Received 30 August 2022, accepted 2 November 2022, date of publication 14 November 2022, date of current version 18 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3221800

RESEARCH ARTICLE

Neural Network-Based Fixed-Complexity Precoder Selection for Multiple Antenna Systems

JAEKWON KIM AND HYO-SANG LIM, (Member, IEEE)
Division of Software, Yonsei University, Wonju 26493, South Korea

Corresponding author: Hyo-Sang Lim (hyosang@yonsei.ac.kr)

ABSTRACT In this paper, we propose a neural network-based precoder selection method for multiple antenna systems that are equipped with maximum likelihood detectors. We train a fully connected neural network by supervised learning with novel soft labels that are derived from the error probability of maximum likelihood detection. The dimension of the input data is reduced by QR decomposition of the channel matrices, thereby reducing the number of nodes of the input layer. Furthermore, the dimension reduction improves the network accuracy. The number of connections between the layers are reduced by applying the network pruning technique, after which the surviving connections are retrained to recover the degraded accuracy due to the pruning. We also optimize the regularization method, considering not only network overfitting but also pruning and retraining. Our method achieves a near optimal bit error performance of the previous sphere decoding (SD)-based symbolic algorithm, of which complexity fluctuates depending on channel matrices. Unlike the conventional SD-based method, the complexity of the proposed method is fixed by the intrinsic characteristic of neural network, which is desirable from the perspective of hardware implementation. And the fixed complexity is lowered by pruning unimportant connections of the networks. With the aid of computer simulations, we show that the fixed complexity of the proposed method is close to the average complexity of the conventional SD-based symbolic algorithm, allowing only negligible degradation of the error performance.

INDEX TERMS Precoder selection, neural network, network pruning, fixed complexity, multiple antenna.

I. INTRODUCTION

Precoding techniques with a codebook are known to improve the performance of multiple antenna systems without requiring feedback of full channel information. Various symbolic algorithms have been developed such as limited search space method [1] and the lattice reduction based method [2]. In [3], the concept of sphere decoding (SD) was exploited to select the optimal precoder for multiple antenna systems in a computationally efficient manner. Compared with the limited search space method and the lattice reduction based method, the SD-based precoder selection technique achieves the optimal error performance with a lower computational complexity. Fixed complexity as well as low average complexity is desirable for hardware implementation of mobile applications such as portable mobile phones. Unfortunately, the worst case complexity of the SD-based precoder selection is several times higher than the average complexity.

From the perspective of hardware implementation of a function, neural networks are preferable to traditional symbolic algorithms due to the intrinsic characteristic of fixed complexity. Once the networks are trained offline to learn the underlying connections between the inputs and outputs of the target function, only the parameter values at the end of machine learning are needed in the mobile devices to perform the function, regardless of how the parameters were updated in the learning phase. The fixed number of parameters implies the fixed required complexity of the function in the inference phase. In this spirit, neural networks have been applied to physical layer of wireless communications: for example, signal detection [4], [6], channel estimation [7], and end-to-end communications [8], [9].

In [10] and [11], precoders were designed for multiple antenna systems using convolutional neural networks. In [12]

The associate editor coordinating the review of this manuscript and approving it for publication was Jose Saldana.

and [13], neural networks were used for joint antenna selection and precoder design. Although antennas at the receiver side were selected in [12] and [13], it is logically equivalent to precoding from the perspective of implementation as neural network, having the label vector with 1 at the optimal antenna set position and 0s at the other set positions. Here the number of sets is the number of combinations choosing a fixed number of antenna out of all antenna installed in the devices.

However, when the number of optimal precoders is not fixed, the antenna selection approaches can not be directly applied to the precoder selection problem addressed in this paper. In an attempt to resolve the varying number of optimal precoders, we propose the concept of soft label that is derived from the error rate expression of the maximum likelihood detection. The use of soft label significantly reduces the length of label vectors, i.e., the number of nodes of the output layer of the neural network. We also reduce the dimension of the input layer by QR decomposition of channel matrices. The dimension reduction not only reduces the number of connections but also offers additional improvement of the network accuracy. Furthermore, in the previous works [12] and [13], the network pruning technique has not been adopted, which is known to reduce the number of weights, thereby reducing the fixed complexity of neural network without significant performance loss. In [14] and [15], it was shown that connections of neural network can be classified into important and unimportant ones, which means we can prune away unimportant connections, thereby reducing the number of parameters without significant reduction of accuracy. As is known, the main purpose of regularization is to prevent the overfitting of the network. In an effort to further improve the accuracy of pruned neural network, we optimize the regularization method and amount, considering not only network overfitting but also pruning and retraining.

In the simulation section, we demonstrate the efficacy of the proposed technique, comparing with the SD-based algorithm in terms of error performance and the computational complexity. We also show the effect of dimension reduction and regularization optimization.

II. PRECODED MULTIPLE ANTENNA SYSTEM

In this section, we describe the precoded spatially multiplexed multiple antenna systems [3]. Letting N_T and N_R denote the number of transmitting and receiving antennas, respectively, the relationship between the transmitted and received symbol vectors can be expressed as follows:

$$\mathbf{y} = \frac{1}{\sqrt{N_S P_\Omega}}\mathbf{H}\mathbf{F}_n\mathbf{x} + \mathbf{z} \tag{1}$$

where $\mathbf{x} = [x_1\ x_2 \cdots x_{N_S}]^T$ denotes the transmitted symbol vector with $N_S(\leq \min(N_T, N_R))$ symbols, and $\mathbf{y} = [y_1\ y_2 \cdots y_{N_R}]^T$ with y_j denoting the received signal at the j-th antenna. $\mathbf{H}$ denotes an $N_R \times N_T$ channel matrix, in which h_{ji} denotes the standard unit power of the Rayleigh-fading complex gain between the i-th transmitting antenna and the j-th receiving antenna, whereas $\mathbf{z} = [z_1\ z_2 \cdots z_{N_R}]^T$ with z_j denoting the additive white Gaussian noise with zero mean and variance σ_z^2 at the j-th receiving antenna. Finally, $\mathbf{F}_n$ denotes an $N_T \times N_S$ precoder in a codebook $\mathcal{F} = \{\mathbf{F}_1, \mathbf{F}_2, \cdots, \mathbf{F}_N\}$ that is available at both the transmitter and the receiver sides.

In this paper, we assume the square quadrature amplitude modulation (QAM) constellations

$$\begin{aligned}\Omega = \Big\{R + jI \Big| R &= -\sqrt{|\Omega|}+1, -\sqrt{|\Omega|}+3,\\ &\cdots, \sqrt{|\Omega|}-3, \sqrt{|\Omega|}-1,\\ I &= -\sqrt{|\Omega|}+1, -\sqrt{|\Omega|}+3,\\ &\cdots, \sqrt{|\Omega|}-3, \sqrt{|\Omega|}-1\Big\},\end{aligned} \tag{2}$$

from which $x_i, i = 1, 2, \cdots, N_S$, are drawn. $|\Omega|$ denotes the cardinality of Ω, and P_Ω denotes the average power of Ω.

