energies

MDPI

Article

Off-Grid DoA Estimation via Two-Stage Cascaded Neural Network

Hyeonjin Chung [1], Hyeongwook Seo [1], Jeungmin Joo [2], Dongkeun Lee [2] and Sunwoo Kim [1,*]

[1] Department of Electronics and Computer Engineering, Hanyang University, Seoul 04763, Korea; hyeonjingo@hanyang.ac.kr (H.C.); rlrprl@hanyang.ac.kr (H.S.)
[2] Agency for Defense Development, Yuseong P.O. Box 35, Daejeon 34186, Korea; gangsang@add.re.kr (J.J.); tbomg00@hanmail.net (D.L.)
* Correspondence: remero@hanyang.ac.kr

Abstract: This paper introduces an off-grid DoA estimation via two-stage cascaded network which can resolve a mismatch between true direction-of-arrival (DoA) and discrete angular grid. In the first-stage network, the initial DoAs are estimated with a convolutional neural network (CNN), where initial DoAs are mapped on the discrete angular grid. To deal with the mismatch between initially estimated DoAs and true DoAs, the second-stage network estimates a tuning vector which represents the difference between true DoAs and nearest discrete angles. By using tuning vector, the final DoAs are estimated by moving initially estimated DoAs as much as the difference between true DoAs and nearest discrete angles. The limitation on estimation accuracy induced by the discrete angular grid can be resolved with the proposed two-stage network so that the estimation accuracy can be further enhanced. Simulation results show that adding the second-stage network after the first-stage network helps improve the estimation accuracy by resolving mismatch induced by the discretized grid. In the aspect of the implementation of machine learning, results also show that using CNN and using PReLU as the activation function is the best option for accurate estimation.

Keywords: off-grid direction-of-arrival (DoA) estimation; machine learning; cascaded neural network; convolutional neural network (CNN); deep neural network (DNN); sparse representation

Citation: Chung, H.; Seo, H.; Joo, J.; Lee, D.; Kim, S. Off-Grid DoA Estimation via Two-Stage Cascaded Neural Network. *Energies* **2021**, *14*, 228. https://doi.org/10.3390/en14010228

Received: 8 December 2020
Accepted: 30 December 2020
Published: 4 January 2021

1. Introduction

Direction-of-arrival (DoA) estimation is one of the long-studied research topics in array signal processing. DoA estimation algorithms have been adopted in various applications, such as localization and radar [1]. The DoA estimation can also take essential role in cooperative localization for vehicular networks, where the cooperative localization between vehicles requires the relative distances and DoAs of neighboring vehicles [2,3]. Using the aforementioned information, the position of vehicles that are derived from a global positioning system (GPS) can be estimated using cooperative localization [2,3]. As the commercialization of an automotive multiple-input multiple-output (MIMO) radar progresses [4], it is possible for vehicles to harness DoAs of other vehicles using the DoA estimation algorithm.

Traditional DoA estimation algorithms such as MUSIC [5] have high estimation accuracy and resolution. However, they require a large number of snapshots and cannot properly estimate DoAs of coherent signal sources. To overcome these disadvantages, compressive sensing (CS)-based DoA estimation algorithms have been proposed in [6–8], where the CS-based DoA estimation exploits the discrete grid which consists of angular bases. Regardless of the type of CS, such as basis pursuit (BP) [9] and sparse Bayesian learning (SBL) [10], CS-based DoA estimation algorithms are commonly robust against low signal-to-noise ratio (SNR) and fewer snapshots [11]. CS-based DoA estimation exploits the discrete angular grid which consists of angular bases; however, there is a high possibility that the true DoAs do not correspond perfectly with the angular bases in practice. The

mismatch between the DoAs and the angular bases is generally referred to as a grid-mismatch and induces an inevitable estimation error [12].

Aside from the aforementioned studies, an attempt to exploit the machine learning to the DoA estimation has first been proposed in [13,14], and continues to be studied as the deep learning theory and methods are developing fast [15,16]. In [13,14], a support vector regression (SVR) approximates the function that maps the received signals to DoAs. Results in [13,14] show that the DoA can be successfully estimated with SVR, where SVR is a machine learning technique that is mainly used for pattern recognition problems. However, the estimation accuracy of algorithms in [13,14] is inferior to the traditional DoA estimation algorithm such as MUSIC [5]. As computational resources that can support large datasets are not viable currently, small amounts of data are used for training, and this causes low estimation accuracy.

After the introduction of a neural network (NN) [16], DoA estimation algorithms based on various types of NN were proposed in [17–21], where these algorithms are trained and tested with the massive amounts of data. Although deep neural network (DNN)-based algorithms in [17,18] have higher estimation accuracy and resolution than traditional DoA estimation algorithms, they can only estimate the fixed number of DoAs as the number of DoAs that DNN can estimate is also fixed. In practice, the number of signal sources is unfixed and unknown so that the number of DoAs can vary. Thus, algorithms in [17,18] are not suitable for practical situations. Unlike in [17,18], the algorithms in [19,20] can cope with the varying number of DoAs by setting the output of network as an angular grid, where the idea of [19,20] is motivated by the CS-based DoA estimation. The angular grid represents the discretized angular domain and maps where DoAs belong within the grid. In this case, DoAs can be successfully estimated if every DoA matches with one of the discrete angles within the grid, regardless of the number of DoAs. However, the algorithms in [19,20] suffer from the identical problem of CS-based DoA estimation, as a true DoA may not perfectly match with the discrete angles in practice. Thus, the inevitable estimation error occurs when estimating DoAs with algorithms in [19,20].

