Direction-of-Arrival Estimation Based on Deep Neural Networks With Robustness to Array Imperfections

Zhang-Meng Liu, Chenwei Zhang, and Philip S. Yu, *Fellow, IEEE*

***Abstract*— Lacking of adaptation to various array imperfections is an open problem for most high-precision direction-of-arrival (DOA) estimation methods. Machine learning-based methods are data-driven, they do not rely on prior assumptions about array geometries, and are expected to adapt better to array imperfections when compared with model-based counterparts. This paper introduces a framework of the deep neural network to address the DOA estimation problem, so as to obtain good adaptation to array imperfections and enhanced generalization to unseen scenarios. The framework consists of a multitask autoencoder and a series of parallel multilayer classifiers. The autoencoder acts like a group of spatial filters, it decomposes the input into multiple components in different spatial subregions. These components thus have more concentrated distributions than the original input, which helps to reduce the burden of generalization for subsequent DOA estimation classifiers. The classifiers follow a one-versus-all classification guideline to determine if there are signal components near preseted directional grids, and the classification results are concatenated to reconstruct a spatial spectrum and estimate signal directions. Simulations are carried out to show that the proposed method performs satisfyingly in both generalization and imperfection adaptation.**

***Index Terms*— Array imperfection, deep neural network (DNN), direction-of-arrival (DOA) estimation, multitask autoencoder, one-versus-all classification, supervised learning.**

I. Introduction

DIRECTION-of-arrival (DOA) estimation is a widely studied problem in various areas, including wireless communications, astronomical observation, radar, and sonar [1]. A main trend of the research in DOA estimation is improving precision and superresolution, enhancing adaptation to demanding scenarios with limited snapshots, low signal-to-noise ratio (SNR), and so on [2]. Various methods have been proposed to meet these requirements, such as beamformers [3]–[5], subspace-based methods [6]–[8], sparsity-inducing methods [9]–[13], and maximum likelihood methods [14]–[16]. There have been long-lasting developments in DOA estimation performance [2]. A common feature of the above-mentioned methods is that they are parametric and formulate forward mappings from signal directions to array outputs and assume that the mappings are reversible. Based on this assumption, the array outputs can be matched by the preformulated mappings to realize direction estimation. Different matching criteria result in different methods, e.g., manifold correlation for beamformers [3]–[5], superplane fitting for subspace-based methods [6]–[8], raw array output reconstruction on overcomplete dictionaries for sparsity-inducing methods [9]–[13], and raw array output fitting for maximum likelihood methods [14]–[16]. Performances of these parametric methods depend heavily on the consistency between the two mappings, the forward mapping from signal directions to array outputs during data collection and the inverse mapping from array outputs to signal directions for DOA estimation.

Various imperfections may exist in array systems due to nonideal sensor design and manufacture, array installation and intersensor mutual interference, background radiation, and so on [17]. As a result, the forward mapping from signal directions to array outputs in practical systems is far more complicated than its backward counterparts used in parametric DOA estimation methods [18], [19]. Some of these imperfections are too complicated to be modeled accurately, and the inaccurate modeling may pose significant negative influence on the performance of DOA estimation [20], [21]. To facilitate method implementation, simplified models are established to describe the effects of various imperfections, and autocalibration processes are proposed to improve DOA estimation precision [22]–[28]. Most of the simplifications on array imperfections are made from mathematical perspectives approximately with various additional assumptions, such as uniform linearity/circularity array geometries [22]–[24], constrained sensor location errors within a particular line or plane [25], [26], and intersensor independence of gain and phase errors [27], [28].

Autocalibration DOA estimation methods have been proved to be effective via simulations in previous studies [22]–[28]. However, in the simulations, array outputs are generated based on artificially simplified models, and the formulations of

Manuscript received April 24, 2018; revised September 15, 2018; accepted September 19, 2018. Date of publication October 8, 2018; date of current version November 30, 2018. This work was supported by the National Science Foundation of China under Grant 61771477. *(Corresponding author: Zhang-Meng Liu.)*

Z.-M. Liu is with the State Key Laboratory of Complex Electromagnetic Environment Effects on Electronics and Information System, National University of Defense Technology, Changsha 410073, China (e-mail: liuzhangmeng@nudt.edu.cn).

C. Zhang and P. S. Yu are with the Department of Computer Science, The University of Illinois at Chicago, Chicago, IL 60607 USA (e-mail: czhang99@uic.edu; psyu@uic.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAP.2018.2874430

imperfections are assumed to be known beforehand with some unknown variables only. These simplifications and assumptions deviate practice with different degrees, and it is still an open question to clearly explain how the autocalibration methods behave in practical systems, especially when additional assumptions, such as linear/circular array geometries, are deviated. Moreover, the combined effects of multiple kinds of imperfections, which probably exist in practical systems, are much more difficult to be modeled precisely and calibrated automatically. Only a small number of studies have made in-depth discussions on this problem [29]–[31].