III. RELATED WORK

In this section, we review previous symbolic algorithms for precoder selection. Then, we also review neural network based connectionist approaches.

A. SYMBOLIC PRECODER SELECTION ALGORITHMS

To minimize the error rate, a precoder can be selected as

$$\mathbf{F}_{\text{opt}} = \arg\max_{\mathbf{F}_q \in \mathcal{F}} d_{\min}(\mathbf{H}\mathbf{F}_n) \tag{3}$$

$$\text{with } d_{\min}(\mathbf{H}\mathbf{F}_n) = \min_{\substack{\mathbf{x}_p \in \Omega^{N_s}, \mathbf{x}_q \in \Omega^{N_s} \\ \mathbf{x}_p \neq \mathbf{x}_q}} \left\|\mathbf{H}\mathbf{F}_n(\mathbf{x}_p - \mathbf{x}_q)\right\| \tag{4}$$

where Ω^{N_s} denotes the set of transmitted symbol vectors. The precoder selection presented in (3) is optimal from the viewpoint of error performance. However, its complexity is prohibitive due to the exhaustive search over $|\Omega|^{N_s}(|\Omega|^{N_s} - 1)$ combinations of $\mathbf{x}_p$ and $\mathbf{x}_q$ for each $\mathbf{F}_n$.

In [3], the optimal precoder selection of (3) and (4) were equivalently expressed as follows:

$$\mathbf{F}_{\text{opt}} = \arg\max_{\mathbf{F}_l \in \mathcal{F}} \min_{\Delta\mathbf{x} \neq \mathbf{0}} \left\|\mathbf{H}\mathbf{F}_n \Delta\mathbf{x}\right\|. \tag{5}$$

with $\Delta\mathbf{x} = (\mathbf{x}_p - \mathbf{x}_q)/2$. Then, the high complexity was successfully reduced by (1) exploiting the symmetric structure of QAM constellations, (2) adopting the concept of SD, (3) eliminating the last stage of SD, and (4) performing an SD-like process in a selective manner. When compared with other techniques such as the limited search space approach [1] and the lattice reduction based approach [2], the SD-based approach offered better error performance with less complexity.

However, the drawback of the SD-based precoder selection technique is that although average complexity is lower than the other algorithms, as will be shown in the simulation section, the worst case complexity is quite high, which is not desirable for hardware implementation.

B. NEURAL NETWORK BASED CONNECTIONIST APPROACH

From the perspective of hardware implementation of precoder selection, neural networks are preferable to the conventional symbolic algorithms in Sec. III-A due to the intrinsic characteristic of fixed complexity of neural networks. Once the networks are trained offline to learn the underlying connections between the channel (input) and the optimal precoder index (output), only the weight values at the end of machine learning are needed in the mobile devices for the precoder selection. The fixed number of weights implies the fixed required complexity.

In [10] and [11], precoders were designed for multiple antenna systems using convolutional neural networks. In [12] and [13], neural networks were used for joint antenna selection and precoder design. Antennas at the receiver side were selected in [12] and [13], which is mathematically described as the multiplication of channel matrix and antenna selection matrix that is composed of 1s at the selected antenna positions and 0s at the other positions, the latter matrix needs to be at the left side of the channel matrix. The antenna selection at the transmitter side is also described as the multiplication of channel matrix and antenna selection matrix, now the latter matrix needs to be at the right side of the channel matrix, thus antenna selection can be thought of as precoder selection. From the perspective of neural network-based implementation, antenna selection at both sides is the same in having the label vector with 1 at the optimal antenna set and 0s at the other sets. Here the number of sets is the number of combinations choosing a fixed number of antenna out of all antenna installed in the devices.

However, when the number of optimal precoders is not fixed, i.e., the optimal precoder in a codebook is unique for some channels and varying number of multiple precoders are simultaneously optimal for the other channels, the antenna selection approaches in [12] and [13] can not be directly applied. Besides, antenna selection criterion in [12] and [13] was spectral efficiency. We use error performance as the precoder selection criterion with maximum likelihood signal detection at the receiver side. Furthermore, in the previous works [12] and [13], network pruning techniques have not been adopted, which are known to reduce the number of weights, thereby reducing the fixed complexity of neural network without significant performance loss. The computational complexity of a pruned neural network is analyzed in the next section.

IV. COMPLEXITY ANALYSIS OF NEURAL NETWORK

In this section, we describe a fully connected neural network and the network pruning with emphasis on the computational complexity. We address the complexity of fully connected neural network in detail and the complexity reduction by using the pruning technique.

A. COMPLEXITY OF FULLY CONNECTED NEURAL NETWORK

We consider a fully connected neural network with $L+1$ layers. The number of nodes of the l-th layer is denoted as $N^{(l)}$, $l = 0, 1, \cdots, L$. The input and output layers are denoted as the 0-th and the L-th layer, respectively.

1) FORWARD COMPUTATION

The weighted sum vector $\mathbf{s}^{(l)} = [s_1^{(l)}\ s_2^{(l)} \cdots s_{N^{(l)}}^{(l)}]^T$ of the l-th layer, $l = 1, 2, \cdots, L$, is

$$\mathbf{s}^{(l)} = \mathbf{W}^{(l)}\mathbf{f}^{(l-1)} \tag{6}$$

where $\mathbf{f}^{(l-1)} = [f_1^{(l-1)}\ f_2^{(l-1)} \cdots f_{N^{(l-1)}}^{(l-1)}]^T$ denotes the features of the $(l-1)$-th layer, $\mathbf{W}^{(l)} \in \mathbb{R}^{N^{(l)} \times N^{(l-1)}}$ denotes the weight matrix of which (i, j)-th element $w_{i,j}^{(l)}$, $i = 1, 2, \cdots, N^{(l)}$, $j = 1, 2, \cdots, N^{(l-1)}$, denotes the connection strength between the i-th node of the l-th layer and the j-th feature of the $(l-1)$-th layer.

In the forward computation and backward learning phase, the data are processed in the unit of mini-batch that is composed of N_{MB} samples. The weighted sum vector $\mathbf{s}^{(l)}$ and the feature vector $\mathbf{f}^{(l-1)}$ corresponding to the m-th sample is denoted as $\mathbf{s}^{(l)[m]}$ and $\mathbf{f}^{(l-1)[m]}$, $m = 1, 2, \cdots, N_{\mathrm{MB}}$, respectively. Then (6) changes to

$$\mathbf{s}^{(l)[m]} = \mathbf{W}^{(l)}\mathbf{f}^{(l-1)[m]}. \tag{7}$$

We note that the mini-batch index is omitted in (7) for the sake of simplicity of notation. We also note that the weight matrix $\mathbf{W}^{(l)}$ is constant with regard to the sample index m, which means the weight update in the unit of mini-batch.