To prevent the inevitable estimation error induced by mismatch between true DoAs and discrete angles within the grid, we propose a novel off-grid DoA estimation algorithm based on the two-stage cascaded NN. A design of the two-stage cascaded NN is motivated by a two-step approach in [22]. In the first stage, the DoAs are initially estimated with the convolutional neural network (CNN). As the initial DoAs estimated by the first-stage network are mapped on the discrete angular grid, the initially estimated DoAs contain the estimation error induced by the mismatch. In the second stage, the mismatch between the true DoA and angles within the grid is resolved by DNN. The second-stage network estimates a tuning vector which represents the difference between true DoAs and nearest discrete angles. The final estimated DoA can be obtained by moving initial DoA as much as the difference between the true DoA and the nearest discrete angle. By using this two-stage cascaded NN, the estimation accuracy is no longer limited by the discrete angular grid so that the estimation accuracy can be further enhanced. Simulation results prove that the two-stage cascaded NN can achieve higher estimation accuracy than using a single network by resolving the mismatch induced by the discrete angular grid.

Notations: We use lower case and upper case bold characters to, respectively, represent vectors and matrices throughout this paper. $(\cdot)^T$, $(\cdot)^H$, and $(\cdot)^*$, respectively, denote the transpose, conjugate transpose, and complex conjugation. $\otimes$ denotes Kronecker product. Trace$(\cdot)$ denotes the trace of a matrix. real$(\cdot)$ and imag$(\cdot)$, respectively, denote the real part and imaginary part. $\mathbf{a}(i)$ denotes the i-th element in a vector $\mathbf{a}$, and $\mathbf{A}(i,j)$ denotes the (i,j)-th element in a matrix $\mathbf{A}$. $\|\mathbf{a}\|$ denotes the L2 norm of $\mathbf{a}$. $\mathbf{0}_N$ denotes a $N \times 1$ zero vector, and $\mathbf{I}_N$ denotes a $N \times N$ identity matrix. diag$(\cdot)$ denotes a vector whose entries are diagonal elements of a given matrix. $\mathbb{C}^{M\times N}$, $\mathbb{R}^{M\times N}$, and $\mathbb{R}_+^{M\times N}$, respectively, denote $M \times N$ matrix whose elements are complex, real, and real positive numbers. If $N = 1$, they are $M \times 1$ vectors.

2. Signal Model

We assume that P uncorrelated narrowband signals impinging on the uniform linear array (ULA) with M elements. Here, P narrowband signals have same carrier frequency. The spacing between adjacent antennas is set to half-wavelength $\lambda/2$, where λ denotes the wavelength of the signals. DoAs of P signals are denoted as $\mathbf{\Theta}$, where $\mathbf{\Theta} = [\theta_1, \ldots, \theta_P]^T$. When the narrowband signal impinges on the antenna array, an identical signal arrives at each antenna with a different time delay, where the time delay is dependent on the DoA of the signal. For narrowband signal, the time delay can be translated as a phase shift, and a vector that models the phase shift of each antenna according to DoA is generally known as array manifold vector or steering vector [23]. An array manifold vector whose DoA is θ, $\mathbf{a}(\theta)$ can be given by follows as in [23].

$$\mathbf{a}(\theta) = \left[1, e^{j\pi\cos\theta}, \ldots, e^{j(M-1)\pi\cos\theta}\right]^T \in \mathbb{C}^{M\times 1}. \tag{1}$$

An array manifold matrix for P signal sources, $\mathbf{A}(\mathbf{\Theta})$ is

$$\mathbf{A}(\mathbf{\Theta}) = [\mathbf{a}(\theta_1), \ldots, \mathbf{a}(\theta_P)] \in \mathbb{C}^{M\times P}. \tag{2}$$

The received signal $\mathbf{X}$ can be written as

$$\mathbf{X} = \mathbf{A}(\mathbf{\Theta})\mathbf{S} + \mathbf{N} \in \mathbb{C}^{M\times D}, \tag{3}$$

where D is the number of snapshots. $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_P]^T$, where $\mathbf{s}_p \in \mathbb{C}^{D\times 1}$ denotes the p-th signal vector. $\mathbf{N}$ is an additive white Gaussian noise matrix whose columns follow $\mathcal{CN}(\mathbf{0}_M, \sigma^2\mathbf{I}_M)$, where $\mathbf{0}_M$ is a $M \times 1$ zero vector, $\mathbf{I}_M$ is a $M \times M$ identity matrix, and σ^2 denotes the power of the noise. $\mathbf{R}_\mathbf{X}$, the covariance matrix of $\mathbf{X}$, can be defined as

$$\mathbf{R}_\mathbf{X} = \mathbb{E}\left[\mathbf{X}\mathbf{X}^H\right] = \mathbf{A}(\mathbf{\Theta})\mathbf{R}_\mathbf{S}\mathbf{A}(\mathbf{\Theta})^H + \sigma^2\mathbf{I}_M \approx \frac{\mathbf{X}\mathbf{X}^H}{D} \in \mathbb{C}^{M\times M}, \tag{4}$$

where $\mathbf{R}_\mathbf{S} = \mathbb{E}\left[\mathbf{S}\mathbf{S}^H\right]$. As every signal source is uncorrelated with each other, $\mathbf{R}_\mathbf{S}$ is a diagonal matrix where its p-th diagonal element equals to the power of the p-th signal source. We define $\mathbf{z} \in \mathbb{R}_+^{P\times 1}$ such that $\mathbf{z} = \mathrm{diag}(\mathbf{R}_\mathbf{S})$. The (i,j)-th element of $\mathbf{R}_\mathbf{X}$, $\mathbf{R}_\mathbf{X}(i,j)$ can be given by

$$\mathbf{R}_\mathbf{X} = \sum_{p=1}^{P} \mathbf{z}(p)\mathbf{a}(\theta_p)\mathbf{a}(\theta_p)^H + \sigma^2\mathbf{I}_M, \tag{5}$$

where $\mathbf{z}(p)$ is the p-th element of $\mathbf{z}$, which denotes the power of the p-th signal source and satisfies that $\mathbf{z}(p) = \mathbb{E}\left[\mathbf{s}_p^H\mathbf{s}_p\right]$. A vectorized covariance matrix $\mathbf{y}$ can be given by

