IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 6 September 2023, accepted 25 September 2023, date of publication 9 October 2023, date of current version 13 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3322679

RESEARCH ARTICLE

Neural Network With Binary Cross Entropy for Antenna Selection in Massive MIMO Systems: Convolutional Neural Network Versus Fully Connected Network

JAEKWON KIM AND HYO-SANG LIM, (Member, IEEE)
Division of Software, Yonsei University, Wonju 26493, Republic of Korea
Corresponding author: Hyo-Sang Lim (hyosang@yonsei.ac.kr)

ABSTRACT This paper makes several contributions to antenna selection techniques for massive multiple-input multiple-output (mMIMO) systems using artificial neural networks. First, binary cross entropy is adopted as the loss function for network training instead of the conventional cross entropy, which reduces the number of nodes in the output layer from $\binom{N_R}{N_{RS}}$ to N_R, where N_R and N_{RS} are the number of candidate antennas and the number of selected antennas, respectively. In mMIMO systems, which have a large number of antennas, binary cross entropy is essential. We also demonstrate that the channel matrix is practically sufficient information to train the network, excluding the signal-to-noise ratio (SNR) factor present in the capacity formula. Since a single label is generated for a given mMIMO channel regardless of SNR, the size of training data is reduced significantly. When the channel matrix without pre-processing is inputted into a neural network for feature extraction, which is referred to as pure connectionist feature extraction, we show that the convolutional neural network (CNN) extracts features more successfully than the fully connected network (FCN). We also show that hybrid feature extraction, in which features are first extracted symbolically from the channel matrix and then connectionist features are extracted from the symbolic features, offers significant performance improvement over pure connectionist feature extraction from the raw data. However, when features are extracted in a hybrid manner, FCN achieves marginally better performance than CNN, contrary to the pure connectionist feature extraction. Finally, when the networks in the hybrid feature extraction are pruned to be suitable for deployment in mobile devices, we show that FCN is a better choice, as it is more robust to severe pruning than CNN. We conducted computer simulations to demonstrate the effectiveness of the proposed approaches.

INDEX TERMS Antenna selection, neural network, massive MIMO.

I. INTRODUCTION

When 30-300 GHz frequency band, known as mmWave band, is used with multiple-input multiple-output (MIMO) systems for wireless communications, wide bandwidth and multiple data streams enable higher data rates. However, the benefit comes with an increased cost of the RF chains that include analog-to-digital converters (ADCs) of which operating speed is in proportion to the spectral bandwidth [1]. Antenna selection techniques retain the benefits of massive MIMO (mMIMO) systems by activating only a subset of antennas instead of full antenna arrays, thereby requiring a small number of the costly RF chains [2].

Various symbolic approaches to antenna selection have been proposed. In [3], the binary particle swarm optimization technique was used to select a subset of transmit and receive antennas. In [4], the lattice reduction-based selection method was proposed assuming maximum likelihood detection at

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

the receiver side. In [5], the branch and bound method was used for efficient antenna selection, pruning away unnecessary branches of the search tree based on the concept of maximizing the minimum singular value. In [6], it was shown that most of the channel capacity can be achieved by acquiring a small portion of channel state information. In [7], antenna selection techniques are proposed based on greedy algorithms. In [8], an efficient antenna selection method was proposed by using the concept of minimizing cross-entropy which originated in machine learning.

Besides the aforementioned symbolic approaches, connectionist approaches have been proposed using neural networks in selecting antennas. The main benefit of the artificial neural network-based technique is that a large portion of the computational burden of antenna selection is shifted to off-line training [9]. Furthermore, the complexity in the inference stage of the antenna selection is fixed, thus amenable to hardware implementation fulfilling delay constraints [10]. Two neural network architectures have been used for the purpose of antenna selection: the convolutional neural network (CNN) [1] and the fully connected network (FCN) [9], [11]. Although other machine learning techniques such as k-nearest neighbors (k-NN) and support vector machine (SVM) algorithms can also be applied to antenna selection [12], we focus on CNN and FCN in this paper. Unlike in [13] and [14], where the antenna selection and beamformer design were jointly addressed using the neural network, we focus on the antenna selection.

In this paper, we first show that the adoption of binary cross entropy is quite essential in training the neural networks for the purpose of antenna selection, given a large number of antennas of massive multiple-input multiple-output (mMIMO) systems, thus a large number of the candidate set of selected antennas. Adopting the binary cross entropy, we can generate short labels for the supervised learning of the networks. When the conventional cross-entropy is used, the length of labels is $\binom{N_R}{N_{RS}}$, where N_R and N_{RS} are the number of candidate antennas and the number of selected antennas, respectively, however, the length of labels is as short as N_R when the binary cross entropy is adopted. The channel capacity formula that is maximized by selecting antennas is a function of the signal-to-noise ratio (SNR) and the channel matrix. We show that the channel matrix excluding the SNR factor is practically sufficient information in generating the short labels.

When applied to antenna selection, neural networks are composed of a feature extraction module and a classifier module [15]. In this paper, we consider two feature extraction strategies: pure connectionist feature extraction and hybrid symbolic-connectionist feature extraction. In the pure connectionist feature extraction, the CNN or FCN extracts features from the raw data, i.e., the channel matrix. In the hybrid feature extraction, the symbolic features are first extracted from the raw data, then the CNN or FCN extracts features from the symbolic features. Then the final features from the pure or hybrid feature extraction module are passed to the classifier module to select a set of antennas. We show that CNN functions better than FCN as a pure connectionist feature extractor. However, when features are extracted in the hybrid manner, the FCN achieves a marginally better performance than the CNN. Finally, when the networks are pruned to be suitable for deployment in mobile devices [16], [17], the hybrid symbolic-connectionist feature extraction using FCN is shown to be quite a better choice, being more robust to severe pruning. We performed a set of computer simulations to demonstrate the effectiveness of the proposed approach.

II. MASSIVE MIMO SYSTEMS WITH ANTENNA SELECTION

In this section, we describe the spatially multiplexed multiple antenna systems with antenna selection. Letting N_T and N_R denote the number of transmit and receive antennas, respectively, the relationship between the transmitted and received symbol vectors can be expressed as follows:

$$\mathbf{y}^{\text{sel}} = \frac{1}{\sqrt{N_T}}\mathbf{P}\mathbf{H}\mathbf{x} + \mathbf{z}^{\text{sel}} \tag{1}$$

where $\mathbf{x} = [x_1\ x_2 \cdots x_{N_T}]^T$ denotes the transmitted symbol vector, $\mathbf{y}^{\text{sel}} = [y^{\text{sel}}_{(1)}\ y^{\text{sel}}_{(2)} \cdots y^{\text{sel}}_{(N_{RS})}]^T$ with $N_{RS} \le N_R$, and $y^{\text{sel}}_{(s)}$ denotes the received signal at the s-th ($1 \le s \le N_{RS}$) selected antenna. The matrix $\mathbf{H}$ denotes an $N_R \times N_T$ channel, in which h_{ji} denotes the standard unit power Rayleigh-fading complex gain between the i-th transmit antenna and the j-th receive antenna, whereas $\mathbf{z}^{\text{sel}} = [z^{\text{sel}}_{(1)}\ z^{\text{sel}}_{(2)} \cdots z^{\text{sel}}_{(N_{RS})}]^T$ with $z^{\text{sel}}_{(s)}$ denoting the additive white Gaussian noise with zero mean and variance σ_z^2 at the s-th selected receiving antenna. Finally, an $N_{RS} \times N_R$ antenna selection matrix is given as

$$\mathbf{P} = \left[\mathbf{e}^{\text{sel}}_{(1)}\ \mathbf{e}^{\text{sel}}_{(2)}\ \cdots\ \mathbf{e}^{\text{sel}}_{(N_{RS})}\right]^T \tag{2}$$

where $\mathbf{e}^{\text{sel}}_{(s)}$ denotes the unit column vector of length N_R with 1 at the s-th selected position and 0s at the other positions. There are $\binom{N_R}{N_{RS}}$ possible antenna selection matrices, and we let $\mathcal{P}$ denote the set of all the selection matrices. We assume $E\{|x_i|^2\} = 1, i = 1, 2, \cdots, N_T$, thus the signal-to-noise ratio per receive antenna is $\rho = \frac{1}{N_T\sigma_z^2}$.

The optimal antenna selection is described as

$$\begin{aligned}\mathbf{P}^{\text{opt}}_{\text{SpecEff+}} &= \arg\max_{\mathbf{P}\in\mathcal{P}} \log_2 \det\left(\mathbf{I}_{N_{RS}} + \rho\mathbf{P}\mathbf{H}\mathbf{H}^H\mathbf{P}^H\right) \\ &= \arg\max_{\mathbf{P}\in\mathcal{P}} \det\left(\mathbf{I}_{N_{RS}} + \rho\mathbf{P}\mathbf{H}\mathbf{H}^H\mathbf{P}^H\right).\end{aligned} \tag{3}$$

When the i-th antenna set that corresponds to $\mathbf{P}_i \in \mathcal{P}$ is the optimal for the given channel $\mathbf{H}$, the label for the network training is given as

$$\mathbf{d}^{\text{long}} = \left[\mathbf{0}_{i-1}\ \ 1\ \ \mathbf{0}_{\binom{N_R}{N_{RS}}-i}\right] \tag{4}$$

where $\mathbf{0}_a$ denotes a row vector with a zeros. We note that the length of the label vector is as long as $\binom{N_R}{N_{RS}}$. The length of the label vector is $\binom{N_R}{N_{RS}}$. The network can be considered as a classification of channel matrices $\mathbf{H}$s into $\binom{N_R}{N_{RS}}$ classes, and each class corresponds to $\mathbf{P}_i$.