Recent research introduces machine learning techniques to solve the DOA estimation problem [32]–[36]. Although ideas alike date back to the 1990s [37]–[39], they may be resuscitated due to fast developing deep learning theory and methods [40]–[43], which have largely enhanced modeling capability than their shallow counterparts and other machine learning techniques. Methods falling in this category establish training data sets with DOA labels first, and then derive a mapping from array outputs to signal directions with existing machine learning techniques, such as radial basis function (RBF) [39] and support vector regression (SVR) [32], [33]. The derived mapping is then used on test data to estimate signal directions. These methods are data-driven and do not rely on preassumptions about array geometries and whether they are calibrated or not. They have been demonstrated to be computationally more efficient than subspace-based methods in simulations [32], [39] and perform comparably with them in experiments [33].

Performances of the RBF- and SVR-based DOA estimation methods [32], [33], [39] rely heavily on the generalization characteristic of the machine learning techniques. They perform satisfyingly when the training and test data have nearly identical distributions [44], [45]. However, in most DOA estimation problems, it is very difficult or even impossible to establish a large enough training data set to cover the distributions of all test data. That is because too many unknown parameters are present in the array output model, such as signal number, signal directions, SNR, signal waveforms, and noise samples.

In the past few years, some researchers have also introduced deep learning techniques to solve DOA estimation and source localization problems with microphone arrays [46]–[50]. Very demanding scenarios with dynamic acoustic signals [46], reverberant environments [47], [48], and wideband signals [49], [50] are considered. It is very hard to establish analytical signal propagation models in such applications, and parametric methods may encounter great difficulties in solving these problems. However, deep learning-based methods are able to reconstruct complicated propagation models based on training data sets, and then estimate source directions and locations. In spite of the methods' successes in single-signal scenarios [47], [48] or in the area of acoustic signal processing [46], [49], [50], they can hardly be used directly for general DOA estimation. That is because super-resolution of multiple temporally overlapped signals is usually required for array signal processing methods, and acoustic signals generally last for seconds and contain redundant time-frequency features. Original acoustic signals are transformed to the time-frequency domain first in the methods, and the transformed signals are then treated as inputs of deep neural networks (DNNs) [46], [49], [50]. DOA estimation of acoustic signals is finally realized in a similar way as pattern recognition of images. Nonetheless, in general, DOA estimation problems, the snapshot number is usually on scales of tens or hundreds, which is not large enough for time-frequency transformation as that for acoustic signals. Moreover, most of the deep learning methods locate sources on very coarse grids with intergrid spacings of 5° [46], [49] or even 10° [50]. Such coarse DOA estimates do not meet the precision requirements in most general DOA estimation applications.

In this paper, we propose a hierarchical framework of DNNs to deal with the general DOA estimation problem. The covariance vector of the array outputs is computed and used as the input of the DNN, and a multitask autoencoder is introduced before multilayer classifiers to decompose the signal components in the input into spatial subregions. Afterward, a series of multilayer classifiers is introduced to realize DOA estimation. Major contributions of this paper are threefold.

1) A DNN is established and an end-to-end method is proposed for general DOA estimation. It does not need to transform array outputs to the frequency domain, and therefore is different from existing deep learning-based DOA estimation methods for acoustic signals [46], [49], [50].
2) An autoencoder is introduced to preprocess original array outputs like spatial filters. This preprocessing step helps to reduce distribution divergences of the input data of DOA estimation NNs, which largely enhances the generalization of the proposed method in unseen scenarios.
3) If the DNN framework is trained using outputs of a certain array, the corresponding DOA estimation method will be robust to various kinds of imperfections in the steering vector of the array without using any prior information about them. Analyses are carried out to support such predominance, and simulation results provide further evidence.

The rest of this paper consists of five parts. Section II formulates the array output model. Section III presents a new DNN framework and interprets how it fits DOA estimation requirements. Section IV introduces training strategies of the hierarchical DNN and highlights its behavior in array imperfection adaptation. Section V carries out simulations to demonstrate the predominance of the proposed method in generalization over previous machine learning-based methods, and its predominance in imperfection adaptation over parametric methods. Section VI concludes this paper.

II. Problem Formulation

Assume that K-independent signals impinge onto an M-element array consisting of omni-directional sensors, the incident directions of the signals are $\theta_1, \ldots, \theta_K$, respectively. The waveforms of the kth signal is $s_k(t)$, and the array output is sampled at N uniquely spaced time instants

$t_1, \ldots, t_N$ to obtain snapshots $\boldsymbol{X} = [\boldsymbol{x}(t_1), \ldots, \boldsymbol{x}(t_N)]$. The array outputs are contaminated by the zero-mean Gaussian noise $\boldsymbol{v}(t)$.

In most academic works on array signal processing, various kinds of practical array imperfections are overlooked, and the mappings from signal directions to array responding functions are supposed to be deterministic and known beforehand. Denote the imperfection-free mapping by $\theta \mapsto \boldsymbol{a}(\theta)$, the array outputs can be formulated as follows:

$$\boldsymbol{x}(t_n) = \sum_{k=1}^{K} \boldsymbol{a}(\theta_k) s_k(t_n) + \boldsymbol{v}(t_n), \quad \text{for } n = 1, \ldots, N \tag{1}$$

where $\boldsymbol{a}(\theta)$ and its perturbed variant $\boldsymbol{a}(\theta, \boldsymbol{e})$ are assumed to be unitary vectors, i.e., $\|\boldsymbol{a}(\theta)\|_2 = \|\boldsymbol{a}(\theta, \boldsymbol{e})\|_2 = 1$ with $\|\boldsymbol{\alpha}\|_2$ representing the l_2-norm of vector $\boldsymbol{\alpha}$.

