Deep CNN based Pilot Allocation Scheme in Massive MIMO systems

Kwihoon Kim[1], Joohyung Lee[2]
[1] Department of Artificial Intelligence Convergence Education, Korea National University of Education
Cheongju 28173, South Korea
[e-mail: kimkh@knue.ac.kr]
[2] Department of Software, Gachon University
Seongnam 13120, South Korea
[e-mail: j17.lee@gachon.ac.kr]
*Corresponding author: Joohyung Lee

Received October 25, 2019; revised March 15, 2020; accepted June 28, 2020; published October 31, 2020

Abstract

This paper introduces a pilot allocation scheme for massive MIMO systems based on deep convolutional neural network (CNN) learning. This work is an extension of a prior work on the basic deep learning framework of the pilot assignment problem, the application of which to a high-user density nature is difficult owing to the factorial increase in both input features and output layers. To solve this problem, by adopting the advantages of CNN in learning image data, we design input features that represent users' locations in all the cells as image data with a two-dimensional fixed-size matrix. Furthermore, using a sorting mechanism for applying proper rule, we construct output layers with a linear space complexity according to the number of users. We also develop a theoretical framework for the network capacity model of the massive MIMO systems and apply it to the training process. Finally, we implement the proposed deep CNN-based pilot assignment scheme using a commercial vanilla CNN, which takes into account shift invariant characteristics. Through extensive simulation, we demonstrate that the proposed work realizes about a 98% theoretical upper-bound performance and an elapsed time of 0.842 ms with low complexity in the case of a high-user-density condition.

Keywords: Deep Learning, CNN, MLP, pilot contamination, pilot assignment, massive MIMO, SIR

1. Introduction

The Multi-Input Multi-Output (MIMO) technology is the most important technology in terms of network capacity in mobile communication [1,2]. The base stations (BS) for 6G technology have considered in using a number of antennas for the large capacity improvements [3,4]. This is because the use of more antennas in massive MIMO technologies than conventional MIMO technologies significantly improves the radiative energy efficiency and throughput by the elimination of intra-cell interference and concentrates power in narrow areas of space [5,6]. The Massive MIMO is generally known to be more than 100 times more efficient in radiant energy, as well as 10 times more capacity growth. In order to mitigate these negative impacts from adopting massive MIMO technology such as "contamination of pilot", which is the inter-cell interference effect [7,8], it requires the difficult task of assigning the pilots for users in mobile communication [9,10].

There are various approaches for realizing efficient pilot assignment solutions that improve the capacity of the overall systems [15]. It is challenging due to the pilot allocation problem with a large complexity, and it is difficult to formulate a mathematical model [11,12]. Firstly, Marzetta proposed a random pilot assignment scheme (R-PAS) as a basic approach that can be easily deployed in a practical system [7]. The R-PAS provides an minimum elapsed time owing to its fixed time complexity. However, the overall network capacity of the R-PAS is limited as compared to the theoretical optimal bound. This is owing to the fact that the R-PAS does not take into consideration the users' different channel qualities. In [16], a smart pilot assignment scheme (S-PAS) was invented, which allocates the pilot sequence with the smallest inter-cell interference to the worst channel condition's user in a sequential manner. In a multi-cell environment, all the cells are required to iteratively solve their own optimization problems until the solution is converged [13,14]. Owing to such an iterative procedure in the multi-cell environment, the S-PAS results in an increased computing delay as the number of cells increases. Here, the computing delay also depends on its convergence step size where there is a trade-off between performance and computing delay cost according to the step size. Recently, as a first approach to adopting deep learning techniques, which have recently emerged as a useful tool for solving complex and difficult problems with a large search space, in our preliminary work [17], we proposed a deep-learning-based pilot assignment scheme (DL-PAS). The DL-PAS achieves almost a 99.38% ideal performance with low complexity within a computing time of 0.92 ms on average. Even though this preliminary work suggests a novel method for adopting machine learning technology into a wireless networking problem, it is restrictively applicable only to low-density users due to the factorial increase in input features and output layers [18,19]. Thus, a performance study on high-density users was not conducted. Accordingly, the previous studies are based on heuristic approaches with sub-optimal performance or are hard to deploy in a practical system owing to their high time/space complexity [20,21]. Thus, a practical solution to the pilot assignment problem is required, which motivated the works [22].

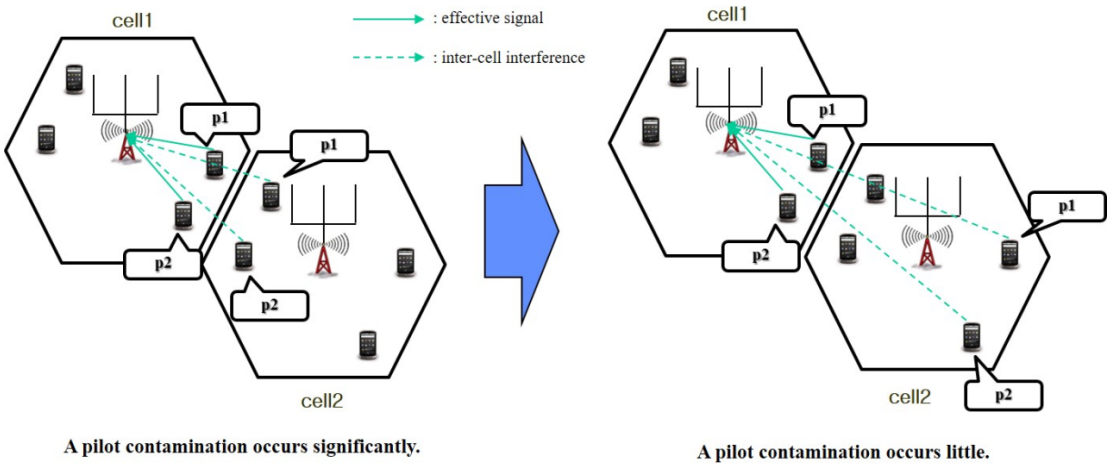


Fig. 1. Importance of assigning the pilot

This paper introduces a deep convolutional-neural-network-based (CNN-based) pilot assignment scheme (DC-PAS) for the massive MIMO systems. The proposed work is an extension of a prior work and adopts the advantages of CNN in learning image data [23-25]. The important contributions are summarized as follows.

• In our scheme, by applying the CNN mechanism to the pilot assignment problem, we design new input features of the users' locations in all cells as the image data with a two-dimensional fixed-size matrix.

• Instead of using a sigmoid function, we construct output layers with linear space complexity according to the number of users. Here, in order to improve the accuracy of the proposed work, we utilize a sorting mechanism to map the pilot index into the user index in the ascending order.

• A theoretical framework for the network capacity model of massive MIMO is derived by extending the model presented in [7], and we apply it into the training process.

• We deploy the proposed DC-PAS under a commercial vanilla CNN by considering shift invariant characteristics. Using simulation, we check that the proposed work realizes about a 98 % theoretical upper-bound performance and an elapsed time of 0.842 ms with low complexity in the case of a high-userdensity condition.• Our research is not limited to this vanilla CNN. There are CNNs that have been modified recently [23, 25, 28], and it is easy to apply to reflect modified CNNs. Later, it will be the future work to apply the revised CNN to improve performance.