III. RELATED WORKS

In this section, we review previous antenna selection techniques using artificial neural networks. We consider neural networks with $L+1$ layers. The layer index is denoted as $l = 0, 1, \cdots, L$, the input and output layers are denoted as the 0-th and the L-th layer, respectively. We note that a layer means a set of a fully connected (or a convolutional) layer, a batch normalization layer, and an activation layer.

A. INPUT DATA OF THE NEURAL NETWORKS

In previous neural network(NN)-based antenna selection techniques, various data were inputted to the artificial NN. The channel matrix $\mathbf{H}$ in (1) is composed of $N_R \times N_T$ complex numbers. In [12], the input data is composed of 2 matrices $\mathrm{Re}\{\mathbf{H}\}$ and $\mathrm{Im}\{\mathbf{H}\}$; real part and imaginary part matrices, thus the three dimensional size of the input data is $N_R \times N_T \times 2$. In [1], [13], and [14], the input data is composed of 3 matrices or channels; real part matrix, imaginary part matrix, and absolute value matrix, thus the data size is $N_R \times N_T \times 3$. In [11], $N_R \times N_R$ symmetric matrix is constructed as $\mathbf{H}\mathbf{H}^H$, then the input data is composed of 1 channel with the absolute values of each entry. In [9], each $N_R \times 1$ channel vector is expanded into a $N_R \times N_R$ matrix of which the lower triangular part is ignored due to the symmetry. Thus, the input data is composed of $N_T \times N_R^2$ real numbers.

In this paper, the $N_R \times N_T \times 2$ real numbers of the channel matrix $\mathbf{H}$ are referred to as raw data because no data processing is performed. We name $|\mathbf{H}|$ or $|\mathbf{H}\mathbf{H}^H|$ symbolic features because $\mathbf{H}$ is processed symbolically without the aid of neural networks.

B. FULLY CONNECTED NETWORK-BASED APPROACH

In [11], a fully connected network (FCN) was used for the antenna selection. The weighted sum $s_i^{(l)}$ of the i-th node of the l-th layer, $i = 1, 2, \cdots, N^{(l)}$, $l = 1, 2, \cdots, L$, is

$$s_i^{(l)} = \sum_{j=1}^{N^{(l-1)}} w_{i,j}^{(l)} f_j^{(l-1)} \tag{5}$$

where $f_j^{(l-1)}$ denotes the j-th feature value of the $(l-1)$-th layer, and $w_{i,j}^{(l)}$, $i = 1, 2, \cdots, N^{(l)}$, $j = 1, 2, \cdots, N^{(l-1)}$, denotes the connection strength between the i-th node of the l-th layer and the j-th feature value of the $(l-1)$-th layer.

Then the weighted sums are normalized, scaled, and shifted to be

$$\tilde{s}_i^{(l)} = \gamma_i^{(l)} \left(\frac{s_i^{(l)} - \mu_i^{(l)}}{\sigma_i^{(l)}} \right) + \beta_i^{(l)} \tag{6}$$

where $\gamma_i^{(l)}$ and $\beta_i^{(l)}$ are optimized in the backward learning, minimizing the loss function of the network [18], [19]. Note that the bias parameters are not present in (5) due to the shifting parameter $\beta_i^{(l)}$ in (6) ([20], p.343).

Then, an activation function produces the feature values of the l-th layer as

$$f_i^{(l)} = \begin{cases} \tilde{s}_i^{(l)}, & \text{if } \tilde{s}_i^{(l)} \geq 0 \\ \exp\{\tilde{s}_i^{(l)}\} - 1, & \text{else} \end{cases}, \tag{7}$$

where we assumed exponential linear unit (ELU) activation function with parameter $\alpha = 1$ for the hidden layers ($l = 1, 2, \cdots, L-1$) ([20], p.336). The activation function of the final layer is addressed in Section III-D.

C. CONVOLUTIONAL NETWORK-BASED APPROACH

In [1], a convolutional neural network (CNN) was used for the antenna selection. A CNN is composed of a number of convolutional layers and a few fully connected layers. We assume that the l-th layer is a convolutional layer and the previous $(l-1)$-th layer outputs $N_{\mathrm{fmap}}^{(l-1)}$ feature maps $f_{i,j,k}^{(l-1)}$, $i = 1, 2, \cdots, N_{\mathrm{hght}}^{(l-1)}$, $j = 1, 2, \cdots, N_{\mathrm{wdth}}^{(l-1)}$, $k = 1, 2, \cdots, N_{\mathrm{fmap}}^{(l-1)}$. Each two dimensional feature map of the $(l-1)$-th layer is assumed to be of size $N_{\mathrm{hght}}^{(l-1)} \times N_{\mathrm{wdth}}^{(l-1)}$. We denote the $N_{\mathrm{fmap}}^{(l)}$ convolution kernels as $w_{u,v,k',k}^{(l)}$, $u = 1, 2, \cdots, W_{\mathrm{hght}}^{(l)}$, $v = 1, 2, \cdots, W_{\mathrm{wdth}}^{(l)}$, $k' = 1, 2, \cdots, N_{\mathrm{fmap}}^{(l-1)}$, $k = 1, 2, \cdots, N_{\mathrm{fmap}}^{(l)}$. Each three dimensional convolution kernel of the l-th layer is of size $W_{\mathrm{hght}}^{(l)} \times$, $W_{\mathrm{wdth}}^{(l)} \times N_{\mathrm{fmap}}^{(l-1)}$. Then, the weighted sum $s_{i,j,k}^{(l)}$ of the (i,j)-th neuron of the k-th feature map of the l-th layer, $i = 1, 2, \cdots, N_{\mathrm{hght}}^{(l)}$, $j = 1, 2, \cdots, N_{\mathrm{wdth}}^{(l)}$, $k = 1, 2, \cdots, N_{\mathrm{fmap}}^{(l)}$, is

$$s_{i,j,k}^{(l)} = \sum_{u=1}^{W_{\mathrm{hght}}^{(l)}} \sum_{v=1}^{W_{\mathrm{wdth}}^{(l)}} \sum_{k'=1}^{N_{\mathrm{fmap}}^{(l-1)}} w_{u,v,k',k}^{(l)} f_{u'(i,u),v'(j,v),k'}^{(l-1)} \tag{8}$$

with $u'(i,u) = (i-1) \times s_H^{(l)} + u$, $v'(i,v) = (j-1) \times s_V^{(l)} + v$. The symbols $s_H^{(l)}$ and $s_V^{(l)}$ denote the horizontal and vertical strides of the l-th layer, respectively. There are cases when $u'(N_{\mathrm{hght}}^{(l)}, W_{\mathrm{hght}}^{(l)}) > N_{\mathrm{hght}}^{(l-1)}$ or $v'(N_{\mathrm{wdth}}^{(l)}, W_{\mathrm{wdth}}^{(l)}) > N_{\mathrm{wdth}}^{(l-1)}$. In those cases, we can either adopt smaller $N_{\mathrm{hght}}^{(l)}$ and $N_{\mathrm{wdth}}^{(l)}$ and discard a few features of the previous layers, or let those cases happen and use 0 feature values, which is referred to as zero padding. In this paper, we adopt the zero padding option.

Once the weighted sums are calculated, the mini-batch normalization and the activation are performed similarly as in the fully connected network, producing $N_{\mathrm{fmap}}^{(l)}$ feature maps, each of size $N_{\mathrm{hght}}^{(l)} \times N_{\mathrm{wdth}}^{(l)}$. In general, a pooling layer is inserted between the convolutional weighted sum in (8) and the batch normalization, however, it is skipped in the paper.[1] If the l-th convolution layer is followed by a fully connected layer, the feature values are flattened into one dimensional

[1]We tried the inclusion of the pooling layers in the simulations but observed only degraded performance.

vector of length $N_{\text{hght}}^{(l)} \times N_{\text{wdth}}^{(l)} \times N_{\text{fmap}}^{(l)}$. Batch normalization in the final fully connected layer is also the same as (6) but the corresponding activation is the softmax addressed in the following Section.

D. CROSS ENTROPY AS A LOSS FUNCTION

The activation function of the output layer (the L-th layer) for both CNN in Section III-B and FCN in Section III-C, is assumed to be the softmax, producing the following features $f_i^{(L)[m]}$, $i = 1, 2, \cdots, N^{(L)}$, $m = 1, 2, \cdots, N_{\text{MB}}$

$$f_i^{(L)[m]} = \frac{\exp\{\tilde{s}_i^{(L)[m]}\}}{\sum_{k=1}^{N^{(L)}} \exp\{\tilde{s}_k^{(L)[m]}\}}, \tag{9}$$

where N_{MB} denotes the batch size. Using the label $\mathbf{d}^{\text{long}} = [d_1\ d_2\ \cdots\ d_{N^{(L)}}]$ in (4) for the supervised learning, the cross-entropy between the network output $\mathbf{f}^{(L)}$ and label $\mathbf{d}^{\text{long}}$ is estimated as

$$H(\mathbf{f}^{(L)}, \mathbf{d}^{\text{long}}) \approx \frac{1}{N_{\text{MB}}} \sum_{m=1}^{N_{\text{MB}}} \sum_{i=1}^{N^{(L)}} \left\{ -d_i^{[m]} \ln f_i^{(L)[m]} \right\}. \tag{10}$$

We note that $N^{(L)} = \binom{N_R}{N_{RS}}$.