$$\mathbf{y} = \sum_{p=1}^{P} \mathbf{z}(p)\mathbf{a}(\theta_p)^* \otimes \mathbf{a}(\theta_p) + \sigma^2\mathbf{i} = \mathcal{A}(\mathbf{\Theta})\mathbf{z} + \sigma^2\mathbf{i} = \mathrm{vec}(\mathbf{R}_\mathbf{X}) \in \mathbb{C}^{M^2\times 1}, \tag{6}$$

where $\mathrm{vec}(\cdot)$ denotes the vectorization of the matrix, $\mathcal{A}(\mathbf{\Theta}) = [\mathbf{a}(\theta_1)^* \otimes \mathbf{a}(\theta_1), \ldots, \mathbf{a}(\theta_P)^* \otimes \mathbf{a}(\theta_P)] \in \mathbb{C}^{M^2\times P}$, and $\mathbf{i} = \mathrm{vec}(\mathbf{I}_M)$.

Following CS-based DoA estimation and a machine learning-based DoA estimation framework [6–8,19,20], $\mathbf{y}$ can be represented as a product of a discretized angular grid and a sparse vector. Here, the discretized angular grid covers the potential DoAs that range between $[0°, 180°]$. The discretized angular grid $\mathbf{\Phi}$ can be given by

$$\mathbf{\Phi} = [\mathbf{a}(\vartheta_1)^* \otimes \mathbf{a}(\vartheta_1), \ldots, \mathbf{a}(\vartheta_G)^* \otimes \mathbf{a}(\vartheta_G)] \in \mathbb{C}^{M^2\times G}, \tag{7}$$

where G denotes the size of the angular grid, and ϑ_g denotes the g-th discrete DoA such that $\vartheta_g = (180g/G)^\circ$. Assuming $\theta_1, \ldots, \theta_P$ corresponds with one of $\vartheta_1, \ldots, \vartheta_G$, $\mathbf{y}$ can be rewritten as

$$\mathbf{y} = \mathbf{\Phi}\boldsymbol{\eta} + \sigma^2\mathbf{i} \in \mathbb{C}^{M^2 \times 1}. \tag{8}$$

$\boldsymbol{\eta} \in \mathbb{R}^{G \times 1}$ denotes the sparse vector, where the majority of the elements of $\boldsymbol{\eta}$ are 0. If $\theta_p = \vartheta_g$, $\boldsymbol{\eta}(g) = \mathbf{z}(p)$.

3. Off-Grid DoA Estimation via Cascaded Convolutional Neural Network

In practice, $\theta_1, \ldots, \theta_P$ may not perfectly correspond to the potential DoAs in $\mathbf{\Phi}$. For example, when $\theta_p = 75.3^\circ$ and $G = 180$, none of the potential DoAs in $\mathbf{\Phi}$ matches perfectly with θ_p. Thus, we propose the DoA estimation algorithm based on the two-stage cascaded NN, which can resolve the mismatch between the true DoAs and the discrete angular grid. In the first stage, the discrete angles that are nearest to the true DoAs are estimated by CNN. In the second stage, the difference from the true DoA and the nearest discrete angle is estimated by DNN. The overall structure of the two-stage cascaded NN is given in Figure 1. Throughout the paper, the term *first-stage network* denotes the network that initially estimates discrete angles, and the term *second-stage network* denotes the network that estimates the difference from the true DoA and the nearest discrete angles.

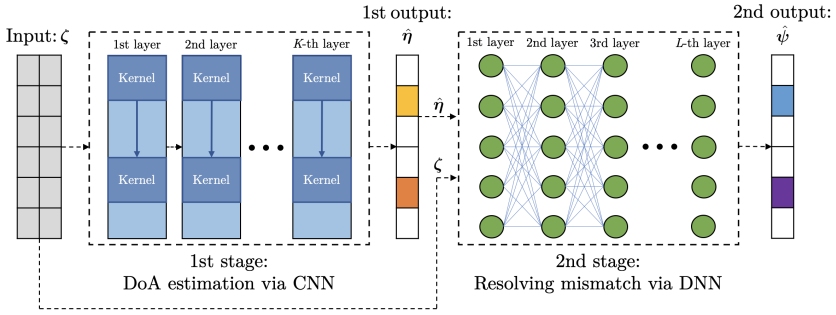


Figure 1. The structure of the two-stage cascaded NN. 1D convolutional layers construct the first-stage network, and dense layers construct the second-stage network.

3.1. First Stage: DoA Estimation via Convolutional Neural Network

The CNN used in the first stage follows the network structure in [20], where 1D convolutional layers in the CNN extract the features from the input and reconstruct $\boldsymbol{\eta}$. As the CNN in [20] does not use pooling, the number of rows of the input should be equal to the size of the output, where the size of the output is G. To form the input whose number of rows equals to G, $\tilde{\boldsymbol{\eta}}$ is formulated as follows.

$$\tilde{\boldsymbol{\eta}} = \mathbf{\Phi}^\dagger \mathbf{y} = \mathbf{\Phi}^H(\mathbf{\Phi}\mathbf{\Phi}^H)\mathbf{y} \in \mathbb{C}^{G \times 1}. \tag{9}$$

As all elements of the input have to be real numbers, the input $\boldsymbol{\zeta}$ is given by

$$\boldsymbol{\zeta} = [\mathrm{real}(\tilde{\boldsymbol{\eta}}), \mathrm{imag}(\tilde{\boldsymbol{\eta}})] \in \mathbb{R}^{G \times 2}. \tag{10}$$

An output of the k-th 1D convolutional layer, $\mathbf{c}_k$, can be represented as in [20]:

$$\mathbf{c}^{(k)} = \mathcal{Z}\left\{ f_{\mathrm{Act}}\left(\mathbf{W}^{(k-1)} * \mathbf{c}^{(k-1)} + \mathbf{b}^{(k-1)} \right) \right\}, \text{for } k = 1, \ldots, K, \tag{11}$$

where $*$ denotes the convolution, $\mathbf{c}^{(0)} = \boldsymbol{\zeta}$, and K denotes the number of convolutional layers. $\mathbf{W}^{(k)}$ and $\mathbf{b}^{(k)}$ denote the convolution kernel and the bias of the k-th convolutional layers. $f^k_{\mathrm{Act}}(\cdot)$ denotes the activation function that the CNN is using. The padding operator $\mathcal{Z}(\cdot)$ adds zeros on both sides of the output of the activation function so that the size of $\mathbf{c}_k$ remains constant for $k = 1, \ldots, K$.