Various imperfections exist in practical sensor arrays. Among the imperfections, gain and phase inconsistencies, sensor position errors, and mutual coupling are widely studied ones. These imperfections cause deviations to the array responding function $\boldsymbol{a}(\theta)$, and the mapping between signal directions and array outputs in (1) does not hold any longer.

Denote the imperfection parameters by $\boldsymbol{e}$, the array outputs should be modified accordingly as follows:

$$\boldsymbol{x}(t_n) = \sum_{k=1}^{K} \boldsymbol{a}(\theta_k, \boldsymbol{e}) s_k(t_n) + \boldsymbol{v}(t_n), \quad \text{for } n = 1, \ldots, N. \tag{2}$$

Different kinds of array imperfections cast different influences on the array responding function, and it is still an open problem to figure out a precise formulation for $\boldsymbol{a}(\theta, \boldsymbol{e})$ in practical arrays. Only after moderate simplifications, analytical mappings between $(\theta, \boldsymbol{e})$ and $\boldsymbol{a}(\theta, \boldsymbol{e})$ can be formulated approximately. However, such simplifications and approximations adapt only to particular array geometries and applications, and most existing autocalibration methods can hardly be generalized to other geometries and applications when the simplifications and approximations do not hold.

III. Deep Neural Network Framework for DOA Estimation

In this section, we design a DNN framework for DOA estimation.

A. DNN Structure

The DNN framework has two parts, a multitask autoencoder that behaves as spatial filters, and a group of parallel multilayer classifiers that realize spatial spectrum construction. A sketch of the DNN structure is shown in Fig. 1.

The autoencoder denoises the DNN input and decomposes its components into P spatial subregions. If some signals located in the pth subregion impinge onto the array (possibly together with some other signals located in the other $P-1$ subregions), the output of the pth decoder equals the DNN input when the other signals are absent. If no signal impinges from this subregion, the output of the pth decoder equals zero.

A fully connected multilayer NN is designed for each subregion afterward. Each of them behaves as a multiclass classifier to determine if there are signals on a list of refined directional grids within the spatial subregion. If a signal is located on a certain grid or between two adjacent grids, the outputs of the corresponding NN node(s) will be nonzero, and the values of the node outputs indicate how close the signal direction is to this grid. The grids are preseted properly to ensure that any two signals do not coexist between two neighbor grids.

In this DNN framework, each decoder output of the autoencoder contains signals impinging from a much narrower space and has more concentrated distributions. Successive classifiers thus do not have to consider signal components located in other subregions, making the training of them much easier and enhancing their generalization to previously unseen scenarios.

B. Autoencoder for Spatial Filtering

The autoencoder first compresses the input vector to a lower dimensional one to extract the principal components in the original input, and then recover it to the original dimension via multitask decoding, with the components belonging to different subregions recovered in different decoders. The encoding–decoding process helps to reduce the influence of noise and perturbations in the autoencoder input [40].

In the sketch shown in Fig. 1, suppose that each of the encoder and decoders has L_1 layers, the vectors $\boldsymbol{c}$ in the $(L_1 - l_1)$th and the $(L_1 + l_1)$th layers have the same dimensions for $0 < l_1 \leq L_1$, and generally $|\boldsymbol{c}_{l_1}^{(p)}| < |\boldsymbol{c}_{l_1-1}^{(p)}|$, with $|\boldsymbol{c}|$ denoting the dimension of vector $\boldsymbol{c}$. Neighbor layers of the autoencoder are fully connected according to feedforward computations, that is,

$$\begin{aligned} \mathbf{net}_{l_1}^{(p)} &= \boldsymbol{U}_{l_1, l_1-1}^{(p)} \boldsymbol{c}_{l_1-1}^{(p)} + \boldsymbol{b}_{l_1}^{(p)} \\ p &= \begin{cases} 1 & \text{for } l_1 = 1, \ldots, L_1 \\ 1, \ldots, P & \text{for } l_1 = L_1 + 1, \ldots, 2L_1 \end{cases} \end{aligned} \tag{3}$$

$$\boldsymbol{c}_{l_1}^{(p)} = f_{l_1}\left[\mathbf{net}_{l_1}^{(p)}\right] \tag{4}$$

where P denotes the spatial subregion number, superscript $(\bullet)^{(p)}$ denotes the variables associated with the pth subregion and the pth autoencoder task, subscripts $(\bullet)_{l_1}$ and $(\bullet)_{l_1-1}$ denote the layer indexes, $\boldsymbol{c}_{l_1}^{(p)}$ stands for the l_1th layer output of the pth autoencoder, the superscript $(\bullet)^{(p)}$ can be ignored when $l_1 \leq L_1$, and we also define $\boldsymbol{c}_0 = \boldsymbol{r}$ as the input of the autoencoder. $\boldsymbol{U}_{l_1, l_1-1}^{(p)} \in \mathcal{R}^{|\boldsymbol{c}_{l_1}^{(p)}| \times |\boldsymbol{c}_{l_1-1}^{(p)}|}$ is the weight matrix from the $(l_1 - 1)$th layer to the l_1th layer of the pth task, $\boldsymbol{b}_{l_1}^{(p)} \in \mathcal{R}^{|\boldsymbol{c}_{l_1}^{(p)}| \times 1}$ is the additive bias vector in the l_1th layer, $f_{l_1}[\bullet]$ represents the elementwise activation function in the l_1th layer.