IV. COMPLEXITY ANALYSIS OF NEURAL NETWORKS

From the perspective of implementation as an embedded system in mobile devices, the inference complexity of a neural network is important. In order to compare the performance of CNN and FCN with various input shapes, the inference complexities need to be the same for a fair comparison. In this section, we analyze the inference complexities of FCN and CNN, and then, use the analysis results in the following Section to set the network parameters such as the number of neurons or the kernels so that the complexity stays the same regardless of the adoption of CNN or FCN with various input data shapes.

A. COMPLEXITY OF FULLY CONNECTED NETWORK

The complexity of the l-th layer is $N^{(l)} \times N^{(l-1)}$ multiplications in (5), about $N^{(l)}/2$ exponential function calculations in (7) assuming half of $\tilde{s}_i^{(l)}$, $i = 1, 2, \cdots, N^{(l)}$ are less than 0. Once the two values $\mu_i^{(l)}$ and $\sigma_i^{(l)}$ are fixed in the inference phase, and $\gamma_i^{(l)}$ and $\beta_i^{(l)}$ are fixed at the end of the machine learning, the process in (6) can be implemented using a single multiplication and a single addition. 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

$$\text{CompFC} = \sum_{l=1}^{L} \left\{ N^{(l)} \times \left(N^{(l-1)} + 2 \right) \right\}. \tag{11}$$

Assuming that all hidden layers have the same number of nodes, i.e. $N^{(l)} = N_{\text{FC}}$, $l = 1, 2, \cdots, L-1$, the complexity is expressed as

$$\text{CompFC} = (L-2)N_{\text{FC}}^2 + \left\{ N^{(0)} + N^{(L)} + 2L - 2 \right\} \times N_{\text{FC}} + 2N^{(L)}. \tag{12}$$

B. COMPLEXITY OF CONVOLUTIONAL NETWORK

We assume L_{CN} convolutional layers, that is followed by L_{FC} fully connected layers, thus $L_{CN} + L_{FC} + 1$ layers in total, including the input layer.

Assuming $N_{\text{fmap}}^{(l-1)}$ feature maps of the $(l-1)$-th layer, the complexity of the weighted sum in (8) requires $W_{\text{hght}}^{(l)} \times W_{\text{wdth}}^{(l)} \times N_{\text{fmap}}^{(l-1)}$ for each neuron of a feature map, thus $W_{\text{hght}}^{(l)} \times W_{\text{wdth}}^{(l)} \times N_{\text{fmap}}^{(l-1)} \times N_{\text{hght}}^{(l)} \times N_{\text{wdth}}^{(l)} \times N_{\text{fmap}}^{(l)}$ considering all $N_{\text{fmap}}^{(l)}$ feature maps. We note that the number of parameters in $N_{\text{fmap}}^{(l)}$ convolution kernels is $W_{\text{hght}}^{(l)} \times W_{\text{wdth}}^{(l)} \times N_{\text{fmap}}^{(l-1)} \times N_{\text{fmap}}^{(l)}$, and each kernel is used $N_{\text{hght}}^{(l)} \times N_{\text{wdth}}^{(l)}$ times. In this paper, we adopt small kernels with $W_{\text{hght}}^{(l)} = 2$, $W_{\text{wdth}}^{(l)} = 2$ for all convolutional layers. With these small kernels and small strides $s_H^{(l)} = 1, 2$ and $s_V^{(l)} = 1, 2$, we have $N_{\text{hght}}^{(l)} = N_{\text{hght}}^{(l-1)}/s_V^{(l)}$ and $N_{\text{wdth}}^{(l)} = N_{\text{wdth}}^{(l-1)}/s_H^{(l)}$, assuming the zero padding and $N_{\text{hght}}^{(l-1)}$ and $N_{\text{wdth}}^{(l-1)}$ being multiples of 2.

$$\text{CompCN} = \sum_{l=1}^{L_{CN}} \left\{ \left(2 + 4 \times N_{\text{fmap}}^{(l-1)} \right) \times \frac{N_{\text{hght}}^{(l-1)}}{s_V^{(l)}} \times \frac{N_{\text{wdth}}^{(l-1)}}{s_H^{(l)}} \times N_{\text{fmap}}^{(l)} \right\}. \tag{13}$$

$$\text{CompFC} = \sum_{l=L_{CN}+1}^{L} \left\{ N^{(l)} \times \left(N^{(l-1)} + 2 \right) \right\}. \tag{14}$$

V. PROPOSED ANTENNA SELECTION TECHNIQUE FOR MASSIVE MIMO SYSTEMS USING ARTIFICIAL NEURAL NETWORK

In this section, we propose an antenna selection technique using a neural network with binary cross entropy and hybrid symbolic-connectionist feature extraction.

A. ADOPTION OF BINARY CROSS ENTROPY

Instead of conventional cross entropy using (9) and (10) as the loss function for network training, we adopt binary cross entropy that assumes the following short label.

$$\mathbf{d}^{\text{short}} = [d_1\ d_2\ \cdots\ d_{N^{(L)}}], \text{ with } N^{(L)} = N_R. \tag{15}$$

The short label is composed of N_{RS} ones that denote the selected antenna indices and $N_R - N_{RS}$ zeros. Note that the length of the long label in (4) is $N^{(L)} = \binom{N_R}{N_{RS}}$ and the length of the short label is only N_R. If $N_R = 16$ and $N_{RS} = 8$, the long label $\mathbf{d}^{\text{long}}$ is as long as 12, 870 but the short label is of length 16. If binary cross entropy is adopted, sigmoid

activation is used in the output layer.

$$f_i^{(L)[m]} = \frac{1}{1+\exp\left\{-\tilde{s}_i^{(L)[m]}\right\}}, i = 1, 2, \cdots, N_R. \quad (16)$$

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

$$H(\mathbf{f}^{(L)}, \mathbf{d}^{\text{short}}) \approx \frac{1}{N_{\text{MB}}} \sum_{m=1}^{N_{\text{MB}}} \sum_{i=1}^{N_R} \left\{-d_i^{[m]} \ln f_i^{(L)[m]}\right\}. \quad (17)$$

Given a large number of antennas in massive MIMO systems, the adoption of binary cross entropy and consequently the use of the short label is almost essential.

B. SHORT LABEL GENERATION

The optimal antenna selection in (3) can be considered as the following nonlinear function.

$$\mathbf{P}_{\text{capacity}}^{\text{opt}} = \Phi_{\text{capacity}}(\rho, \mathbf{H}), \quad (18)$$

with domain $\mathbb{R}\times\mathbb{C}^{N_R\times N_T}$ and range $\mathcal{P}$. In an effort to facilitate the short label generation, we simplify the objective function in (3) or (18) as

$$\begin{aligned}\mathbf{P}_{\text{simplified}}^{\text{opt}} &= \arg\max_{\mathbf{P}\in\mathcal{P}} \det\left(\mathbf{P}\mathbf{H}\mathbf{H}^H\mathbf{P}^H\right) \\ &= \Phi_{\text{simplified}}(\mathbf{H}). \end{aligned} \quad (19)$$

Thus the optimal precoder $\mathbf{P}_{\text{simplified}}^{\text{opt}}$ is independent of SNR. Fig. 1 compares the capacity achieved by the optimal antenna selection in (3) and the simplified antenna selection in (19) when $N_T = 8$, $N_R = 16$, $N_{RS} = 8$. It can be observed that the capacity achieved by $\mathbf{P}_{\text{simplified}}^{\text{opt}}$ is very close to the optimal performance. There is practically no performance degradation when SNR is higher than 0[dB], and only a negligible performance degradation occurs by the simplification when the SNR is lower than 0[dB]. Based on the observations, we use henceforth the simplified optimization (19) in generating the short labels.

C. CNN VS FCN FOR PURE CONNECTIONIST FEATURE EXTRACTION

Not only in [1] but also in [21] and [22], CNN was used for the classification of non-image data. The convolutional neural network (CNN) is known to successfully extract visual features. Due to the inherent characteristic of convolutional structure, the local visual features are extracted by CNN regardless of their spatial positions in the samples, e.g., **H**s in the antenna selection problem addressed in the paper. The channel matrices **H**s are generated as independent identical Gaussian random variables, which means there is no statistical spatial correlation. When a specific channel **H** is considered, however, there might be underlying connections between the visual features and the antenna selection matrix **P**. In this regard, it is worth comparing CNN and FCN

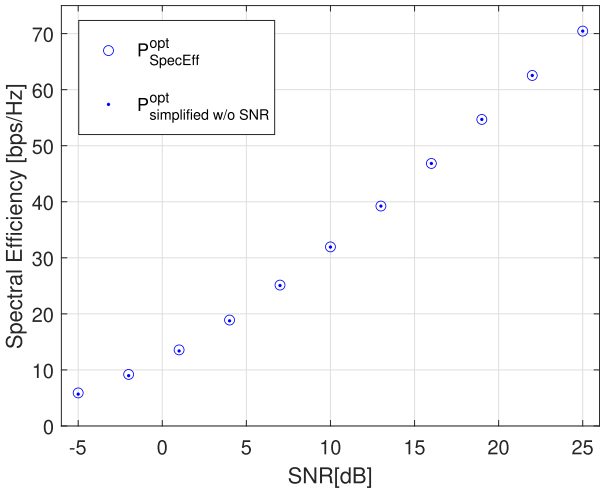


FIGURE 1. Spectral efficiency with the optimal antenna selection in (3) or (18) that uses both SNR and channel matrix and the simplified antenna selection in (19) that uses only the channel matrix when $N_T = 8$, $N_R = 16$, $N_{RS} = 8$. The SNR information is not essential from the perspective of the antenna selection.

when raw data Re(**H**) and Im(**H**) is inputted. An example of the random matrix with $N_R = 8$ and $N_T = 4$ is given in Fig. 2(a). Fig. 2(b) is the $8 \times 4 \times 2$ input sample that corresponds to the random matrix in Fig. 2(a). We call Re(**H**) and Im(**H**) raw data because no data processing is applied to **H**.