We can thus develop a practical learning-based pilot assignment scheme that is applicable to high-density users and suggest a method by which the deep learning mechanism could improve the network performance in the case of complex problems.

Section II presents a overview of the massive MIMO. In Section III, we formulate the problem as a deep learning model and present the novel DC-PAS. Numerical results and performance analysis are explained in Section IV. We summarize and conclude this work in Section V.

2. System Model

For the reader's convenience, in **Table 1**, we present a list of the major symbols that are used in this paper with their definitions.

We consider the massive MIMO systems that consist of L multiple cells. Each cell has a BSBS with M omni-directional antennas, where M is generally assumed to be very large in the massive MIMO systems such that $M \to \infty$. Pilot contamination occurs owing to a user assigned with the same pilot assignment from another cell. As illustrated in **Fig. 1**, the pilot contamination is small when assigning the same pilot to another cell's users who are far from the cell. Within each cell, there are multiple terminals with a single antenna that conducts the uplink transmission. In order to model this in a manner similar to our preliminary work [17], we define I as the set of cells, where $|I|=L$ denotes the total number of cells, and K_i as the set of terminals in the i-th cell for each $i \in I$, where $|K_i|=U_i$ denotes the total number of terminals in the cell. For the channel estimation, a set of orthogonal pilots is assigned and transmitted to the terminals, where the total number of orthogonal pilots used in each cell is P. Correspondingly, the BSBS can measure the channel condition for up to P terminals, which means that the maximum number of terminals within a cell that can simultaneously transmit data is only up to P terminals. The channel is composed of large- and small-scale propagation effects. The large-scale propagation effect comprises pathloss and shadowing. For each $i, j \in I$ and $k \in K_j$, we define $r_{i,k}$ and $z_{i,k}$, which denote the distance and shadowing, respectively, between the BSBS of the i-th cell and k-th terminal, which may reside in the j-th cell. Accordingly, as in [7, 8], the large-scale propagation effect can be modeled using $\beta_{i,k} = z_{i,k} r_{i,k}^{-\gamma}$, where γ is the pathloss exponent. In addition, in the case of the small-scale propagation effect, the fading coefficient between the m-th antenna of the BS in i-th cell and k-th terminal is denoted by $h_{m,i,k}$, where the coherence time T_c is assumed to be constant over the fading coefficient $h_{m,i,k}$. Finally, the overall channel gain $g_{m,i,k}$ between the m-th antenna of the BS in i-th cell and k-th terminal is denoted by $g_{m,i,k} = h_{m,i,k}\sqrt{\beta_{i,k}}$. As assumed in other literatures on massive MIMO, we assume that $\{\beta_{i,k}\}$ values are known at the BSs such that it can be learned in the network. This is because $\{\beta_{i,k}\}$ are varying *much more slowly* than the small-scale fading coefficients, and they are independent of frequency [8]. In contrast, the fading coefficient $h_{m,i,k}$ should be estimated for every coherence time T_c. For the estimation of $h_{m,i,k}$, it is assumed that a set of P orthogonal pilot sequences is used for each cell wherein P depends on the channel and system parameters such as the maximum delay spread, channel coherence time, etc. In addition, the *orthogonality* of the pilots can be implemented using a few methods: i) each pilot can be transmitted in the form of one orthogonal frequency division multiplexed (OFDM) symbol in one subcarrier or ii) each pilot can be transmitted in the form of an orthogonal sequence over the time–frequency domain. Here, the proposed work is not restricted to any specific implementation; only P the number of pilots is important in our analysis.

Moreover, as in [7], we assume that all other cells in the network use the same pilot set. We investigated the effect of using different orthogonal pilot sequences in other cells, and it was demonstrated that essentially the same performance was obtained.

Table 1. List of major symbols

Symbol	Description
L	The total number of cells
M	The total number of omni-directional antennas of a BS
I	The set of cells
K_i	The set of terminals in the i-th cell
U_i	The number of terminals in the i-th cell
P	The total number of pilots of a BS
i	The i-th cell
m	The m-th antenna
k	The k-th terminal
p	The p-th pilot
$r_{i,j,k}$	The distance between the i-th cell and k-th terminal in the j-th cell
$z_{i,j,k}$	The shadowing between the i-th cell and k-th terminal in the j-th cell
$\beta_{i,j,k}$	The large-scale fading between the i-th cell and k-th terminal in the j-th cell
γ	The pathloss exponent
$h_{m,i,k}$	The small-scale fading between the m-th antenna of the BS in the i-th cell and k-th terminal
T_c	The coherence time
$g_{m,i,k}$	The overall channel gain between the m-th antenna of the BS in the i-th cell and k-th terminal
$SIR_{i,k}^{u}$	The uplink signal-to-interference ratio (SIR) of the k-th terminal in the i-th cell
B_w	The bandwidth
α	The frequency reuse factor
T_{block}	The OFDM symbols of one frame
T_{pilot}	The time for sending reverse pilots
T_u	The data transmission interval
T_s	The OFDM symbol interval
C_i^u	The total uplink capacity of i-th cell
$C_{network}^{u}$	The total uplink capacity of the entire network comprising L cells

The channel vector from the k-th terminal in the j-th cell to the m-th antenna of the BS in the i-th cell is given by

$$\mathrm{g}_{m,i,j,k} = \mathrm{h}_{m,i,j,k}\sqrt{\beta_{\mathrm{i,j,k}}} = \mathrm{h}_{m,i,j,k}\sqrt{\mathrm{z}_{\mathrm{i,j,k}}\mathrm{r}_{\mathrm{i,j,k}}^{-\gamma}}. \quad (1)$$

where $h_{m,i,j,k}$ is the small-scale fading between the m-th antenna of the BS in the i-th cell and k-th terminal in the j-th cell, $\beta_{i,j,k}$ is the large-scale fading between the i-th cell and k-th terminal in the j-th cell, $z_{i,j,k}$ is the shadowing between the i-th cell and k-th terminal in the j-th cell, $r_{i,j,k}$ is the distance between the i-th cell and k-th terminal in the j-th cell, and γ is the pathloss exponent.

The uplink SIR of the k-th terminal in the i-th cell is obtained as

$$SIR_{i,k}^{u} = \frac{|g_{m,i,i,k}^{H} g_{m,i,i,k}|^2}{\sum_{j \neq i} |g_{m,i,j,k}^{H} g_{m,i,j,k}|^2 + \frac{|\epsilon_{i,k}^{u}|^2}{\rho_u}}.$$

With $M \to \infty$, we obtain

$$\underline{M \to \infty} \quad \frac{\beta_{i,i,k}^2}{\sum_{j \neq i} \beta_{i,j,k}^2}. \tag{2}$$

In the uplink, the capacity of the k-th terminal in the i-th cell is given by

$$C_{i,k}^{u} = B_w \left(\frac{1}{\alpha}\right) \left(\frac{T_{block} - T_{pilot}}{T_{block}}\right) \left(\frac{T_u}{T_s}\right) \log_2\left(1 + SIR_{i,k}^{u}\right) \left(\frac{\frac{bps}{terminal}}{cell}\right) \tag{3}$$

where B_w is the bandwidth, α is the frequency reuse factor, T_{block} is the time for the OFDM symbols of one frame, T_{pilot} is the time for sending reverse pilots, T_u is the data transmission interval, and T_s is an OFDM symbol interval.

