IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 16, 2022, accepted March 26, 2022, date of publication April 5, 2022, date of current version May 20, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3164897

Robust DoA Estimation Using Denoising Autoencoder and Deep Neural Networks

DAWEI CHEN[1], (Student Member, IEEE), SHUO SHI[1], (Member, IEEE), XUEMAI GU[1], (Member, IEEE), AND BYONGHYO SHIM[2,3], (Senior Member, IEEE)

[1]School of Electronics and Information Engineering, Harbin Institute of Technology, Harbin 150006, China
[2]School of Electrical and Computer Engineering, Seoul National University, Seoul 08826, South Korea
[3]Institute of New Media and Communications, Seoul National University, Seoul 08826, South Korea

Corresponding author: Xuemai Gu (guxuemai@hit.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 62171158, in part by the National Research Foundation (NRF) funded by the Korean Government through the Ministry of Science, ICT and Future Planning (MSIP) under Grant 2020R1A2C2102198, and in part by the Ministry of Science and ICT (MSIT) through the Information Technology Research Center (ITRC) Program supervised by the Institute for Information & Communications Technology Promotion (IITP) under Grant IITP-2019-2017-0-01637.

ABSTRACT As one of the most critical technology in array signal processing, direction of arrival (DoA) estimation has received a great deal of attention in many areas. Traditional methods perform well when the signal-to-noise ratio (SNR) is high and the receiving array is perfect, which are quite different from the situation in some real applications (e.g., the marine communication scenario). To get satisfying performance of DoA estimation when SNR is low and the array is inaccurate (mutual coupling exist), this paper introduces a scheme consisting of denoising autoencoder (DAE) and deep neural networks (DNN), referred to as DAE-DNN scheme. DAE is used to reconstruct a clean ''repaired'' input from its corrupted version to increase the robustness, and then divide the input into multiple parts in different sub-areas. DNN is used to learn the mapping between the received signals and the refined grids of angle in each sub-areas, then the outputs of each sub-areas are concatenated to perform the final DoA estimation. By simulations in different SNR regimes, we study the performance of DAE-DNN in terms of the different snapshots, batch size, learning rate, and epoch. Our results demonstrate that the proposed DAE-DNN scheme outperforms traditional methods in accuracy and robustness.

INDEX TERMS DoA, SNR, denoising autoencoder, deep neural networks, mutual coupling.

I. INTRODUCTION

Direction of arrival (DoA) estimation is long-standing yet still one of the most important problems in array signal processing. DoA estimation has a variety of applications such as mobile communications [1], [2], vehicular communications [3], airborne radar recognition [4], source localization [5], unmanned aerial vehicle [6], and sonar navigation [7], to name just a few. Recently, as the demand for high-quality DoA estimation increases, the high accuracy DoA estimation technique has received much attention. To meet the demand for both civil and military use in target detection, various DoA estimation techniques have been proposed [8]–[11]. Roughly speaking, these approaches fall into three categories. The first category is the spectral estimation technique where the DoA is estimated by the numerical search. Popular approaches in this class include Capon and Bartlett methods [8]. However, it is challenging to achieve high angular resolution using this approach since we need to compute the spatial spectrum and estimate DoA by numerical search of local maxima of the spectrum. The second category is the subspace-based technique such as MUltiple SIgnal Classification (MUSIC) [9] and Estimation Of Signal Parameter via Rotational Invariance Techniques (ESPRIT) [10], where the DoA estimation problem is described by the matrix model. Due to the dependence on array characteristics and computational burden of the eigenvalue decomposition (EVD) and singular value decomposition, this approach is not so effective in coping with the inaccuracies of receiving array. The third category is the probabilistic model-based technique, such as the Maximum-Likelihood (ML) [11] and maximum a posterior estimation [12]. This approach is

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

effective in handling the deterministic signal but it is in general difficult to obtain a prior information on the signal model, especially for low-elevation targets. The more prior information on the target signal model or noise distribution we have, the better performance one can achieve. Also, when the number of sources is large and the range of angles is wide (i.e., when the search grid is large), the computational complexity of ML estimation grows exponentially with the number of sources.

In recent years, various methods have been proposed to solve these problems, however, the advantages are limited, especially in low SNR regimes. Considering the mutual coupling matrix of the uniform linear array (ULA) modeled as a banded symmetric Toeplitz matrix, MUSIC-Like and ESPRIT-Like algorithms are proposed to estimate DoA [13], [14]. However, these methods lead to the array aperture loss. In [15], a new DoA estimation algorithm based on the orthogonality of a specific eigenvector has been proposed, but it needs the array response to be quiescent. In [16], a reweighted regularized sparse recovery algorithm has been proposed for the DoA estimation with the unknown mutual coupling of the ULA, assuming the signal number is known in high SNR (around 10dB). In [17], an estimation method based on the spatial spectrum with a fixed eigenvalue order has been proposed to estimate DoA under the circumstance of high SNR (above 0dB), without considering the mutual coupling and the number of sources. For these reasons mentioned, when the number of signals is unknown, the receiving array is inaccurate and the SNR is low, conventional approaches are not so effective and comprehensive, in particular for super-resolution DoA estimation.

In this paper, we propose an entirely new DoA estimation technique based on deep learning to obtain the high-resolution DoA estimation in the complex environment, such as low SNR regime, an unknown number of sources, inaccurate array (mutual coupling), or a combination of these problems. For this purpose, two objectives have been identified. The first objective is to reduce the noise of noisy input signals. Literature [18] has shown that denoising autoencoders (DAEs) can achieve higher detection accuracy than the basic autoencoders. Specifically, DAE [19] can learn more robust nonlinear representation from signal against noise and fluctuation, has powerful generalization capability, and produce the state-of-the-art performance on many challenging feature learning tasks, such as wireless sensor networks [20], autonomous fault detection and analysis [21]–[23], unmanned aerial vehicle networks [24], biomedical science [25], etc. Motivated by these successful applications and the excellent properties of the DAE, we adopted DAE to deal with target signals with noisy data. Our goal is to provide a robust signal reconstruction in the case of perturbations or disturbances presented in the sensor data while capturing nonlinear correlations embedded in multiple signals. The other objective is to capture DoA features of the inaccurate array when the number of sources is unknown. Although machine learning has shown a great potential in DoA estimation, such as radial basis function (RBF) [26], which can reduce the computation complexity and perform well in high-resolution, yet its performance degrades rapidly when the number of sources is unknown and the array is inaccurate. Recently, as a key enabler for future wireless communications [27]–[29], deep neural networks (DNN) has been successfully applied to various wireless systems such as multiple-input and multiple-output (MIMO) [30], wireless scheduling [31], and active user detection [32]. In these works, DNN is used to learn a desired nonlinear function (e.g., classification and decision) through the training process. Benefited from the powerful representation of the mapping relationship between high-dimensional random vector elements, DNN has shown new ways of obtaining useful feature representation that provides better performance than those traditional feature extractors. Here, DNN is used to capture the abstractions of DoA features, obtain distributed representations and further improve the generalization of the whole system when the array is inaccurate and the number of sources is unknown.

In a nutshell, we exploit DAE to increase the robustness at lower SNR, and DNN to improve detection accuracy when the number of sources is unknown and the receiving array is inaccurate. To be specific, the proposed DoA estimation scheme in our framework, henceforth referred to as the DAE-DNN, DAE extracts useful features from the corrupted received signal and learns the more robust mapping between the received signal and each sub-areas. In each sub-areas, DNN further learns the complicated mapping between the received signals with different inter-signal angles and the refined grids of angle.

Two key ingredients of the DAE-DNN technique accomplishing this mission are 1) DAE dividing the angle range into several sub-areas and 2) DNN to get a more refined estimation of DoA in each sub-areas. In the DAE stage, we extract the main features of data from the corrupted input data by encoder and decoder. After this, DNN is applied to refine the DoA estimation result in each sub-areas. According to the *universal approximation theorem*, DNN processed by the deeply stacked hidden layers well approximate the desired function [33]. In our context, this means that the properly designed DAE-DNN system with multiple hidden layers can handle the whole DoA estimation process, resulting in an accurate DoA estimation.

The main contributions of this paper are as follows:

1) **We propose a DoA estimation system composed of DAE algorithm and DNN framework.** Rather than transferring the original covariance matrix of the received data to the frequency domain to get the input data, the DAE algorithm processes the DoA sensor array output in the time domain directly.
2) **We obtain the high-resolution of DoA estimation in very noisy and reverberant environments with array inaccuracies.** After the denoising of the corrupted data at the receiver, a high-resolution DoA estimation is performed by training procedures based on

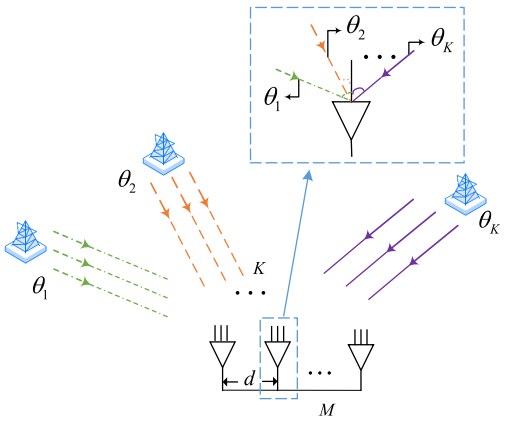

FIGURE 1. Framework of DoAs estimation of ULA.

one-versus-all classification, which decides the existence of a signal near the refined grids of angle.

3) **We provide a performance analysis of the proposed system under different parameters and cases.** By simulations, we identify the optimal snapshot, batch size, learning rate, and epoch of the DAE-DNN system. We also examine the root mean square error (RMSE) of the DoA in low SNR regime for assessing the accuracy of DoA estimation. From extensive simulations empirical evaluations test, we demonstrate the efficiency and robustness of the proposed DAE-DNN.

The rest of this paper is organized as follows. In Section II, we present a DoA model with multiple signal sources, DAE model and DNN model. In Section III, we discuss the DAE-DNN estimation system. We describe the architecture of DAE-DNN system and its corresponding training strategies in detail in Section IV. We provide the performance analysis of DAE-DNN, including snapshot, batch size, learning rate, epoch, and the comparison to traditional methods in Section V. Finally, we conclude the paper in Section VI.

Notations: throughout the paper, matrices are noted by bold capital letters, while vectors and scalers are denoted by boldface small letters and small letters, respectively. $(\cdot)^*$, $(\cdot)^T$, $(\cdot)^H$, and $\|\cdot\|_2$ represent conjugate, transpose, conjugate transpose and l_2 norm operator of a matrix, respectively. Also, $\boldsymbol{E}[\cdot]$, $\mathbb{H}[\cdot]$ are given as the expectation and entropy operator, $re[\cdot]$ and $im[\cdot]$ represent the real and imaginary parts of a complex-valued entity, respectively.

II. PROBLEM FORMULATION

A. DoA MODEL WITH MULTIPLE SIGNAL SOURCES

We consider a ULA with M antenna elements where the element spacing is $d = \lambda/2$ (λ is the wavelength). We assume that the plane waves impinging on the receiving array are parallel since the sources are located in the far-field areas. We also assume that there is no correlation between sources. Let θ_k be the kth angle of arrival in the K-independent signal impinging on the M-element array from direction $\{\theta_1, \theta_2, \cdots, \theta_K\}$ (see Fig. 1). The medium through which the wave propagates is assumed to be homogenous, isotropic, and non-dispersive.