The CNN is illustrated in Fig.3(a) with the raw data as input. The input data is of dimension $N_R \times N_T \times 2$, and the hidden layers that extract connectionist features are composed of three convolutional layers and a fully connected layer. For each layer, batch normalization is done right before the activations. The parameters for exponential moving averages for the batch normalization are $\beta_\mu = \beta_\sigma = 0.99$. The stride parameter of the first ($l = 1$) convolutional layer is 2, and the strides are set to 1 for the second ($l = 2$) and the third ($l = 3$) convolutional layers. Then a fully connected layer with sigmoid activation is used as the classifier of the input samples using the connectionist features from the hidden layers. We note that the pooling layers are not utilized in Fig.3(a), which means that the rotated and/or scaled visual features are not extracted.

When FCN is used for the connectionist feature extraction, the input data is the one dimensional vector of length $N_R \times N_T \times 2$, and the hidden layers are composed of four fully connected layers. In order to compare CNN and FCN in a fair manner, we adjust the number of kernels of CNN and the number of neurons of FCN using (11)-(14) so that the inference complexities of the two structures are the same. Table 1 shows the number of kernels of the convolutional layer and the number of neurons of the fully connected layer so that both networks require about 370,000 multiplications for inference from a sample.

$$\mathbf{H} = \begin{bmatrix} 0.5377 - 1.0689i & 3.5784 - 0.1022i & -0.1241 + 1.0933i & 0.4889 - 0.7697i \\ 1.8339 - 0.8095i & 2.7694 - 0.2414i & 1.4897 + 1.1093i & 1.0347 + 0.3714i \\ -2.2588 - 2.9443i & -1.3499 + 0.3192i & 1.4090 - 0.8637i & 0.7269 - 0.2256i \\ 0.8622 + 1.4384i & 3.0349 + 0.3129i & 1.4172 + 0.0774i & -0.3034 + 1.1174i \\ 0.3188 + 0.3252i & 0.7254 - 0.8649i & 0.6715 - 1.2141i & 0.2939 - 1.0891i \\ -1.3077 - 0.7549i & -0.0631 - 0.0301i & -1.2075 - 1.1135i & -0.7873 + 0.0326i \\ -0.4336 + 1.3703i & 0.7147 - 0.1649i & 0.7172 - 0.0068i & 0.8884 + 0.5525i \\ 0.3426 - 1.7115i & -0.2050 + 0.6277i & 1.6302 + 1.5326i & -1.1471 + 1.1006i \end{bmatrix}$$

(a)

$$\mathrm{Re}(\mathbf{H}) = \begin{bmatrix} 0.5377 & 3.5784 & -0.1241 & 0.4889 \\ 1.8339 & 2.7694 & 1.4897 & 1.0347 \\ -2.2588 & -1.3499 & 1.4090 & 0.7269 \\ 0.8622 & 3.0349 & 1.4172 & -0.3034 \\ 0.3188 & 0.7254 & 0.6715 & 0.2939 \\ -1.3077 & -0.0631 & -1.2075 & -0.7873 \\ -0.4336 & 0.7147 & 0.7172 & 0.8884 \\ 0.3426 & -0.2050 & 1.6302 & -1.1471 \end{bmatrix} \quad \mathrm{Im}(\mathbf{H}) = \begin{bmatrix} -1.0689 & -0.1022 & 1.0933 & -0.7697 \\ -0.8095 & -0.2414 & 1.1093 & 0.3714 \\ -2.9443 & 0.3192 & -0.8637 & -0.2256 \\ 1.4384 & 0.3129 & 0.0774 & 1.1174 \\ 0.3252 & -0.8649 & -1.2141 & -1.0891 \\ -0.7549 & -0.0301 & -1.1135 & 0.0326 \\ 1.3703 & -0.1649 & -0.0068 & 0.5525 \\ -1.7115 & 0.6277 & 1.5326 & 1.1006 \end{bmatrix}$$

(b)

$$\frac{|\mathbf{H}\mathbf{H}^H|}{\max|\mathbf{H}\mathbf{H}^H|} = \begin{bmatrix} 0.8571 & 0.6858 & 0.2795 & 0.4660 & 0.2095 & 0.1005 & 0.0417 & 0.0755 \\ 0.6858 & 0.8631 & 0.5634 & 0.6325 & 0.2927 & 0.3326 & 0.1394 & 0.2486 \\ 0.2795 & 0.5634 & 1.0000 & 0.4589 & 0.0453 & 0.3285 & 0.2134 & 0.4956 \\ 0.4660 & 0.6325 & 0.4589 & 0.8143 & 0.2784 & 0.2091 & 0.2673 & 0.1742 \\ 0.2095 & 0.2927 & 0.0453 & 0.2784 & 0.2462 & 0.1569 & 0.1720 & 0.2020 \\ 0.1005 & 0.3326 & 0.3285 & 0.2091 & 0.1569 & 0.2948 & 0.1415 & 0.1302 \\ 0.0417 & 0.1394 & 0.2134 & 0.2673 & 0.1720 & 0.1415 & 0.2217 & 0.2079 \\ 0.0755 & 0.2486 & 0.4956 & 0.1742 & 0.2020 & 0.1302 & 0.2079 & 0.5796 \end{bmatrix}$$

(c)

FIGURE 2. Illustration of raw data and symbolic features when $N_T = 4$, $N_R = 8$, $N_{RS} = 4$. (a) an example of the random channel matrix, (b) $8 \times 4 \times 2$ raw data that is composed of real and imaginary matrices, (c) 8×8 symbolic features of the channel matrix.

The Adam optimizer ([20], page 356) is used for weight updates in the training phase. In training the networks, we scheduled the learning rate as

$$\eta^{\text{train}} = \begin{cases} 5 \times 10^{-3}, & 1 \le \text{epoch} < 10 \\ 5 \times 10^{-4}, & 10 \le \text{epoch} < 15 \\ 5 \times 10^{-5}, & 15 \le \text{epoch} < 20. \end{cases} \tag{20}$$

We generated $1,600,000$ random channels and the corresponding short labels, which are divided into $1,560,000$ training samples and $40,000$ testing samples.

Upon completion of the training, we have the following two nonlinear inference functions with about 370,000 parameters, approximating (19).

$$\hat{\mathbf{P}}^{\text{opt,CNN}}_{\text{simplified}} = \hat{\Phi}^{\text{CNN}}_{\text{simplified}}(\mathbf{H}). \tag{21}$$

$$\hat{\mathbf{P}}^{\text{opt,FCN}}_{\text{simplified}} = \hat{\Phi}^{\text{FCN}}_{\text{simplified}}(\mathbf{H}). \tag{22}$$

Fig. 4 compares CNN and FCN in terms of spectral efficiency for 3 sets of $\{N_T, N_R, N_{RS}\}$. The spectral efficiency by the simplified optimal antenna selection $\log_2 \det(\mathbf{I}_{N_{RS}} + \rho \mathbf{P}^{\text{opt}}_{\text{simplified}} \mathbf{H}\mathbf{H}^H \mathbf{P}^{\text{opt}H}_{\text{simplified}})$ is presented as the solid line, and the spectral efficiency by not choosing the antennas corresponds to the dotted line which is obtained by fixed $\mathbf{P}_{\text{no selection}} = [\mathbf{e}_1\ \mathbf{e}_2 \cdots \mathbf{e}_{N_{RS}}]$, i.e., selecting the first N_{RS} antennas regardless of $\mathbf{H}$s. Performance of the CNN using (21) and the FCN using (22) are denoted as circles and small dots, respectively. Fig. 4 shows that the CNN extracts features better from the raw data than the FCN. It is interesting to observe that the random channels contain visual features that are underlying connections to the short labels.