The sum capacity of i-th cell is given by

$$C_i^u = \sum_{k=1}^{U_i} C_{i,k}^u \quad \left(\frac{bps}{cell}\right) \tag{4}$$

The total capacity of the entire network comprising L cells is given by

$$C_{network}^u = \sum_{i=1}^{L} C_i^u \quad (bps) \tag{5}$$

For achieving maximum capacity of the entire systems, the optimization over *all the cells* in the network should be conducted *simultaneously*. Here, the large-scale propagation effect $\beta_{i,j,k}$ is considered as a correlated parameter as it results in inter-cell interference.

In [7], considering only the special case of $P = U_i, \forall i$, the capacity of the entire systems was studied. In practice, however, the assumption of $P = U_i, \forall i$ is not realistic because the actual number of terminals U_i can be varied from different cells, and moreover, it can change with time. That is, the number of actual terminals U_i cannot be generalized as P. This has motivated our work.

Considering the general case of an arbitrary number of terminals U_i, our objective is to improve the capacity of the massive MIMO systems. The case of $U_i > P$ can be considered as a high-user-density scenario; in this case, only P terminals out of the U_i terminals must be selected for data transmission in each time slot, which will be referred to as *terminal selection*. In contrast, the case of $U_i \leq P$ can be considered as a low-user-density scenario; in this case, all U_i terminals should be selected for data transmission, which means that no terminal selection is required. However, U_i pilots should be selected for channel estimation out of P pilots. This is referred to as *pilot selection*. Finally, when the terminals or pilots are selected, the (selected) pilots must then be assigned to the (selected) terminals, which is referred to as *pilot assignment*. It should be noted that pilot assignment is required for both overloaded and underloaded cases. When there are U_i terminals and P pilots in a cell, $\min(U_i, P)$ means the maximum number of pilots, which can be assigned to terminals in a cell. Specifically, $\min(U_i, P)$ pilots are assigned to $\min(U_i, P)$ terminals. We first derive the capacity as a function of the selected terminals, selected pilots, and pilot assignment. Using the capacity expression, we then propose an optimal equation.

3. Proposed DC-PAS : Deep CNN Learning Framework

The pilot assignment with our proposed work is based on a deep CNN learning that could be suitable for large neuron networks with shift-invariant characteristics such as the image recognition area [23].

3.1 Preliminaries

There is a limitation of the DL-PAS in applications to a high-user-density nature owing to the factorial increase in both output nodes and multi-layer perceptions (MLP) weight.

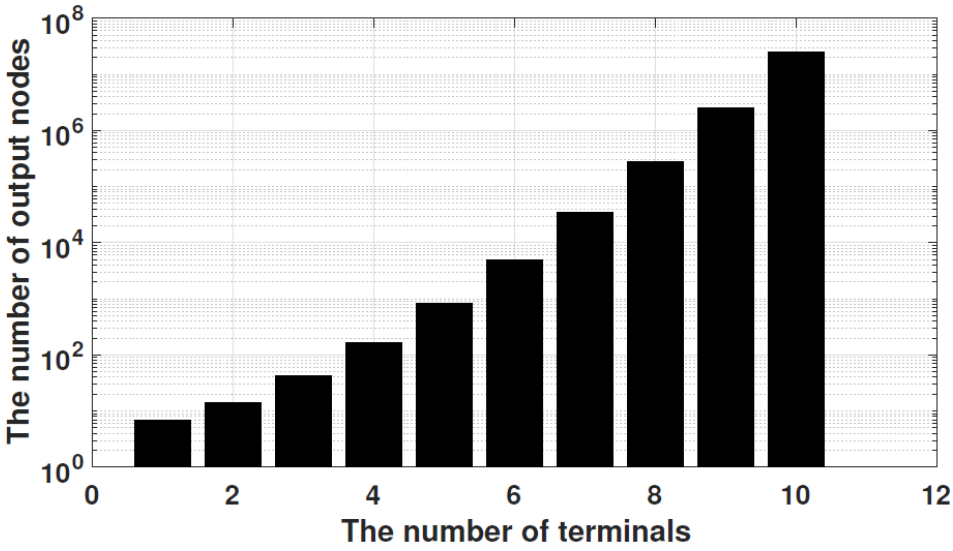


Fig. 2. Problem of high density in DL-PAS

As shown in **Fig. 2**, even though the DL-PAS performs well in the case of a low-user-density, it requires a significant amount of output layers even in the case of nine users. For ten users, almost 25,401,600 output nodes are required. That is, it is impossible to implement the DL-PAS in a practical system for handling a high-user-density.

In machine learning, the deep CNN is a image oriented neural network in many areas. The deep CNN outperforms other classification algorithms with little preprocessing by reflecting the location characteristics of pictures. Such advantages have inspired us to apply the deep CNN mechanism to the pilot assignment problem in order to solve the problem focused on in prior works. The MLP is a mesh-shaped neuron network structure that is easy to create. Even though it performs well with almost a 100 % accuracy in learning for small-sized neuron networks, it requires high complexity and a huge amount of data to learn for realizing a relevant accuracy. That is, it is only suitable for small-sized neuron networks. Conversely, deep CNN is suitable for large-sized neuron networks. For instance, the ResNet, which is a good example of a deep CNN, consists of more than 100 layers and a high number of input nodes as the number of image pixels in the picture [23]. It should be noted that deep CNN is less precise in learning in the case of a small-sized neuron network.

3.2 Proposed DC-PAS algorithm

We introduce the entire deep learning process to apply to the proposed system.
Deep learning can be adopted in the pilot assignment scheme by following procedure:

3.2.1 Getting data for machine learning

In the case of $U_l > P$, only P terminals must be selected out of U_l for data transmission. The set of selected terminals is defined as follows.

Definition 1 *(Set of Selected Terminals):* Let Ω_l denote the index set of the selected terminals in the l-th cell (out of all U_l terminals) with $|\Omega_l| = \min(P, U_l)$. □

