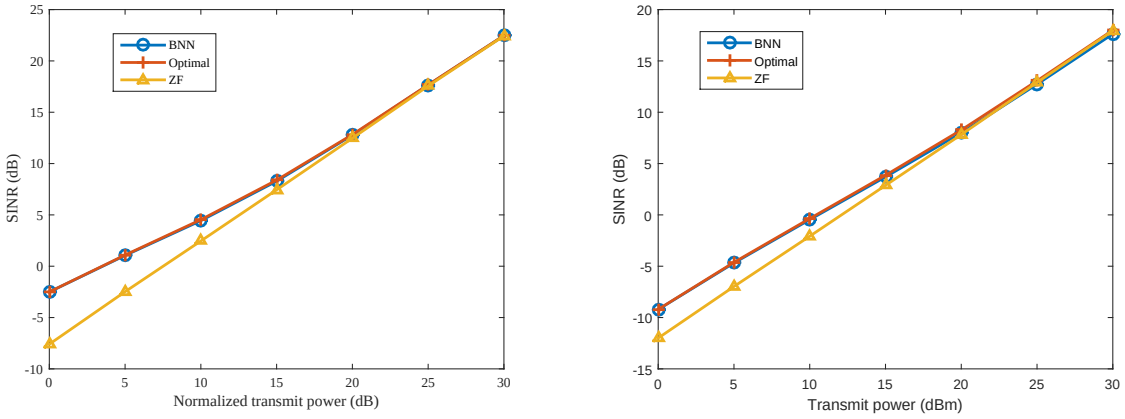
To evaluate the performance of the proposed BNNs, we carry out numerical simulations to compare the BNNs with several benchmark solutions (when available), including the optimal beamforming, the ZF beamforming [58], the RZF beamforming [59], and the WMMSE algorithm. We consider a downlink transmission scenario where the BS is equipped with $N = 6$ antennas and its coverage is a disc with a radius of 500 m. There are $K = 4$ single-antenna users and these users are distributed uniformly within the coverage of the BS. Note that none of these users is closer to the BS than 100 m. The channel of user k is modelled as $\mathbf{h}_k = \sqrt{d_k}\tilde{\mathbf{h}}_k \in \mathbb{C}^{N\times 1}$ where $\tilde{\mathbf{h}}_k \sim \mathcal{CN}(\mathbf{0}, \mathbf{I}_N)$ is the small-scale fading [60] and $d_k = 128.1 + 37.6\log_{10}(\omega)$[dB] denotes the pathloss between user k and the BS [61] with ω representing the distance in km. Here, shadow fading is omitted for simplicity. The noise power spectral density is -174 dBm/Hz and the total system bandwidth is 20 MHz. For simplicity, we assume all the sub-streams have the same importance and all the users have the same priority, i.e., $\rho_k = 1, \forall k$, and $\alpha_k = 1, \forall k$. Besides, perfect CSI is assumed to be available at the BS.

In our simulation, we prepare 20000 training samples and 5000 testing samples, respectively. The validation split is set to 0.2 and the training data is randomly shuffled at each epoch. All the BNNs have the same structure as shown in Table I. The fully-connected layer in the BNNs for problems **P1** and **P2** has K neurons but that in the BNN for problem **P3** has $2K$ neurons. The Glorot normal initializer [62] is used for weight initialization and biases are initialized to 0. Adam optimizer [63] is used with the MSE metric-based loss function. However, in the second stage of the BNN for problem **P3**, the metric of the loss function becomes the sum rate. The last activation layer is the sigmoid function so that the target output in the training and testing samples should be normalized into (0,1] by dividing a factor. Also, the channel coefficients are normalized by the noise power before being fed into the BNNs to avoid entering the insensitive area of the sigmoid function. The proposed BNN solutions are implemented in Python 3.6.5 with Tensorflow 1.2.1 and Keras 2.2.2 on a computer with 1 Intel i7-7700U CPU Core and RAM of 32GB, and the benchmarks are also implemented in Python 3.6.5 with a popular library **numpy**. Note that unless explicitly mentioned otherwise, all the neural network modules adopt the default setting in Table I and a separate neural network model is trained for each different case.

TABLE I

PARAMETERS OF THE NEURAL NETWORK MODULES.

Layer	Parameter
Layer 1 (input)	Input of size $2 \times NK$, batch of size 200, 100 epochs
Layer 2 (convolutional)	8 kernels of 3×3, zero padding 1, stride 1
Layer 3 (batch normalization)	Momentum=0.99, $\epsilon = 0.001$
Layer 4 (activation)	ReLU
Layer 5 (convolutional)	8 kernels of 3×3, zero padding 1, stride 1
Layer 7 (batch normalization)	Momentum=0.99, $\epsilon = 0.001$
Layer 6 (activation)	ReLU
Layer 8 (flatten)	
Layer 9 (fully-connected)	K or $2K$ neurons
Layer 10 (activation)	Sigmoid
Layer 11 output layer	Adam optimizer, learning rate of 0.001, MSE metric



(a) (b)

Fig. 5. The SINR performance averaged over 5000 samples in two different cases: (a) without large-scale fading and (b) with large-scale fading under $\{K = 4, N = 6\}$.