The steering vector $\boldsymbol{a}(\theta_k)$ can be expressed as

$$\boldsymbol{a}(\theta_k) = \left[1, e^{-j2\pi d \sin\theta_k/\lambda}, e^{-j2\pi d\cdot 2\sin\theta_k/\lambda}, \cdots, e^{-j2\pi d\cdot(M-1)\sin\theta_k/\lambda}\right]^T, k = 1, 2, \cdots, K \quad (1)$$

The array inaccuracies will cause deviations in the mapping from θ_k to $\boldsymbol{a}(\theta_k)$. In this paper, $\boldsymbol{a}(\theta_k)$ and its perturbed variant $\boldsymbol{a}(\theta_k, \boldsymbol{e})$ are assumed to be unitary vectors (i.e., $\|\boldsymbol{a}(\theta_k)\|_2 = \|\boldsymbol{a}(\theta_k, \boldsymbol{e})\|_2 = 1$).

When sampled with equally spaced interval at $\{t_1, t_2, \cdots, t_I\}$ (I is the snapshot), the received signal $\boldsymbol{X}(t) = [\boldsymbol{x}(t_1), \ldots, \boldsymbol{x}(t_I)] \in \mathbb{C}^{M\times I}$ can be formulated as

$$\boldsymbol{X}(t) = \boldsymbol{A}(\theta)\boldsymbol{S}(t) + \boldsymbol{N}(t) \quad (2)$$

where $\boldsymbol{A}(\theta) \in \mathbb{C}^{M\times K}$ is the *steering matrix* of the array, $\boldsymbol{S}(t) \in \mathbb{C}^{K\times I}$ is the incident signal waveforms and $\boldsymbol{N}(t) \in \mathbb{C}^{M\times I}$ is the zero-mean Gaussian noise, which are given, respectively, by

$$\begin{aligned} \boldsymbol{A}(\theta) &= [\boldsymbol{a}(\theta_1, \boldsymbol{e}), \boldsymbol{a}(\theta_2, \boldsymbol{e}), \cdots, \boldsymbol{a}(\theta_K, \boldsymbol{e})] \\ \boldsymbol{S}(t) &= [\boldsymbol{s}_1(t), \boldsymbol{s}_2(t), \cdots, \boldsymbol{s}_K(t)]^T \\ \boldsymbol{N}(t) &= [\boldsymbol{n}_1(t), \boldsymbol{n}_2(t), \cdots, \boldsymbol{n}_M(t)]^T \end{aligned} \quad (3)$$

Note that the elements of $\boldsymbol{X}(t)$ can be expressed as

$$\boldsymbol{x}(t_i) = \sum_{k=1}^{K} \boldsymbol{a}(\theta_k, \boldsymbol{e})\boldsymbol{s}_k(t_i) + \boldsymbol{n}(t_i), \text{ for } i = 1, \ldots, I \quad (4)$$

The more information will be obtained with the increasing of I. The covariance matrix $\boldsymbol{R}_x(t)$ of the received signals is

$$\boldsymbol{R}_x(t) = \boldsymbol{E}[\boldsymbol{X}(t)\boldsymbol{X}(t)^H] = \boldsymbol{A}\boldsymbol{R}_{\boldsymbol{S}}\boldsymbol{A}^H + \sigma^2 \cdot \boldsymbol{I} \quad (5)$$

where $\boldsymbol{R}_{\boldsymbol{S}}$ is the covariance matrix of signals $\boldsymbol{S}(t)$, σ^2 is the variance of noise, and $\boldsymbol{I}$ is an identity matrix.

B. DENOISING AUTOENCODER AND DEEP NEURAL NETWORKS

1) DENOISING AUTOENCODER

In contrast to the conventional autoencoder used primarily in a mid or high SNR regime [34], DAE is used to recover an input signal from a corrupted version, and we can obtain a more robust representation of the input. Two ideas explain this approach. Firstly, as a higher representation of autoencoder, DAE is stable and robust when the input is corrupted; Secondly, in the denoising phase, we expected that more useful features could be extracted from the input distribution.

Formally, the initial input $\boldsymbol{r}$ is corrupted by Gaussian Noise and corresponding corrupted version is denoted as $\boldsymbol{r_0}$, which is generated according to a stochastic mapping $\boldsymbol{r}_0 \sim \boldsymbol{q}_{\mathcal{D}}(\boldsymbol{r}_0|\boldsymbol{r})$. Then the corrupted input $\boldsymbol{r_0}$ is mapped to the hidden representation $\boldsymbol{d}_0$ via encoder f_ϑ. To get the reconstructed value $\boldsymbol{d}$, we use the decoder $g_{\vartheta'}$ given by

$$\boldsymbol{d}_0 = f_\vartheta(\boldsymbol{r_0}) = s(\boldsymbol{w}\boldsymbol{r_0} + \boldsymbol{b}) \quad (6)$$

$$\boldsymbol{d} = g_{\vartheta'}(\boldsymbol{d}_0) = s\left(\boldsymbol{w}'\boldsymbol{d}_0 + \boldsymbol{b}'\right) \quad (7)$$

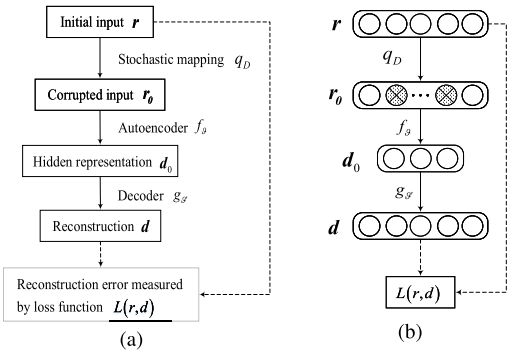

FIGURE 2. The framework of DAE: (a) flow chart of DAE, (b) neural network representation of DAE.

where $\boldsymbol{w}$, $\boldsymbol{w}'$ are weight matrices and $\boldsymbol{b}$, $\boldsymbol{b}'$ are offset vectors in parameter sets $\vartheta = \{\boldsymbol{w}, \boldsymbol{b}\}$ and $\vartheta' = \{\boldsymbol{w}', \boldsymbol{b}'\}$, $f(\cdot)$ and $g(\cdot)$ are a nonlinear activation function of the encoder and the decoder. Reconstruction error is measured by the loss function $L(\boldsymbol{r}, \boldsymbol{d})$. Well-known loss functions include the squared error loss $L_2(\boldsymbol{r}, \boldsymbol{d}) = \|\boldsymbol{r} - \boldsymbol{d}\|^2$ and the cross-entropy loss $L_H(\boldsymbol{r}, \boldsymbol{d}) = -\sum_j \left[r_j \log d_j + (1 - r_j) \log (1 - d_j)\right]$. The framework of denoising autoencoder is shown in Fig. 2.

In the training phase of Fig. 2, the parameter sets ϑ and ϑ' are trained to minimize $L(\boldsymbol{r}, \boldsymbol{d})$ over a training set, that is, to have $\boldsymbol{d}$ as close as possible to the uncorrupted input $\boldsymbol{r}$. Then the reconstructed value $\boldsymbol{d}$ is now obtained by applying deterministic mapping f_ϑ to a corrupted input $\boldsymbol{r_0}$ instead of initial input $\boldsymbol{r}$. Parameters are initialized at random and then optimized by stochastic gradient descent. Note that each time a training example $\boldsymbol{r}$ is presented, a different corrupted version $\boldsymbol{r_0}$ of it is generated according to $\boldsymbol{r}_0 \sim \boldsymbol{q}_{\mathcal{D}}(\boldsymbol{r}_0|\boldsymbol{r})$. Therefore, the mapping forces DAE to learn a far more robust mapping instead of identity: a mapping that extracts useful features for denoising [35].

2) DEEP NEURAL NETWORKS

We consider the basic model of DNN, a linear input-output model given by

$$f_{dnn}(\boldsymbol{x}, \boldsymbol{w}) = \sum_{\hat{i}=1}^{n} x_{\hat{i}} w_{\hat{i}} \tag{8}$$

where $\boldsymbol{x}$ and $\boldsymbol{w}$ represent the inputs and weights of the neural networks, respectively, and n is the number of input. Note that $f_{dnn}(\boldsymbol{x}, \boldsymbol{\omega})$ used in the classification depends on the sign of its corresponding value.

In the deep learning field, DNN is considered as one of the most promising generative models since it can deal with many non-convex and non-linear mappings and also learn the characteristics of data in a high-dimensional space. In the data processing of DNN, there are multiple neurons in each hidden layer and we can obtain an output with a weighted sum of these neurons operated by a nonlinear function. The activation function is used in the process of DNN to realize recognition and representation operation. Generally, we usually choose the *Sigmoid* function and the *ReLU* function in the nonlinear operation, which are defined as

$$f_{sigmoid}(x) = \frac{1}{1 + \exp(-x)} \tag{9}$$

$$f_{ReLU}(x) = \max(0, x) \tag{10}$$

III. DEEP NEURAL NETWORKS FRAMEWORK FOR DoA ESTIMATION

In this section, we present the DAE-DNN architecture. In contrast to the previous works focusing on the high SNR scenario, the proposed DAE-DNN is designed to handle the DoA estimation in the low SNR regime. Benefited from multiple types of autoencoder, DAE is robust to the noise like Gaussian Noise and Salt and Pepper Noise [35]. As shown in Fig. 3, the proposed DAE-DNN consists of three parts. The first part in the green rectangle above is the data preprocessing phase of the arrival angle θ of impinging plane waves, where the covariance matrix $\boldsymbol{R}_{rr}(t)$ of signals is computed from the steering vector $\boldsymbol{a}(\theta_k)$. The second part is the DAE phase shown in the red rectangle middle. In this phase, DAE denoises the input data and divide the whole range of θ into J spatial sub-areas. The last part in the blue rectangle below is multi-layer classifiers, and they determine which sub-area the impinging signal is located.

A. DENOISING AUTOENCODER

Main purpose of DAE is to reconstruct a clean repaired input from a corrupted version of train data. In the encoding phase, useful features in a better higher-level representation are extracted. In other words, an input data is compressed to a low dimensional vector by extracting the principal component feature of the uncorrupted version. In the decoding phases, the low dimensional vector of original data can be recovered via the decoder. This step helps to reduce interference of noise and distribution divergences of the input data.