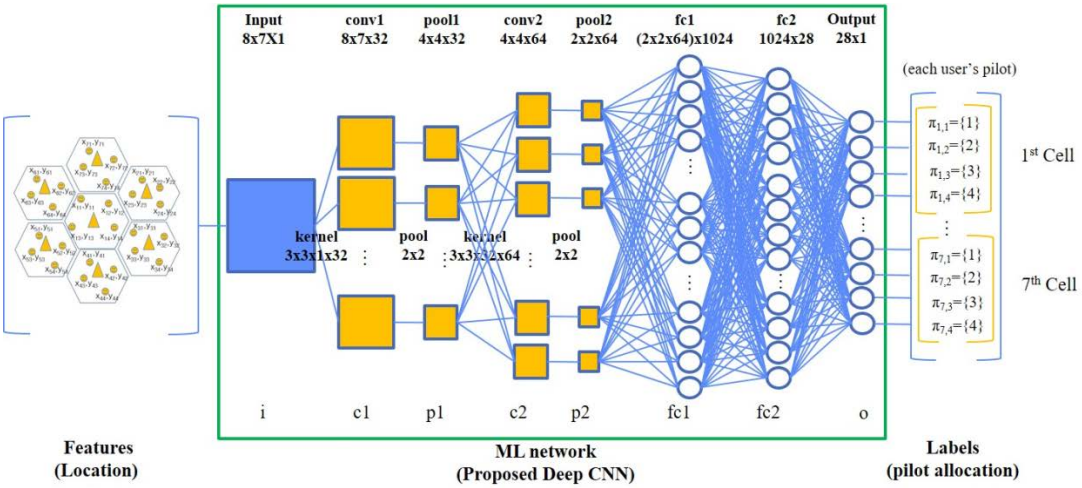


Fig. 3. Proposed deep CNN-based pilot assignment scheme

The next issue is pilot selection: when $U_l \leq P$, only U_l pilots must be selected out of P for the channel estimation of U_l terminals. The set of selected pilots is defined as follows.

Definition 2 *(Set of Selected Pilots):* Let Γ_l denote the set of selected pilots in the l-th cell (out of all P pilots) with $|\Gamma_l| = \min(P, U_l)$. □

Once the terminals and pilots are selected, they must be one-to-one mapped. This pilot assignment is defined by a one-to-one mapping $\Pi_l(\cdot)$ between the terminal indices in Ω_l and the pilot indices in Γ_l.

Definition 3 *(Mapping between Terminals and Pilot (pilot assignment)):* When $p \in \Gamma_l$, $\Pi_l(p)$ denotes the index of the terminal in Ω_l, which p-th pilot is allocated to. When $p \notin \Gamma_l$, $\Pi_l(p)$ does not give any index of the terminal in Ω_l, and it is denoted as $\Pi_l(p) = 0$, which means that the p-th pilot is *not* assigned to any terminal in l-th cell. □

Using the expression of $C^u_{network}$ in equation (5), we propose the following optimization problem in order to maximize the network capacity:

$$\left(\Omega_l^{opt}, \Gamma_l^{opt}, \Pi_l^{opt}\right) = \arg\max_{(\Omega_l, \Gamma_l, \Pi_l)} C_{network}(\Omega_l, \Gamma_l, \Pi_l). \quad (6)$$

It should be noted that (6) is a *joint* optimization over all *three* variables $(\Omega_l, \Gamma_l, \Pi_l)$. However, if we consider the cases of $U_l > P$ and $U_l \leq P$ individually, the problem is reduced to a simpler joint optimization of (Ω_l, Π_l) or (Γ_l, Π_l).

Subsequent to extracting the equation of the joint optimization for the terminal selection and pilot assignment according to the location of each terminal, we use that equation as the learning data for the deep CNN model. 1) Induce joint optimization for terminal selection and pilot assignment in an equation according to the terminal location. 2) Calculate the joint optimization for the terminal selection and pilot assignment according to the random terminal location. 3) Classify the data using the deep CNN model. The input value is considered as the terminal location, and the output is considered as the result of the joint optimization for the terminal selection and pilot assignment. 4) Data could be obtained from the result of the joint optimization for the terminal selection and pilot assignment equation.

3.2.2 Building the deep CNN model and the proposed pilot assignment scheme

We use 32 kernels (3x3 size) and max-pooling (2x2 size). Finally, we use two fully connected layers to distinguish the output as shown in **Fig. 3**.

The algorithm for designing the input feature is explained in Algorithm 1. We initialize the input data from the coordinates of all the cells and terminals by mapping two-dimensional coordinates onto a one-dimensional array. Based on this input data, we create a two-dimensional image matrix for the CNN.

Algorithm 1 Algorithm for designing the input feature based on the square matrix imagination method

Initialization :
(a) Obtaining $(x_{i,k}, y_{i,k})$, which are the coordinates of the i-th cell and k-th terminal.

Two-dimensional coordinates onto one-dimensional array
(a) Ordering the cells and terminals.
- 1st cell : the center of the cell is first.
- 2nd cell~7th cell : the other outer cells are clockwise.

(b) Ordering the terminals in the cells.
(c) One-dimensional input plat array ($in_{plat_{i,k}}$) :
for i = 1 to L for k = 1 to U_l
$in_{plat_{U_l*(i-1)+2*(k-1)+1}} = x_{i,k}$;
$in_{plat_{U_l*(i-1)+2*(k-1)+2}} = y_{i,k}$;
end for end for

Preparing the input feature : two-dimensional image matrix
(a) Width of input square matrix (S) : $\lceil \sqrt{L * U_l * 2} \rceil$
(b) Input two-dimensional image matrix ($in_{a,b}$) :
for a = 1 to S for b = 1 to S
if (a*S+b > $L * U_l$) then $in_{a,b} = 0.0$;
else $in_{a,b} = in_{plat_{a*S+b}}$; end if

end for end for

Algorithm 2 Algorithm for designing the output label and assigning the pilot for each terminal based on the pilot value sorting method

Initialization :
(a) Obtaining $\Pi_{i,k}$, which is the pilot assignment of the i-th cell and k-th terminal.

Preparing the output label for training
(a) Output label (out_{train_i}) :
for i = 1 to L for k = 1 to U_l
$out_{train_{U_l*(i-1)+k}} = \Pi_{i,k}$;
end for end for

Assigning the pilot for each terminal
(a) Predicted output label ($out_{predicted_i}$)
(b) Divided output label ($out_{divided_{i,k}}$) in the i-th cell and k-th terminal
(c) Assigned pilot ($out_{assigned_{i,k}}$) in the i-th cell and k-th terminal :
for i = 1 to L
sorted_index = sort($out_{divided_{i,k}}$, ascending);
for k = 1 to U_l
$out_{assigned_{i,sorted_{index[k]}}} = k$;
end for end for

The algorithm for designing the output label and assigning the pilot for each terminal is presented in Algorithm 2. We set the pilot assignment results and called it as the "pilot value" as an output label. For instance, for four pilots, there are four pilot values for identifying those pilots with an index such that {3,2,1,4}. Each location means terminal's order. That is, 3rd pilot is mapped to 1st terminal, 2nd pilot is 2nd terminal, 1st pilot is 3rd terminal, and 4th pilot is mapped to 4th terminal, respectively. Here, the order of the output is a combination of the assigned pilot values for each cell and the number of terminals residing in the cell. It should be noted that during the prediction, the output value might return a real number, such as {1.2,1.9,3.4,4.1}, and not an integer. Thus, the post-processing is necessary for mapping a real number value into an actual pilot value. In Algorithm 2, this post-processing is based on an ascending sorting of the real number values and then mapping them into the integer-valued pilot values in the ascending order. For example, each cell should first be found. For the first cell, we can sort the predicted values in the increasing order. Their order will be $\Pi_{1,1}$={1.2} < $\Pi_{1,2}$={1.9} < $\Pi_{1,3}$={3.4} < $\Pi_{1,4}$={4.1}. We then assign pilots to $\Pi_{1,1}$={1} < $\Pi_{1,2}$={2} < $\Pi_{1,3}$={3} < $\Pi_{1,4}$={4} for each terminal in the first cell. Similarly, for the seventh cell, the expected value is first debited. Their order will be $\Pi_{7,1}$={0.9} < $\Pi_{7,3}$={2.1} < $\Pi_{7,2}$={3.3} < $\Pi_{7,4}$={3.9}. We then assign the pilot value to $\Pi_{7,1}$={1} < $\Pi_{7,3}$={2} < $\Pi_{7,2}$={3} < $\Pi_{7,4}$={4} for each terminal in the seventh cell. The manipulation of training data sets consists of three stages: the creation of training sets, the design of key metrics, and the labeling. First, training samples shall be designed for the generation of training sets. It is going to adopt location of devices as training set in order to detect major effects such as large-scale fading. For L cells, U_l terminals, and P pilots ($U_l = P$ for balance), the input size is 2 (position/terminal coordination) x U_l (terminal/cell) x L (cell). For example, suppose each of the seven cells has four terminals, the input feature vector is [8 x 8 x 1]. Second, major performance metrics