A. *BNN for the SINR Balancing Problem*

We first consider the BNN for the SINR balancing problem **P1**, which updates network parameters in a supervised learning way. The iterative algorithm in [12, Table 1] is used to generate the training and testing samples. The ZF beamforming is achieved by allocating power to make all the users have the same SINR value under a total power constraint. Fig. 5 shows the SINR performance averaged over 5000 samples in two cases: one only considering the small-scale fading but the other considering both the small-scale fading and large-scale fading. In both

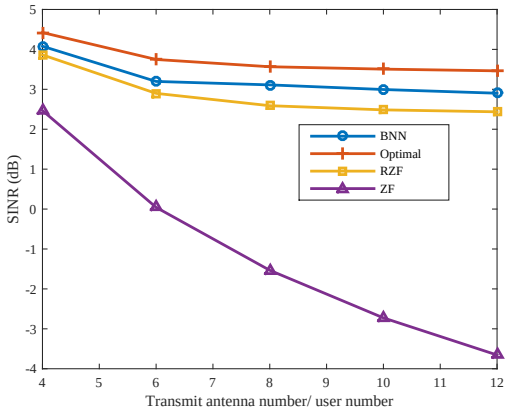


Fig. 6. Comparison of four different beamforming solutions, i.e., the optimal solution, the ZF beamforming, the RZF beamforming, and the BNN solution under $\{K = N, P_{max} = 20$ dBm$\}$.

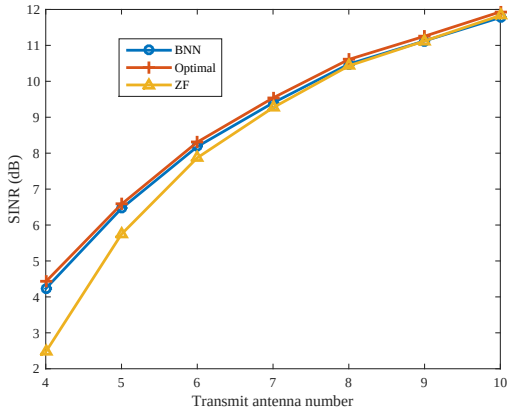


Fig. 7. The SINR performance versus different transmit antenna numbers using the same trained BNN under $\{K = 4, N = 10, P_{max} = 20$ dBm$\}$.

cases, the SINR performance of the proposed BNN solution is very close to that of the optimal solution [12]. It is observed that there is an obvious gap between the optimal solution and the ZF beamforming in the low normalized transmit-power ($\frac{P_{max}}{\sigma^2}$) regime of Fig. 5(a) as well as the low transmit-power regime of Fig. 5(b). However, the gap decreases as the (normalized) transmit power increases.

To further compare the SINR performance of the optimal solution, the ZF beamforming, the RZF beamforming whose regularization parameter is set as $\frac{P_{max}}{K}$, and the BNN solution, we evaluate the output SINR in Fig. 6 assuming that the number of users is the same as the number of BS antennas, i.e., $K = N$, and they increase together. It is shown that the BNN solution has some performance loss compared to the optimal solution due to the estimation error, but the BNN solution always achieves a better performance than the ZF beamforming and RZF beamforming. This fact indicates the application prospect of the BNN: the computational complexity and time of the BNN solution is similar to those of the ZF beamforming and RZF beamforming, but is much lower than that of the optimal solution because the optimal solution relies on an iterative process. Besides, we also find that the SINR performance of the four solutions decrease as the transmit antenna number (user number) increases and among the four solutions the ZF beamforming suffers most from the performance loss.

Table II presents the comparison of two input formats, i.e., **I/Q transformation** and **P/M**

TABLE II

I/Q TRANSFORMATION VERSUS P/M TRANSFORMATION.