The mean vector $[\mu_1^{(l)}\ \mu_2^{(l)}\ \cdots\ \mu_{N^{(l)}}^{(l)}]^T$ and the standard deviation vector $[\sigma_1^{(l)}\ \sigma_2^{(l)}\ \cdots\ \sigma_{N^{(l)}}^{(l)}]^T$ of the mini-batch are calculated as

$$\mu_i^{(l)} = \frac{1}{N_{\mathrm{MB}}}\sum_{m=1}^{N_{\mathrm{MB}}} s_i^{(l)[m]}, \tag{8}$$

$$\sigma_i^{(l)} = \sqrt{\frac{1}{N_{\mathrm{MB}}}\sum_{m=1}^{N_{\mathrm{MB}}} (s_i^{(l)[m]} - \mu_i^{(l)})^2}, \tag{9}$$

where $s_i^{(l)[m]}$ denotes the i-th element of the column vector $\mathbf{s}^{(l)[m]}$.

Then the mini-batch $[\mathbf{s}^{(l)[1]}\ \mathbf{s}^{(l)[2]}\ \cdots\ \mathbf{s}^{(l)[N_{\mathrm{MB}}]}]$ is normalized, scaled and shifted to be $[\tilde{\mathbf{s}}^{(l)[1]}\ \tilde{\mathbf{s}}^{(l)[2]}\ \cdots\ \tilde{\mathbf{s}}^{(l)[N_{\mathrm{MB}}]}]$.

$$\tilde{s}_i^{(l)[m]} = \gamma_i^{(l)}\left(\frac{s_i^{(l)[m]} - \mu_i^{(l)}}{\sigma_i^{(l)}}\right) + \beta_i^{(l)} \tag{10}$$

where $\gamma_i^{(l)}$ and $\beta_i^{(l)}$ are optimized in the backward learning, minimizing the loss function of the network. Note that the bias parameters are not present in (6) and (7) due to the shifting parameter $\beta_i^{(l)}$ in (10) ([16], p.343).

Then, an activation function act$(\cdot)$ produces the features of the l-th layer $\mathbf{f}^{(l)} = [f_1^{(l)}\ f_2^{(l)} \cdots f_{N^{(l)}}^{(l)}]^T$.

$$f_i^{(l)[m]} = \text{act}(\tilde{s}_i^{(l)[m]}) = \begin{cases} \tilde{s}_i^{(l)[m]}, & \text{if } \tilde{s}_i^{(l)[m]} \geq 0 \\ \exp\{\tilde{s}_i^{(l)[m]}\} - 1, & \text{else,} \end{cases} \tag{11}$$

where we assumed exponential linear unit (ELU) activation function with parameter $\alpha = 1$ for the hidden layers ($l = 1, 2, \cdots, L-1$) ([16], p.336). The activation function of the output layer (the L-th layer) is assumed to be the softmax, producing the following features.

$$f_i^{(L)[m]} = \frac{\exp\{\tilde{s}_i^{(L)[m]}\}}{\sum_{k=1}^{N^{(L)}} \exp\{\tilde{s}_k^{(L)[m]}\}}. \tag{12}$$

2) BACKWARD LEARNING

Denoting the label for the supervised learning as $\mathbf{d} = [d_1\ d_2\ \cdots\ d_{N^{(L)}}]$, the cross entropy between the network output $\mathbf{f}^{(L)}$ and label $\mathbf{d}$ is estimated as

$$H(\mathbf{f}^{(L)}, \mathbf{d}) \approx \frac{1}{N_{\text{MB}}} \sum_{m=1}^{N_{\text{MB}}} \sum_{i=1}^{N^{(L)}} \left\{ -d_i^{[m]} \ln f_i^{(L)[m]} \right\}. \tag{13}$$

Including the L_2 regularization term to prevent the network overfitting, we have the following loss function.

$$f_{\text{loss}} = H(\mathbf{f}^{(L)}, \mathbf{d}) + \frac{\lambda}{2} \sum_{l=1}^{L} \|\mathbf{W}^{(l)}\|_F^2. \tag{14}$$

We denote the differentiation of the loss function (14) with regard to $\tilde{s}_i^{(l)}$ and $s_i^{(l)}$ as $\tilde{\delta}_i^{(l)}$ and $\delta_i^{(l)}$, respectively, and their estimated values considering the m-th sample are denoted as $\tilde{\delta}_i^{(l)[m]}$ and $\delta_i^{(l)[m]}$. Then we have

$$\delta_i^{(l)[m]} = \tilde{\delta}_i^{(l)[m]} \frac{\gamma_i^{(l)}}{\sigma_i^{(l)}}. \tag{15}$$

Once the estimated values $\tilde{\delta}_i^{(l)[m]}$ and $\delta_i^{(l)[m]}$, $m = 1, 2, \cdots, N_{\text{MB}}$ are available, the following partial derivatives can be estimated.

$$\frac{\partial f_{\text{loss}}}{\partial \gamma_i^{(l)}} \approx \frac{1}{N_{\text{MB}}} \sum_{m=1}^{N_{\text{MB}}} \tilde{\delta}_i^{(l)[m]} \frac{(s_i^{(l)[m]} - \mu_i^{(l)})}{\sigma_i^{(l)}}. \tag{16}$$

$$\frac{\partial f_{\text{loss}}}{\partial \beta_i^{(l)}} \approx \frac{1}{N_{\text{MB}}} \sum_{m=1}^{N_{\text{MB}}} \tilde{\delta}_i^{(l)[m]}. \tag{17}$$

$$\frac{\partial f_{\text{loss}}}{\partial w_{i,j}^{(l)}} \approx \frac{1}{N_{\text{MB}}} \sum_{m=1}^{N_{\text{MB}}} \delta_i^{(l)[m]} f_j^{(l-1)[m]} + \lambda w_{i,j}^{(l)}. \tag{18}$$

The above partial derivatives are used to update the learnable parameters for each mini-batch adopting various optimization techniques such as stochastic gradient descent (SGD) and adaptive moment estimation (Adam).

Assuming the loss function (14) and softmax activation at the L-th layer, we have

$$\tilde{\delta}_i^{(L)[m]} = f_i^{(L)[m]} - d_i^{[m]}. \tag{19}$$

Then $\delta_i^{(L)[m]}$ can be calculated using (15), and all the other $\tilde{\delta}_i^{(l)[m]}$ and $\delta_i^{(l)[m]}$, $l = 1, 2, \cdots, L-1$, are calculated as follows, which is known as the back-propagation.

$$\tilde{\delta}_i^{(l)[m]} = \begin{cases} \sum_{j=1}^{N^{(l+1)}} w_{j,i}^{(l+1)} \delta_j^{(l+1)[m]}, & \text{if } \tilde{s}_i^{(l)[m]} \geq 0 \\ \exp\{\tilde{s}_i^{(l)[m]}\} \sum_{j=1}^{N^{(l+1)}} w_{j,i}^{(l+1)} \delta_j^{(l+1)[m]}, & \text{else} \end{cases} \tag{20}$$

where we assumed ELU activation in (11).