should be designed. This is adopted as a key metric to maximize total uplink network capacity$C_{network}(\Omega_l, \Gamma_l, \Pi_l)$. Finally, we should label on the basis of this key performance indicator. Then the label for each set of training is calculated (e.g. pilot assignment). Common deep learning algorithms, such as CNN, choose one of the combinations of the total result values on the label. That is, the number of output-layer neurons is the number of combinations that get the total result. For DC-PAS, L cell, U_l terminal, and P pilot ($U_l = P$ for balance), it is not practical to deploy as the terminal U_l number increases because the output label is $(U_l!)$ x L. Thus, to address this problem, a new version of the deep learning network is proposed, with minor modifications to the conventional deep learning network, to reduce the number of labels. The key idea is to resize the output layer to the number of pilot allocation cases per cell. This is because each cell has its own pilot task. Thus, for L cell, U_l terminal, and P pilot, the size of the output label is (U_l) x L. For example, assuming that each of the seven cells has 10 terminals, the combination of the results is required to be $(10!)$ x $7 = 25{,}401{,}600 \approx 2.54$ x 10^7. This means that the size of the output label is $[2.54 \text{ x } 10^7]$ and the number of neurons in the output layer is $[2.54 \text{ x } 10^7]$. For example, if each of the seven cells has four terminals, 10 x $7 = 70$, only a combination of result values is required. That is, while the traditional method requires $[2.54 \text{ x } 10^7]$, the number of neurons in the output layer is only $[70]$. Therefore, the proposed DC-PAS would reduce the number of neurons in the order of 10^6. Afterwards, extract $\Pi_{i,k}$=$\{p\}$ label, where $\Pi_{i,k}$ is a pilot allocation with the i-th cell and the k-th terminal.

4. Performance Evaluation

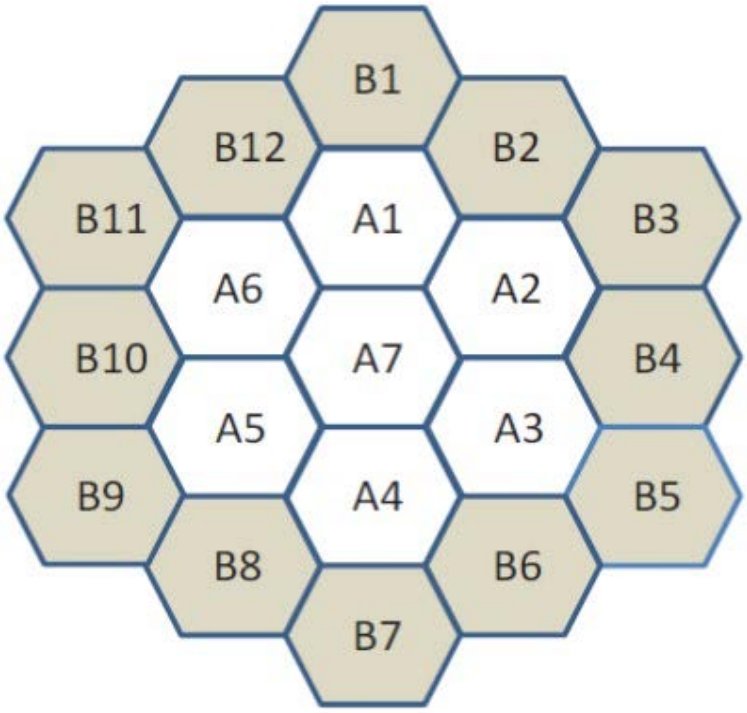


Fig. 4. Simulation setting

Here, the detailed performance evaluation of the suggested DC-PAS algorithm is explained. We compare the proposed DC-PAS with the traditional schemes such as a deep-learning-based pilot assignment scheme (DL-PAS), a random pilot assignment scheme (R-PAS), a smart pilot assignment scheme (S-PAS) and a exhaustive-search pilot assignment scheme (ES-PAS). The performance measures used are the deep learning accuracy, normalized capacity gain, elapsed time, and training time. The deep learning accuracy is considered as a metric for evaluation of classification models. Specifically, accuracy has the following definition: Accuracy=Number of correct predictions/Total number of predictions [26, 27]. Accordingly, in this paper, we measured this learning accuracy by comparing between exhaustive based pilot assignment (theoretical upper-bound = correct predictions) and prediction based pilot assignment.

Table 2. Deep learning system environment

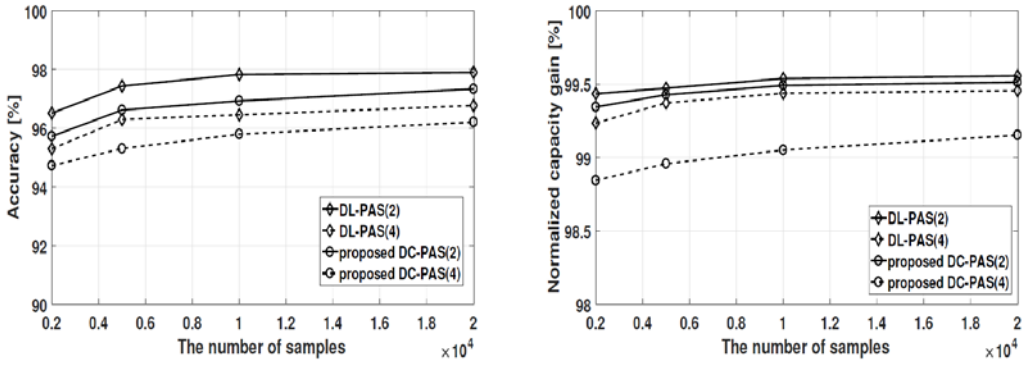
	Description
DL toolkit	Tensorflow 1.12
Optimization	Adam optimizer
Loss function	Mean square error
Learning rate	10^{-4}
Dropout	Train : 0.5, Test : 1.0
Activation	Elu

Table 3. Mobile network model environment

	Description
Network	19 hexagonal cells : seven inner cells and twelve outer cells
Cell	Hexagonal cell, r_c : 1600m, r_h : 100m at the center of cell
Terminal	Uniformly, terminals are generated within the hexagonal cells
Wireless	Frequency re-use factor : 1, path-loss exponent : 4, standard deviation : 8.0 dB

Table 2 lists the details of the deep learning system environment with the TensorFlow with the Adam optimizer, the loss function of the mean square error, the learning rate of 10^{-4}, the activation of Elu and the dropout which are 0.5 for train and 1.0 for test. We get the simulation results for a mobile network consisting of twelve outer cells in the outer ring and nineteen hexagonal cells consisting of seven inner cells. Twelve external cells are used as boundaries of the network, and the capacity is measured only in seven internal cells in **Fig. 4** and **Table 3**. As in [7], we suppose that the cell radius is 1600 m and that there is a hole in the center of the cell with a radius 100 m. For each cell, the terminals are created uniformly within each cell except in the central holl area. Assume that the frequency reuse factor is 1, the path-loss exponent is 4, and the shadowing standard deviation is 8.0 dB.