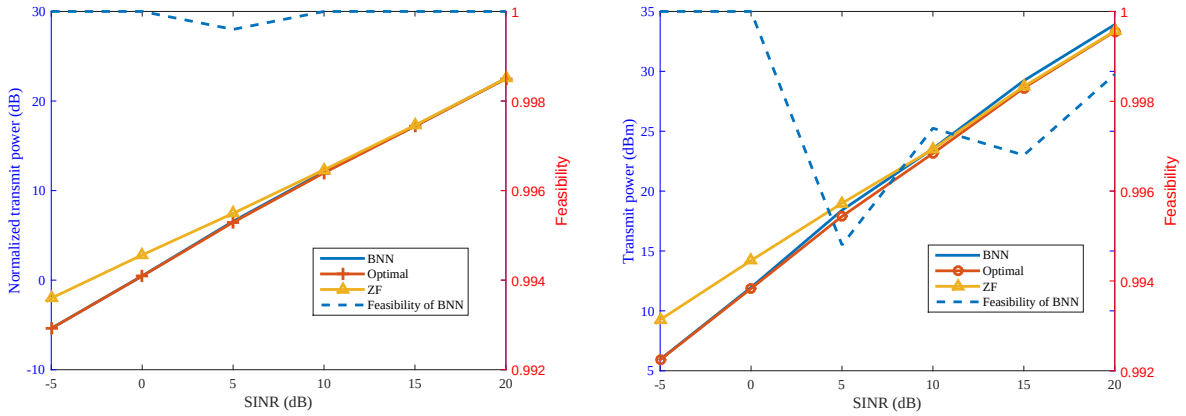
K/N		4	6	8	10	12
I/Q transformation	MSE	0.084	0.038	0.022	0.014	0.010
	MAE	0.223	0.147	0.111	0.088	0.075
P/M transformation	MSE	0.086	0.039	0.022	0.014	0.010
	MAE	0.225	0.149	0.111	0.087	0.073

transformation, in terms of the MSE performance and MAE performance of the predicted normalized power under the case with $K = N$ and $P_{max} = 20$ dBm. As shown in Table II, **I/Q transformation** and **P/M transformation** have close performance.

In Fig. 7, we demonstrate the generality of the proposed BNN by fixing the user number as $K = 4$ and the transmit power as $P_{max} = 20$ dBm and show the SINR performance versus different transmit antenna settings. We train only a single BNN with $\{K = 4,\ N = 10\}$, but allow the number of transmit antennas to vary from 4 to 10 when using the trained BNN. Then the redundant entries at the inputs and outputs are filled with 0's. It can be seen that these predicted results are very close to that of the optimal solution. This fact suggests the generality of the BNN, i.e., we can train a large BNN with more antennas which will also work for the cases with less antennas without re-training. This will be useful when some transmit antennas of the BS are malfunctioning or turned off.

B. BNN for the Power Minimization Problem

In this subsection, we consider the BNN for the power minimization problem **P2**, which also updates network parameters in a supervised learning way. The iterative algorithm in [5] is used to generate the training and testing samples. The ZF beamforming for comparison is achieved by minimizing the power for each user with a QoS constraint since there is no inter-user interference. We first investigate the effect of the SINR constraints of users on the power consumption. For convenience of comparison, we assume the SINR constraints of all users are the same, i.e. $\Gamma_k = \Gamma, \forall k$. In Fig. 8, we compare the power performance of the optimal beamforming, the ZF beamforming, and the beamforming obtained by the BNN. Note that both Figs. 8(a) and 8(b) have two Y-axes where the left Y-axis is used to measure the (normalized) transmit power averaged over the feasible sample set of the BNN solution and the right Y-axis is used to show



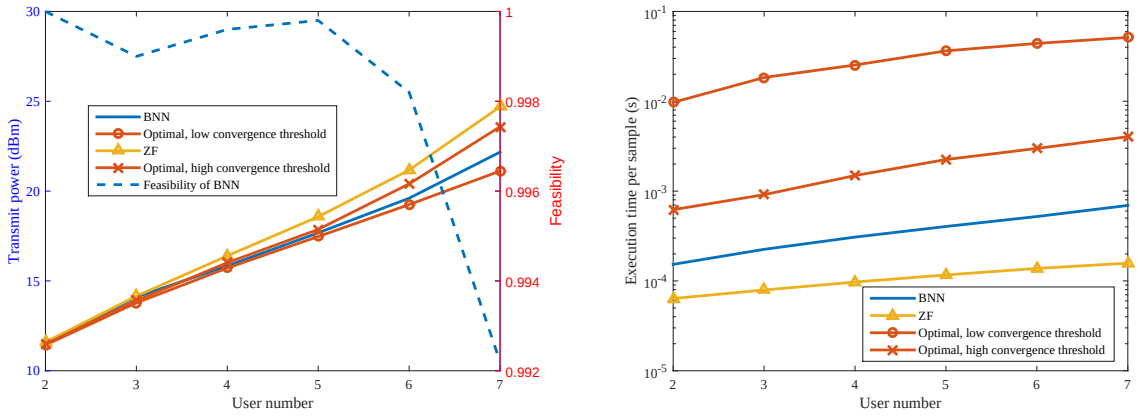
(a) (b)

Fig. 8. The power performance averaged over the feasible sample set of the BNN solution in two different cases: (a) without large-scale fading and (b) with large-scale fading under $\{K = 4,\ N = 6\}$.

the feasibility of the BNN. As mentioned in Section V, the BNN may fail to find a feasible solution to problem **P2** if the prediction error is unacceptable.