The loss function of the CNN is given by mean squared error (MSE), $\|\hat{\boldsymbol{\eta}} - \boldsymbol{\eta}\|^2$, where $\hat{\boldsymbol{\eta}}$ denotes the estimation of $\boldsymbol{\eta}$. The convolution kernel and the bias that minimize the loss function can be given by follows as in [20].

$$\left\{\hat{\mathbf{W}}^{(k)}, \hat{\mathbf{b}}^{(k)}\right\}_{k=1}^{K} = \underset{\left\{\mathbf{W}^{(k)}, \mathbf{b}^{(k)}\right\}_{k=1}^{K}}{\operatorname{argmin}} \|\hat{\boldsymbol{\eta}} - \boldsymbol{\eta}\|^2, \tag{12}$$

where $\hat{\mathbf{W}}^{(k)}$ and $\hat{\mathbf{b}}^{(k)}$ denotes the convolution kernel and the bias of the k-th convolutional layer that minimize the loss function.

The input data and the output data of the first-stage network can be summarized as follows.

- Input data: $\boldsymbol{\zeta} \in \mathbb{R}^{G \times 2}$
- Output data: $\hat{\boldsymbol{\eta}} \in \mathbb{R}^{G \times 1}$, trained by $\boldsymbol{\eta}$

3.2. Second Stage: Resolving Mismatch via Deep Neural Network

The second-stage network followed by the first-stage network resolves the mismatch by estimating the tuning vector $\boldsymbol{\psi} \in \mathbb{R}^{G \times 1}$. $\boldsymbol{\psi}$ represents the difference between true DoAs and nearest discrete angles. After obtaining $\hat{\boldsymbol{\eta}}$ from the first-stage network, the second-stage network yields $\hat{\boldsymbol{\psi}}$ with $\boldsymbol{\zeta}$ and $\hat{\boldsymbol{\eta}}$, where $\hat{\boldsymbol{\psi}}$ denotes the estimation of $\boldsymbol{\psi}$. Figure 2 explains how the DoAs are estimated when $G = 180$ and $\boldsymbol{\Theta} = [80.4°, 89.8°]^T$. If the estimation is accurate, the 80-th and the 90-th elements of $\hat{\boldsymbol{\eta}}$, respectively, are $\mathbf{z}(1)$ and $\mathbf{z}(2)$, and the 80-th and the 90-th elements of $\hat{\boldsymbol{\psi}}$ are $+0.4°$ and $-0.2°$, respectively. To obtain the final estimated DoAs, the initial DoAs are estimated from the discretized angular grid by finding peaks in $\hat{\boldsymbol{\eta}}$. Then, the initial DoAs are moved as much as the difference between true DoAs and nearest discrete angles by using $\hat{\boldsymbol{\psi}}$.

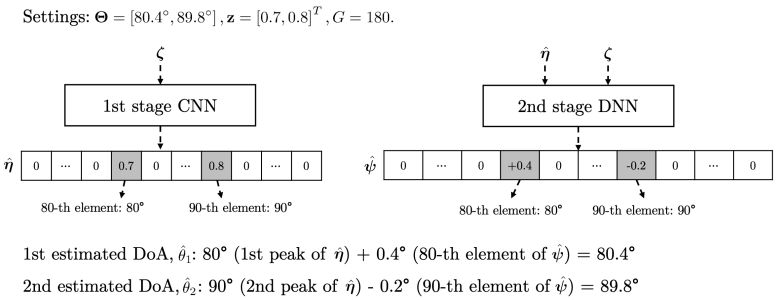


Figure 2. Determination of final estimated DoAs by using the output of the first-stage network $\hat{\boldsymbol{\eta}}$ and the output of the second-stage network $\hat{\boldsymbol{\psi}}$.

As the input of the DNN must be a real vector, the input of the DNN $\boldsymbol{\chi}$ is formulated as

$$\boldsymbol{\chi} = \left[\operatorname{vec}(\boldsymbol{\zeta})^T, \hat{\boldsymbol{\eta}}^T\right]^T = \left[\operatorname{real}(\tilde{\boldsymbol{\eta}})^T, \operatorname{imag}(\tilde{\boldsymbol{\eta}})^T, \hat{\boldsymbol{\eta}}^T\right]^T \in \mathbb{R}^{3G \times 1}. \tag{13}$$

Letting L, $\mathbf{d}^{(l)}$, and $I^{(l)}$, respectively, denote the number of dense layers, the output of the l-th layer, and the size of $\mathbf{d}^{(l)}$, $\mathbf{d}^{(l)}(j)$ can be given by

$$\mathbf{d}^{(l)}(j) = \tanh\left(\sum_{i=1}^{I^{(l-1)}} \mathbf{U}^{(l)}(j,i)\mathbf{d}^{(l-1)}(j) + \mathbf{v}^{(l)}(j)\right), \; l = 1, \ldots, L, \tag{14}$$

where $\mathbf{d}^{(0)} = \boldsymbol{\chi}$. $\mathbf{U}^{(l)}$ and $\mathbf{v}^{(l)}$ denote the weights and the bias of the l-th layers, respectively. In the second stage, a hyperbolic tangent function tanh is employed for the activation function. The reason for selecting tanh as the activation function is that the output vector of the second-stage network must be capable of representing a negative value. For example,

if $G = 180$ so that the angular grid is discretized into units of 1°, elements of $\hat{\psi}$ should be able to range between -0.5° and $+0.5^\circ$. Note that ReLU and sigmoid cannot be employed as the activation function of the second-stage network as ReLU and sigmoid can only yield the value equal to or greater than 0.