The multitask autoencoder aims at decomposing the inputs into P spatial subregions. A straightforward strategy for defining the subregions is choosing $P+1$ particular directions $\theta^{(0)} < \theta^{(1)} < \cdots < \theta^{(P)}$, which satisfy $\theta^{(1)} - \theta^{(0)} = \theta^{(2)} - \theta^{(1)} = \cdots = \theta^{(P)} - \theta^{(P-1)}$ and $[\theta^{(0)}, \theta^{(P)})$ spans the potential scope of the incident signals. If a signal component impinging from the pth subregion is used as the input to the autoencoder, the output of the pth decoder, which is also

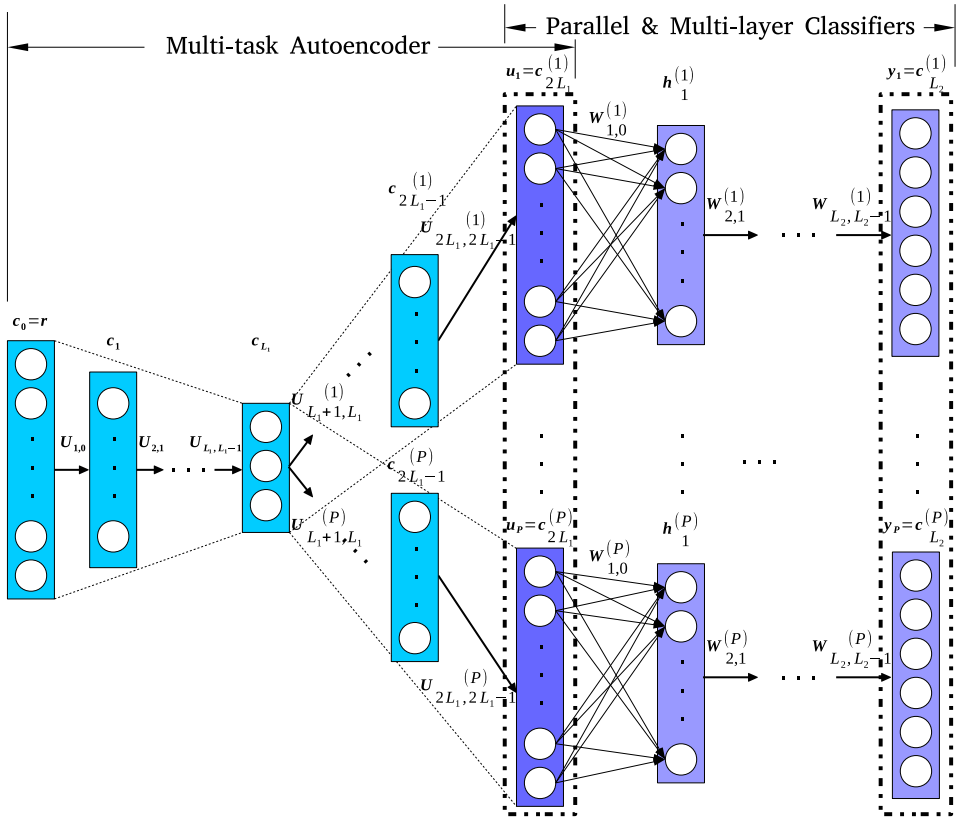


Fig. 1. Structure of proposed DNN for DOA estimation. The network consists of two parts, one is a multitask autoencoder for spatial filtering and the other is a fully connected multilayer NN for spatial spectrum estimation.

denoted as $\boldsymbol{u}_p = \boldsymbol{c}_{2L_1}^{(p)}$ is expected to be equal to the input $\boldsymbol{r}$, while the other decoder outputs equals zero.[1]

There are many ways to design spatial filterlike autoencoders, so that $F^{(p)}(\boldsymbol{r}) = \boldsymbol{r}$ if the signal direction $\theta \in [\theta^{(p-1)}, \theta^{(p)})$ and $F^{(p)}(\boldsymbol{r}) = \boldsymbol{0}$ otherwise approximately, where $F^{(p)}(\bullet)$ is the overall function of the pth autoencoder task. Furthermore, an additional requirement is necessary in the autoencoder structure for the DOA estimation application, i.e., $F^{(p)}(\boldsymbol{r}_1 + \boldsymbol{r}_2) = F^{(p)}(\boldsymbol{r}_1) + F^{(p)}(\boldsymbol{r}_2)$. The additive property is required, because when multiple signals located in different subregions impinge onto the array simultaneously, the autoencoder should be able to decompose the input vector to different decoder outputs. In order to make the autoencoder additive, the activation functions $f_{l_1}^{(p)}[\bullet]$ should be linear, therefore, we replace it with the unit function instead, that is,

$$\boldsymbol{c}_{l_1}^{(p)} = \mathbf{net}_{l}^{(p)}. \tag{5}$$

As there is no nonlinear transformations in the autoencoder hidden layers, each of the multilayer encoding and decoding processes can be simplified to a single layer, i.e., $L_1 = 1$, and the autoencoder can be rewritten as

$$\boldsymbol{c}_1 = \boldsymbol{U}_{1,0}\boldsymbol{r} + \boldsymbol{b}_1 \tag{6}$$

$$\boldsymbol{u}_p = \boldsymbol{U}_{2,1}^{(p)}\boldsymbol{c}_1 + \boldsymbol{b}_2^{(p)}, \quad p = 1, \ldots, P. \tag{7}$$

[1]The denoising ability of the autoencoder can be further enhanced by using the perturbation-free counterpart of the input as the expected output of the corresponding decoder. However, as it is not an easy task to collect perturbation-free counterparts of the training data set, we use the original noisy inputs as the outputs directly in this paper.