D. CNN VS FCN FOR HYBRID FEATURE EXTRACTION

In [11], a neural network based antenna selection technique was proposed for a multi-user MISO (multiple input single output) downlink scenario, where the following symbolic features were inputted to the neural network for the purpose of antenna selection.

$$\frac{\left|\mathbf{H}\mathbf{H}^H\right|}{\max\left|\mathbf{H}\mathbf{H}^H\right|}, \tag{23}$$

where $|\cdot|$ means the element-wise absolute values of the argument matrix, and $\max|\cdot|$ denotes the maximum value among the $N_R \times N_R$ absolute values. Fig. 2(c) is the 8×8 symbolic features that correspond to the random matrix in Fig. 2(a). It is necessary to check if the information $|\mathbf{H}\mathbf{H}^H|$ keeps the essential features of $\mathbf{H}$ in (19). In an attempt to justify the use of (23), we checked the capacity achieved by the following antenna selection.

$$\mathbf{P}^{\text{opt}}_{\text{simplified absolute}} = \arg\max_{\mathbf{P} \in \mathcal{P}} \det\left(\mathbf{P}^H \left|\mathbf{H}\mathbf{H}^H\right| \mathbf{P}\right). \tag{24}$$

Comparing (19) and (24), we can observe that the $N_R \times N_R$ complex values ($N_R \times N_R \times 2$ real values) $\mathbf{H}\mathbf{H}^H$ in (19) are reduced to $N_R \times N_R$ real values in (24), which is a dimension reduction [23].

Fig. 5 compares the spectral efficiency with the optimal antenna selection in (3) and the simplified antenna selection in (24) when $N_T = 8$, $N_R = 16$, $N_{RS} = 8$. It can be

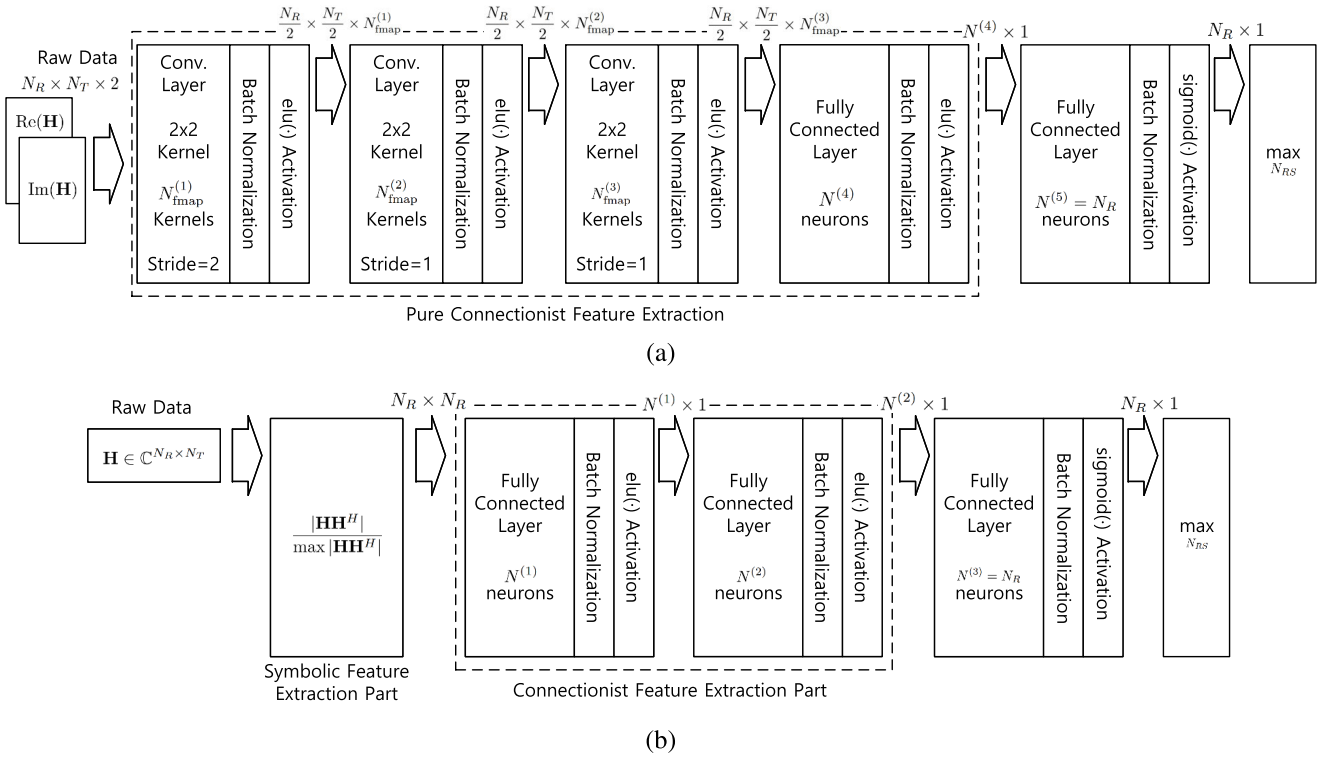


FIGURE 3. Neural network architecture is chosen for antenna selection depending upon whether the features are extracted in a pure connectionist manner or in a hybrid symbolic-connectionist manner: (a) Convolutional neural network is chosen when raw data is inputted and the features are extracted in a pure connectionist manner (b) Fully connected neural network is preferred when the features are extracted in a hybrid manner, i.e., the symbolically extracted features are inputted to the neural network then the connectionist features are extracted from the symbolic features.

observed that there exists only a slight spectral efficiency reduction in the high SNR regime due to taking absolute values. We argue that the information $|\mathbf{H}\mathbf{H}^H|$ in (24) keeps the essential features of $\mathbf{H}$ or $\mathbf{H}\mathbf{H}^H$ in (19). Furthermore, the $N_R \times N_R$ numbers in $|\mathbf{H}\mathbf{H}^H|$ are good features from the perspective of antenna selection.

Fig. 3(b) illustrates a hybrid feature extraction that combines the symbolic and connectionist feature extractions. The raw data $\mathbf{H} \in \mathbb{C}^{N_R \times N_T}$ is composed of $N_R \times N_T \times 2$ real numbers. The symbolic features in (23) that are composed of $N_R \times N_R$ real numbers, are extracted from $\mathbf{H}$. Then the symbolic features are inputted into the neural network. In Fig. 3(b), two fully connected layers are used for the connectionist feature extraction, then another fully connected layer is used for the classification using the features from the second hidden layer. Finally, N_{RS} antenna indices are selected from the output vector with length N_R of the last fully connected layer. In this hybrid symbolic and connectionist feature extraction scenario, the CNN can also be adopted for the connectionist part. Table 2 shows the number of kernels of the convolutional layer and the number of neurons of the fully connected layer so that both networks require the same number of multiplications for inference from a sample. We note that about 370,000 multiplications are required for all the networks in Table 1 and Table 2.

Upon completion of training adopting the same training optimizer and learning rate schedule as in Section V-C, we have the following two nonlinear inference functions with about 370,000 parameters, approximating (19).

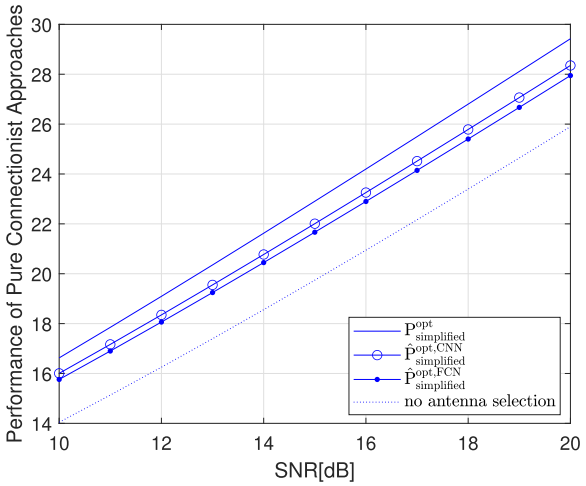
$$\hat{\mathbf{P}}^{\text{opt,Hybrid-CNN}}_{\text{simplified}} = \hat{\Phi}^{\text{Hybrid-CNN}}_{\text{simplified}}\left(\frac{|\mathbf{H}\mathbf{H}^H|}{\max|\mathbf{H}\mathbf{H}^H|}\right). \quad (25)$$

$$\hat{\mathbf{P}}^{\text{opt,Hybrid-FCN}}_{\text{simplified}} = \hat{\Phi}^{\text{Hybrid-FCN}}_{\text{simplified}}\left(\frac{|\mathbf{H}\mathbf{H}^H|}{\max|\mathbf{H}\mathbf{H}^H|}\right). \quad (26)$$

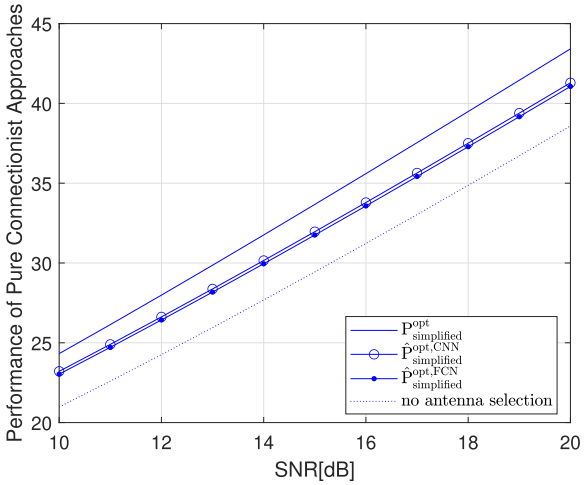
Fig. 6 compares the Hybrid-CNN and the Hybrid-FCN for 3 sets of $\{N_T, N_R, N_{RS}\}$. The two networks offer practically the same spectral efficiency for each set of $\{N_T, N_R, N_{RS}\}$ when the symbolic features are inputted to the connectionist feature extraction parts. Comparing Fig. 4 and Fig. 6, we can also observe that the hybrid feature extraction strategy in Fig. 3(b) outperforms the pure connectionist feature extraction in Fig. 3(a).