3) INFERENCE

In the phase of inference, there is no concept of mini-batch. Thus the mean vector and the standard deviation vector can not be calculated as in (8) and (9). Instead, the two vectors need to be estimated in the learning phase, for an example, by exponential moving average as follows.

$$\mu_i^{(l)\text{infer}} \leftarrow \beta_\mu \mu_i^{(l)\text{infer}} + (1 - \beta_\mu)\mu_i^{(l)}, \tag{21}$$

$$\sigma_i^{(l)\text{infer}} \leftarrow \beta_\sigma \sigma_i^{(l)\text{infer}} + (1 - \beta_\sigma)\sigma_i^{(l)} \tag{22}$$

where $\mu_i^{(l)}$ and $\sigma_i^{(l)}$ are the ones in (8) and (9). The initial values of $\mu_i^{(l)\text{infer}}$ and $\sigma_i^{(l)\text{infer}}$ are set to be 0s.

Once the two values $\mu_i^{(l)\text{infer}}$ and $\sigma_i^{(l)\text{infer}}$ are calculated, and $\gamma_i^{(l)}$ and $\beta_i^{(l)}$ are fixed at the end of the machine learning, the process in (10) can be performed as follows, requiring a single multiplication and a single addition.

$$\tilde{s}_i^{(l)} = a_i^{(l)} s_i^{(l)} + b_i^{(l)} \tag{23}$$

where $a_i^{(l)}$ and $b_i^{(l)}$ are calculated offline before the inference as follows.

$$a_i^{(l)} = \frac{\gamma_i^{(l)}}{\sigma_i^{(l)\text{infer}}}, \tag{24}$$

$$b_i^{(l)} = \beta_i^{(l)} - \frac{\gamma_i^{(l)} \mu_i^{(l)\text{infer}}}{\sigma_i^{(l)\text{infer}}}. \tag{25}$$

The complexity of the l-th layer is $N^{(l)} \times N^{(l-1)}$ multiplications in (6), about $N^{(l)}/2$ exponential function calculations in (11) assuming half of $\tilde{s}_i^{(l)}, i = 1, 2, \cdots, N^{(l)}$ are less than 0, and $N^{(l)}$ multiplications in (23). Thus, translating one exponential function calculation into 2 multiplications for the sake of simplicity, the overall complexity of the fully connected neural network in the phase of inference is given as

$$\text{FC} - \text{Complexity} = \sum_{l=1}^{L} \left\{ N^{(l)} \times \left(N^{(l-1)} + 2 \right) \right\}. \tag{26}$$

B. COMPLEXITY OF PRUNED NEURAL NETWORK

In order to decrease the complexity of the fully connected network, network pruning on completion of the training can be used [14]. The number of connections between the l-th and the $(l-1)$-th layers is $N^{(l)} \times N^{(l-1)}$, which is the dimension of the weight matrix $\mathbf{W}^{(l)}$.

We set a threshold as follows.

$$\mathrm{Th}_s^{\mathrm{prune}} = |w_{(s)}^{(l)}|, \quad s \in [0, 1] \tag{27}$$

where $w_{(s)}^{(l)}$ denotes the $(N^{(l)} \times N^{(l-1)} \times s)$-th smallest element among $|w_{i,j}^{(l)}|, i = 1, 2, \cdots, N^{(l)}, j = 1, 2, \cdots, N^{(l-1)}$. Then, the entries of $\mathbf{W}^{(l)}$ are set as

$$w_{i,j}^{(l)} = \begin{cases} 0, & \text{if } |w_{i,j}^{(l)}| \le \mathrm{Th}_s^{\mathrm{prune}} \\ w_{i,j}^{(l)}, & \text{else.} \end{cases} \tag{28}$$

The zero-forcing of entries means the pruning of their corresponding connections. Now, the number of non-zero elements of $\mathbf{W}^{(l)}$ is reduced to $N^{(l)} \times N^{(l-1)}(1-s)$, thus s is referred to as sparsity.

Obviously, increased sparsity induces more loss in the network accuracy. The network can be retrained after pruning so that the survived non-zero weights compensate for the zero-forced weights. Note that once a connection is pruned, its weight is not updated in the retraining. In [15], it was shown that the pruning and retraining can be performed iteratively in order to successfully recover the pruning-induced degradation. The sparsity at the i-th iteration, $i = 0, 1, \cdots, N_P$, is set as

$$s_i = s_t + (s_0 - s_t)\left(1 - \frac{i}{N_P}\right)^3. \tag{29}$$

where s_0 and s_t are the initial and the target sparsity values, respectively.

V. PROPOSED NN-BASED PRECODER SELECTION

In this section, we describe our proposed NN-based technique that includes (1) the use of soft label, (2) the dimension reduction of the input data, (3) regularization optimization considering pruning as wells as overfitting.

A. SOFT LABEL

In order to train the neural network, we need to generate labels $\mathbf{d} = [d_1\ d_2\ \cdots\ d_{N^{(L)}}]$ for each channel realization. However, the label generation is not straightforward. Table 1 shows the codebook of LTE(Long-Term Evolution)-Advanced systems when the number of transmit antenna and the number of data streams are 4 and 2, respectively.

If the optimal precoder is unique for a given channel $\mathbf{H}$, the label would be a unit vector of length 16 with 1 at the optimal precoder index and 0s at the other indices. Unfortunately, there are cases when 2 and 3 precoders are simultaneously optimal.

Let us assume that $\mathbf{F}_5$ is the optimal precoder for a given channel $\mathbf{H}$ and $\Delta x^{\min|\mathbf{F}_5} = [-1\ \ 0]^T$ minimizes the norm. Then, we have

$$\begin{aligned} \left\|\mathbf{HF}_5\begin{bmatrix}-1\\0\end{bmatrix}\right\| &= \frac{1}{2\sqrt{2}}\left\|\mathbf{H}\begin{bmatrix}-\sqrt{2}\\-1-j\\-j\sqrt{2}\\1-j\end{bmatrix}\right\| \\ &= \frac{1}{2\sqrt{2}}\left\|\mathbf{H}\begin{bmatrix}-\sqrt{2}\\-1-j\\-j\sqrt{2}\\1-j\end{bmatrix}\times j\right\| \\ &= \frac{1}{2\sqrt{2}}\left\|\mathbf{H}\begin{bmatrix}-j\sqrt{2}\\1-j\\\sqrt{2}\\1+j\end{bmatrix}\right\| \\ &= \left\|\mathbf{HF}_7\begin{bmatrix}0\\1\end{bmatrix}\right\|. \end{aligned} \tag{30}$$

Equation (30) shows that if $\mathbf{F}_5$ is the optimal precoder with minimizer $\Delta x^{\min|\mathbf{F}_5} = [-1\ \ 0]^T$, then $\mathbf{F}_7$ can be also optimal with minimizer $\Delta x^{\min|\mathbf{F}_7} = [0\ \ 1]^T$. There are even cases when 3 precoders are optimal.