The loss function of the DNN is given by MSE, $\|\hat{\boldsymbol{\psi}} - \boldsymbol{\psi}\|^2$. The weights and the bias that minimize the loss function can be denoted as

$$\left\{\hat{\mathbf{U}}^{(l)}, \hat{\mathbf{v}}^{(l)}\right\}_{l=1}^{L} = \underset{\left\{\mathbf{U}^{(l)}, \mathbf{v}^{(l)}\right\}_{l=1}^{L}}{\operatorname{argmin}} \|\hat{\boldsymbol{\psi}} - \boldsymbol{\psi}\|^2, \tag{15}$$

where $\hat{\mathbf{U}}^{(l)}$ and $\hat{\mathbf{v}}^{(l)}$, respectively, denote the weights and the bias of the l-th dense layer that minimize the loss function.

The input data and the output data of the second stage DNN can be summarized as follows.

- Input data: $\boldsymbol{\chi} \in \mathbb{R}^{3G \times 1}$ ($\boldsymbol{\zeta}$ and $\hat{\boldsymbol{\eta}}$)
- Output data: $\hat{\boldsymbol{\psi}} \in \mathbb{R}^{G \times 1}$, trained by $\boldsymbol{\psi}$

4. Simulation Results and Discussions

4.1. Simulation Settings

The performance of the proposed algorithm and L1-SVD [6] are compared in this section. Even though the proposed algorithm is motivated by the work in [22], the comparison with the work in [22] is omitted as the algorithm in [22] exploits the co-prime array [24] and targets for the wideband signal. $M = 8$ and $P = 2$. For all algorithms, a size of the grid G is set to 180 so that a discrete grid spacing is set to 1°. The signal-to-noise ratio (SNR) is defined as

$$\text{SNR} = \frac{\text{Trace}(\mathbf{R_S})}{\sigma^2 M} = \frac{1}{\sigma^2 M} \sum_{p=1}^{P} \mathbf{z}(p). \tag{16}$$

The root mean square error (RMSE) is defined as

$$\text{RMSE} = \sqrt{\frac{1}{PQ} \sum_{q=1}^{Q} \left\{ \sum_{p=1}^{P} \left(\hat{\theta}_p^q - \theta_p^q \right)^2 \right\}}, \tag{17}$$

where Q is the number of Monte Carlo trials for RMSE calculation, and θ_p^q and $\hat{\theta}_p^q$, respectively, denote the true DoA and the estimated DoA of the p-th signal source on the q-th trial.

For training, a total of 106,800 training data are used, and 20 percent of the training data is used for the test. When generating the received signal for the training, the SNR is randomly set between 0 and 10 dB while the number of snapshots is set to 256. DoAs are randomly set between $[30^\circ, 150^\circ]$, and $\mathbf{z} = [1, 1]^T$. Once the DoAs and SNR are determined, the received signal $\mathbf{X}$ is generated as (3), and the covariance matrix $\mathbf{R_X}$ is formulated as (4). Using $\mathbf{R_X}$, the input data of the first-stage network $\boldsymbol{\zeta}$ can be formulated by following the procedure of (6)–(8) and (10). The output data of the first-stage network used for training $\boldsymbol{\eta}$ can be generated by $\mathbf{z}$ and DoAs. If $\boldsymbol{\Theta} = [80.4^\circ, 89.8^\circ]^T$, and $\mathbf{z} = [1, 1]^T$, $\boldsymbol{\eta}$ becomes a sparse vector whose 80-th and 90-th elements are 1. Finally, the first-stage network is trained by 85,440 different sets of $\boldsymbol{\zeta}$ and $\boldsymbol{\eta}$. The input data of the second-stage network $\boldsymbol{\chi}$ are generated as (13), where $\boldsymbol{\chi}$ is a combination of input and output of the first-stage network. The output data of the second-stage network used for training $\boldsymbol{\psi}$ can be generated by DoAs. If $\boldsymbol{\Theta} = [80.4^\circ, 89.8^\circ]^T$, $\boldsymbol{\psi}$ becomes a sparse vector whose 80-th and 90-th elements are, respectively, $+0.4^\circ$ and -0.2°. The second-stage network is trained by 85,440 different sets of $\boldsymbol{\chi}$ and $\boldsymbol{\psi}$.

In the first-stage network, the DNN is implemented as well as the CNN to compare the performance according to the type of the network. When the DNN is used for the first

stage, the input $\boldsymbol{\zeta} \in G \times 2$ is vectorized as the DNN can only support vector for both input and output. The output is identical to $\hat{\boldsymbol{\eta}}$ in both CNN- and DNN-based first-stage networks. Regardless of the type of network that the first stage is using, three types of activation functions—PReLU, ReLU, and tanh—are used, and the number of convolutional layers and dense layers in the first-stage network is set to 4. If the CNN is used at the first stage, the kernel size of each layer is set to 25×12, 15×6, 5×3, and 3×1. If the DNN is used at the first stage, the number of nodes in every layer is set to 200. The number of dense layers in the second-stage network, L equals 4. Here, the number of nodes in every layer is set to 200. Number of epochs of the first-stage network and the second-stage network are, respectively, set to 300 and 500, and the batch size is uniformly set to 64. The overall training settings are organized in Table 1.

Table 1. Simulation settings for training.