In the preprocessing stage, due to the influence of noise, a part of the input components is erased while leaving others uncorrupted, we can obtain the corrupted version $\boldsymbol{r_0}$ generated by stochastic mapping $\boldsymbol{r}_0 \sim \boldsymbol{q}_{\mathcal{D}}(\boldsymbol{r}_0|\boldsymbol{r})$. Considering this, we should focus on the corrupted components instead of all components of the input data. To achieve this, we give a different weight for the reconstructed values, β_1 for the corrupted components, and β_2 for the uncorrupted components. β_1 and β_2 are considered hyper-parameters. For the training data $\boldsymbol{r}$ and reconstructed value $\boldsymbol{d}$, the squared loss function is

$$L(\boldsymbol{r}, \boldsymbol{d}) = \beta_1 \cdot \sum_{j \in \Im(\boldsymbol{r}_0)} \left(\boldsymbol{r}_j - \boldsymbol{d}_j\right)^2 + \beta_2 \cdot \sum_{j \notin \Im(\boldsymbol{r}_0)} \left(\boldsymbol{r}_j - \boldsymbol{d}_j\right)^2 \tag{11}$$

where $\Im(\boldsymbol{r}_0)$ is the indices of the components of $\boldsymbol{r}_0$ that are corrupted. The cross-entropy loss is given by

$$\begin{aligned} L_{\beta_1}(\boldsymbol{r}, \boldsymbol{d}) = &-\beta_1 \cdot \sum_{j \in \Im(\boldsymbol{r}_0)} \left[\boldsymbol{r}_j \log \boldsymbol{d}_j + (1 - \boldsymbol{r}_j) \log (1 - \boldsymbol{d}_j)\right] \\ &- \beta_2 \cdot \sum_{j \notin \Im(\boldsymbol{r}_0)} \left[\boldsymbol{r}_j \log \boldsymbol{d}_j + (1 - \boldsymbol{r}_j) \log (1 - \boldsymbol{d}_j)\right] \end{aligned} \tag{12}$$

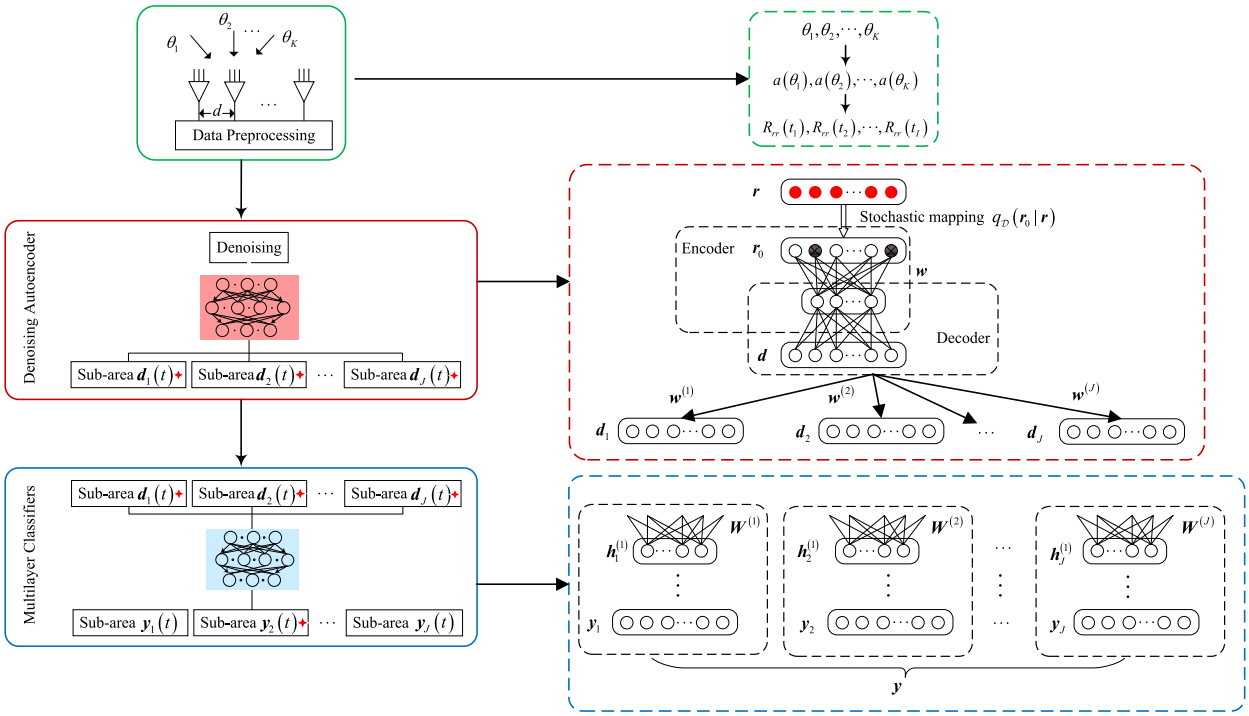

FIGURE 3. Structure of proposed DAE-DNN scheme.

which is called *emphasized DAE*. A special case when $\beta_1 = 1$, $\beta_2 = 0$ is called *full emphasis*, where we only consider the prediction error of the corrupted elements. Here, we set part of the elements of the original input as zeroes randomly and leaves the remaining elements uncorrupted, resulting in a "corrupted" version $\boldsymbol{r}_0$ of the original input $\boldsymbol{r}$. Comparing to $\boldsymbol{r}$, the "blank" elements of $\boldsymbol{r}_0$ will reduce many more information of $\boldsymbol{r}$. Benefit from DAE phase, we can try to fill the missing information by learning the mapping between $\boldsymbol{r}$ and $\boldsymbol{d}$, and then the extracted features can reflect the features of the original input $\boldsymbol{r}$ better.

Assuming that the number of layers in encoding and decoding are all equivalent to L_1. Setting the layer index is l_1 $(0 < l_1 \leq L_1)$, we generally have $|\boldsymbol{r}_{l_1}| < |\boldsymbol{r}_{l_1-1}|$ and $|\boldsymbol{r}_{L_1-l_1}| = |\boldsymbol{r}_{L_1+l_1}|$, where $|\boldsymbol{r}_{l_1}|$ is the dimension of $\boldsymbol{r}_{l_1}$. In the encoding phase, the hidden representation is given by

$$\boldsymbol{r}_{l_1} = f_{l_1}(\boldsymbol{r}_0) = f\left(\boldsymbol{w}_{l_1,l_1-1}\boldsymbol{r}_{l_1-1} + \boldsymbol{b}_{l_1}\right) \tag{13}$$

where l_1 and $l_1 - 1$ are the layer indices, $\boldsymbol{w}_{l_1,l_1-1} \in \mathbb{R}^{|r_{l_1}|\times|r_{l_1-1}|}$ is the weight matrix from the (l_1-1)th layer to the l_1th layer, and $\boldsymbol{b}_{l_1} \in \mathbb{R}^{|r_{l_1}|\times 1}$ is the additive bias vector in the l_1th layer, f_{l_1} represents the element-wise activation function in the l_1th layer.

In the decoding process, the number of decoders j is $1 \leq j \leq J$. The phase of decoding has the same hidden representation to (13) in each sub-areas. If we define the potential scope of the incident signals as $\left[\theta^{[0]}, \theta^{[J]}\right)$, $\varpi^{[j]}$ denote the angle scope $\left[\theta^{[j-1]}, \theta^{[j]}\right)$ of j, then we have

$$\theta^{[0]} < \theta^{[1]} < \cdots < \theta^{[J]} \tag{14a}$$

$$\theta^{[1]} - \theta^{[0]} = \theta^{[2]} - \theta^{[1]} = \cdots = \theta^{[J]} - \theta^{[J-1]} \tag{14b}$$

Figure 4 shows the relationship between the angle scope and sub-areas of DAE-DNN. If there are some signals located in the jth sub-areas, the output of the jth decoder, denoted as $\boldsymbol{d}_j = \boldsymbol{r}_{2L_1}^{[j]}$, is equivalent to the input $\boldsymbol{r}$ while the output of the other decoders is zero. Furthermore, we take the total outputs of DAE as the input of the multi-classifier in the next phase and further obtain the final estimation. For this reason, some additional requirements are needed in the DAE structure for DoA estimation, which can be described as

$$U^{[j]}(\boldsymbol{r}_1 + \boldsymbol{r}_2) = U^{[j]}(\boldsymbol{r}_1) + U^{[j]}(\boldsymbol{r}_2) \tag{15a}$$

$$\begin{cases} U^{[j]}(\boldsymbol{r}_1) = \boldsymbol{r}, & \theta \in \left[\theta^{[0]}, \theta^{[J]}\right) \\ U^{[j]}(\boldsymbol{r}_1) = 0, & otherwise \end{cases} \tag{15b}$$

where $U^{[j]}(\cdot)$ is the function of j-th DAE task.

From (15), in case of multiple signals located in different sub-areas impinge onto the array simultaneously, we should assure the additive property of DAE to classify the input data in different angle range to the corresponding different decoder outputs. Thus, we use the linear activation functions $f_{l_1}(\cdot)$ given by

$$\boldsymbol{r}_{l_1} = \boldsymbol{w}_{l_1,l_1-1} \cdot \boldsymbol{r}_{l_1-1} + \boldsymbol{b}_{l_1} \tag{16}$$

B. MULTILAYER CLASSIFIERS

In the phase of multilayer classifiers, a list of one-versus-all classifiers is applied in the DAE-DNN system. The final estimation of DoA with parameter θ is composed of J DAE, where there is a fixed number of angle values in each sub-areas output. Then each angle value is used as a refined grid of angle. According to the principle of the one-versus-all

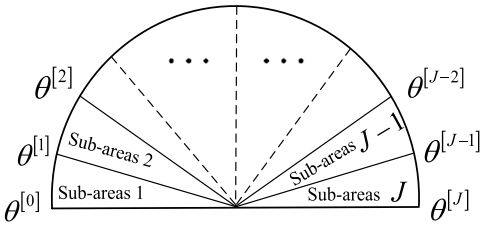

FIGURE 4. The relationship between angle scope and sub-areas of DAE-DNN.

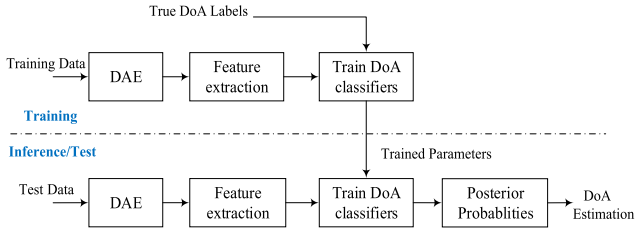

FIGURE 5. Basic structure of training and testing in DAE-DNN.

method, the final output of angle values represents the probability of a signal located in its neighborhood. Furthermore, the DoA of signals from off-grid directions can be estimated by interpolation between two adjacent refined grids.

From the blue rectangle below in Fig. 3, there are J parallel classifiers with the same number of sub-areas, which use the outputs of DAEs as inputs. We can obtain the DoA estimation based on the refined grids of the angle of all sub-areas, only the grid being close to the actual signal directions will generate a positive valued output while all others are zero.

The feed-forward computations of the classifiers are given by

$$\begin{aligned} \boldsymbol{y}_{l_2}^{[j]} &= g\left(\boldsymbol{W}_{l_2,l_2-1}^{[j]}\boldsymbol{h}_{l_2-1}^{[j]} + \boldsymbol{p}_{l_2}^{[j]}\right), \\ &\text{with} \quad j = 1,\ldots,J;\, l_2 = 1,\ldots,L_2 \end{aligned} \tag{17}$$

where l_2 and l_2-1 are the layer indices in the multi-classifiers, $\boldsymbol{y}_{l_2}^{[j]}$ is the output vector in the l_2th layer of jth classifier, with $\boldsymbol{h}_0^{[j]} = \boldsymbol{d}_j$ and $\boldsymbol{h}_{L_2}^{[j]} = \boldsymbol{y}_j$, $\boldsymbol{W}_{l_2,l_2-1}^{[j]} \in \mathbb{R}^{\left|h_{l_2}^{[j]}\right| \times \left|h_{l_2-1}^{[j]}\right|}$ is the weight matrix with fully connected feed-forward property from the (l_2-1)th layer to the l_2th layer, $\boldsymbol{p}_{l_2}^{[j]} \in \mathbb{R}^{\left|h_{l_2}^{[j]}\right| \times 1}$ is the additive bias vector in the l_2th layer, and g_{l_2} represents the element-wise activation function for the input of the l_2th layer nodes.