$$\begin{aligned} \left\|\mathbf{HF}_1\begin{bmatrix}0\\1\end{bmatrix}\right\| &= \frac{1}{2}\left\|\mathbf{H}\begin{bmatrix}1\\-1\\-1\\1\end{bmatrix}\right\| \\ &= \frac{1}{2}\left\|\mathbf{H}\begin{bmatrix}1\\-1\\-1\\1\end{bmatrix}\times(-1)\right\| \\ &= \frac{1}{2}\left\|\mathbf{H}\begin{bmatrix}-1\\1\\1\\-1\end{bmatrix}\right\| \\ &= \left\|\mathbf{HF}_3\begin{bmatrix}0\\1\end{bmatrix}\right\| \\ &= \left\|\mathbf{HF}_{11}\begin{bmatrix}1\\0\end{bmatrix}\right\| \end{aligned} \tag{31}$$

Equation (31) shows that three precoders $\mathbf{F}_1$, $\mathbf{F}_3$, and $\mathbf{F}_{11}$ can be simultaneously optimal for a given channel $\mathbf{H}$, assuming $\Delta x^{\min|\mathbf{F}_1} = [0\ \ 1]^T$, $\Delta x^{\min|\mathbf{F}_3} = [0\ \ 1]^T$, $\Delta x^{\min|\mathbf{F}_{11}} = [1\ \ 0]^T$. We note that equation (31) does not necessarily mean the simultaneous optimality of the three precoders every time when $\mathbf{F}_1$ with $\Delta x^{\min|\mathbf{F}_1} = [0\ \ 1]^T$ is optimal, because $\Delta x^{\min|\mathbf{F}_3}$ can be different from $[0\ \ 1]^T$, and $\Delta x^{\min|\mathbf{F}_{11}}$ can be different from $[1\ \ 0]^T$. We generated $5,000,000$ random $\mathbf{H}$ to check how often multiple optimizers occur. When 4-QAM is assumed, the optimal precoders were unique for about 97.6825% of $\mathbf{H}$, about 2.3114% of $\mathbf{H}$ had 2 optimal precoders, about 0.0065% of $\mathbf{H}$ had 3 optimal precoders, and there were no cases when a $\mathbf{H}$ had more than 3 optimal precoders.

For a given $\mathbf{H}$, $\mathbf{F}_n$, and $\mathbf{y}$, the vector error probability of maximum likelihood detection is given as

$$\begin{aligned} \mathrm{P}(\mathbf{x} \neq \hat{\mathbf{x}}_{\mathrm{ML}}) &\le \frac{1}{2}\mathrm{E}_{\mathbf{x}}\left[\exp\left\{-c\left\|\mathbf{HF}_n(\mathbf{x}-\hat{\mathbf{x}}_{\mathrm{ML}})\right\|^2\right\}\right] \\ &\approx \frac{1}{2}\exp\left\{-c\min_{\mathbf{x}}\left\|\mathbf{HF}_n(\mathbf{x}-\hat{\mathbf{x}}_{\mathrm{ML}})\right\|^2\right\} \\ &= \frac{1}{2}\exp\left\{-c d_{\min}(\mathbf{HF}_n)^2\right\} \end{aligned} \tag{32}$$

TABLE 1. Codebook of LTE-A systems for 4 transmit antennas and 2 data streams.

$$\mathbf{F}_1=\tfrac{1}{2}\begin{bmatrix}1&1\\1&-1\\1&-1\\1&1\end{bmatrix},\mathbf{F}_2=\tfrac{1}{2}\begin{bmatrix}1&-j\\j&1\\-1&-j\\-j&1\end{bmatrix},\mathbf{F}_3=\tfrac{1}{2}\begin{bmatrix}1&-1\\-1&1\\1&1\\-1&-1\end{bmatrix},\mathbf{F}_4=\tfrac{1}{2}\begin{bmatrix}1&j\\j&1\\-1&j\\j&1\end{bmatrix}$$

$$\mathbf{F}_5=\tfrac{1}{2\sqrt{2}}\begin{bmatrix}\sqrt{2}&-1-j\\1+j&j\sqrt{2}\\j\sqrt{2}&-1+j\\-1+j&\sqrt{2}\end{bmatrix},\mathbf{F}_6=\tfrac{1}{2\sqrt{2}}\begin{bmatrix}\sqrt{2}&1-j\\-1+j&-j\sqrt{2}\\-j\sqrt{2}&1+j\\1+j&\sqrt{2}\end{bmatrix},\mathbf{F}_7=\tfrac{1}{2\sqrt{2}}\begin{bmatrix}\sqrt{2}&-j\sqrt{2}\\-1-j&1-j\\j\sqrt{2}&\sqrt{2}\\1-j&1+j\end{bmatrix},\mathbf{F}_8=\tfrac{1}{2\sqrt{2}}\begin{bmatrix}\sqrt{2}&j\sqrt{2}\\1-j&-1-j\\-j\sqrt{2}&\sqrt{2}\\-1-j&-1+j\end{bmatrix}$$

$$\mathbf{F}_9=\tfrac{1}{2}\begin{bmatrix}1&1\\1&1\\-1&1\\-1&1\end{bmatrix},\mathbf{F}_{10}=\tfrac{1}{2}\begin{bmatrix}1&-j\\j&-1\\1&j\\j&1\end{bmatrix},\mathbf{F}_{11}=\tfrac{1}{2}\begin{bmatrix}1&-1\\-1&-1\\-1&1\\1&1\end{bmatrix},\mathbf{F}_{12}=\tfrac{1}{2}\begin{bmatrix}1&1\\-j&j\\1&1\\-j&j\end{bmatrix}$$

$$\mathbf{F}_{13}=\tfrac{1}{2}\begin{bmatrix}1&1\\1&1\\1&-1\\-1&1\end{bmatrix},\mathbf{F}_{14}=\tfrac{1}{2}\begin{bmatrix}1&-1\\1&1\\-1&1\\1&1\end{bmatrix},\mathbf{F}_{15}=\tfrac{1}{2}\begin{bmatrix}1&1\\-1&1\\1&1\\1&-1\end{bmatrix},\mathbf{F}_{16}=\tfrac{1}{2}\begin{bmatrix}1&-1\\-1&1\\-1&-1\\-1&-1\end{bmatrix}$$

where $\mathbf{x}$ denotes the true transmitted signal vector, $\hat{\mathbf{x}}_{\mathrm{ML}}$ denotes the maximum likelihood detected symbol vector, c is a constant reflecting the number of streams and power of the adopted constellation in (1), and $\mathrm{E}_{\mathbf{x}}\{\cdot\}$ denotes the expectation with regard to random $\mathbf{x}$ [18].

Considering (32), we propose to use the following soft labels.

$$d_i = \frac{\exp\left\{d_{\min}(\mathbf{HF}_i)^2\right\}}{\sum_{n=1}^{N}\exp\left\{d_{\min}(\mathbf{HF}_n)^2\right\}}, \quad i = 1, 2, \cdots, N^{(L)}. \tag{33}$$

We named it soft label because the values can assume continuous values in the closed interval [0, 1]. Note that the soft label can be considered as softmax of $d_{\min}(\mathbf{HF}_i)^2, i = 1, 2, \cdots, N^{(L)}$.