E. CNN VS FCN WHEN NETWORKS ARE PRUNED

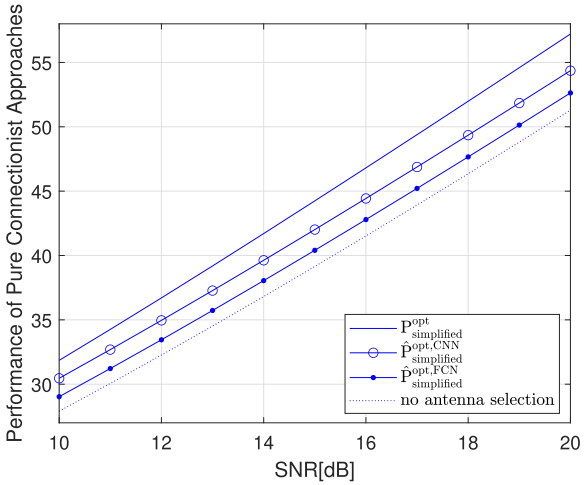
A small number of parameters are desirable when the nonlinear inference functions are deployed in mobile devices. In this subsection, we prune to reduce the about 370,000 trained parameters of (25) and (26). We note that the kernel parameters $w^{(l)}_{u,v,k',k}$ in (8) of the convolutional layer



(a) $N_T = 4$, $N_R = 8$, $N_{RS} = 4$



(b) $N_T = 6$, $N_R = 12$, $N_{RS} = 6$



(c) $N_T = 8$, $N_R = 16$, $N_{RS} = 8$

FIGURE 4. Comparison of convolutional neural network (CNN) and fully connected neural network (FCN) in terms of spectral efficiency when raw data is inputted. CNN extracts features better from the raw data than the FCN.

TABLE 1. Number of Kernels and Neurons of CNN and FCN for the Connectionist Feature Extraction. The input data shape for CNN is 2 channels of $N_R \times N_T$, and the input data shape for FCN is a vector of length $N_R \times N_T \times 2$. We assumed $s_V^{(2)} = s_H^{(2)} = 2$ and $s_V^{(l)} = s_H^{(l)} = 1, l \neq 2$ for the CNNs.

layer index l	CNN	FCN
1	$N^{(1)}_{\text{fmap}} = 24$	$N^{(1)} = 157$
2	$N^{(2)}_{\text{fmap}} = 24$	$N^{(2)} = 157$
3	$N^{(3)}_{\text{fmap}} = 24$	$N^{(3)} = 157$
4	$N^{(4)} = 200$	$N^{(4)} = 157$
5(output layer)	$N^{(5)} = 8$	$N^{(5)} = 8$

(a) $N_T = 4$, $N_R = 8$, $N_{RS} = 4$

layer index l	CNN	FCN
1	$N^{(1)}_{\text{fmap}} = 24$	$N^{(1)} = 228$
2	$N^{(2)}_{\text{fmap}} = 24$	$N^{(2)} = 228$
3	$N^{(3)}_{\text{fmap}} = 24$	$N^{(3)} = 228$
4	$N^{(4)} = 200$	$N^{(4)} = 228$
5(output layer)	$N^{(5)} = 12$	$N^{(5)} = 12$

(b) $N_T = 6$, $N_R = 12$, $N_{RS} = 6$

layer index l	CNN	FCN
1	$N^{(1)}_{\text{fmap}} = 24$	$N^{(1)} = 311$
2	$N^{(2)}_{\text{fmap}} = 64$	$N^{(2)} = 311$
3	$N^{(3)}_{\text{fmap}} = 24$	$N^{(3)} = 311$
4	$N^{(4)} = 400$	$N^{(4)} = 311$
5(output layer)	$N^{(5)} = 16$	$N^{(5)} = 16$

(c) $N_T = 8$, $N_R = 16$, $N_{RS} = 8$

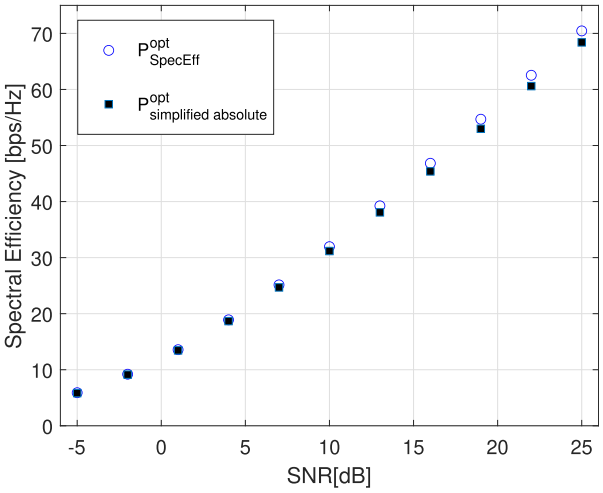
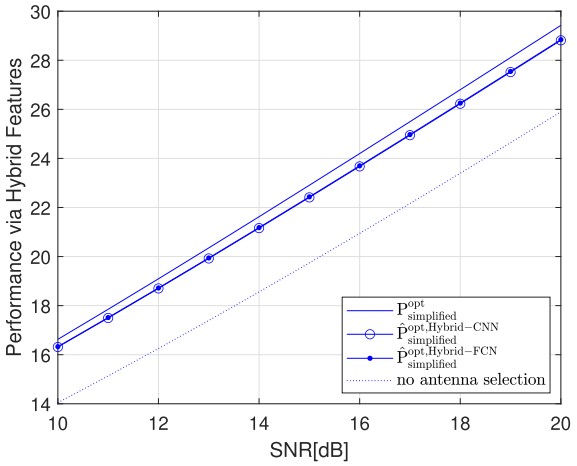
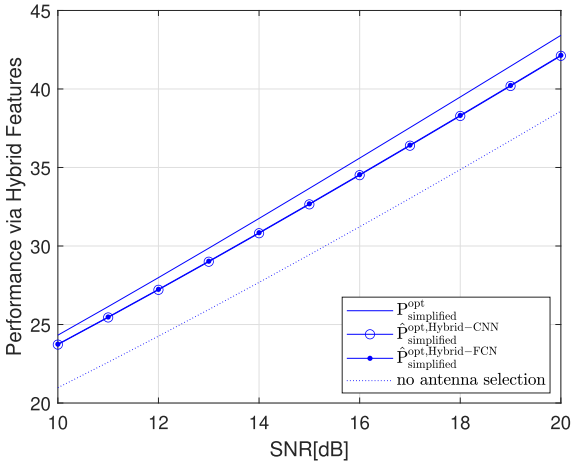


FIGURE 5. Spectral efficiency with the optimal antenna selection in (3) and the simplified antenna selection in (24) when $N_T = 8$, $N_R = 16$, $N_{RS} = 8$. The information $|\mathbf{H}\mathbf{H}^H|$ in (24) seems to keep the essential features of $\mathbf{H}$ or $\mathbf{H}\mathbf{H}^H$ in (19). Furthermore, the $N_R \times N_R$ numbers in $|\mathbf{H}\mathbf{H}^H|$ are the good features from the perspective of antenna selection.

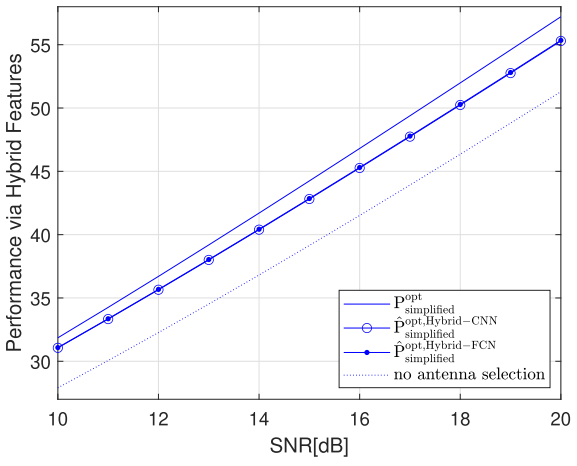
and the neuron parameters $w_{i,j}^{(l)}$ in (5) are pruned keeping the batch normalization parameters. The survived weights after the pruning are retrained with the following learning



(a) $N_T = 4$, $N_R = 8$, $N_{RS} = 4$



(b) $N_T = 6$, $N_R = 12$, $N_{RS} = 6$



(c) $N_T = 8$, $N_R = 16$, $N_{RS} = 8$

FIGURE 6. Comparison of convolutional neural network (CNN) and fully connected neural network (FCN) in terms of spectral efficiency when the symbolic features (or pre-processed data) are inputted. The two networks offer practically the same spectral efficiency when the symbolic features are inputted.