All the outputs of the J parallel classifiers based on the J decoder outputs are obtained as

$$\boldsymbol{y} = \left[\boldsymbol{y}_1^T, \cdots, \boldsymbol{y}_J^T\right]^T \tag{18}$$

where $\boldsymbol{y}_j^T$ is the estimation of spatial spectrum associated with DNN input $\boldsymbol{r}$. Note that $\boldsymbol{y}_j^T$ is concatenated in order from the classifiers number 1 to J. And only the values of refined grids close to the actual signal directions are positive in $\boldsymbol{y_j}$ while all the others are zero.

IV. DoA ESTIMATION BASED ON DAE-DNN

Based on the framework discussed in Section III, we present the basic structure of training and testing of the proposed DAE-DNN. In Fig. 5, DAE is used to mitigate the noise and divide the whole estimated angle range into the designed number sub-areas. At the same time, the parallel multilayer classifiers make a further estimation of the refined grid located by the impinging signal. Generally, DNN approach can achieve non-linear mappings and learn the data's characteristics in a high-dimensional space. In this paper, a new characteristic space is exploited to achieve DoA estimation. In order to avoid a situation where the neural networks get trapped in local minima, we use two completely separated training procedures.

A. DAE TRAINING

As results in Section II indicate that the covariance matrix $\boldsymbol{R}_{rr}$ of receiving array has all information of direction of arrival $\{\theta_1, \theta_2, \cdots, \theta_K\}$, which is given by

$$\boldsymbol{R}_{rr} = \begin{bmatrix} \boldsymbol{R}_{1,1} & \cdots & \boldsymbol{R}_{1,M-1} & \boldsymbol{R}_{1,M} \\ \boldsymbol{R}_{2,1} & \cdots & \boldsymbol{R}_{2,M-1} & \boldsymbol{R}_{2,M} \\ \vdots & \ddots & \vdots & \vdots \\ \boldsymbol{R}_{M-1,1} & \cdots & \boldsymbol{R}_{M-1,M-1} & \boldsymbol{R}_{M-1,M} \\ \boldsymbol{R}_{M,1} & \cdots & \boldsymbol{R}_{M,M-1} & \boldsymbol{R}_{M,M} \end{bmatrix} \tag{19}$$

where $\boldsymbol{R}_{x_1,x_2}$ is the (x_1, x_2)th element of the covariance matrix $\hat{\boldsymbol{R}} = \frac{1}{I}\sum_{i=1}^{I} \boldsymbol{x}(t_i)\boldsymbol{x}^T(t_i)$. Instead of taking the covariance matrix $\boldsymbol{R}_{rr}$ as the input of DAE directly, in this paper, we just use the upper-right matrix elements (the color area in (19)). The lower left elements are unnecessary because it is the conjugate replicas of the upper right ones, and the diagonal covariance elements are not included since this elements are associated with unknown noise variance.

Furthermore, we reformulate the complex data (19) to real data by preserving the original information as much as possible by (20) and (21), which can be used as the input of the parallel multilayer classifiers.

$$\hat{\boldsymbol{r}} = \left[\boldsymbol{R}_{1,2}, \boldsymbol{R}_{1,3}, \cdots, \boldsymbol{R}_{1,M}, \boldsymbol{R}_{2,3}, \cdots, \boldsymbol{R}_{2,M}, \cdots, \boldsymbol{R}_{M-1,M}\right]^T \tag{20}$$

$$\boldsymbol{r} = \left[re\left\{\hat{\boldsymbol{r}}^T\right\}, \text{im}\left\{\hat{\boldsymbol{r}}^T\right\}\right]^T / \left\|\hat{\boldsymbol{r}}\right\|_2 \tag{21}$$

Constructing the training dataset $\boldsymbol{r}$ with the signal direction varying in the range of $\left[\theta^{[0]}, \theta^{[J]}\right)$, $\varpi^{[j]}$ $(1 \le j \le J)$ denotes the range of spectrum grid points covered by the jth decoder. Setting the whole grids as $\delta_1, \delta_2, \cdots, \delta_V$ (V is the total number of spaced spectrum grids). So, in each spaced spectrum of DAE, there are $V/J = V_0$ grids. The relationship between spectrum grids and decoder scope is shown in Fig. 6, in which $\theta^{[0]} = \delta_1$, $\theta^{[1]} = \delta_{V_0}$, $\theta^{[2]} = \delta_{2V_0}$, $\theta^{[J-1]} = \delta_{(J-1)\cdot V_0+1}$ and $\theta^{[J]} = \delta_{J\cdot V_0}$.

If there is an impinging signal from the direction δ_v, then the corresponding covariance vector $\boldsymbol{r}(\delta_v)$ is generated,

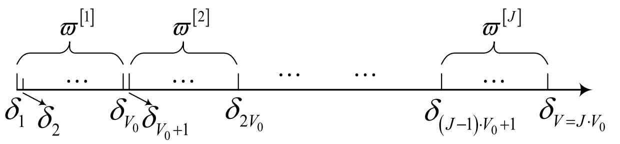

FIGURE 6. **The relationship between spectrum grids and decoder scope.**

which causes the output of decoder is $\boldsymbol{d}(\delta_v)$ as we expect while the outputs of other decoders are zero.

We denote the dataset for DAE training as

$$\boldsymbol{\Gamma} = [\boldsymbol{r}(\delta_1), \boldsymbol{r}(\delta_2), \cdots, \boldsymbol{r}(\delta_V)] \tag{22}$$

and the corresponding DAE label set is $\boldsymbol{\Delta} \in \mathbb{R}^{J\cdot|\boldsymbol{r}|\times V}$, which consist of $\boldsymbol{d}(\delta_v)$ (see (23), as shown at the bottom of the page). The data-label of $(\boldsymbol{\Gamma}, \boldsymbol{\Delta})$ is used as the input and expected output to train the DAE. No matter the signals lies in the same sub-area or not, the data-label of $(\boldsymbol{\Gamma}, \boldsymbol{\Delta})$ contains the key information of signals. The loss function used between the actual output and the expected one is expressed as

$$\varepsilon(\delta_v) = \frac{1}{2}\left\|\boldsymbol{d}(\delta_v) - \hat{\boldsymbol{d}}(\delta_v)\right\|_2^2 \tag{24}$$

where $\hat{\boldsymbol{d}}(\delta_v)$ is the actual output of the DAE, $\boldsymbol{d}(\delta_v)$ is the expected output when $\boldsymbol{r}(\delta_v)$ is the input.

To obtain the updated weight matrices $\boldsymbol{w'}_{l_1,l_1-1}$ and bias vectors $\boldsymbol{b'}_{l_1}$, backpropagated gradient algorithm is used in the loss function.

$$\boldsymbol{w'}_{l_1,l_1-1} = \boldsymbol{w}_{l_1,l_1-1} + \mu_1\frac{\partial\varepsilon(\delta_v)}{\partial\boldsymbol{w}_{l_1,l_1-1}} \tag{25}$$

$$\boldsymbol{b'}_{l_1} = \boldsymbol{b}_{l_1} + \mu_1\frac{\partial\varepsilon(\delta_v)}{\partial\boldsymbol{b}_{l_1}} \tag{26}$$

where $\boldsymbol{w'}_{l_1,l_1-1}$ and $\boldsymbol{b'}_{l_1}$ are the variables after current update, $\boldsymbol{w}_{l_1,l_1-1}$ and $\boldsymbol{b}_{l_1}$ are the variables before update and μ_1 is the learning rate. *Algorithm*1 presents the operational procedure of the proposed DAE algorithm for sub-areas spatial estimation.

B. PARALLEL MULTILAYER CLASSIFIERS TRAINING

After the DAE training phase, its outputs generated from J decoders are taken as the inputs of parallel multilayer classifiers, and the spatial spectrum is estimated simultaneously in each sub-areas. Different from the input $\boldsymbol{r}$ of DAE, which contain the features of all possible directions range, the input of multilayer classifiers $\boldsymbol{d}^{[j]}$ $(j = 1, 2, \cdots, J)$ only contain the features of its own sub-areas direction range. In other words, when compared to $\boldsymbol{r}$, $\boldsymbol{d}^{[j]}$ $(j = 1, 2, \cdots, J)$ have more concentrated distributions. Based on this, we further set the activation function as the hyperbolic tangent function and then get a refined DoA estimation, which is shown as

$$\tanh(\boldsymbol{\kappa}) = [\tanh(\kappa_1), \tanh(\kappa_2), \ldots, \tanh(\kappa_{-1})]^T \tag{27}$$

$$\tanh(\kappa) = \frac{e^{\kappa} - e^{-\kappa}}{e^{\kappa} + e^{-\kappa}} \tag{28}$$

where κ_{-1} is the last element of $\boldsymbol{\kappa}$.

In the whole DAE-DNN estimation system, there are two training processes, DAE training and multilayer classifiers training. When the DAE training is finished, values of the weight matrices and bias vectors are fixed. Then we train the whole end-to-end DAE-DNN framework whose input and output are vector $\boldsymbol{r}$ and reconstructed spectrum $\boldsymbol{y}$, respectively. The weight matrices and bias vectors of the classification neural networks should be trained to estimate different directions of multiple signals in different sub-areas. To achieve this goal, we collect another training dataset with multiple signals at the same time.

As mentioned in Section II, there are K-independent signals to be detected at the receiver, among these we denote the inter signal angles as $\boldsymbol{c} = \{c_m\}_m^{K-1}$ $(m = 1, 2, \cdots, K-1)$. Then we can get the input vectors $\boldsymbol{r}(\theta, c_1, \cdots, c_m)$, which represents the K signals from different directions θ, $\theta + c_1, \cdots, \theta + c_m$ with $\theta^{[0]} \le \theta < \theta^{[J]} - c_m$. Meanwhile, as discussed in the previous section, the reconstructed spectrum of multilayer classifiers only has positive values on the grid adjacent to the actual signal direction. Thus, we can obtain the estimation of each signal via linear interpolation between two adjacent refined grids.

Mathematically, the classifier output $\boldsymbol{y}(\theta, c_1, \cdots, c_m)$ corresponding to input $\boldsymbol{r}(\theta, c_1, \cdots, c_m)$ is expressed as

$$\boldsymbol{y}(\theta, c_1, \cdots, c_m) = \begin{cases} \dfrac{\bar{\theta} - \delta_{i-1}}{\delta_i - \delta_{i-1}}, & \delta_{i-1} \le \bar{\theta} < \delta_i \\ \dfrac{\delta_{i+1} - \bar{\theta}}{\delta_{i+1} - \delta_i}, & \delta_i \le \bar{\theta} < \delta_{i+1} \\ 0, & others. \end{cases} \tag{29}$$

where $\bar{\theta} \in \{\theta, \theta + c_1, \cdots, \theta + c_m\}$.