Figs. 8(a) and 8(b) present the (normalized) transmit power performance in the cases without and with consideration of the large-scale fading, respectively. In both cases, the (normalized) transmit power performance of the BNN solution is close to that of the optimal solution, and significantly outperforms the ZF beamforming in the low SINR-constraint regime which is higher than that of the optimal solution. We also find that, according to Fig. 8(b), the BNN solution performs slightly worse than the ZF solution when the SINR constraint is large, this is because the ZF solution becomes closer to the optimal solution as the SINR constraints increase, but the performance of the BNN solution is still close to that of the optimal solution. This fact suggests that when the SINR constraints are high, the ZF solution is a good choice instead of the BNN solution. Besides, we find that the feasibility of the BNN solution in both cases is more than 99.4%.

To further compare the BNN solution with the optimal solution and the ZF beamforming, we plot their power performance and execution time per sample in Figs. 9(a) and 9(b), respectively. Here, we consider two convergence strategies for the optimal iterative algorithm: the high convergence threshold ($\varepsilon_1 = 10^{-2}$) which can be reached with less iterations and the low convergence threshold ($\varepsilon_2 = 10^{-4}$) which requires more iterations for problem **P2**, i.e., $\frac{|\sum_{k=1}^{K}||\mathbf{w}_k^{(t-1)}||^2 - \sum_{k=1}^{K}||\mathbf{w}_k^{(t)}||^2|}{\sum_{k=1}^{K}||\mathbf{w}_k^{(t-1)}||^2} \leq \varepsilon_\kappa, \kappa \in \{1, 2\}$. In Fig. 9, the BS antenna number and SINR target



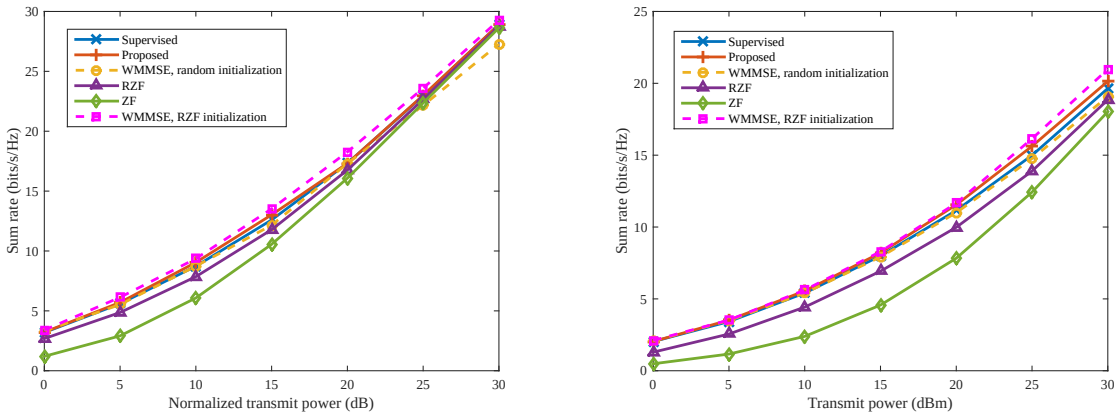
(a) (b)

Fig. 9. Comparison of three different beamforming solutions, i.e., the optimal solution, the BNN solution, and ZF beamforming: (a) power performance and (b) execution time per sample averaged over 5000 samples under $\{\Gamma = 5$ dB, $N = 8\}$.

of users are fixed as $N = 8$ and $\Gamma = 5$ dB. It is observed from Fig. 9(a) that as the user number K increases, the performance gap between the ZF beamforming and the optimal beamforming with the low convergence threshold becomes large because more users share the array gain. The BNN solution, with the feasibility of up to 99%, shows a better performance than the ZF beamforming and the optimal iterative algorithm with the high convergence threshold. Fig. 9(b) demonstrates that compared to the optimal solution with the low convergence threshold, the BNN solution can reduce the execution time per sample by about two orders of magnitude, which is slightly longer than that of the ZF beamforming. This is because the BNN solution and the ZF beamforming are obtained without an iterative process, but the BNN needs to execute the neural network operations as well as the conversion process. We can reduce the iteration times using the high convergence threshold, but this leads to the power performance degradation. According to the results in Figs. 9(a) and 9(b), we can conclude that the BNN solution provides a good balance between the performance and computational complexity.