After the completion of training, the inference is done as follows regardless of the number of optimal precoders.

$$i^{\mathrm{opt}} = \arg\max_i f_i^{(L)}. \tag{34}$$

Then $\mathbf{F}_{i^{\mathrm{opt}}}$ is used for the precoding.

We note that $N^{(L)} = 16$ when the soft label (33) is used. The total number of combinations including the 3 cases of the unique, 2 simultaneous, and 3 simultaneous optimal precoders, is $\sum_{k=1}^{3}\binom{16}{k} = 696$. Thus $N^{(L)}$ would be as large as 696 if one-hot vectors are used as the labels as in [12] and [13].

B. DIMENSION REDUCTION

Using QR decomposition $\mathbf{H} = \mathbf{QR}$, we have

$$d_{\min}(\mathbf{HF}_i)^2 = d_{\min}(\mathbf{RF}_i)^2. \tag{35}$$

The channel matrix $\mathbf{H} \in \mathbb{C}^{N_R \times N_T}$ is composed of $N_R \times N_T \times 2$ real numbers. If $N_R = N_T = 4$, $\mathbf{H}$ is composed of 32 real numbers, which is the input data of the neural network. Since the diagonal entries of $\mathbf{R}$ are real numbers, $\mathbf{R}$ is composed of only 16 real numbers, hence if $\mathbf{R}$ is used as the input data instead of $\mathbf{H}$, the number of nodes of the input layer $N^{(0)}$ is reduced from 32 to 16. The dimension reduction by using $\mathbf{R}$ not only reduces the network complexity but also offers additional gain of network accuracy, which will be demonstrated in the simulation section.

C. REGULARIZATION AND PRUNING

The main purpose of regularization is to prevent the neural network from being overfitted to the training data. By forgetting some weight information drawn from the training data, the network can perform better for the test data. L_2 regularization in the loss function (14) leads to the partial derivative (18), thus the weights are updated as follows, assuming SGD.

$$w_{i,j}^{(l)} \leftarrow (1-\eta\lambda)w_{i,j}^{(l)} - \frac{\eta}{N_{\mathrm{MB}}}\sum_{m=1}^{N_{\mathrm{MB}}}\delta_i^{(l)[m]}f_j^{(l-1)[m]} \tag{36}$$

where η is the learning rate. If L_1 regularization is adopted instead of L_2, i.e., $\frac{\lambda}{2}\|\mathbf{W}^{(l)}\|_F^2$ is replaced by $\lambda\sum_{i,j}|w_{i,j}^{(l)}|$, the above learning changes to

$$w_{i,j}^{(l)} \leftarrow w_{i,j}^{(l)} - \eta\lambda \times \mathrm{sgn}(w_{i,j}^{(l)}) - \frac{\eta}{N_{\mathrm{MB}}}\sum_{m=1}^{N_{\mathrm{MB}}}\delta_i^{(l)[m]}f_j^{(l-1)[m]} \tag{37}$$

where $\mathrm{sgn}(\cdot)$ is the sign function that outputs 1 when the argument is non-negative and outputs -1 otherwise. From (36) and (37), it can be seen that the weights are decreased in absolute value by the presence of λ. In (36), the decreased amount is in proportion to the $|w_{i,j}^{(l)}|$ while the decreased amount is a constant $\eta\lambda$ regardless of $|w_{i,j}^{(l)}|$ in (37). Consequently, the small weights get close to 0 more quickly by L_1 regularization than by L_2 regularization. Therefore, from the perspective of pruning that set small weights to 0, L_1 regularization is more suitable than L_2 regularization. We can say that the regularization plays a role in the network pruning as well as in the overfitting problem. Therefore, we optimize the regularization method and amount considering not only overfitting but also the network pruning.

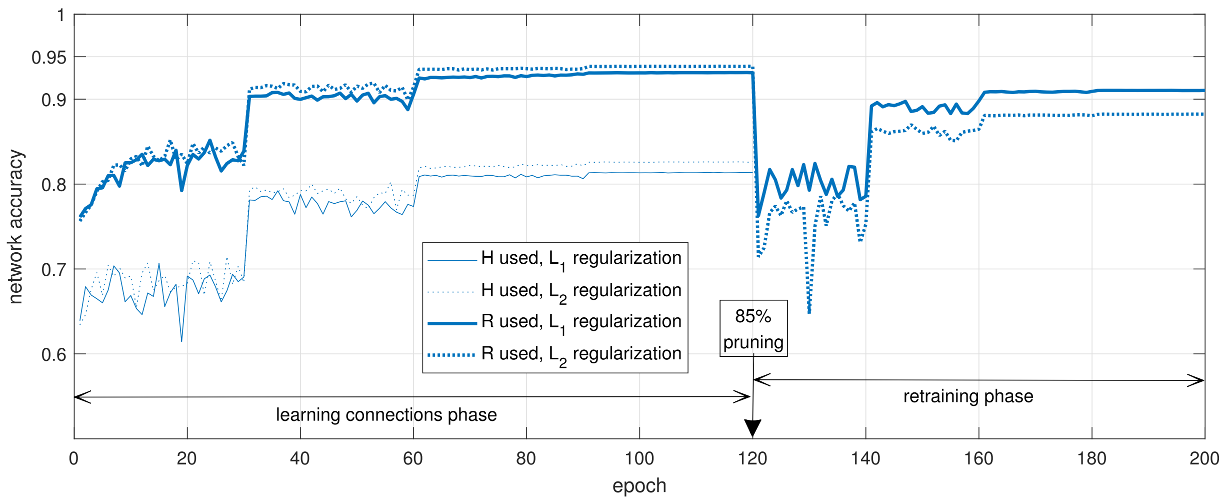


FIGURE 1. The network accuracy versus epoch: the network is trained in three phases. The learning connections (or training) phase is from epoch 1 to epoch 120, on epoch 120 the unimportant (85% in this example) connections are pruned, and the survived (15% in this example) connections are retrained to recover the network accuracy degraded due to the pruning. Note that retraining is done only for networks with R used as the input data.

VI. SIMULATIONS

In this section, we demonstrate the efficacy of the proposed NN-based precoder selection. We consider $N_T = N_R = 4$ and $N_S = 2$ multiple antenna systems. The constellation is 4-QAM. The LTE-A codebook corresponding to the parameters $N_T = 4$ and $N_S = 2$ is given in Table 1 [17]. We considered neural network with 6 layers ($L = 5$) and the number of nodes are as follows: $N^{(0)} = 16$ assuming the use of $\mathbf{R}$, $N^{(1)} = N^{(2)} = N^{(3)} = N^{(4)} = 80$, $N^{(5)} = 16$. We adopted the ELU activation and the batch normalizations in each layer right before the activations as described in Section IV-A. The parameters for exponential moving averages in (21) and (22) are $\beta_\mu = \beta_\sigma = 0.99$. We generated 5,000,000 independent identical complex Gaussian channels of which 80% is used for training and 20% for validation. The Adam optimizer ([16], page 356) is used for both training (or learning connections phase) and retraining. The learning schedule in the phase of learning connections is

$$\eta^{\text{train}} = \begin{cases} 5 \times 10^{-3}, & 1 \leq \text{epoch} < 30 \\ 5 \times 10^{-4}, & 30 \leq \text{epoch} < 60 \\ 5 \times 10^{-5}, & 60 \leq \text{epoch} < 90 \\ 5 \times 10^{-6}, & 90 \leq \text{epoch} < 120. \end{cases} \quad (38)$$

Fig. 1 shows the accuracy versus epoch of the network. The parameter $\lambda = 10^{-5}$ is used for both L_1 and L_2 regularizations. We can observe staircase shaped network accuracy according to the learning rate schedule in (38).