C. Multilayer Classifiers for Spectrum Estimation

Previous RBF-based [39] and SVR-based [32], [33] machine learning methods for DOA estimation fix their output node number as the incident signal number, which is assumed to be known beforehand. The trained models do not work if the incident signal number changes. Therefore, a set of models should be trained for each case of incident signal number. However, these models can hardly be integrated to deal with the DOA estimation problems when the signal number is not known beforehand.

A more flexible way to enhance the generalization of the methods to unknown signal number is to use a list of one-versus-all classifiers instead. In the DOA estimation problem, each node of the classifier output stands for a preseted directional grid, and the final output value on the node represents the probability of a signal locating in the neighborhood of the grid. DOA of signals impinging from off-grid directions can be estimated via interpolation between two adjacent grids.

As shown in Fig. 1, there are P parallel classifiers in total, the pth classifier takes the output of the pth decoder as input and analyzes the components of the input on the preseted grids in the pth spatial subregion. There are no mutual connections between different classifiers. The computations of the classifiers are feedforward

$$\mathbf{net}_{l_2}^{(p)} = \boldsymbol{W}_{l_2,l_2-1}^{(p)}\boldsymbol{h}_{l_2-1}^{(p)} + \boldsymbol{q}_{l_2}^{(p)} \quad p = 1, \ldots, P; l_2 = 1, \ldots, L_2 \tag{8}$$

$$\boldsymbol{h}_{l_2}^{(p)} = g_{l_2}\left[\mathbf{net}_{l_2}^{(p)}\right] \tag{9}$$

where $\boldsymbol{h}_{l_2}^{(p)}$ is the output vector in the l_2th layer of the pth classifier, with $\boldsymbol{h}_0^{(p)} = \boldsymbol{u}_p$ and $\boldsymbol{h}_{L_2}^{(p)} = \boldsymbol{y}_p$; $\boldsymbol{W}_{l_2,l_2-1}^{(p)} \in \mathcal{R}^{|\boldsymbol{h}_{l_2}^{(p)}|\times|\boldsymbol{h}_{l_2-1}^{(p)}|}$ is the fully connected feed-forward weight matrix between the (l_2-1)th layer and the l_2th layer, $\boldsymbol{q}_{l_2}^{(p)}$ is the additive bias vector on the l_2th layer; and $g_{l_2}[\bullet]$ is an elementwise activation function for the inputs of the l_2th layer nodes.

After obtaining all the outputs of the P classifiers in parallel based on the P decoder outputs, the spatial spectrum associated with DNN input $\boldsymbol{r}$ can be estimated by concatenating the P outputs in order, that is,

$$\boldsymbol{y} = \left[\boldsymbol{y}_1^T, \ldots, \boldsymbol{y}_P^T\right]^T. \tag{10}$$

There are totally $|\boldsymbol{y}|$ one-versus-all classifiers in this part of the DNN. In order to realize DOA estimation based on the spectrum estimate, only the grid nodes close to the true signal directions are expected to have positive values in $\boldsymbol{y}$, while all the others have zero values.

IV. DNN-Based Direction-of-Arrival Estimation

Besides the framework described in Section III, training data set structure and training strategy are two other factors that play important roles in the performance of the DNN-based DOA estimator. As the autoencoder and the parallel classifiers implement different functions and work separately during DOA estimation, and training deep NN as a whole gets trapped in undesirable local minima more easily [51], we propose to train the two parts of the DNN in separate procedures.

In order to reduce the variability of the DNN input, which is influenced significantly by uncertain signal waveforms, we follow the guidelines of the RBF- and SVR-based methods to compute the array covariance matrix [32], [33], [39] and reformulate the off-diagonal upper right matrix elements as an input vector to the DNN

$$\overline{\boldsymbol{r}} = [R_{1,2}, R_{1,3}, \ldots, R_{1,M}, R_{2,3}, \ldots, R_{2,M}, \ldots, R_{M-1,M}]^T \in \mathcal{C}^{(M-1)M/2\times 1} \tag{11}$$

$$\boldsymbol{r} = [Real\{\overline{\boldsymbol{r}}^T\}, Imag\{\overline{\boldsymbol{r}}^T\}]^T / \|\overline{\boldsymbol{r}}\|_2 \tag{12}$$

where R_{m_1,m_2} is the (m_1, m_2)th element of the covariance matrix $\hat{\boldsymbol{R}} = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x}(t_n)\boldsymbol{x}^T(t_n)$, and $Real\{\bullet\}$ and $Imag\{\bullet\}$ represent the real and image parts of a complex-valued entity, respectively. The input vector is slightly different from that of the RBF- and SVR-based methods [32], [33], [39]. The diagonal covariance elements are not included due to the unknown noise variance, and the lower left elements are abandoned because they are conjugate replicas of the upper right ones.