C. BNN for the Sum Rate Maximization Problem

In this subsection, we evaluate the performance of the BNN for the sum rate maximization problem **P3** based on the proposed hybrid learning under the assumption that $K = 4$ and $N = 4$. The ZF beamforming with $p_k = \frac{P_{max}}{K}, \forall k$ and the RZF beamforming with $p_k = \lambda_k = \frac{P_{max}}{K}, \forall k$ are introduced as two baseline solutions. Since the performance of the WMMSE algorithm heavily

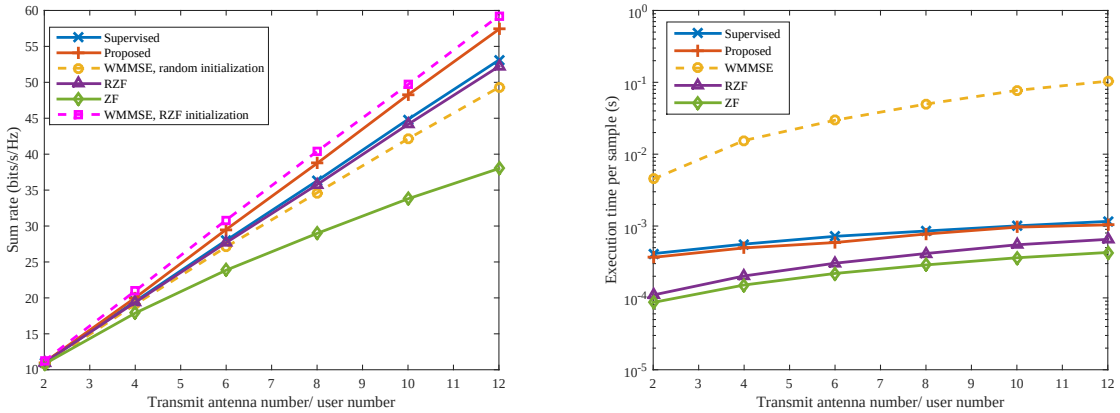


(a) (b)

Fig. 10. The sum rate performance averaged over 5000 samples in two different cases: (a) without large-scale fading and (b) with large-scale fading under $\{K = 4,\ N = 4\}$.

relies on initialization [9, 10], two different initialization methods, the RZF initialization and the random initialization, are considered and the WMMSE algorithm with the RZF initialization is used to generate samples for the supervised learning in the first stage. First, Fig. 10 shows the sum rate performance averaged over 5000 samples in two different cases: the former case in Fig. 10(a) only considers small-scale fading and and the latter case in Fig. 10(b) considers both small-scale fading and large-scale fading. It is shown that the sum rate performance of all solutions increases as the (normalized) transmit power increases and different initialization methods of the WMMSE algorithm have a large performance gap. We observe that in both cases the proposed BNN solution based on the hybrid learning always achieves a performance close to that of the WMMSE algorithm with the RZF initialization, while the performance of the supervised learning-based BNN solution is less satisfactory. This is because the second stage of the hybrid learning method aims to maximize the sum rate and its performance is bounded by the global optimal solution to problem **P3**. But the aim of the BNN solution based on the supervised learning is to achieve as close to the WMMSE solution as possible and its performance is restricted by the WMMSE solution, which is verified in Figs. 10(a) and 10(b).

We further compare the sum rate performance and the computational complexity, in terms of the execution time per sample, of five beamforming solutions in Figs. 11(a) and 11(b), respectively. The iteration number of the WMMSE algorithm is limited to at most 10. We fix the transmit power budget as $P_{max} = 30$ dBm and assume the transmit antenna number is the same



(a) (b)

Fig. 11. Comparison of five different beamforming solutions, i.e., the WMMSE solution, BNN solutions based on the supervised learning and the proposed hybrid learning, respectively, the RZF beamforming, and the ZF beamforming: (a) sum rate performance and (b) execution time per sample averaged over 5000 samples under $\{K = N,\ P_{max} = 30$ dBm$\}$.

as the user number, i.e., $N = K$. As the number of transmit antennas increases, the sum rate performance of all five solutions increases simultaneously. The performance of the proposed BNN solution based on the hybrid learning method is always close to that of the WMMSE algorithm with the RZF initialization, but is superior to those of the other four solutions and the performance gap becomes larger when the number of the transmit antenna increases. According to Fig. 11(b), the execution time per sample of the BNN solutions based on the supervised learning and hybrid learning methods is at the same level, which is slightly longer than that of the ZF beamforming and the RZF beamforming, for the same reason of Fig. 9(b). As expected, the WMMSE algorithm consumes the most time because of its iterative process. Similar to the other proposed BNNs, it proves that the proposed BNN solution to the sum rate problem **P3** provides a good balance between the performance and computational complexity.