	First Stage		**Second Stage**
	CNN	**DNN**	**DNN**
Activation function	PReLU, ReLU, tanh		tanh
Number of training data	106,800 (85,440 for training and 21,360 for test)		
Number of layers (K, L)	4		
Kernel size	25×12, 15×6, 5×3, 3×1	-	
Number of nodes	-	200	
Batch size	64		
The number of epochs	300		500

4.2. Performance Analysis According to Hyperparameters

The performance of the NN varies with hyperparameters, where hyperparameters include the number of epochs and the type of the activation function. The performance of the first-stage network is compared with respect to the number of epochs and the type of the activation function in Figure 3. Figure 3 shows that the test MSE is lower in the order of CNN with PReLU, DNN with PReLU, CNN with tanh, DNN with tanh, and DNN with ReLU. PReLU has the low test MSE while tanh has the high test MSE, where the test MSE means the loss function obtained by test data. Overall, the test MSE tends to converge around the 200-th epoch, except the DNN with tanh which converges around the 700-th epoch. Moreover, CNN has a lower test MSE than DNN if an identical activation function is used. From Figure 3, we can conclude that CNN whose activation function is PReLU is best suited for the DoA estimation at the first stage. Note that the test MSE of the second-stage network according to hyperparameters is omitted as the activation function of the second-stage network is fixed to tanh. In case of the second-stage network, the test MSE converges around the 500-th epoch.

Figure 4 shows outputs of the first-stage network and the second-stage network according to the type of NN and the activation function. Note that only results of CNN with PReLU, CNN with ReLU, and DNN with ReLU are presented as the others have higher test MSE, and figures in the second row show the true DoAs and the final estimated DoAs determined by two-stage cascaded NN. In Figure 4, CNN with PReLU yields distinct peaks around true DoAs at the first stage and achieves the highest estimation accuracy with the help of the second-stage network. However, DNN with PReLU yields obscure peaks at the first stage, and the output of the second stage shows poor performance when using DNN with PReLU. From Figures 3 and 4, we can tell that CNN whose activation function is PReLU is the best option, and the output of the first-stage network affects the output of the second-stage network.

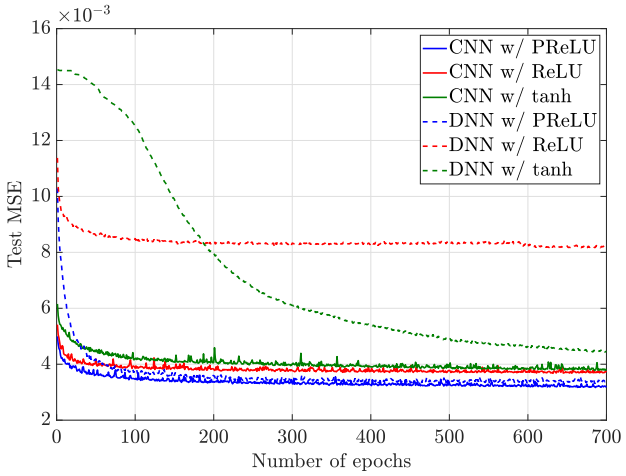


Figure 3. The test mean squared error (MSE) over epochs. The convolutional neural network (CNN)-based first-stage network and the deep neural network (DNN)-based first-stage network are compared, and three types of activation functions—PReLU, ReLU, and tanh—are used.

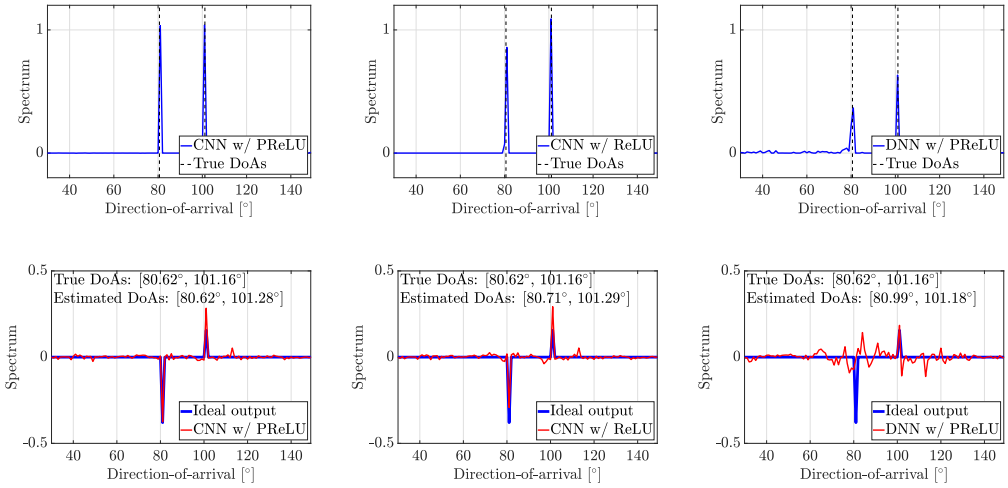


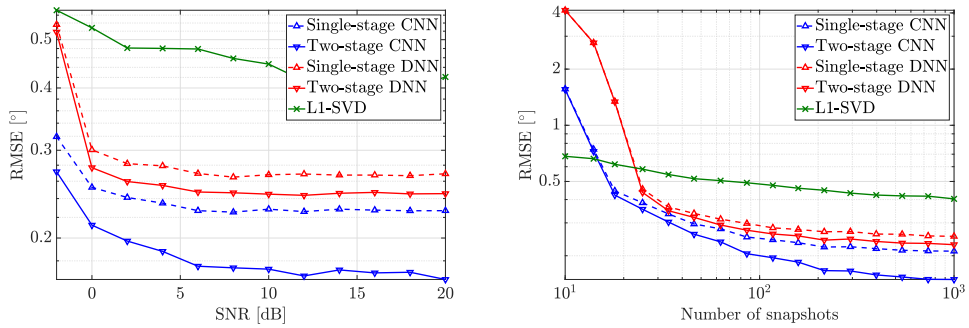
Figure 4. Outputs of the first-stage network and the second-stage network according to the type of NN and the activation function. The first row presents outputs of the first-stage network, and the second row presents outputs of the second-stage network.