A. DIMENSION REDUCTION

Fig. 1 shows the improvement of network accuracy by the dimension reduction by using $\mathbf{R}$ instead of $\mathbf{H}$. For both cases of L_1 and L_2 regularizations, the dimension reduction offers more than 10% improvement of the network accuracy. The dimension reduction not only reduces the number of connections, but also offers significant improvement of network accuracy. We can also observe that L_2 regularization outperforms L_1 regularization in the phase of learning connections. The accuracy gap is about $0.9387 - 0.9311 = 0.76\%$ when $\mathbf{R}$ is used, which means that L_2 regularization prevents the network overfitting more successfully than L_1 regularization in the training of fully connected neural network.

B. REGULARIZATION AND PRUNING

Fig. 2 compares L_1 and L_2 regularizations in terms of network accuracy versus target sparsity of the network pruning. Learning rate schedule for the retraining is

$$\eta^{\text{retrain}} = \begin{cases} 5 \times 10^{-3}, & 121 \leq \text{epoch} < 140 \\ 5 \times 10^{-4}, & 140 \leq \text{epoch} < 160 \\ 5 \times 10^{-5}, & 160 \leq \text{epoch} < 180 \\ 5 \times 10^{-6}, & 180 \leq \text{epoch} \leq 200. \end{cases} \quad (39)$$

We note that $s_0 = s_t$, thus the fully connected network is pruned once, then the sparse network is retrained. The batch normalization parameters were not pruned. When $s_t \leq 0.7$, L_2 regularization is more suitable to the pruning than L_1 regularization, but when $s_t > 0.7$, L_1 regularization outperforms L_2 regularization. The network accuracy recovery by retraining after pruning 85% connections is given in Fig. 1. From Fig. 1 and Fig. 2, we can conclude that L_2 regularization is a better choice than L_1 regularization when a fully connected neural network is to be pruned mildly, however, L_1 regularization is preferable to L_2 regularization when a fully connected neural network needs to be severely pruned.

C. COMPLEXITY AND ERROR PERFORMANCE

Fig. 3 and Fig. 4 compare the conventional SD-based optimal precoder selection and the proposed NN-based precoder selection in terms of complexity and the error performance. We note that new channels that were not used for training, were generated for error performance

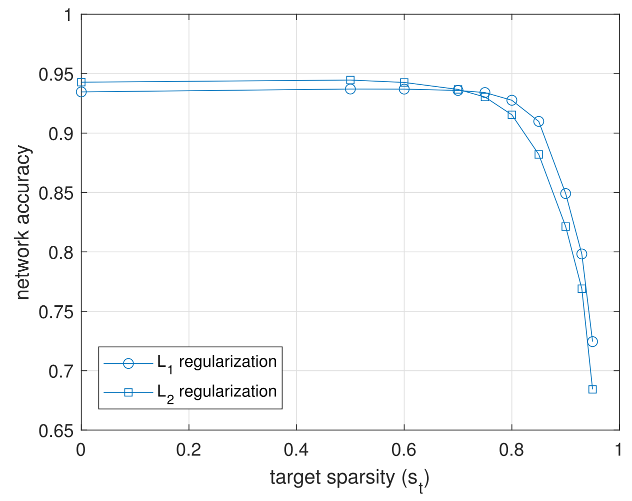


FIGURE 2. Network accuracy versus target sparsity.

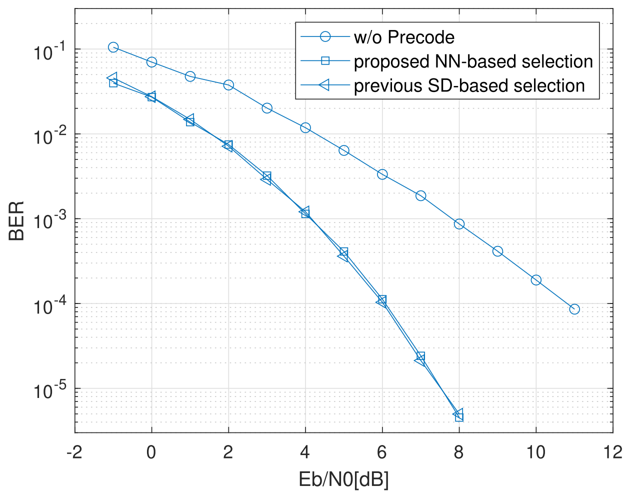


FIGURE 4. BER(Bit Error Rate) performance of the precoded multiple antenna system. $N_S = 2$, $N_T = N_R = 4$, 4-QAM modulation.

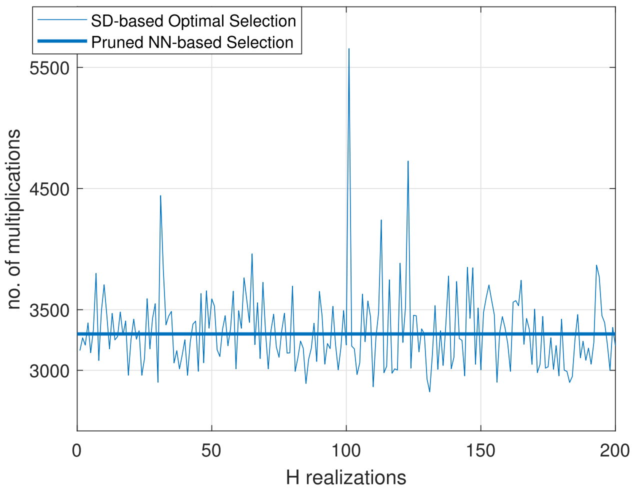


FIGURE 3. Complexity comparison of precoder selection techniques: the previous SD-based optimal selection and the proposed NN-based selection.