VIII. Conclusions

In this paper, we proposed a DL-based framework for fast optimization of the beamforming vectors in the MISO downlink and then devised three BNNs under this framework for the SINR balancing problem under a total power constraint, the power minimization problem under individual QoS constraints, and the sum rate maximization problem under a total power constraint, respectively. The proposed BNNs are based on the CNN structure and expert knowledge. The supervised learning method was adopted for the SINR balancing problem and the power

minimization problem because effective algorithms are available for generating training samples. However, there is no practically useful algorithm to find the optimal solution to the nonconvex sum rate maximization problem, therefore the corresponding BNN adoptes a hybrid learning method which first pre-trains the neural network based on the supervised learning method, and then updates the network parameters with the unsupervised learning method to further improve learning performance. Furthermore, in order to reduce the complexity of prediction, the proposed BNNs take advantage of expert knowledge to extract key features instead of predicting beamforming matrix directly. Simulation results demonstrated that the proposed BNN solutions provided a good balance between the performance and computational complexity.

This work is an attempt to apply the DL technique to beamforming optimization. Actually, a lot of extension works are worth further study. For example, it is unclear so far which input format, I/Q transformation or P/M transformation, is better. In addition, the joint optimization of user selection and beamforming design for the power minimization problem is interesting and it deserves more investigation. Besides, user mobility, machine-type communications, imperfect CSI, and multi-cell scenarios are also interesting extensions for future works.

REFERENCES

[1] W. Xia, G. Zheng, Y. Zhu, J. Zhang, J. Wang, and A. Petropulu, "Deep learning based beamforming neural networks in downlink MISO systems," in *Proc. IEEE Int. Conf. Commun. (ICC) Workshop*, Shanghai, China, May 2019, pp. 1–5.

[2] E. Björnson, M. Bengtsson, and B. Ottersten, "Optimal multiuser transmit beamforming: A difficult problem with a simple solution structure," *IEEE Signal Process. Mag.*, vol. 31, no. 4, pp. 142–148, Jul. 2014.

[3] H. Boche and M. Schubert, "A general duality theory for uplink and downlink beamforming," in *Proc. IEEE Conf. Veh. Technol. Conf. (VTC)*, vol. 1, Vancouver, Canada, Sep. 2002, pp. 87–91.

[4] D. Gerlach and A. Paulraj, "Base station transmitting antenna arrays for multipath environments," *Signal Process.*, vol. 54, no. 1, pp. 59–73, Oct. 1996.

[5] Q. Shi, M. Razaviyayn, M. Hong, and Z. Luo, "SINR constrained beamforming for a MIMO multi-user downlink system: Algorithms and convergence analysis," *IEEE Trans. Signal Process.*, vol. 64, no. 11, pp. 2920–2933, Jun. 2016.

[6] F. Rashid-Farrokhi, K. R. Liu, and L. Tassiulas, "Transmit beamforming and power control for cellular wireless systems," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 8, pp. 1437–1450, Oct. 1998.

[7] A. B. Gershman, N. D. Sidiropoulos, S. Shahbazpanahi, M. Bengtsson, and B. Ottersten, "Convex optimization-based beamforming," *IEEE Signal Process. Mag.*, vol. 27, no. 3, pp. 62–75, May 2010.

[8] A. Wiesel, Y. C. Eldar, and S. Shamai, "Linear precoding via conic optimization for fixed MIMO receivers," *IEEE Trans. Signal Process.*, vol. 54, no. 1, pp. 161–176, Jan. 2006.

[9] Q. Shi, M. Razaviyayn, Z. Luo, and C. He, "An iteratively weighted MMSE approach to distributed sum-utility maximization for a MIMO interfering broadcast channel," *IEEE Trans. Signal Process.*, vol. 59, no. 9, pp. 4331–4340, Sep. 2011.

[10] S. S. Christensen, R. Agarwal, E. D. Carvalho, and J. M. Cioffi, "Weighted sum-rate maximization using weighted MMSE for MIMO-BC beamforming design," *IEEE Trans. Wireless Commun.*, vol. 7, no. 12, pp. 4792–4799, Dec. 2008.

[11] T. Yoo and A. Goldsmith, "On the optimality of multiantenna broadcast scheduling using zero-forcing beamforming," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 3, pp. 528–541, Mar. 2006.

[12] M. Schubert and H. Boche, "Solution of the multiuser downlink beamforming problem with individual SINR constraints," *IEEE Trans. Veh. Technol.*, vol. 53, no. 1, pp. 18–28, Jan. 2004.

[13] Z.-Q. Luo, W.-K. Ma, A. M.-C. So, Y. Ye, and S. Zhang, "Semidefinite relaxation of quadratic optimization problems," *IEEE Signal Process. Mag.*, vol. 27, no. 3, pp. 20–34, May 2010.

[14] M. Bengtsson and B. Ottersten, "Optimal and suboptimal transmit beamforming," *in Handbook of Antennas in Wireless Communications*, CRC Press, Jan. 2001.