A. Autoencoder Training

As the autoencoder is designed to be linear and has additive property, its performance of spatial filtering can be guaranteed if it performs well in the single-signal scenarios. Therefore, we construct the training data set using the $\boldsymbol{r}$'s corresponding to one-signal scenarios, with the signal direction varying from $\theta^{(0)}$ to $\theta^{(P)}$. A straightforward selection of the signal directions is the equally spaced spectrum grids of the classifier outputs, which are denoted by $\vartheta_1, \vartheta_2, \ldots, \vartheta_I$. I is supposed to be dividable by P with $I/P = I_0$ being an integer.

If the covariance vector $\boldsymbol{r}(\vartheta_i)$ corresponding to the signal from direction ϑ_i is inputted to the autoencoder, the output of the p_ith ($p_i = \lceil i/I_0 \rceil$ with $\lceil\alpha\rceil$ equaling the smallest integer not smaller than α) decoder is expeced to be $\boldsymbol{r}(\vartheta_i)$, while the outputs of the other $P-1$ decoders expected to be $\boldsymbol{0}_{\kappa\times 1}$ where $\kappa = |\boldsymbol{r}|$. By concatenating the outputs of all the P decoders, the expected output of the whole autoencoder can be obtained as

$$\begin{aligned}\boldsymbol{u} &= \left[\boldsymbol{u}_1^T, \ldots, \boldsymbol{u}_P^T\right]^T \\ &= \left[\underbrace{\boldsymbol{0}_{\kappa\times 1}^T, \ldots, \boldsymbol{0}_{\kappa\times 1}^T}_{p-1}, \boldsymbol{r}^T(\vartheta_i), \underbrace{\boldsymbol{0}_{\kappa\times 1}^T, \ldots, \boldsymbol{0}_{\kappa\times 1}^T}_{P-p}\right]^T.\end{aligned} \tag{13}$$

When ϑ_i varies from $\theta^{(0)}$ to $\theta^{(P)}$, the p_is corresponding to $i = 1, \ldots, I$ are $\underbrace{1, \ldots, 1}_{I_0}, \underbrace{2, \ldots, 2}_{I_0}, \ldots, \underbrace{P, \ldots, P}_{I_0}$. Denote the autoencoder label corresponding to data $\boldsymbol{r}(\vartheta_i)$ by $\boldsymbol{u}(\vartheta_i)$, then the data set for autoencoder training is

$$\Gamma^{(1)} = [\boldsymbol{r}(\vartheta_1), \ldots, \boldsymbol{r}(\vartheta_I)] \tag{14}$$

and the columnwise label set associated with the data set is

$$\begin{aligned}\Psi^{(1)} &= [\boldsymbol{u}(\vartheta_1), \boldsymbol{u}(\vartheta_2), \ldots, \boldsymbol{u}(\vartheta_I)] \\ &= \begin{bmatrix} \Phi_1 & \boldsymbol{0}_{\kappa\times I_0} & \boldsymbol{0}_{\kappa\times I_0} & \boldsymbol{0}_{\kappa\times I_0} \\ \boldsymbol{0}_{\kappa\times I_0} & \Phi_2 & \boldsymbol{0}_{\kappa\times I_0} & \boldsymbol{0}_{\kappa\times I_0} \\ \boldsymbol{0}_{\kappa\times I_0} & \boldsymbol{0}_{\kappa\times I_0} & \ddots & \boldsymbol{0}_{\kappa\times I_0} \\ \boldsymbol{0}_{\kappa\times I_0} & \boldsymbol{0}_{\kappa\times I_0} & \boldsymbol{0}_{\kappa\times I_0} & \Phi_P \end{bmatrix}\end{aligned} \tag{15}$$

where superscript $(\bullet)^{(1)}$ is used for variables related to the autoencoder, and superscript $(\bullet)^{(2)}$ will be used for the classifiers, and

$$\Phi_p = [\boldsymbol{r}(\vartheta_{(p-1)I_0+1}), \ldots, \boldsymbol{r}(\vartheta_{pI_0})]. \tag{16}$$

The data-label pair of $(\Gamma^{(1)}, \Psi^{(1)})$ is then used as input and expected output to train the autoencoder. The squared l_2-norm distance between the actual output and the expected one is used as the loss function, that is,

$$\boldsymbol{\epsilon}^{(1)}(\vartheta_i) = \frac{1}{2}\|\tilde{\boldsymbol{u}}(\vartheta_i)\|_2^2 \tag{17}$$

where

$$\tilde{\boldsymbol{u}}(\vartheta_i) = \boldsymbol{u}(\vartheta_i) - \hat{\boldsymbol{u}}(\vartheta_i) \tag{18}$$

and $\hat{\boldsymbol{u}}(\vartheta_i)$ is the actual output of the autoencoder when $\boldsymbol{r}(\vartheta_i)$ is inputted.