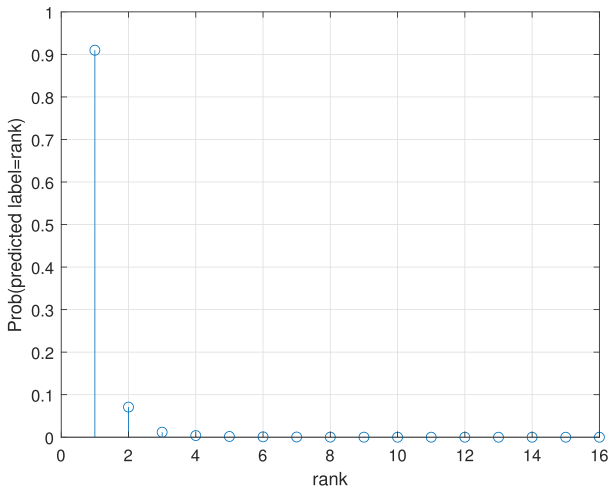


FIGURE 5. The optimality rank of precoders selected by the proposed NN-based method.

measurement. The fixed complexity and error performance in Fig. 3 and Fig. 4 are the ones after the retraining. As can be seen in Fig. 3, the complexity of the SD-based selection fluctuates quite severely, while the complexity of the proposed pruned NN-based selection is fixed at $[(16+2)\times 80 + \{(80+2)\times 80\}\times 3 + (80+2)\times 16](1-0.85) \approx 3,365$ multiplications, where $(1-0.85)$ accounts for the pruning with $s_t = 0.85$. We generated 100,000 channel matrices, and the average complexity of SD-based selection was about $3,317$ multiplications. As can be seen in Fig. 4, the proposed NN-based selection offers a near-optimal error performance that is obtained by averaging over all of the generated channels. As we noted in Section IV-B, pruning incurs performance degradation, however, the degradation in the precoded multi-antenna system is not significant. The almost negligible degradation by the pruning is due to the fact that the second or third best precoders are good enough in terms of the error performance. Fig. 5 shows the ranks of the precoders chosen by the proposed neural network. About 91% of precoders are the optimal, and about 7% of precoders are the second best out of the 16 precoders.

From the simulation results in Fig. 3 and Fig. 4, we argue that the proposed NN-based selection requires a fixed complexity which is about the average complexity of the conventional SD-based selection, allowing only a negligible degradation in the error performance.

VII. CONCLUSION

In this paper, we proposed a neural network based fixed complexity precoder section technique. The technical novel ingredients of the proposed method are (1) the use of soft label, (2) the dimension reduction of the input data, and (3) the optimization of regularization taking into account network pruning and retraining as well as the network overfitting. The fixed complexity of the proposed connectionist NN-based selection, which is amenable to hardware implementation, is close to the average complexity of the previous SD-based symbolic algorithm of which worst case complexity is the several times of the average complexity. The error performance of the proposed technique is close to the optimal performance of the previous SD-based method.

REFERENCES

[1] J.-H. Lee, S.-Y. Jung, and D. Park, "Simplified maximum-likelihood precoder selection for spatial multiplexing systems," *IEEE Trans. Veh. Technol.*, vol. 59, no. 9, pp. 4628–4634, Nov. 2010.

[2] C. Lin and W. Wu, "QRD-based antenna selection for ML detection of spatial multiplexing MIMO systems: Algorithms and applications," *IEEE Trans. Veh. Tech.*, vol. 60, no. 7, pp. 3178–3190, Sep. 2011.

[3] W. Hwang, J. Park, J. Kim, and H.-S. Lim, "Optimal precoder selection for spatially multiplexed multiple-input multiple-output systems with maximum likelihood detection: Exploiting the concept of sphere decoding," *IEEE Access*, vol. 8, pp. 223859–223868, 2020.

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

[5] N. Shlezinger, N. Farsad, Y. C. Eldar, and A. J. Goldsmith, "ViterbiNet: A deep learning based Viterbi algorithm for symbol detection," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3319–3331, May 2020.

[6] M. Khani, M. Alizadeh, J. Hoydis, and P. Fleming, "Adaptive nueral signal detection for massive MIMO," *IEEE Trans. Wireless Commun.*, vol. 19, no. 8, pp. 5635–5648, Aug. 2020.

[7] H. Ye, G. Y. Li, and B.-H. 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.

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

[9] V. Raj and S. Kalyani, "Backpropagating through the air: Deep learning at physical layer without channel models," *IEEE Commun. Lett.*, vol. 22, no. 11, pp. 2278–2281, Nov. 2018.

[10] A. M. Elbir, "CNN-based precoder and combiner design in mmWave MIMO systems," *IEEE Commun. Lett.*, vol. 23, no. 7, pp. 1240–1243, May 2019.

[11] X. Bao, W. Feng, J. Zheng, and J. Li, "Deep CNN and equivalent channel based hybrid precoding for mmWave massive MIMO systems," *IEEE Access*, vol. 8, pp. 19327–19335, 2020.

[12] A. M. Elbir and K. V. Mishra, "Joint antenna selection and hybrid beamformer design using unquantized and quantized deep learning networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 1677–1688, Mar. 2020.

[13] S. Khalid, W. Bin Abbas, and F. Khalid, "Deep learning based joint precoder design and antenna selection for partially connected hybrid massive MIMO systems," 2021, *arXiv:2102.01495*.

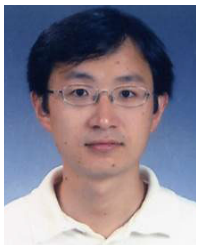
[14] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural networks," 2015, *arXiv:1506.02626*.

[15] M. Zhu and S. Gupta, "To prune, or not to prune: Exploring the efficacy of pruning for model compression," 2017, *arXiv:1710.01878*.

[16] A. Geron, *Hands-on Machine Learning With Scikit-Learn, Keras, and Tensorflow*, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, Sep. 2019.

[17] *Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Channels and Modulation (Release 10)*, document TS 36.211 V.10.0.0, 3GPP, Dec. 2010.

[18] Y. Cho, J. Kim, C. Kang, and W. Yang, *MIMO-OFDM Wireless Communications With MATLAB*. Singapore: Wiley, 2010.



JAEKWON KIM received the B.S. *(summa cum laude)* and M.S. degrees in electrical engineering from Chung-Ang University, Seoul, South Korea, in 1995 and 2000, respectively, and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Texas at Austin, in May 2004. From August 2004 to August 2005, he was at the Samsung Advanced Institute of Technology (SAIT), Gyeonggido, South Korea, where he did research on 4G cellular systems. He was a Visiting Scholar at Purdue University, from September 2014 to August 2015. Since September 2005, he has been a Faculty Member with the Division of Software, Yonsei University, Wonju, South Korea. His research interests include receiver techniques for wireless communication systems, unequal error protection (UEP) techniques for wireless multimedia streaming systems, and machine learning based physical layer techniques. He received the Texas Telecom Consortium (TexTEC) Fellowship, from Fall 2001 to Spring 2002 and from Fall 2002 to Spring 2003.

HYO-SANG LIM (Member, IEEE) received the B.S. degree from Yonsei University, South Korea, and the M.S. and Ph.D. degrees from the Korea Advanced Institute of Science and Technology, all in computer science. He was a Postdoctoral Research Fellow at Purdue University, from 2007 to 2011. He is currently an Associate Professor with the Division of Software, Yonsei University, Wonju, South Korea. His research interests include databases, database systems, data streams, sensor networks, database security, data trustworthiness, and applications of machine learning. He is a member of the Association for Computing Machinery (ACM).