The training dataset of the parallel multilayer classifiers and the associated label set are

$$\Phi = [\Phi_1, \Phi_2, \cdots, \Phi_{K-1}] \tag{30}$$

$$\Psi = [\Psi_1, \Psi_2, \cdots, \Psi_{K-1}] \tag{31}$$

$$\boldsymbol{\Delta} = \left[\boldsymbol{d}\left(\varpi^{[1]}\right), \cdots, \boldsymbol{d}\left(\varpi^{[J]}\right)\right] = \begin{bmatrix} \underbrace{\begin{matrix}\boldsymbol{r}(\delta_1), \cdots, \boldsymbol{r}(\delta_{V_0}) \\ 0 \\ 0 \\ 0\end{matrix}}_{V_0} & \underbrace{\begin{matrix}0 \\ \boldsymbol{r}(\delta_{V_0+1}), \cdots, \boldsymbol{r}(\delta_{2V_0}) \\ 0 \\ 0\end{matrix}}_{V_0} & \begin{matrix}0 \\ 0 \\ \ddots \\ 0\end{matrix} & \underbrace{\begin{matrix}0 \\ 0 \\ 0 \\ \boldsymbol{r}(\delta_{(J-1)\cdot V_0+1}), \cdots, \boldsymbol{r}(\delta_{J\cdot V_0})\end{matrix}}_{V_0} \end{bmatrix} \tag{23}$$

Algorithm 1 DAE Algorithm for Sub-Areas Spatial Estimation

Input: Impinging signal angles $\{\theta_1, \theta_2, \cdots, \theta_K\}$.

Output: Estimated spatial sub-areas $\boldsymbol{d}_1^{[j]}, \boldsymbol{d}_2^{[j]}, \cdots, \boldsymbol{d}_K^{[j]}$.

1: **Step1: Get the input $\boldsymbol{r}$ of DAE from impinging direction $\{\theta_1, \theta_2, \cdots, \theta_K\}$.**
2: The steering vector $\boldsymbol{a}(\theta)$ is generated by $\{\theta_1, \theta_2, \cdots, \theta_K\}$ in (1), then the array output $\boldsymbol{X}(t)$, covariance matrix $\boldsymbol{R}_{xx}(t)$ are obtained in (2) and (5). Also, we can get the $\hat{\boldsymbol{r}}$ and $\boldsymbol{r}$ based on (20) and (21).
3: **Step2: Train the DAE with data-label $(\boldsymbol{\Gamma}, \boldsymbol{\Delta})$.**
4: Generate a set of training sequences of DAE, which includes the input $\boldsymbol{r}$ and output $\boldsymbol{d}^{[j]}$. Also, set the learning rate and the loss rate, as well as the weight matrices and bias vectors. Furthermore, set the error threshold as $\tau = 10^{-6}$.
5: **while** error $\geq 10^{-6}$: **do**
6: Train the DAE based on the given sequences according to the proposed learning policy by backpropagated gradients algorithm;
7: Update the weight matrices $\boldsymbol{w'}_{l_1,l_1-1}$ and bias vectors $\boldsymbol{b'}_{l_1}$ of each layer of DAE by (25) and (26).
8: **end while**
9: **Step3: Obtain the spatial gains of filters $\alpha^{[j]}$.**
10: **for** $v = 1, 2, \cdots, V$ **do**
11: Calculate the conjugate transpose matrix $\hat{\boldsymbol{r}}(\delta_v)^H$ based on the input $\boldsymbol{r}(\delta_v)$.
12: **end for**
13: **for** $j = 1, 2, 3, \cdots, J$ **do**
14: Calculate the complex-valued $\hat{\boldsymbol{d}}^{[j]}$ consist of the first half of $\boldsymbol{d}^{[j]}$ as the real part while the second half as imaginary part.
15: Obtain the spatial gains of filters $\alpha_v^{[j]} = \left|\hat{\boldsymbol{r}}(\delta_v)^H \cdot \hat{\boldsymbol{d}}^{[j]}\right|$.
16: **end for**
17: **return** DAE scheme and the sub-areas spatial estimation $\boldsymbol{d}_1^{[j]}, \boldsymbol{d}_2^{[j]}, \cdots, \boldsymbol{d}_K^{[j]}$ corresponding of impinging signal angles $\{\theta_1, \theta_2, \cdots, \theta_K\}$ according to $\alpha_v^{[j]}$.

where

$$\Phi_m = [\boldsymbol{r}(\delta_1, c_1, \cdots c_m), \cdots, \boldsymbol{r}(\delta_V - \max(\boldsymbol{c}), c_1, \cdots c_m)] \tag{32}$$

$$\Psi_m = [\boldsymbol{y}(\delta_1, c_1, \cdots c_m), \cdots, \boldsymbol{y}(\delta_V - \max(\boldsymbol{c}), c_1, \cdots c_m)] \tag{33}$$

In the training process, we update the parameters of the multilayer classifiers by the backpropagation of the reconstruction error. The loss function $\epsilon(\theta, \boldsymbol{c})$ is defined as the squared l_2-norm of the spectrum reconstruction error:

$$\epsilon(\theta, \boldsymbol{c}) = \frac{1}{2}\left\|\boldsymbol{y}(\theta, \boldsymbol{c}) - \hat{\boldsymbol{y}}(\theta, \boldsymbol{c})\right\|_2^2 \tag{34}$$

where $\hat{\boldsymbol{y}}(\theta, \boldsymbol{c})$ is the actual output of the multi-layer classifiers and $\boldsymbol{y}(\theta, \boldsymbol{c})$ is the expected output.

The elements of the weight matrices $\boldsymbol{W}_{l_2,l_2-1}$ and bias vectors $\boldsymbol{p}_{l_2}$ are then updated as

$$\boldsymbol{W'}_{l_2,l_2-1} = \boldsymbol{W}_{l_2,l_2-1} + \mu_2 \frac{\partial \epsilon(\theta, \boldsymbol{c})}{\partial \boldsymbol{W}_{l_2,l_2-1}} \tag{35}$$

$$\boldsymbol{p'}_{l_2} = \boldsymbol{p}_{l_2} + \mu_2 \frac{\partial \epsilon(\theta, \boldsymbol{c})}{\partial \boldsymbol{p}_{l_2}} \tag{36}$$

where $\boldsymbol{W'}_{l_2,l_2-1}$ and $\boldsymbol{p'}_{l_2}$ are the values of variables after current update, and $\boldsymbol{W}_{l_2,l_2-1}$ and $\boldsymbol{p}_{l_2}$ are the variables before the update, and μ_2 is the learning rate. **Algorithm 2** presents the operational procedure of parallel multilayer classifiers based on DNN.

Considering that the responding function of the array is perturbed by the noise and array inaccuracies, the mapping from signal direction to covariance vectors is $\theta \overset{e}{\mapsto} \boldsymbol{r}_e(\theta)$, where e is the error. When we use the perturbed vector $\boldsymbol{r}_e(\theta)$ as an input to the DAE, the associated label vector is still located in the jth sub-area, even in the environment with strong noise. After this, in the process of parallel multilayer classifiers, we take the signal from direction δ_i as the input, which is embedded in the output of jth decoder. If the associated spectrum label contains a spectrum peak (one or two grids closest to δ_i), we can obtain the DoA estimate of δ_i by interpolation. Therefore, whole DAE-DNN system (the DAE process together with the parallel multilayer classifiers) actually forms an inverse mapping from perturbed output $\boldsymbol{r}_e(\theta)$ to the input θ when the array inaccuracies and environmental noise all exist. Furthermore, this derived inverse mapping $\boldsymbol{r}_e(\theta) \overset{e}{\mapsto} \theta$ under noise effect also adapts to the test data and is expected to obtain accurate DoA estimation even in the low SNR regime.

V. SIMULATION AND RESULTS

In this section, we present the simulation results to demonstrate the effectiveness of the proposed method. For comparison, we test the traditional DoA detection methods in low SNR level. The simulations are implemented on *TensorFlow* [36], and its embedded tools are used to compute the gradients directly.

A. SIMULATION SETTINGS

We consider ULA with M elements are used in the estimation of K signals impinging from the spatial scope of $\left[\theta^{[0]}, \theta^{[J]}\right)$. The inter-element spacing of the ULA is half-wavelength, and the potential space is divided into J sub-areas with equal spatial scopes V_0 grids. The covariance vector $\boldsymbol{r}$ in the training dataset of both DAE and multilayer classifiers are all obtained from I snapshots. The simulation parameters of DAE-DNN system and the value of training data are shown in Table 1.

In the DAE training phase, we obtain a direction set $(\delta_1 = -90°, \delta_2 = -89°, \cdots, \delta_V = 89°)$ through sampling with an interval of δ in the spatial scope $\left[\theta^{[0]}, \theta^{[J]}\right)$ and compute the covariance vectors and associated labels on the basis of (22) and (23).

Algorithm 2 Parallel Multilayer Classifiers Based on DNN Scheme for DoA Estimation (PMC-DNN).

Input: Impinging signal angles $\{\theta_1, \theta_2, \cdots, \theta_K\}$.

Output: DoA estimation $\left\{\theta_1', \theta_2', \cdots, \theta_K'\right\}$.

1: **Step1: Get the input $\boldsymbol{r}$ of DAE from impinging direction $\{\theta_1, \theta_2, \cdots, \theta_K\}$.**
2: It is the same as **Step1** in **Algorithm 1**.
3: **Step2: Train the PMC-DNN with data-label $(\boldsymbol{\Phi}, \boldsymbol{\Psi})$.**
4: After the training in DAE scheme, we can get some parameters (such as $\boldsymbol{w'}_{l_1,l_1-1}$, $\boldsymbol{b'}_{l_1}$), then fix them. Generate training sequences $(\boldsymbol{\Phi}, \boldsymbol{\Psi})$ of PMC-DNN according to (30), (32), (31) and (33). Also, set the learning rate, the loss rate and the error threshold as **Algorithm 1**.
5: **while** error $\geq 10^{-6}$: **do**
6: Train the PMC-DNN based on the given sequences according to the proposed learning policy by back-propagated gradients algorithm;
7: Update the weight matrices $\boldsymbol{W'}_{l_2,l_2-1}$ and bias vectors $\boldsymbol{p'}_{l_2}$ of each layer by (35) and (36).
8: **end while**
9: **Step3: Obtain the DoA estimation $\left\{\theta_1', \theta_2', \cdots, \theta_K'\right\}$.**
10: **for** $v = 1, 2, \cdots, V$ **do**
11: Calculate the conjugate transpose matrix $\hat{\boldsymbol{r}}\,(\delta_v)^H$ based on the input $\boldsymbol{r}\,(\delta_v)$.
12: **end for**
13: **for** $m = 1, 2, 3, \cdots, K-1$ **do**
14: Calculate the PMC-DNN estimation $\boldsymbol{y}\,(\theta, c_1, \cdots, c_m)$ of $\boldsymbol{r}\,(\theta, c_1, \cdots, c_m)$ by interpolating the one or two spectrum peaks. Obtain the DoA estimation $\left\{\theta_1', \theta_2', \cdots, \theta_K'\right\}$ by $\boldsymbol{r}\,(\theta, c_1, \cdots, c_m)$.
15: **end for**
16: **return** DoA estimation $\left\{\theta_1', \theta_2', \cdots, \theta_K'\right\}$.