[15] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[16] F. Rashid-Farrokhi, L. Tassiulas, and K. R. Liu, "Joint optimal power control and beamforming in wireless networks using antenna arrays," *IEEE Trans. Commun.*, vol. 46, no. 10, pp. 1313–1324, Oct. 1998.

[17] E. Nachmani, Y. Be'ery, and D. Burshtein, "Learning to decode linear codes using deep learning," in *Proc. IEEE Annual Allerton Conf. Commun. Control Comput.*, Monticello, USA, Sep. 2016, pp. 341–346.

[18] F. Liang, C. Shen, and F. Wu, "An iterative BP-CNN architecture for channel decoding," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 144–159, Feb. 2018.

[19] C. Fan, X. Yuan, and Y.-J. A. Zhang, "CNN-based signal detection for banded linear systems," *IEEE Trans. Wireless Commun.*, pp. 1–1, 2019.

[20] N. Samuel, T. Diskin, and A. Wiesel, "Learning to detect," *IEEE Trans. Signal Process.*, vol. 67, no. 10, pp. 2554–2564, May 2019.

[21] N. Farsad and A. Goldsmith, "Detection algorithms for communication systems using deep learning," *arXiv preprint arXiv:1705.08044*, 2017.

[22] C.-K. Wen, W.-T. Shih, and S. Jin, "Deep learning for massive MIMO CSI feedback," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 748–751, Oct. 2018.

[23] T. Wang, C.-K. Wen, S. Jin, and G. Y. Li, "Deep learning-based CSI feedback approach for time-varying massive MIMO channels," *IEEE Wireless Commun. Lett.*, vol. 8, no. 2, pp. 416–419, Apr. 2019.

[24] H. Ye, G. Y. Li, and B. Juang, "Power of deep learning for channel estimation and signal detection in OFDM systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 114–117, Feb. 2018.

[25] M. Eisen, C. Zhang, L. F. O. Chamon, D. D. Lee, and A. Ribeiro, "Learning optimal resource allocations in wireless systems," *IEEE Trans. Signal Process.*, vol. 67, no. 10, pp. 2775–2790, May 2019.

[26] K. I. Ahmed, H. Tabassum, and E. Hossain, "Deep learning for radio resource allocation in multi-cell networks," *IEEE Network*, pp. 1–8, 2019.

[27] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for wireless resource management," in *Proc. IEEE Int. Workshop Signal Process. Advances Wireless Commun. (SPAWC)*, Sapporo, Japan, Jul. 2017, pp. 1–6.

[28] F. Liang, C. Shen, W. Yu, and F. Wu, "Towards optimal power control via ensembling deep neural networks," *arXiv preprint arXiv:1807.10025*, 2018.

[29] W. Lee, M. Kim, and D.-H. Cho, "Deep power control: Transmit power control scheme based on convolutional neural network," *IEEE Commun. Lett.*, vol. 22, no. 6, pp. 1276–1279, Jun. 2018.

[30] T. Van Chien, T. N. Canh, E. Björnson, and E. G. Larsson, "Power control in cellular massive MIMO with varying user activity: A deep learning solution," *arXiv preprint arXiv:1901.03620*, 2019.

[31] L. Sanguinetti, A. Zappone, and M. Debbah, "Deep learning power allocation in massive MIMO," in *Proc. Asilomar Conf. Signals, Systems, Computers*, Pacific Grove, CA, USA, Oct. 2018, pp. 1257–1261.

[32] M. Chen, W. Saad, C. Yin, and M. Debbah, "Echo state networks for proactive caching in cloud-based radio access networks with mobile users," *IEEE Trans. Wireless Commun.*, vol. 16, no. 6, pp. 3520–3535, Jun. 2017.

[33] S. Dörner, S. Cammerer, J. Hoydis, and S. t. Brink, "Deep learning based communication over the air," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 132–143, Feb. 2018.

[34] T. J. O'Shea, K. Karra, and T. C. Clancy, "Learning to communicate: Channel auto-encoders, domain specific regularizers, and attention," in *Proc. IEEE Int. Symp. Signal Process. Inf. Technol. (ISSPIT)*, Limassol, Cyprus, Dec. 2016, pp. 223–228.

[35] Z. Zhao, "Deep-waveform: A learned OFDM receiver based on deep complex convolutional networks," *arXiv preprint arXiv:1810.07181*, 2018.

[36] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Commun. Surveys Tutorials*, pp. 1–1, 2019.

[37] A. Zappone, M. Di Renzo, and M. Debbah, "Wireless networks design in the era of deep learning: Model-based, AI-based, or both?" *IEEE Trans. Commun.*, pp. 1–1, 2019.

[38] T. Wang, C.-K. Wen, H. Wang, F. Gao, T. Jiang, and S. Jin, "Deep learning for wireless physical layer: Opportunities and challenges," *China Commun.*, vol. 14, no. 11, pp. 92–111, Nov. 2017.