Fig. 5 shows the performance analysis for low-user-density ($U_l \leq P$). The accuracy of the suggested DC-PAS with various numbers of users, i.e., two users and four users, is denoted by DC-PAS(2) and DC-PAS(4), respectively, as the samples number varies from 2,000 to 20,000. The DL-PAS with various users' numbers, i.e., two users and four users are also denoted by DL-PAS(2) and DL-PAS(4), respectively.



(a) (b)

Fig. 5. Performance in low-user-density case ($U_l \leq P$) for (a) accuracy, (b) normalized capacity gain

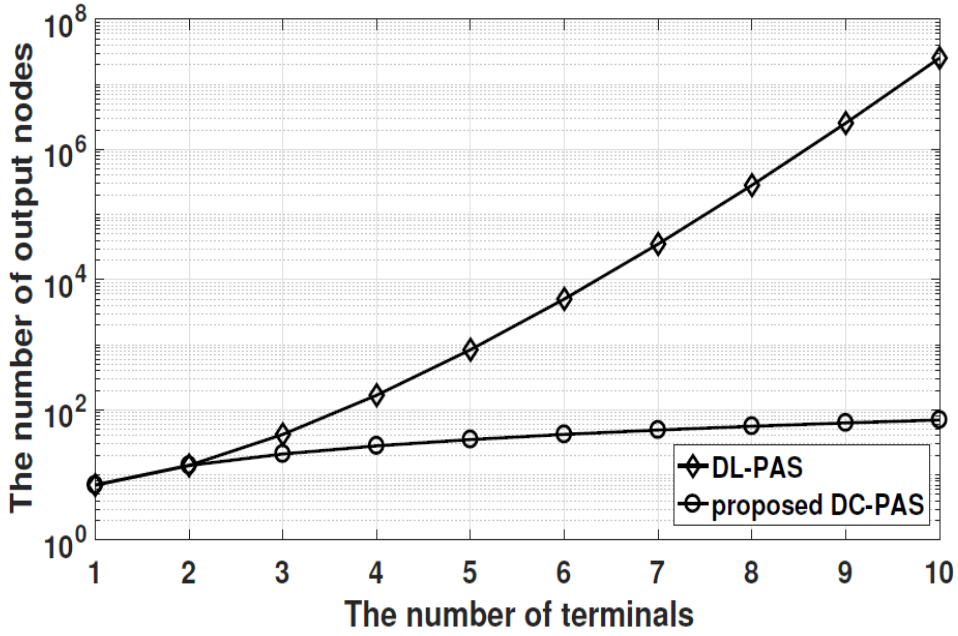


Fig. 6. Comparison of output nodes with the DL-PAS versus that with the proposed DC-PAS

As depicted in **Fig. 5(a)**, the accuracy of the suggested DC-PAS increases as the sample size increases, where the achieved accuracy of the suggested DC-PAS is almost 96.09%, which is similar to the 96.82% of the DL-PAS. For the normalized capacity gain in **Fig. 5(b)**, both the DL-PAS and the proposed DC-PAS can realize a similar performance as 99.44% and 99.22%, respectively. The proposed DC-PAS is characterized by searching for similar patterns but has few patterns for a small number of users, which indicates a poor performance. The DC-PAS is slightly less capable than the DL-PAS but does not differ significantly. That is, the normalized capacity gain of the suggested DC-PAS is similar to that of the DL-PAS.

Fig. 6 shows the comparison of the output nodes with the DL-PAS versus that of the suggested DC-PAS. As expected, as the number of terminals increases, the number of output nodes in the case of the DL-PAS has faster growth rates than those of the suggested DC-PAS, which has linear growth rates. That is, the proposed DC-PAS requires a smaller number of output nodes for a high number of terminals.

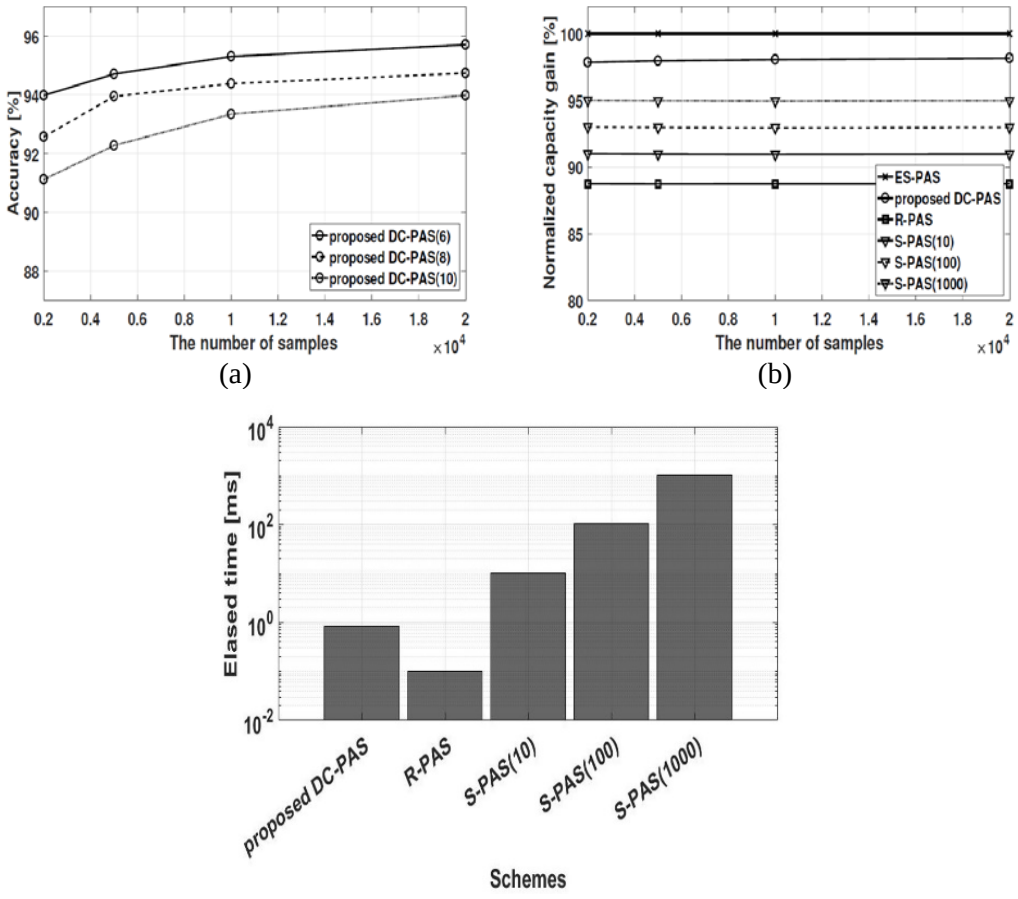


Fig. 7. Performance for the high-user-density case ($U_l > P$) for (a) accuracy of the suggested DC-PAS, (b) normalized capacity gain, and (c) elapsed time