On each direction grid, we compute one covariance vector by the collection of one group of snapshots. For the training by the minibatch training strategy, the dataset shuffled in each epoch with training parameters are: batch size of B and learning rate of μ_1, and $epoch_1$. We set the size of the input layer, hidden layer and output layer as $o = M(M-1) = 90$, $90/2 = 45$ and $oJ = 90 \times 9$, respectively.

In the DNN training phase, we collect another dataset to train the classifiers with more than one signal when the DAE parameters are fixed after the training process. In this setting, we choose K signals. The inter-signal angle $\boldsymbol{c}$ is sampled from the dataset, which covers all scenarios of adjacent signals separated by the double width of the spatial spectrum grid. If one signal direction (denoted by θ) is sampled with an interval of 1° from -90° to $90^\circ - \boldsymbol{c}$, then the other signal direction is $\theta + \boldsymbol{c}$. Then, the covariance vectors are collected, which are used for training with a batch size of B and learning rate of μ_2, and the order of the vector is shuffled during each training epochs of $epoch_2$. In order to obtain a trade-off between the expressivity power (improves with deeper networks) and undertraining risk (aggravates with more network parameters) of the classifiers, we set the size of hidden layers to $L_2 - 1 = 2$. In each classifier, we set the sizes of the hidden layer and output layers to $\lfloor 2/3 \times o \rfloor = 60$, $\lfloor 4/9 \times o \rfloor = 40$, respectively. Based on a uniform distribution between -0.1 and 0.1, we initialize all the weights and biases of the DNN randomly.

TABLE 1. Simulation parameter of DAE training.

Description	Parameters	Values
ULA elements	M	10
Number of signals	K	2
Spatial scope of signals	$\left[\theta^{[0]}, \theta^{[J]}\right)$	$[-90^\circ, 90^\circ)$
Spatial spectrum of grid	δ	1°
Total grids	V	180 grids
Number of sub-areas	J	9
Spatial scope of sub-area	V_0	20 grids
Snapshots	I	500
Batch size	B	32 bits
Learning rate	μ_1, μ_2	0.001
Epochs of DAE	$epoch_1$	1000
Epochs of DNN	$epoch_2$	300
Inter-signal angle	$\boldsymbol{c}$	$[2^\circ, 4^\circ, \cdots, 90^\circ]$

In this paper, we consider mutual coupling as the main source of the array inaccuracy, which occurs often in almost all types of arrays and leads to considerable deterioration in conventional algorithms [37]. The mathematical model in the simulations is shown as

$$\boldsymbol{e} = \rho \times \left[0, \zeta^1, \ldots, \zeta^{M-1}\right]^T \tag{37}$$

where the parameter $\rho \geq 0$ is used to indicate the strength of array inaccuracies, $\zeta = 0.5e^{j\pi/3}$ is the mutual coupling coefficient between adjacent sensors. Further, the perturbed array responding function is

$$\boldsymbol{a}\,(\theta, \boldsymbol{e}) = (\boldsymbol{I}_M + \boldsymbol{E}_{mc}) \times \boldsymbol{a}\,(\theta) \tag{38}$$

where $\boldsymbol{I}_M$ is the $M \times M$ unitary matrix $\boldsymbol{E}_{mc}$ is a toeplitz matrix with parameter vector $\boldsymbol{e}$.

B. ANALYSIS ON THE PARAMETERS AND DoA ESTIMATION

In Fig. 7, we plot the RMSE performance of the DoA estimation against different SNR of the proposed scheme with different snapshot number I. Recall that the snapshot I of receiving array is a controlling parameter in DAE-DNN. We consider $I = 100, 400, 500, 600$, and 800 in the simulation with an ascending order. Note that dB scale of SNR is defined as SNR $= 10\log_{10}\left(P_{signal}/P_{noise}\right)$, where P_{signal} and P_{noise} are the power of signal and noise respectively. We observe that the RMSE of the DoA estimation decreases with SNR, and it becomes stable gradually until the SNR is large enough. Meanwhile, the simulation results demonstrate that the RMSE performance can be enhanced when adopting a large snapshot number. However, as the number of snapshot continues to increase, the accuracy improves substantially.

This result is dedicated to the fact that a larger snapshot of receiving array can stir the effectiveness of the off-line training process of DAE-DNN scheme, which can capture complete information about DoA.

Additionally, Fig. 8 (a)-(c) displays that the RMSE performance against different strength of array inaccuracies parameter ρ for different SNR levels. It demonstrates that as I increases, the RMSE of DAE-DNN scheme decreases with better robustness to the array inaccuracies. However, the large snapshot will result in complex computation and long-time consumption. Thus, in this paper, we choose $I = 500$ to get a trade-off between them.

Figure 9 exhibits the RMSE performance of the DoA estimation in different SNR levels with different batch sizes. In our simulations, we test various batch sizes (16, 32, ... bits). We set the length of the training sequence as 16 bits initially. From Fig. 9, we see that the RMSE decreases with SNR in each batch size setting. When the SNR is higher than -8dB, the RMSE performance decrease first and then increase when the batch size varies from 16 bits to 32 bits, 64 bits, and 128 bits. More details of the RMSE performance in different SNR levels with different batch sizes are shown in Fig. 10 (a)-(d). As is clear from the figure, "Batch size x= 32 bits" has lower RMSE and also more stable when compared to that of other batch sizes (16 bits, 64 bits, and 128 bits). This means that in the training procedure of the DAE-DNN scheme, the small-batch size reduces the convergence rate while the large-batch size enlarges the epochs. Therefore, we choose "Batch size = 32 bits" for optimizing the network between RMSE performance and the stability of the proposed DAE-DNN scheme.

In Fig. 11, we show the RMSE performance of the DoA estimation as a function of SNR of the proposed DAE-DNN system, where the learning rate is set as 0.0001, 0.0005, 0.001, 0.005, and 0.01, respectively. Here, the length of the training sequence is initialized to 32 bits. For "learning rate" is 0.0005 and 0.001, the RMSE performance of the DoA estimation decreases more significantly as SNR increases. However, the RMSE performance in the case of "learning rate = 0.0001, 0.005 and 0.01" shows a slow convergence speed in the low SNR range while performing worse and unstable as SNR increases. It can be concluded from this group of simulation results that selecting an appropriate learning rate is a significant issue for boosting the performance of the DAE-DNN scheme for DoA estimation. When "learning rate" is smaller than 0.001, it takes quite a bit of time to attain good DoA estimation. In contrast, "learning rate" being larger than 0.001 will lead to worse DoA performance improvements. Considering this tradeoff, we set the "learning rate" to be 0.001 in simulations. More details are shown in Fig. 12 (a)-(c), which we observe that "learning rate = 0.001" shows the best performance.

Figure 13 provides an RMSE comparison of the DoA estimation against the SNR, for various epochs. As the SNR increases, the RMSE performance is improved, which is similar to the results in the other parameters above. In case

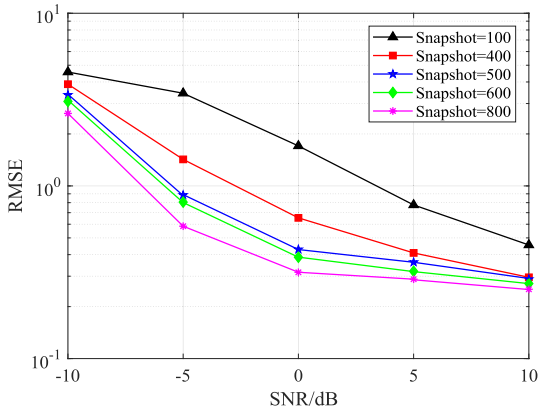

FIGURE 7. RMSE performance of the DoA estimation of the proposed scheme when the snapshot are set as 100, 400, 500, 600, and 800.

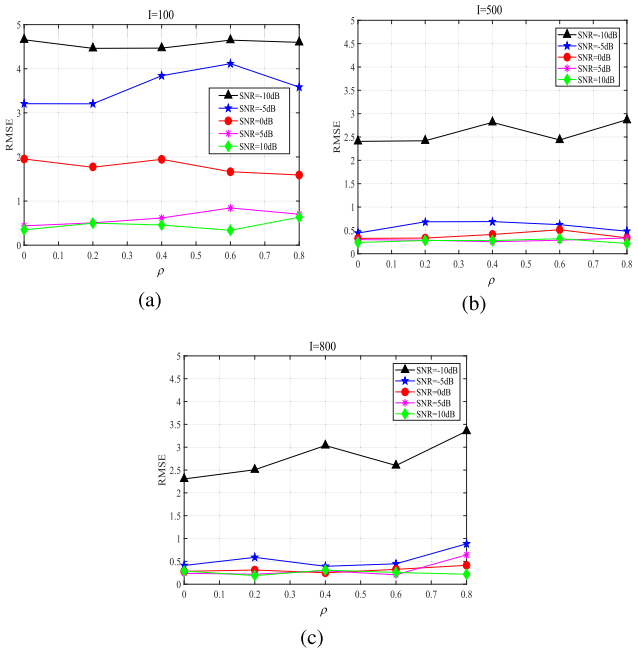

FIGURE 8. RMSE performance of the DoA estimation in different rho values when the snapshot are different. (a) $I = 100$. (b) $I = 500$. (c) $I = 800$.

of "Epoch = 300" and "Epoch = 2000," the RMSE performance are decrease between -10dB and 5dB and increase between 5dB and 10dB. This is because too small epoch and too large epoch will result in under-fitting or over-fitting, which is always different in different DNN-based systems or schemes. Furthermore, compared to the "Epoch = 500" situation, the RMSE performance of "Epoch = 1000" is degraded in the range of -10dB to 10dB. More details are shown in Fig. 14 (a)-(d), when SNR larger than -5dB, all of the epoch number will get a satisfied RMSE performance. This is an expected result since the DAE-DNN scheme is a promising tool for ensuring robustness in low SNR when array inaccuracies exist.

Figure 15 depicts the RMSE performance against different ρ in different SNR levels (e.g., -10dB$<$SNR$<$10dB).

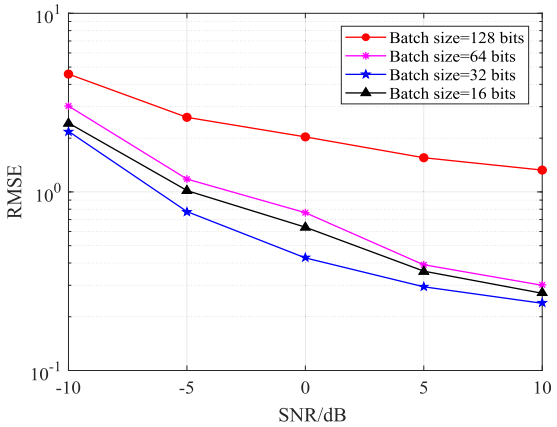

FIGURE 9. RMSE performance of the DoA estimation of the proposed scheme when the batch size set as 128 bits, 64 bits, 32 bits and 16 bits.