The weight matrices and bias vectors are then updated based on the backpropagated gradients of the loss function with respect to the variables. The gradients can be computed via straightforward mathematical derivations and are listed in the

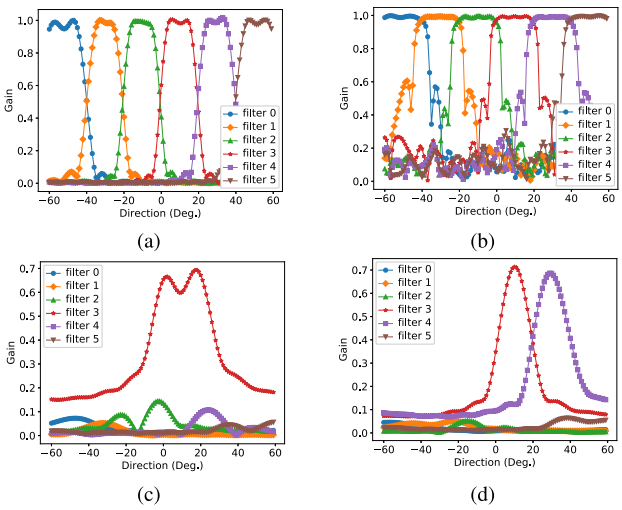


Fig. 2. Performance of multitask autoencoder for spatial filtering. (a) Gain responses. (b) Phase responses. (c) Filter outputs of two signals in the same subregion ($\theta_1 = 5°$, $\theta_2 = 15°$). (d) Filter outputs of two signals in different subregions ($\theta_1 = 10°$, $\theta_2 = 30°$).

following:

$$\frac{\partial \boldsymbol{\epsilon}^{(1)}(\vartheta_i)}{\partial [\boldsymbol{U}_{2,1}]_{i_1,i_2}} = [\tilde{\boldsymbol{u}}(\vartheta_i)]_{i_1}[\boldsymbol{U}_{1,0}\boldsymbol{r}(\vartheta_i) + \boldsymbol{b}_1]_{i_2} \tag{19}$$

$$\frac{\partial \boldsymbol{\epsilon}^{(1)}(\vartheta_i)}{\partial [\boldsymbol{U}_{1,0}]_{i_1,i_2}} = \tilde{\boldsymbol{u}}^T(\vartheta_i)[\boldsymbol{U}_{2,1}]_{:,i_1}[\boldsymbol{r}(\vartheta_i)]_{i_2} \tag{20}$$

$$\frac{\partial \boldsymbol{\epsilon}^{(1)}(\vartheta_i)}{\partial [\boldsymbol{b}_1]_l} = \tilde{\boldsymbol{u}}^T(\vartheta_i)[\boldsymbol{U}_{2,1}]_{:,l} \tag{21}$$

$$\frac{\partial \boldsymbol{\epsilon}^{(1)}(\vartheta_i)}{\partial [\boldsymbol{b}_2]_l} = [\tilde{\boldsymbol{u}}(\vartheta_i)]_l \tag{22}$$

where $[\boldsymbol{\alpha}]_l$ represents the lth element of vector $\boldsymbol{\alpha}$, and $[\boldsymbol{A}]_{i_1,i_2}$ represents the (i_1, i_2)th element of matrix $\boldsymbol{A}$. The variants are then updated iteratively as

$$\alpha_{\text{new}} = \alpha_{\text{old}} + \mu_1 \frac{\partial \boldsymbol{\epsilon}^{(1)}(\vartheta_i)}{\partial \alpha} \tag{23}$$

where α can be any element of matrices $\boldsymbol{U}_{1,0}$, $\boldsymbol{U}_{2,1}$ or vectors $\boldsymbol{b}_1$, $\boldsymbol{b}_2$, μ_1 is the learning rate, and α_{old} and α_{new} denote the values of the variables before and after current update, respectively.

Fig. 2(a)–(d) show the spatial responses of six trained filters, i.e., $P = 6$, in the spatial scope of $[-60°, 60°)$.[2] Fig. 2(a) shows the spatial gains of the filters, that is,

$$g_a^{(p)} = |\overline{\boldsymbol{r}}^H(\vartheta_i)\overline{\boldsymbol{u}}_p|, \quad \text{for } p = 1, \ldots, P; \ i = 1, \ldots, I \tag{24}$$

where superscript $(\bullet)^H$ represents the conjugate transpose of matrices and vectors, $\overline{\boldsymbol{u}}_p$ is the complex-valued variant of $\boldsymbol{u}_p$ by taking the first half of the vector as the real part and the second half as imaginary part. Fig. 2(b) shows the phase responses of the filters, that is,[3]

$$g_b^{(p)} = \frac{|\overline{\boldsymbol{r}}^H(\vartheta_i)\overline{\boldsymbol{u}}_p|}{\|\overline{\boldsymbol{r}}(\vartheta_i)\|_2 \|\overline{\boldsymbol{u}}_p\|_2}, \quad \text{for } p = 1, \ldots, P; \ i = 1, \ldots, I. \tag{25}$$

The gain $g_b^{(p)}$ describes how much the phase shifts between the elements of $\overline{\boldsymbol{r}}(\vartheta_i)$ are kept unchanged after being filtered by the autoencoder, and $g_a^{(p)}$ also combines the effect of amplitude attenuation of different filters. It can be seen in Fig. 2(a) and (b) that the consistency between the phase shifts of the autoencoder input and its expected output is satisfying within the divided subregions, and the gain attenuates fast near the edges of the subregions. The outputs of the decoders other than the assigned one are minor in amplitude and inconsistent with the input in phase.