4.3. Analysis on Estimation Accuracy and Resolution

Before presenting simulation results, an explanation of the terms that are used in figures and analysis is given to avoid unclear understanding. The term *single-stage* denotes the DoA estimation without the second-stage network, and the term *two-stage* means that the second-stage network is added after the first-stage network. A type of the first-stage network is described after the number of stage that the algorithm is using. Here, the type of the first-stage network is either CNN or DNN. For example, *two-stage CNN* means that there is the second-stage network followed by the first-stage network and the first-stage network is CNN. Since Figure 3 shows that PReLU outperforms other types of activation functions, the activation function of the first-stage network is fixed to PReLU.

Figure 5 shows the root mean square error (RMSE) of DoA estimation algorithms when the SNR and the number of snapshots vary. Figure 5 shows that the proposed two-stage CNN presents the best estimation accuracy among other algorithms, except when fewer snapshots are used. When training the network, the number of snapshots is fixed to 256 so that the condition with the fewer snapshots is not considered. Thus, the RMSE of L1-SVD becomes lower than that of two-stage CNN and single-stage CNN when the number of snapshots is below 18. In Figure 5a,b, the RMSE of single-stage CNN,

single-stage DNN, and L1-SVD does not fall below 0.2°. When the second-stage network is added after CNN, the RMSE falls below 0.2° by resolving the mismatch between true DoAs and discrete angular grid. However, when DoAs are incorrectly estimated at the first stage, the second-stage network does not improve the RMSE as the incorrect output of the first-stage network affects the performance of the second-stage network.



(**a**) (**b**)

Figure 5. (**a**) The RMSE for different SNR. $Q = 1000$ and $D = 256$. (**b**) The RMSE for different number of snapshots. $Q = 1000$ and the SNR is set to 10 dB.

To analyze the resolution of DoA estimation algorithms, we present the RMSE when the difference between two adjacent DoAs varies. We define a distance between two adjacent DoAs τ as $\tau = |(\cos\theta_1 - \cos\theta_2)/2|$, where τ is used as a criterion of resolution in [25–27]. Figure 6 shows the RMSE versus τ and presents that the second-stage network helps improve RMSE. The RMSE of the single-stage CNN and the two-stage CNN converges when $\tau = 0.02$, and the RMSE of the single-stage DNN and the two-stage DNN converges when $\tau = 0.03$. The RMSE of L1-SVD is higher than that of the CNN-based DoA estimation and the DNN-based DoA estimation at every τ. From this result, we can conclude that the CNN-based DoA estimation has the highest resolution, and the second-stage network helps to increase the estimation accuracy furthermore.

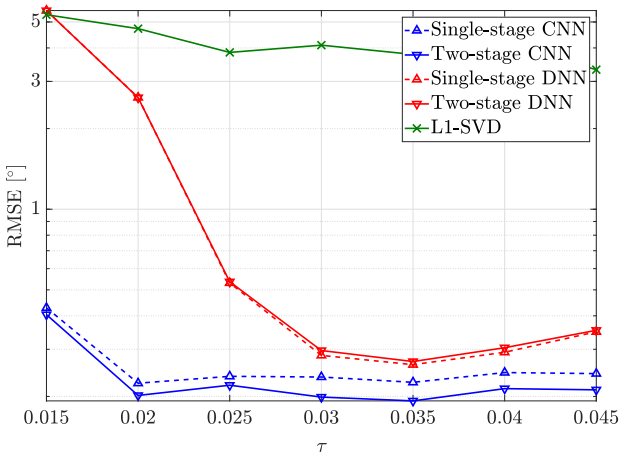


Figure 6. The RMSE versus τ. $Q = 1000$ and the SNR is set to 10 dB. $|\theta_1 - \theta_2|$ is about 1.7° around 90° when $\tau = 0.015$ and is about 5.2° around 90° when $\tau = 0.045$.

5. Conclusions

In this paper, we propose the off-grid DoA estimation algorithm which can resolve mismatch induced by discretized angular grid via two-stage cascaded NN. The first-stage network employs CNN, and a resized covariance matrix to fit the output dimension of CNN is used as an input. The first-stage network estimates initial DoAs that can be represented as discrete angles. Thus, there is a mismatch between true DoAs and initially

estimated DoAs. The second-stage network employs DNN, and both input and output of the first-stage network are used as input. The output of the second-stage network is the tuning vector which represents the difference between true DoAs and nearest discrete angles. By using outputs of the first-stage network and the second-stage network, final estimated DoAs can be obtained. Simulation results show that using the cascaded NN improves the RMSE by resolving mismatch induced by the discretized grid. Furthermore, results show that using CNN and using PReLU as the activation function is the best option for DoA estimation considering both estimation accuracy and resolution. The proposed algorithm can be an attractive option for applications such as far-field localization and far-field radar. These applications require high estimation accuracy since a small DoA estimation error may cause large performance degradation. However, the performance of the proposed algorithm is analyzed only when the number of signal sources is fixed. In practice, the number of signal sources can vary so that the performance of the proposed algorithm may decrease. Thus, a method that addresses this issue needs to be studied for the practical implementation of the NN-based DoA estimation algorithm.