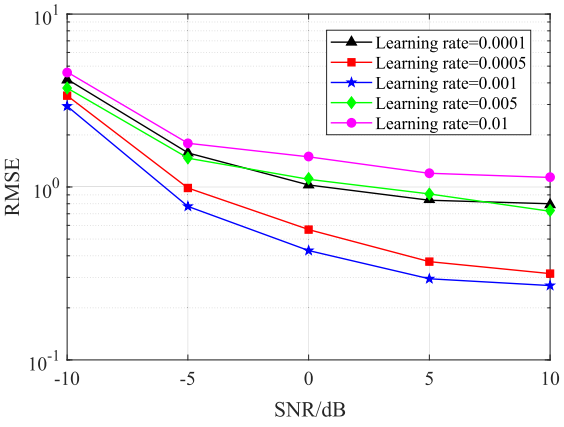

FIGURE 11. RMSE performance of the DoA estimation of the proposed scheme when the learning rate are set as 0.0001, 0.0005, 0.001, 0.005 and 0.01.

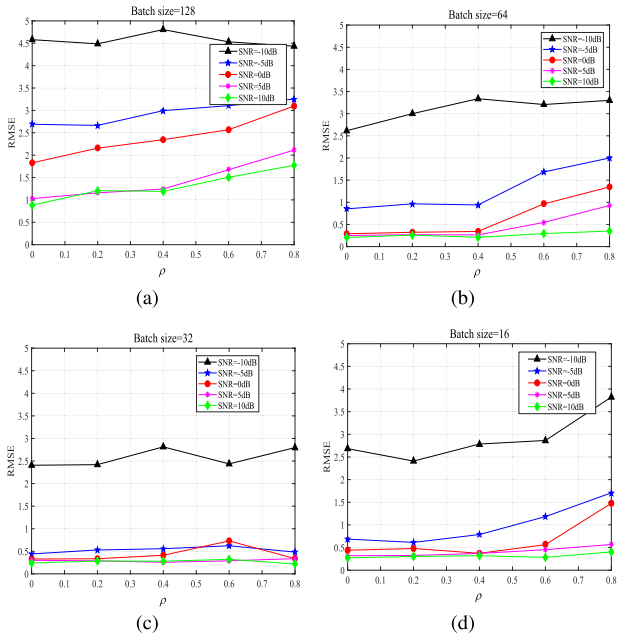

FIGURE 10. RMSE performance of the DoA estimation in different batch size. (a) Batch size=128 bits. (b) Batch size = 64 bits. (c) Batch size = 32 bits. (d) Batch size=16 bits.

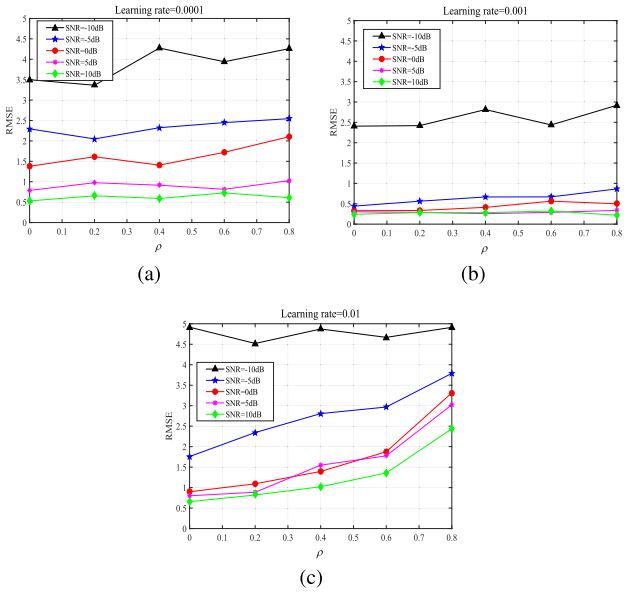

FIGURE 12. RMSE performance of the DoA estimation in different learning rate. (a) Learning rate = 0.0001. (b) Learning rate = 0.001. (c) Learning rate = 0.01.

As the ρ increases, all RMSE values fluctuate around the mean value of each SNR. When SNR $>$ 0dB, DAE-DNN scheme will perform good achieving RMSE under 0.5. In the low SNR regime (e.g., -5dB$<$SNR$<$0dB), the DAE-DNN system maintains its robustness even when ρ increases. While in the lower SNR regime (e.g., -10dB$<$SNR<-5dB), the correlation associated with SNR increases, causing severe degradation of the RMSE performance. The results in Fig. 15 show that the performance of RMSE is dominated by the value of SNR rather than ρ. Furthermore, Fig. 15 shows the relationship between RMSE and SNR of DAE-DNN scheme when ρ is fixed. As the SNR increases, DAE-DNN shows better robustness performance. When SNR is as low as -5 dB, DAE-DNN remains to demonstrate small RMSE performance.

Furthermore, in order to compare the performance of the proposed scheme and the previous schemes in the complexity scenarios (low SNR or mutual coupling when the signal number is unknown), we plot the estimated RMSE against different mutual coupling and SNR levels in Fig. 16 and Fig. 17, respectively. Fig. 16 compares the performance of ESPRIT, CLASSICAL MUSIC, ROOT-MUSIC, I-MUSIC, and DAE-DNN when SNR $=$ 0 dB, $M = 10$, $K = 2$, $I = 500$ and the spacing of the sensors equals half of the wavelength. DAE-DNN exhibited the satisfying performance in the whole SNR range in terms of RMSE, while MUSIC and its derivative algorithms were best only when $\rho < 0.2$. We should mention here again that it seems that the performance of ESPRIT, MUSIC and its derivative algorithms is dominated by the quality of array responding function.

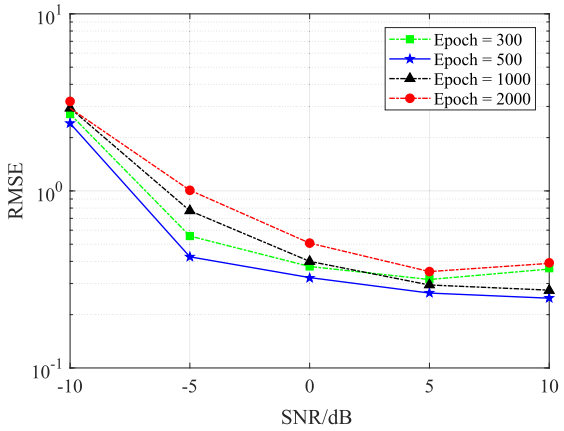

FIGURE 13. RMSE performance of the DoA estimation of the proposed scheme when the epoch are set as 300, 500, 1000 and 2000.

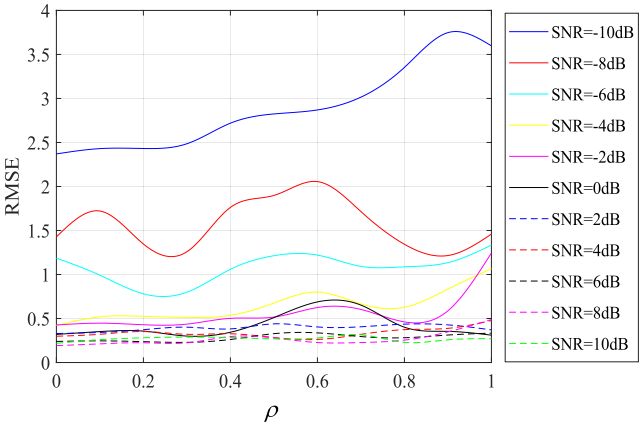

FIGURE 15. The relationship between ρ and RMSE for different SNR levels.

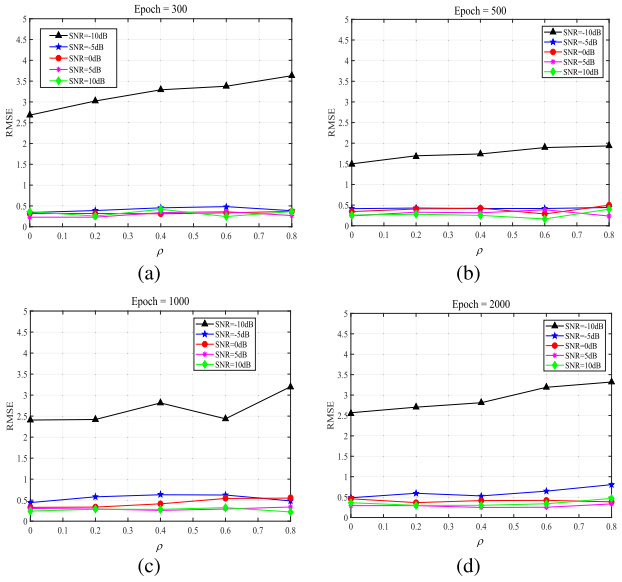

FIGURE 14. RMSE performance of the DoA estimation in different epoch number. (a) Epoch = 300. (b) Epoch = 500. (c) Epoch = 1000. d) Epoch = 2000.

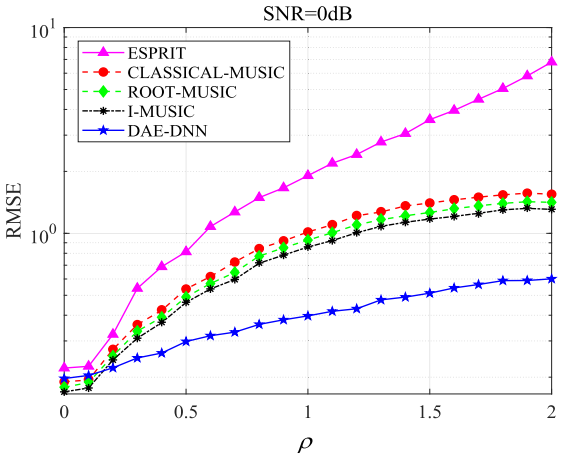

FIGURE 16. The comparison results of traditional methods and DAE-DNN when SNR = 0 dB.

When ρ increases, these traditional methods perform worse than DAE-DNN due to the lack of receiving array information.

For performance comparison, we choose the classical method (MUSIC-like) since it has low complexity computational. Also, we choose RBF since it is a typical machine learning method. As a similar method to DAE-DNN, DOA-AI can obtain satisfying performance when SNR is high. In Fig. 17, we plot the RMSE performance of the MUSIC-like [15], RBF [26], DOA-AI [34] and DAE-DNN against different SNR levels (e.g., -10dB$<$SNR<10dB) when $\rho = 0.5$. As the SNR increases, all RMSE values become stable. Considering the inaccurate array and unknown signal number in the complex scenarios, the methods of MUSIC-like and RBF all show unsatisfying DoA estimation performance in the whole SNR range. Additionally, the method in [34] can obtain a high-resolution DoA estimation in the complex scenarios and exhibit comparable performance to the proposed DAE-DNN scheme. However, it shows less robustness when SNR is lower than 0dB.