[39] P. de Kerret and D. Gesbert, "Robust decentralized joint precoding using team deep neural network," in *Proc. Int. Symp. Wireless Commun. Systems (ISWCS)*, Lisbon, Portugal, Aug. 2018.

[40] A. Alkhateeb, S. Alex, P. Varkey, Y. Li, Q. Qu, and D. Tujkovic, "Deep learning coordinated beamforming for highly-mobile millimeter wave systems," *IEEE Access*, vol. 6, pp. 37 328–37 348, 2018.

[41] Y. Shi, A. Konar, N. D. Sidiropoulos, X. Mao, and Y. Liu, "Learning to beamform for minimum outage," *IEEE Trans. Signal Process.*, vol. 66, no. 19, pp. 5180–5193, Oct. 2018.

[42] H. Huang, W. Xia, J. Xiong, J. Yang, G. Zheng, and X. Zhu, "Unsupervised learning based fast beamforming design for downlink MIMO," *IEEE Access*, vol. 7, pp. 7599–7605, 2019.

[43] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[44] D.-X. Zhou, "Universality of deep convolutional neural networks," *Applied Computational Harmonic Analysis*, Jun. 2019.

[45] M. Lee, Y. Xiong, G. Yu, and G. Y. Li, "Deep neural networks for linear sum assignment problems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 6, pp. 962–965, Dec. 2018.

[46] W. Cui, K. Shen, and W. Yu, "Spatial deep learning for wireless scheduling," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1248–1261, Jun. 2019.

[47] W. Yu and T. Lan, "Transmitter optimization for the multi-antenna downlink with per-antenna power constraints," *IEEE Trans. Signal Process.*, vol. 55, no. 6, pp. 2646–2660, Jun. 2007.

[48] M. Kulin, T. Kazaz, I. Moerman, and E. D. Poorter, "End-to-end learning from spectrum data: A deep learning approach for wireless signal identification in spectrum monitoring applications," *IEEE Access*, vol. 6, pp. 18 484–18 501, 2018.

[49] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Machine Learning (ICML)*, Lille, France, Jul. 2015, pp. 448–456.

[50] A. Botchkarev, "Performance metrics (error measures) in machine learning regression, forecasting and prognostics: Properties and typology," *arXiv preprint arXiv:1809.03006*, 2018.

[51] B. Matthiesen, A. Zappone, E. A. Jorswieck, and M. Debbah, "Deep learning for optimal energy-efficient power control in wireless interference networks," *arXiv preprint arXiv:1812.06920*, 2018.

[52] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *Proc. IEEE Conf. Computer Vision Pattern Recognition (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 5353–5360.

[53] W. Yang and G. Xu, "Optimal downlink power assignment for smart antenna systems," in *Proc. IEEE Int. Conf. Acoustics, Speech Process. (ICASSP)*, vol. 6, Seattle, USA, May 1998.

[54] E. Seneta, *Non-negative matrices and Markov chains*. Springer Science & Business Media, 2006.

[55] J. M. Ortega, *Numerical analysis: a second course*. SIAM, 1990.

[56] E. Visotsky and U. Madhow, "Optimum beamforming using transmit antenna arrays," in *Proc. IEEE Veh. Technol. Conf.*, vol. 1, Houston, TX, USA, May 1999, pp. 851–856.

[57] E. Björnson and E. Jorswieck, "Optimal resource allocation in coordinated multi-cell systems," *Foundations and Trends® in Communications and Information Theory*, vol. 9, no. 2–3, pp. 113–381, 2013.

[58] M. Joham, W. Utschick, and J. A. Nossek, "Linear transmit processing in MIMO communications systems," *IEEE Trans. Sig. Process.*, vol. 53, no. 8, pp. 2700–2712, Aug. 2005.

[59] Q. H. Spencer, A. L. Swindlehurst, and M. Haardt, "Zero-forcing methods for downlink spatial multiplexing in multiuser MIMO channels," *IEEE Trans. Signal Process.*, vol. 52, no. 2, pp. 461–471, Feb. 2004.

[60] H. Zhu and J. Wang, "Chunk-based resource allocation in OFDMA systems-Part II: Joint chunk, power and bit allocation," *IEEE Trans. Commun.*, vol. 60, no. 2, pp. 499–509, Feb. 2012.

[61] H. Dahrouj and W. Yu, "Coordinated beamforming for the multicell multi-antenna wireless system," *IEEE Trans. Wireless Commun.*, vol. 9, no. 5, pp. 1748–1759, May 2010.

[62] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. int. conf. artificial intelligence statistics (AISTATS)*, Sardinia, Italy, May 2010, pp. 249–256.

[63] J. Ba and D. Kingma, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learning Representations (ICLR)*, San Diego, USA, May 2015, pp. 1–15.