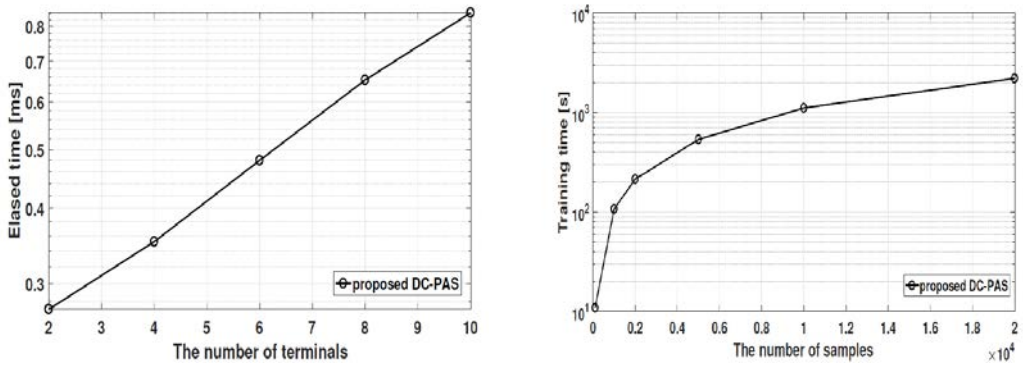
Fig. 7 shows performance analysis for a high-user-density ($U_l > P$). The proposed accuracy of DC-PAS is expressed as DC-PAS (6), DC-PAS (8), and DC-PAS (10) because each of the different sample numbers, along with six users. As **Fig. 7(a)** shows, the proposed accuracy of the DC-PAS increases to an average of 93.84% of the proposed work. When user densities are high, DL-PAS should be complex in implementation and excluded from evaluation. **Fig. 7(b)** illustrates the capacity gains for the various schemes. We define the *normalized capacity gain* G_n as

$$\mathrm{G}_n = \frac{C^u_{network}}{C^u_{network-upper}} \tag{7}$$

where the theoretical upper-bound capacity $C^u_{network-upper}$ can be calculated from ES-PAS.

In this study, various iterations of S-PAS(1000) were tested with S-PAS(10), S-PAS(100), and S-PAS(1000) repeated 10 times, 100 times, and 1000 times, respectively.
The normalized capacity gain is ES-PAS(100%) > DC-PAS(98%) > S-PAS(1000)(95%) > S-PAS(100)(93%) > S-PAS(10)(91%) > R-PAS(89%) for each scheme. The DC-PAS realizes a nearly 98.15% normalized capacity gain of in 20,000 samples and ten terminals. As

shown in **Fig. 7(a)**, the proposed normalized capacity increase in DC-PAS increases slightly with the number of samples. On the other hand, other existing schemes are independent of the number of samples because they do not adopt a learning mechanism. As a result, it is clear that the proposed work exceeded the existing pilot assignment system.Specifically, R-PAS, which randomly assigns pilots to users, shows the worst performance despite the complexity and ease of deployment. Normalised gains from S-PAS increase with the number of repetitions. It should be noted that increasing the number of iterations of the S-PAS significantly increases the elapsed time and constrains the S-PAS. **Fig. 7(c)** shows performance in terms of time elapsed from various pilot allocation plans. where R-PAS has the lowest elapsed time because of its low complexity. Only 0.842 ms is required for the proposed DC-PAS. It should be noted that the computational ability and training time of the in-depth learning process can be further analyzed as a practical concern. For S-PAS, the elapsed time increases with the number of repetitions such as S-PAS(1000) > S-PAS(100) > S-PAS(10). It also evaluates the elapsed time of the ES-PAS returning the theoretical capacity upper limit. Despite notable performance, the elapsed time in this case is nearly 13 hours in the same environment. This means that this evaluation is not included in the figure because the ES-PAS cannot be placed in the actual system.



(a) (b)

Fig. 8. Performance analysis (a) elapsed time of the suggested DC-PAS, (b) training time

Fig. 8 shows the performance in terms of the elapsed time and training time for the proposed DC-PAS. We observe that the elapsed time increases according to the number of terminals as shown in **Fig. 8(a)**. This is because, as the number of terminals increases, the number of output nodes increases linearly, thus resulting in a larger model size and greater complexity. In addition, we further observe that the training time also increases according to the number of samples as shown in **Fig. 8(b)**. This is because a larger number of samples requires a larger amount of data for processing in the training phases.

5. Conclusion

We have proposed a DC-PAS in order to reduce the difficulty of applying a deep-learning-based scheme to a high-user-density nature because of the factorial increase in both input features and output layers. Specifically, the input and labels are considered as the users' positions in all the cells and the pilot allocations, respectively. The pre-trained optimal pilot allocations with the given users' positions are provided through a mathematical calculation for a high-user-density as the training data set, and a near-optimal pilot assignment

is returned from analyzing the data set. The obtained results demonstrate that our proposed DC-PAS realizes an nearly 98% performance with low complexity and requiring a computing time of only 0.842 ms. In the future, in order to reduce the load changes and change the minimum pilot assignment, the current work can be expanded by utilizing reinforcement learning considering the previous state.

References

[1] M. A. Wijaya, K. Fukawa and H. Suzuki, “Intercell-Interference Cancellation and Neural Network Transmit Power Optimization for MIMO Channels," in *Proc. of IEEE Vehicular Technology Conference Fall*, Sep. 2015. Article (CrossRef Link).

[2] S. Vishwanath, N. Jindal, and A. Goldsmith, “Duality, achievable rates, and sum-rate capacity of Gaussian MIMO broadcast channels," *IEEE Trans Inf. Theory*, vol. 49, pp. 2658-2668, Oct. 2003. Article (CrossRef Link).

[3] H. Weingarten, Y. Steinberg, and S. Shamai, “The capacity region of the Gaussian multiple-input multiple-output broadcast channel," *IEEE Trans. Inf. Theory*, vol. 52, pp. 3936-3964, Sep. 2006. Article (CrossRef Link).

[4] Q. H. Spencer, A. L. Swindlehurst, and M. Haardt, “Zero-forcing methods for downlink spatial multiplexing in multiuser MIMO channels," *IEEE Trans. Signal Process.*, vol. 52, pp. 461-471, Feb. 2004. Article (CrossRef Link).