TABLE 2. Number of Kernels and Neurons of CNN and FCN for Hybrid Symbolic and Connectionist Feature Extraction. The input data shape for CNN is 1 channel of $N_R \times N_R$, and the input data shape for FCN is a vector of length $N_R \times N_R$. We assumed $s_V^{(2)} = s_H^{(2)} = 2$ and $s_V^{(l)} = s_H^{(l)} = 1, l \neq 2$ for the CNNs.

layer index l	CNN	FCN
1	$N^{(1)}_{\text{fmap}} = 17$	$N^{(1)} = 258$
2	$N^{(2)}_{\text{fmap}} = 17$	$N^{(2)} = 258$
3	$N^{(3)}_{\text{fmap}} = 17$	$N^{(3)} = 8$
4	$N^{(4)} = 144$	-
5	$N^{(5)} = 8$	-

(a) $N_T = 4$, $N_R = 8$, $N_{RS} = 4$

layer index l	CNN	FCN
1	$N^{(1)}_{\text{fmap}} = 17$	$N^{(1)} = 367$
2	$N^{(2)}_{\text{fmap}} = 17$	$N^{(2)} = 367$
3	$N^{(3)}_{\text{fmap}} = 17$	$N^{(3)} = 12$
4	$N^{(4)} = 143$	-
5	$N^{(5)} = 12$	-

(b) $N_T = 6$, $N_R = 12$, $N_{RS} = 6$

layer index l	CNN	FCN
1	$N^{(1)}_{\text{fmap}} = 24$	$N^{(1)} = 491$
2	$N^{(2)}_{\text{fmap}} = 32$	$N^{(2)} = 491$
3	$N^{(3)}_{\text{fmap}} = 24$	$N^{(3)} = 16$
4	$N^{(4)} = 200$	-
5	$N^{(5)} = 16$	-

(c) $N_T = 8$, $N_R = 16$, $N_{RS} = 8$

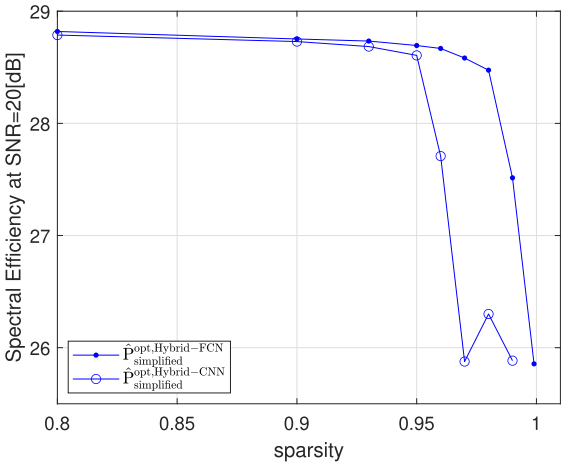
rate schedule.

$$\eta^{\text{retrain}} = \begin{cases} 5 \times 10^{-3}, & 20 \leq \text{epoch} < 30 \\ 5 \times 10^{-4}, & 30 \leq \text{epoch} < 35 \\ 5 \times 10^{-5}, & 35 \leq \text{epoch} < 40. \end{cases} \tag{27}$$

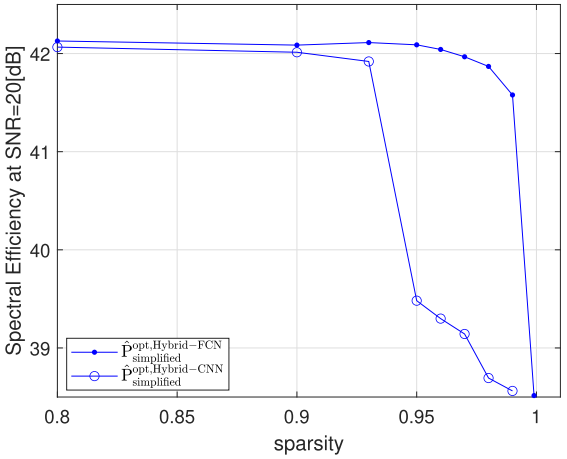
Fig. 7 compares the Hybrid-CNN and the Hybrid-FCN when the networks are pruned for 3 sets of $\{N_T, N_R, N_{RS}\}$. The signal-to-noise ratio per receive antenna ρ is fixed at 20[dB]. The performance by the Hybrid-CNN starts to degrade when the sparsity is 0.8, and the performance degradation becomes significant when the sparsity is bigger than 0.9. The Hybrid-FCN, however, maintains the performance when the sparsity is as high as 0.95. From Fig. 7, we can state that the Hybrid-FCN is more robust to network pruning than the Hybrid-CNN.

F. FURTHER COMMENTS ON THE COMPLEXITY

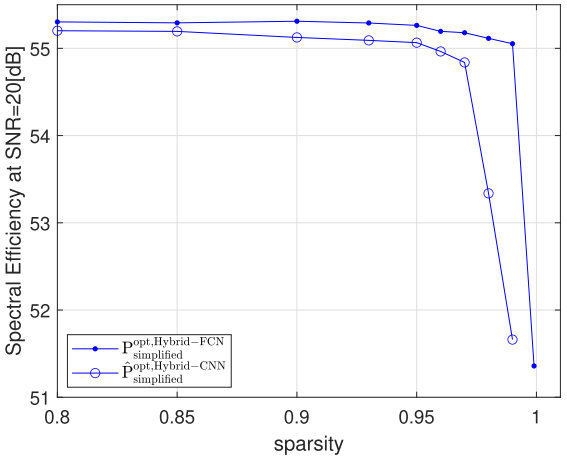
In this subsection, we elaborate on how we set the hyper-parameters in Table 1 and Table 2 so that 4 neural networks have the same complexity for a given N_T, N_R, and N_{RS}. **Step1**: The hyper-parameters are set for the CNN in Table 1 with input data shape of $N_R \times N_T \times 2$ based on a grid search for the hyper-parameters with extensive learning and evaluations.



(a) $N_T = 4, N_R = 8, N_{RS} = 4$



(b) $N_T = 6, N_R = 12, N_{RS} = 6$



(c) $N_T = 8, N_R = 16, N_{RS} = 8$

FIGURE 7. Comparison of convolutional neural network (CNN) and fully connected neural network (FCN) in terms of spectral efficiency when the networks are pruned. The symbolic features are inputted into the two networks. The signal-to-noise ratio per receive antenna ρ is fixed at 20[dB]. The FCN is more robust to network pruning than the CNN.

In the grid search for the hyper-parameters, the number of CNN kernels (thus the number of feature maps) of all layers is restricted to be the same, i.e., $N^{(l)}_{\text{fmap}} = N_{\text{fmap}}$. (We note that the number of CNN kernels was not fixed for the case of $N_T = 8, N_R = 16$.) Furthermore, the convolution kernel size is restricted to be 2×2 for all layers, the strides are also restricted to be 1 or 2. Table 1(b) shows the result of the grid search when $N_T = 6, N_R = 12$, and $N_{RS} = 6$. Using (13) and (14), we can calculate the required number of multiplications as CompCN + CompFC = 104544 + 89436 = 193980. The hyper-parameters of the other 3 neural networks are chosen so that the required complexities are close to 193980. The complexities can not be exactly the same because the hyper-parameters are constrained to be integers. We note that the following **Step2**-**Step4** can be taken in a different order. **Step2**: We choose the hyper-parameters of FCN. Applying $N^{(0)} = 6 \times 12 \times 2 = 144, N^{(L)} = 12$ to (12), we can express the complexity of the FCN as $(L-2)N^2_{\text{FC}} + (154 + 2L)N_{\text{FC}} + 24$. Constraining the complexity to be 193980 and choosing the number of layers to be $L = 5$, we have $3N^2_{\text{FC}} + 164N_{\text{FC}} + 24 = 193980$. Solving the polynomial equation, we find $N_{\text{FC}} = N^{(l)} = 228, l = 1, 2, 3, 4$ to be the integer as shown in Table 1(b). **Step3**: Then the hyper-parameters of CNN in Table 2(b) with input data shape of $N_R \times N_R$ is determined so that CompCN is the same as that of CNN in Table 1(b). Let us first elaborate on the complexity of the convolution layers of CNN in Table 2(b). Given the input data shape of $N_R \times N_R$, the kernel size of 2×2, the vertical and horizontal strides of 1s, and the number of kernels of N_{fmap}, the first hidden layer ($l = 1$) has the complexity of $N_R \times N_R \times 2 \times 2 \times N_{\text{fmap}} + N_R \times N_R \times N_{\text{fmap}} \times 3$ multiplications, where $N_R \times N_R \times N_{\text{fmap}} \times 3$ represents the complexity of the batch normalizations and the activations. Given the input data shape (i.e., the output data shape of the first hidden layer) of $N_R \times N_R \times N_{\text{fmap}}$, the kernel size of 2×2, the vertical and horizontal strides of 2s, and the number of kernels of N_{fmap}, the second hidden layer ($l = 2$) has the complexity of $N_R/2 \times N_R/2 \times N_{\text{fmap}} \times 2 \times 2 \times N_{\text{fmap}} + N_R/2 \times N_R/2 \times N_{\text{fmap}} \times 3$ multiplications, where $N_R/2 \times N_R/2 \times N_{\text{fmap}} \times 3$ represents the complexity of the batch normalizations and the activations. If additional convolution layers are added with the kernel size of 2×2, the vertical and horizontal strides of 1s, and the number of kernels of N_{fmap}, its complexity becomes $N_R \times N_R \times N^2_{\text{fmap}} + N_R/2 \times N_R/2 \times N_{\text{fmap}} \times 3$ multiplications. If L_{CN} convolution layers are adopted, the additional convolution layers excluding the first and the second layers is $L_{CN} - 2$, thus the total complexity of the convolution layers is expressed as $(L_{CN} - 1) \times N^2_R \times N^2_{\text{fmap}} + 7 \times N^2_R \times N_{\text{fmap}} + \frac{3}{4}(L_{CN} - 1)N^2_R \times N_{\text{fmap}}$. Constraining the complexity to be the same as CompCN = 104544 in Table 1(b) and choosing L_{CN} of 3, we obtain $N_{\text{fmap}} = 17$ as in Table 2(b). Although the number of convolution layers in Table 2(b) is the same as in Table 1(b), the number of