Author Contributions: Conceptualization, H.C.; Methodology, H.C. and H.S.; Software, H.C. and H.S.; Validation, H.C. H.S., and S.K.; Formal analysis, H.C.; Investigation, H.C. and H.S.; Resources, H.C., H.S., J.J., D.L., and S.K.; Data curation, H.C. and H.S.; Writing—original draft preparation, H.C. and S.K.; Visualization, H.C.; Supervision, S.K.; Project administration, J.J., D.L., and S.K.; Funding acquisition, S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Agency for Defense Development.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chen, J.C.; Yao, K.; Hudson, R.E. Source localization and beamforming. *IEEE Signal Process. Mag.* **2002**, *19*, 30–39. [CrossRef]
2. Wymeersch, H.; Seco-Granados, G.; Destino, G.; Dardari, D.; Tufvesson, F. 5G mmWave positioning for vehicular networks. *IEEE Wirel. Commun.* **2017**, *24*, 80–86. [CrossRef]
3. Eom, J.; Kim, H.; Lee, S.H.; Kim, S. DNN-assisted cooperative localization in vehicular networks. *Energies* **2019**, *12*, 2758. [CrossRef]
4. Hasch, J.; Topak, E.; Schnabel, R.; Zwick, T.; Weigel, R.; Waldschmidt, C. Millimeter-wave technology for automotive radar sensors in the 77 GHz frequency band. *IEEE Trans. Microw. Theory Tech.* **2012**, *60*, 845–860. [CrossRef]
5. Schmidt, R. Multiple emitter location and signal parameter estimation. *IEEE Trans. Antennas Propag.* **1986**, *34*, 276–280. [CrossRef]
6. Malioutov, D.; Cetin, M.; Willsky, A.S. A sparse signal reconstruction perspective for source localization with sensor arrays. *IEEE Trans. Signal Process.* **2005**, *53*, 3010–3022. [CrossRef]
7. Gorodnitsky, I.F.; Rao, B.D. Sparse signal reconstruction from limited data using FOCUSS: A re-weighted minimum norm algorithm. *IEEE Trans. Signal Process.* **1997**, *45*, 600–616. [CrossRef]
8. Yang, Z.; Xie, L.; Zhang, C. Off-grid direction of arrival estimation using sparse Bayesian inference. *IEEE Trans. Signal Process.* **2013**, *61*, 38–43. [CrossRef]
9. Chen, S.S.; Donoho, D.L.; Saunders, M.A. Atomic decomposition by basis pursuit. *SIAM J. Sci. Comput.* **1999**, *20*, 33–61. [CrossRef]
10. Wipf, D.P.; Rao, B.D. Sparse Bayesian learning for basis selection. *IEEE Trans. Signal Process.* **2004**, *52*, 2153–2164. [CrossRef]
11. Shen, Q.; Liu, W.; Cui, W.; Wu, S. Underdetermined DOA estimation under the compressive sensing framework: A review. *IEEE Access* **2016**, *4*, 8865–8878. [CrossRef]
12. Chi, Y.; Scharf, L.L.; Pezeshki, A.; Calderbank, A.R. Sensitivity to basis mismatch in compressed sensing. *IEEE Trans. Signal Process.* **2011**, *59*, 2182–2195. [CrossRef]
13. Pastorino, M.; Randazzo, A. A smart antenna system for direction of arrival estimation based on a support vector regression. *IEEE Trans. Antennas Propag.* **2005**, *53*, 2161–2168. [CrossRef]
14. Randazzo, A.; Abou-Khousa, M.A.; Pastorino, M.; Zoughi, R. Direction of arrival estimation based on support vector regression: Experimental validation and comparison with MUSIC. *IEEE Antennas Wirel. Propag. Lett.* **2007**, *6*, 379–382. [CrossRef]
15. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
16. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef] [PubMed]
17. Huang, H.; Yang, J.; Huang, H.; Song, Y.; Gui, G. Deep learning for super-resolution channel estimation and DOA estimation based massive MIMO system. *IEEE Trans. Veh. Technol.* **2018**, *67*, 8549–8560. [CrossRef]

18. Guo, Y.; Zhang, Z.; Huang, Y.; Zhang, P. DOA estimation method based on cascaded neural network for two closely spaced sources. *IEEE Signal Process. Lett.* **2020**, *27*, 570–574. [CrossRef]
19. Liu, Z.; Zhang, C.; Yu, P.S. Direction-of-arrival estimation based on deep neural networks with robustness to array imperfections. *IEEE Trans. Antennas Propag.* **2018**, *66*, 7315–7327. [CrossRef]
20. Wu, L.; Liu, Z.; Huang, Z. Deep convolution network for direction of arrival estimation with sparse prior. *IEEE Signal Process. Lett.* **2019**, *26*, 1688–1692. [CrossRef]
21. Xiang, H.; Chen, B.; Yang, T.; Liu, D. Improved de-multipath neural network models with self-paced feature-to-feature learning for DOA estimation in multipath environment. *IEEE Trans. Veh. Technol.* **2020**, *69*, 5068–5078. [CrossRef]
22. Shen, Q.; Cui, W.; Liu, W.; Wu, S.; Zhang, Y.D.; Amin, M.G. Underdetermined wideband DOA estimation of off-grid sources employing the difference co-array concept. *Signal Process.* **2017**, *130*, 299–304. [CrossRef]
23. Krim, H.; Viberg, M. Two decades of array signal processing research: The parametric approach. *IEEE Signal Process. Mag.* **1996**, *13*, 67–94. [CrossRef]
24. Vaidyanathan, P.P.; Pal, P. Sparse sensing with co-prime samplers and arrays. *IEEE Trans. Signal Process.* **2011**, *59*, 573–586. [CrossRef]
25. Candès, E.; Fernandez-Granda, C. Towards a mathematical theory of super-resolution. *Commun. Pure Appl. Math.* **2014**, *67*. [CrossRef]
26. Tang, G.; Bhaskar, B.N.; Shah, P.; Recht, B. Compressed sensing off the grid. *IEEE Trans. Inf. Theory* **2013**, *59*, 7465–7490. [CrossRef]
27. Chi, Y.; Ferreira Da Costa, M. Harnessing sparsity over the continuum: Atomic norm minimization for superresolution. *IEEE Signal Process. Mag.* **2020**, *37*, 39–57. [CrossRef]