[5] G. J. Foschini, M. K. Karakayali, and R. A. Valenzuela, “Coordinating multiple antenna cellular networks to achieve enormous spectral efficiency," *Proceedings of the IEEE*, vol. 153, pp. 548-555, Aug. 2006. Article (CrossRef Link).

[6] D. Vesbert, S. Hanly, H. Huang, S. Shamai Shitz, O. Simeone, and W. Yu, “Multi-cell MIMO cooperative networks: a new look at interference," *IEEE J. Sel. Areas Commun.*, vol. 28, pp. 1380-1408, Dec. 2010. Article (CrossRef Link).

[7] T. L. Marzetta, “Noncooperative cellular wireless with unlimited numbers of base station antennas," *IEEE Trans. Wireless Commun.*, vol. 9, pp. 3590-3600, Nov. 2010. Article (CrossRef Link).

[8] F. Rusek, D. Persson, B. K. Lau, E. G. Larsson, T. L. Marzetta, O. Edfors, and F. Tufvesson, “Scaling up MIMO: opportunities and challenges with very large arrays," *IEEE Signal Proces. Mag.*, vol. 30, no. 1, pp. 40-60, 2013. Article (CrossRef Link).

[9] J. Hoydis, S. t. Brink, and M. Debbah, “Massive MIMO: How many antennas do we need?," in *Proc. of IEEE 49th Allerton Conf. on Communication, Control, and Computing (Allerton)*, pp. 545-550, 2011. Article (CrossRef Link).

[10] X. Gao, O. Edfors, F. Rusek, and F. Tufvesson, “Linear pre-coding performance in measured very-large MIMO channels," in *Proc. of IEEE Vehicular Technology Conference Fall*, 2011. Article (CrossRef Link).

[11] H. Q. Ngo and E. G. Larsson, “EVD-Based Channel Estimations for Multicell Multiuser MIMO with Very Large Antenna Arrays," in *Proc. of IEEE Int’l Conf. on Acoustics, Speed and Signal Processing (ICASSP)*, Mar. 2012. Article (CrossRef Link).

[12] G. J. Foschini and J. J. Gans, “On limits of wireless communications in a fading environment when using multiple antennas," *Wireless Personal Communications*, vol. 6, pp. 311-355, 1998. Article (CrossRef Link).

[13] J. Jose, A. Ashikhmin, T. L. Marzetta, S. Vishwanath, “Pilot contamination problem in multi-cell TDD systems," in *Proc. of ISIT 2009*, Jul. 2009. Article (CrossRef Link).

[14] J. Jose, A. Ashikhmin, T. L. Marzetta, S. Vishwanath, “Pilot contamination and precoding in multi-cell TDD systems," *IEEE Trans. Wireless Commun.*, vol. 10, pp. 2640-2651, Aug. 2011. Article (CrossRef Link).

[15] E. Larsson, O. Edfors, F. Tufvesson, T. L. Marzetta, “Massive MIMO for next generation wireless systems," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 186-195, Feb. 2014. Article (CrossRef Link).

[16] X. Zhu, Z. Wang, L. Dai, and C. Qian, “Smart Pilot Assignment for Massive MIMO," *IEEE Commun. Letters*, vol. 19, Sep. 2015. Article (CrossRef Link).
[17] K. Kim, J. Lee, and J. Choi, “Deep Learning based Pilot Allocation Scheme (DL-PAS) for 5G Massive MIMO Systems," *IEEE Commun. Letters*, vol. 22, no. 4, pp. 828-831,Apr. 2018. Article (CrossRef Link).
[18] L. Mekinda and L. Muscariello, “Supervised Machine Learning-Based Routing for Named Data Networking," in *Proc. of 2016 IEEE GLOBELOM*, vol. 20, pp. 1-6, Dec. 2016. Article (CrossRef Link).
[19] J. Joung, “Machine Learning-based Antenna Selection in Wireless Communications," *IEEE Commun. Letters*, vol. 20, pp. 2241-2244, Nov. 2016. Article (CrossRef Link).
[20] M. Alsheikh, S. Lin, D. Niyato and H. Tan, “Machine Learning in Wireless Sensor Networks:Algorithms, Strategies, and Applications," *IEEE Commun. Sur. & Tut.*, vol. 16, No. 4, pp. 1996-2018, 2014. Article (CrossRef Link).
[21] F. Karim, “LSTM Fully Convolutional Networks for Time Series Classification," *IEEE Access*, vol. 6, pp. 1662-1669, 2017. Article (CrossRef Link).
[22] R. A. Alshinina, “A Highly Accurate Deep Learning Based Approach for Developing Wireless Sensor Network Middleware," in *Proc. of 2018 Wireless Telecommunications Symposium (WTS)*, 2018. Article (CrossRef Link).
[23] K. He, X. Zhang, S. Ren and J. Sun, “Deep Residual Learning for Image Recognition," in *Proc. of The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770-778, 2016. Article (CrossRef Link).
[24] J. Deng, W. Dong, R. Socher, L. Li, K. Li and L. Fei-Fei, “ImageNet: A Large-Scale Hierarchical Image Database," in *Proc. of The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. Article (CrossRef Link).
[25] E. Park, W. Liu, O. Russakovsky, J. Deng, L. Fei-Fei and A. Berg, “ILSVRC-2017," *URL http://www.image-net.org/challenges/LSVRC/2017/*, 2017.
[26] Olson et al., *Advanced Data Mining Techniques, 1st edition*, Springer, pp. 138, Feb. 2008.
[27] Powers et al., “Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation," *Journal of Machine Learning Technologies*, pp. 37-63, 2011.
[28] R. Khan, X. Zhang, and R. Kumar, “Analysis of ResNet and GoogleNet models for malware detection,” *Journal of Computer Virology and Hacking Techniques*, vol. 15, pp. 29–37, 2019. Article (CrossRef Link).

Kwihoon Kim is currently an Assistant Professor in the Department of Artificial Intelligence Convergence Education, Korea National University of Education (KNUE), South Korea. He received the B.S, M.S. and Ph.D. degrees from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea in 1998, 2000 and 2019, respectively. He worked in LG DACOM 2000~2005. From 2005 to 2020, he was a Principle Researcher with Electronics and Telecommunications Research Institute (ETRI). He is an editor and rapporteur of ITU-T SG11 since 2006. His interested fields are Fog/edge computing, Internet of Things, 5G/IMT2020, deep learning, machine learning, reinforcement learning, GAN and knowledge-converged intelligent service.

Joohyung Lee is currently an Assistant Professor in the School of AI Software, Gachon University, South Korea. He received the B.S, M.S. and Ph.D. degrees from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2008, 2010 and 2014, respectively. From 2012 to 2013, he was a Visiting Researcher with the Information Engineering Group, Department of Electronic Engineering, City University of Hong Kong, Hong Kong. From 2014 to 2017, he was a Senior Engineer with Samsung Electronics. He has contributed several articles to the International Telecommunication Union Telecommunication (ITU-T) and the 3rd Generation Partnership Project (3GPP). His research work is resource management at the intersection of mobile systems and machine learning focusing on edge computing architectures to optimize the trade-off between latency, energy, bandwidth and accuracy for data analytics.