kernels can be determined accordingly if a different number of hidden convolution layers (L_{CN}) is adopted. After setting the hyper-parameters for the convolution layers, the $N^{(4)}$ in Table 2(b) is calculated so that CompFC in Table 2(b) is the same as that in Table 1(b). **Step4**: The final step is to determine $N_{\text{FC}} = N^{(1)} = N^{(2)}$ in Table 2(b) in a similar manner as in **Step2** but now with choosing the number of layers $L = 3$. Note that the complexity of FCN in both Table 1 and Table 2 is $\mathcal{O}((L-2)N_{\text{FC}}^2)$. The complexities of CNN in Table 1 and Table 2 are $\mathcal{O}((L_{CN}-1)\times N_R \times N_T \times N_{\text{fmap}}^2 + (L-L_{CN}-1)\times N_R \times N_T \times N_{\text{fmap}} \times N_{\text{FC}})$ and $\mathcal{O}((L_{CN}-1)\times N_R^2 \times N_{\text{fmap}}^2 + (L-L_{CN}-1)\times N_R^2 \times N_{\text{fmap}} \times N_{\text{FC}})$, respectively. We emphasize that all $\mathcal{O}$ expressions are based on the assumption that $N^{(L)} \ll N_{\text{FC}}$.

We measured the time taken to train the model and perform inference using the trained models. All experiments are conducted in an environment equipped with an Intel i7-6700K CPU operating at 4.00 GHz and 16.0GB of memory, running the Windows 10 Pro x64 operating system. The development tools used include Python 3.9.13, Tensorflow 2.11.0, and Keras 2.11.0. For the case of $\{N_T = 6, N_T = 12, N_{RS} = 6\}$, For inference on 100,000 samples, CNN takes 6.2 seconds for raw data and 7.2 seconds for preprocessed data, while FCN takes 5.2 seconds for raw data and 5.1 seconds for preprocessed data. The slight variation in inference times despite having the same number of multiplications can be attributed to the inherent differences in algorithmic behavior (initialization, optimization, etc.) between the software-implemented CNN and FCN.

We emphasize that it has been a conventional practice to evaluate the time performance of an algorithm based on the number of multiplications in the telecommunication field. This practice is particularly appropriate when the trained model is ultimately implemented in hardware (as a system semiconductor) on mobile devices. Compared to the number of adders and comparators, the number of multiplicators that is required by an algorithm dominantly determines the time performance of the corresponding semiconductor. Therefore, reducing the number of multiplications directly benefits in shortening the execution time which is important for real-time processing of the algorithm. Our approach aligns with this industry practice, focusing on a metric that has practical and direct implications for time performance in hardware implementation. Thus, although a strict validation for the time performance is not provided, we argue that the number of multiplications in Equations (11), (12), (13), and (14) are self-validating to a good degree.

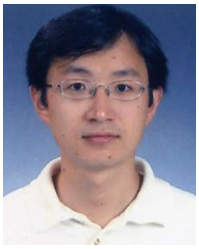
VI. CONCLUSION

In this paper, we showed that the use of binary cross entropy is quite essential for an artificial neural network that selects antennas in mMIMO systems that are equipped with a large number of antenna arrays. We also showed that the spectral efficiency formula as the objective function can be simplified by ignoring the SNR parameter practically without performance degradation. Omitting the SNR factor, we could assign a unique label to a mMIMO channel regardless of the SNR, which in turn facilitates the training data generation. When the channel matrix excluding the SNR factor is inputted to the neural networks and pure connectionist features are extracted, the CNN outperformed the FCN. However, in the hybrid scenario where good symbolic features are extracted first then the connectionist features extracted from the symbolic features, the Hybrid-CNN and the Hybrid-FCN offered practically the same performance. Finally, when the networks are pruned to be amenable to mobile devices, the Hybrid-FCN turned out to be preferable to the Hybrid-CNN.

REFERENCES

[1] 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.
[2] Y. Cho, J. Kim, C. Kang, and W. Yang, *MIMO-OFDM Wireless Communications With MATLAB*. Singapore: Wiley, 2010.
[3] M. Naeem and D. C. Lee, "Low-complexity joint transmit and receive antenna selection for MIMO systems," *Eng. Appl. Artif. Intell.*, vol. 24, no. 6, pp. 1046–1051, Sep. 2011.
[4] C. Lin and W. Wu, "QRD-based antenna selection for ML detection of spatial multiplexing MIMO systems," *IEEE Trans. Veh. Tech.*, vol. 60, no. 7, pp. 3178–3190, Sep. 2011.
[5] Y. Gao and T. Kaiser, "Antenna selection in massive MIMO systems: Full-array selection or subarray selection?" in *Proc. IEEE Sensor Array Multichannel Signal Process. Workshop (SAM)*, Jul. 2016, pp. 1–5.
[6] C. Ouyang and H. Yang, "Massive MIMO antenna selection: Asymptotic upper capacity bound and partial CSI," 2018, *arXiv:1812.06595*.
[7] M. O. K. Mendonça, P. S. R. Diniz, T. N. Ferreira, and L. Lovisolo, "Antenna selection in massive MIMO based on greedy algorithms," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 1868–1881, Mar. 2020.
[8] X. Li, Y. Huang, W. Heng, and J. Wu, "Machine learning-inspired hybrid precoding for mmWave MU-MIMO systems with domestic switch network," *Sensors*, vol. 21, no. 9, p. 3019, Apr. 2021.
[9] M. S. Ibrahim, A. S. Zamzam, X. Fu, and N. D. Sidiropoulos, "Learning-based antenna selection for multicasting," in *Proc. IEEE 19th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Jun. 2018, pp. 1–5.
[10] J. Kim and H.-S. Lim, "Neural network-based fixed-complexity precoder selection for multiple antenna systems," *IEEE Access*, vol. 10, pp. 120343–120351, 2022.
[11] T. X. Vu, S. Chatzinotas, V.-D. Nguyen, D. T. Hoang, D. N. Nguyen, M. D. Renzo, and B. Ottersten, "Machine learning-enabled joint antenna selection and precoding design: From offline complexity to online performance," *IEEE Trans. Wireless Commun.*, vol. 20, no. 6, pp. 3710–3722, Jun. 2021.
[12] J. Joung, "Machine learning-based antenna selection in wireless communications," *IEEE Commun. Lett.*, vol. 20, no. 11, pp. 2241–2244, Nov. 2016.
[13] A. M. Elbir, "CNN-based precoder and combiner design in mmWave MIMO systems," *IEEE Commun. Lett.*, vol. 23, no. 7, pp. 1240–1243, Jul. 2019.
[14] 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.
[15] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[16] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural networks," 2015, *arXiv:1506.02626*.
[17] M. Zhu and S. Gupta, "To prune, or not to prune: Exploring the efficacy of pruning for model compression," 2017, *arXiv:1710.01878*.
[18] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.
[19] C. Summers and M. J. Dinneen, "Four things everyone should know to improve batch normalization," 2019, *arXiv:1906.03548*.

[20] A. Geron, *Hands-on Machine Learning With Scikit-Learn, Keras, and Tensorflow*, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, Sep. 2019.
[21] A. Sharma, E. Vans, D. Shigemizu, K. A. Boroevich, and T. Tsunoda, "DeepInsight: A methodology to transform a non-image data to an image for convolution neural network architecture," *Sci. Rep.*, vol. 9, no. 1, Aug. 2019.
[22] A. Sharma and D. Kumar, "Classification with 2-D convolutional neural networks for breast cancer diagnosis," 2020, *arXiv:2007.03218*.
[23] I. K. Fodor, "A survey of dimension reduction techniques," U.S. Dept. Energy, Washington, DC, USA, Tech. Rep., UCRL-ID-148494, May 2002.



JAEKWON KIM received the B.S. (summa cum laude) and M.S. degrees in electrical engineering from Chung-Ang University, Seoul, Republic of Korea, in 1995 and 2000, respectively, and the Ph.D. degree from the Department of Electrical and Computer Engineering, The University of Texas at Austin, in May 2004. He received the Texas Telecom Consortium (TexTEC) Fellowship, from Fall 2001 to Spring 2002 and from Fall 2002 to Spring 2003. From August 2004 to August 2005, he was with the Samsung Advanced Institute of Technology (SAIT), Gyeonggi-do, Republic of Korea, where he did research on 4G cellular systems. Since September 2005, he has been a Faculty Member with the Division of Software, Yonsei University, Wonju, Republic of Korea. He was a Visiting Scholar with Purdue University, from September 2014 to August 2015. 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.

HYO-SANG LIM (Member, IEEE) received the B.S. degree from Yonsei University, Republic of Korea, and the M.S. and Ph.D. degrees in computer science from the Korea Advanced Institute of Science and Technology. He was a Postdoctoral Research Fellow with Purdue University, from 2007 to 2011. He is currently an Associate Professor with the Division of Software, Yonsei University, Wonju, Republic of 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).