C. COMPUTATIONAL COMPLEXITY ANALYSIS

In this subsection, we focus on the computational complexity of the proposed scheme. In essence, the proposed DAE-DNN scheme mainly consists of following operations: matrix multiplications, element-wise, and convolution operations. Since the number of input units, hidden units, output units and total number of layers are n_1, n_2, ..., n_L, L, respectively, the computational complexity of the DAE-DNN scheme is $\mathcal{O}\left(\sum_{l=2}^{L} n_{l-1}n_l\right)$. To compare the proposed scheme with existing techniques, we consider the MUSIC [9] and MUSIC-based schemes [15], which we assume that the ULA contains M sensors, and there exist K-independent signals. In Table 2, we compare the computational complexity of various schemes, where N' is the number of iterations. Since the EVD computation is unnecessary, when compared to the

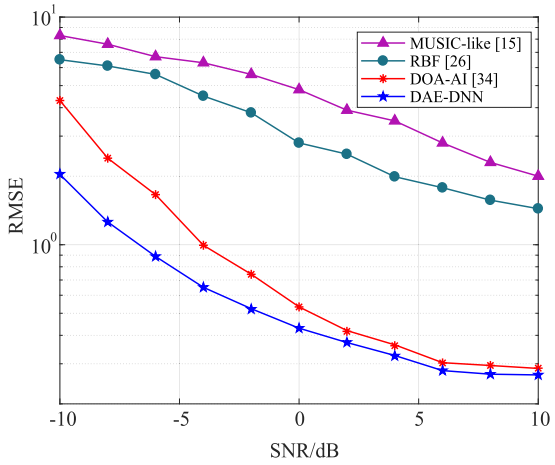

FIGURE 17. The comparison results of MUSIC-like [15], RBF [26], DOA-AI [34] and the proposed DAE-DNN scheme when $\rho = 0.5$.

TABLE 2. Comparison of the computational complexity.

Algorithm	The computational complexity
MUSIC [9]	$\mathcal{O}\left(KM^2 + (M+1)(M-K)N'\right)$
MUSIC-like [15]	$\mathcal{O}\left(N'M^2(K+2) + N'(M+1)\right)$
DAE-DNN	$\mathcal{O}\left(\sum_{l=2}^{L} n_{l-1}n_l\right)$

MUSIC-based schemes, the DAE-DNN scheme has lower computational complexity and exhibits a similar computational complexity level to RBF [26] and DOA-AI [34]. Due to the DAE phase in the proposed scheme, the computational complexity of DAE-DNN is higher than DOA-AI [34]. However, the proposed scheme achieves enhanced robustness in the low SNR regimes and provides superior DoA estimation performance when the array is inaccurate and the signal number is unknown.

VI. CONCLUSION

We proposed a novel DoA estimation scheme, DAE-DNN based on denoising autoencoder and deep neural networks for multi-signal estimation in an array inaccuracies scenario. This method is different from the usual DoA estimation methods since it takes advantage of DoA sensor array output in the time domain directly. We have shown that the performance of DAE-DNN is satisfied in the whole SNR range. Another advantage of DAE-DNN is that it shows better robustness in case of array inaccuracies exist. Previous methods are dominated by the array information to find DoAs, so the effect of array inaccuracies is not considered. The performance of the estimator can be influenced by the array inaccuracy even in the high SNR case. To reduce those degradations, we apply the training data to the properly designed DAE-DNN scheme, which learns the nonlinear mapping between the data of the receiving array and the estimation. Simulations demonstrate that the proposed DAE-DNN scheme is effective in high-resolution DoA estimation. We believe that DAE-DNN represents a new way of processing multi-signal and would be able to improve DoA estimates not only for the weak signal with strong noise but also for a complex signal having multiple harmonics. The training phase in the DAE-DNN scheme will cost some time and produce a certain amount of calculation. However, with the improvement of the hardware's computing power, the calculation problem will be effectively resolved. Therefore, we also believe that there are many interesting applications of the proposed approaches, such as mmWave channel estimation and MIMO detection.

REFERENCES

[1] L. C. Godara, "Application of antenna arrays to mobile communications. II. Beam-forming and direction-of-arrival considerations," *Proc. IEEE*, vol. 85, no. 8, pp. 1195–1245, Aug. 1997.

[2] J. W. Choi, B. Shim, Y. Ding, B. Rao, and D. I. Kim, "Compressed sensing for wireless communications: Useful tips and tricks," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1527–1550, 3rd Quart., 2017.

[3] C. Guo, L. Liang, and G. Y. Li, "Resource allocation for vehicular communications with low latency and high reliability," *IEEE Trans. Wireless Commun.*, vol. 18, no. 8, pp. 3887–3902, Aug. 2019.

[4] D. Bonacci, F. Vincent, and B. Gigleux, "Robust DoA estimation in case of multipath environment for a sense and avoid airborne radar," *IET Radar, Sonar Navigat.*, vol. 11, no. 5, pp. 797–801, May 2017.

[5] J. Wang, J. Chen, and D. Cabric, "Cramer-rao bounds for joint RSS/DoA-based primary-user localization in cognitive radio networks," *IEEE Trans. Wireless Commun.*, vol. 12, no. 3, pp. 1363–1375, Mar. 2013.

[6] J. Yang, X. Xu, D. Yin, Z. Ma, and L. Shen, "A space mapping based 0–1 linear model for onboard conflict resolution of heterogeneous unmanned aerial vehicles," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7455–7465, Aug. 2019.

[7] A. B. Gershman, V. I. Turchin, and V. A. Zverev, "Experimental results of localization of moving underwater signal by adaptive beamforming," *IEEE Trans. Signal Process.*, vol. 43, no. 10, pp. 2249–2257, Oct. 1995.

[8] J. Capon, "High-resolution frequency-wavenumber spectrum analysis," *Proc. IEEE*, vol. 57, no. 8, pp. 1408–1418, Aug. 1969.

[9] R. O. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. Antennas Propag.*, vol. AP-34, no. 3, pp. 276–280, Mar. 1986.

[10] R. Roy and T. Kailath, "Esprit-estimation of signal parameters via rotational invariance techniques," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 37, no. 7, pp. 984–995, Jul. 1989.

[11] H. L. Trees, *Detection, Estimation, and Modulation Theory, Part IV, Optimum Array Processing*. Hoboken, NJ, USA: Wiley, 2002.

[12] B. Chen and J. C. Principe, "Maximum correntropy estimation is a smoothed MAP estimation," *IEEE Signal Process. Lett.*, vol. 19, no. 8, pp. 491–494, Jun. 2012.

[13] X. Liu and G. Liao, "Direction finding and mutual coupling estimation for bistatic MIMO radar," *Signal Process.*, vol. 92, no. 2, pp. 517–522, Feb. 2012.

[14] Z. Zheng, J. Zhang, and J. Zhang, "Joint DOD and DOA estimation of bistatic MIMO radar in the presence of unknown mutual coupling," *Signal Process.*, vol. 92, no. 12, pp. 3039–3048, Dec. 2012.

[15] Y. Zhang and B. P. Ng, "MUSIC-like DOA estimation without estimating the number of sources," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1668–1676, Mar. 2010.

[16] X. P. Wang, D. D. Meng, and M. X. Huang, "Reweighted regularized sparse recovery for DOA estimation with unknown mutual coupling," *IEEE Commun. Lett.*, vol. 23, no. 2, pp. 290–293, Feb. 2019.

[17] F. Chen, D.-S. Yang, D. Wu, C. Gui, and S.-Q. Mo, "A novel direction-of-arrival estimation algorithm without knowing the source number," *IEEE Commun. Lett.*, vol. 25, no. 3, pp. 985–989, Mar. 2021.

[18] T. Grozdić and S. T. Jovičić, "Whispered speech recognition using deep denoising autoencoder and inverse filtering," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 25, no. 12, pp. 2313–2322, Dec. 2017.

[19] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Mach. Learn.*, New York, NY, USA, Jun. 2008, pp. 1096–1103.

[20] G. Li, S. Peng, C. Wang, J. Niu, and Y. Yuan, "An energy-efficient data collection scheme using denoising autoencoder in wireless sensor networks," *Tsinghua Sci. Technol.*, vol. 24, no. 1, pp. 86–96, Feb. 2019.
[21] G. Jiang, P. Xie, H. He, and J. Yan, "Wind turbine fault detection using a denoising autoencoder with temporal information," *IEEE/ASME Trans. Mechatronics*, vol. 23, no. 1, pp. 89–100, Feb. 2018.
[22] X. Wu, G. Jiang, X. Wang, P. Xie, and X. Li, "A multi-level-denoising autoencoder approach for wind turbine fault detection," *IEEE Access*, vol. 7, pp. 59376–59387, 2019.
[23] G. Kang, S. Gao, L. Yu, and D. Zhang, "Deep architecture for high-speed railway insulator surface defect detection: Denoising autoencoder with multitask learning," *IEEE Trans. Instrum. Meas.*, vol. 68, no. 8, pp. 2679–2690, Aug. 2019.
[24] T. Yu, X. Wang, and A. Shami, "UAV-enabled spatial data sampling in large-scale IoT systems using denoising autoencoder neural network," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1856–1865, Apr. 2019.
[25] Y. H. Lai, F. Chen, S.-S. Wang, X. Lu, Y. Tsao, and C.-H. Lee, "A deep denoising autoencoder approach to improving the intelligibility of vocoded speech in cochlear implant simulation," *IEEE Trans. Biomed. Eng.*, vol. 64, no. 7, pp. 1568–1578, Jul. 2017.
[26] A. H. E. Zooghby, C. G. Christodoulou, and M. Georgiopoulos, "A neural network-based smart antenna for multiple source tracking," *IEEE Trans. Antennas Propag.*, vol. 48, no. 5, pp. 768–776, May 2000.
[27] U. Challita, H. Ryden, and H. Tullberg, "When machine learning meets wireless cellular networks: Deployment, challenges, and applications," *IEEE Commun. Mag.*, vol. 58, no. 6, pp. 12–18, Jun. 2020.
[28] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Artificial neural networks-based machine learning for wireless networks: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3039–3071, Apr. 2019.
[29] A. Azari, M. Ozger, and C. Cavdar, "Risk-aware resource allocation for URLLC: Challenges and strategies with machine learning," *IEEE Commun. Mag.*, vol. 57, no. 3, pp. 42–48, Mar. 2019.
[30] H. Huang, J. Yang, H. Huang, Y. Song, and G. Gui, "Deep learning for super-resolution channel estimation and DOA estimation based massive MIMO system," *IEEE Trans. Veh. Tech.*, vol. 67, no. 9, pp. 8549–8560, Jun. 2018.
[31] W. Cui, K. Shen, and W. Yu, "Spatial deep learning for wireless scheduling," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1248–1261, Jun. 2019.
[32] W. Kim, Y. Ahn, and B. Shim, "Deep neural network-based active user detection for grant-free NOMA systems," *IEEE Trans. Commun.*, vol. 68, no. 4, pp. 2143–2155, Apr. 2020.
[33] Z. A. Eu and H.-P. Tan, "Adaptive opportunistic routing protocol for energy harvesting wireless sensor networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2012, pp. 318–322.
[34] Z.-M. Liu, C. Zhang, and P. S. Yu, "Direction-of-arrival estimation based on deep neural networks with robustness to array imperfections," *IEEE Trans. Antennas Propag.*, vol. 66, no. 12, pp. 7315–7327, Dec. 2018.
[35] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, Mar. 2010.
[36] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, and S. Ghemawat, "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," 2016, *arXiv:1603.04467*.
[37] B. Liao, Z.-G. Zhang, and S.-C. Chan, "DOA estimation and tracking of ULAs with mutual coupling," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 48, no. 1, pp. 891–905, Jan. 2012.