We then input vectors $\boldsymbol{r}$ corresponding to two-signal scenarios to the autoencoder to test the additive property of the autoencoder. First, two signals impinge from directions $\theta_1 = 5°$ and $\theta_2 = 15°$ in the same subregion $[0°, 20°)$, the gain response $g_a^{(p)}$ of the six decoder outputs are shown in Fig. 2(c). The spatial gain response of the assigned decoder looks very like basic beamformers, while that of the other decoders are very small. Second, we set the directions of the two signals to be $\theta_1 = 10°$ and $\theta_2 = 30°$, so that they locate in two adjacent subregions, then $g_a^{(p)}$ of the six decoder outputs is shown in Fig. 2(d). The two-signal components are well separated by the filters, and the irrelevant filters' outputs are negligible.

B. Parallel Classifier Training

The P parallel classifiers take the decoders' outputs as inputs and estimate the spatial spectrum in the corresponding subregions separately. When compared with $\boldsymbol{r}$, each of the autoencoder outputs $\boldsymbol{u}_p$ $(p = 1, \ldots, P)$ contains signals located only in a much smaller subregion. As spatially closer signal components in the array outputs generally have more similar steering vectors, $\boldsymbol{u}_p$s thus have much more concentrated distributions than $\boldsymbol{r}$. In the layer of parallel classifiers, we use more than one hidden layers and add nonlinear activations to enhance expressivity, so as to realize refined DOA estimation. In order to retain the polarity of the inputs at each layer of the classifiers, we use an elementwise hyperbolic tangent function for activation, that is,

$$tanh(\boldsymbol{\alpha}) = [tanh(\alpha_1), tanh(\alpha_2), \ldots, tanh(\alpha_{-1})]^T \tag{26}$$

$$tanh(\alpha) = \frac{e^{\alpha} - e^{-\alpha}}{e^{\alpha} + e^{-\alpha}} \tag{27}$$

where α_{-1} is the last element of $\boldsymbol{\alpha}$.

When the training of the autoencoder has been completed, we keep its weights and biases fixed, then the input vector $\boldsymbol{r}$ and the reconstructed spectrum $\boldsymbol{y}$ form a new end-to-end NN framework. The weights and biases of the classification NNs should be trained to detect and estimate directions of coexisting signals in different subregions. To achieve this goal, we recollect another training data set with two simultaneous signals.[3]

[2]Detailed descriptions of the simulation settings are delayed to Section V.

[3]This data set can also be established by adding two $\boldsymbol{r}(\theta)$s in the training data set of the multitask autoencoder.

We choose several intersignal angles $\boldsymbol{\Delta} = \{\Delta_j\}_{j=1}^{J}$ and form input vectors $\boldsymbol{r}(\theta, \Delta_j)$ corresponding to two incident signals from directions θ and $\theta + \Delta_j$, where $\theta^{(0)} \leq \theta < \theta^{(P)} - \Delta_j$ and $j = 1, \ldots, J$. The expected classifier output corresponding to input $\boldsymbol{r}(\theta, \Delta_j)$ is $\boldsymbol{y}(\theta, \Delta_j)$ with

$$[\boldsymbol{y}(\theta, \Delta_j)]_l = \begin{cases} \dfrac{\overline{\theta} - \vartheta_{l-1}}{\vartheta_l - \vartheta_{l-1}}, & \vartheta_{l-1} \leq \overline{\theta} < \vartheta_l,\ \overline{\theta} \in \{\theta, \theta + \Delta_j\} \\ \dfrac{\vartheta_{l+1} - \overline{\theta}}{\vartheta_{l+1} - \vartheta_l}, & \vartheta_l \leq \overline{\theta} < \vartheta_{l+1},\ \overline{\theta} \in \{\theta, \theta + \Delta_j\} \\ 0, & \text{otherwise.} \end{cases} \tag{28}$$

That is, the reconstructed spectrum is expected to have nonzero positive values only on the grids adjacent to the true signal directions, and the direction of each signal can be estimated precisely via linear amplitude interpolation between the two adjacent grids.

The training data set of the classifiers can be written as

$$\Gamma^{(2)} = \left[\Gamma_1^{(2)}, \ldots, \Gamma_J^{(2)}\right] \tag{29}$$

where

$$\Gamma_j^{(2)} = [\boldsymbol{r}(\vartheta_1, \Delta_j), \ldots, \boldsymbol{r}(\vartheta_I - \Delta_j, \Delta_j)] \tag{30}$$

and the associated label set is

$$\Psi^{(2)} = \left[\Psi_1^{(2)}, \ldots, \Psi_J^{(2)}\right] \tag{31}$$

where

$$\Psi_j^{(2)} = [\boldsymbol{y}(\vartheta_1, \Delta_j), \ldots, \boldsymbol{y}(\vartheta_I - \Delta_j, \Delta_j)]. \tag{32}$$

During the training process, the reconstruction error of the spatial spectrum is backpropagated to update the NN parameters of the parallel classifiers. Denote the expected and actual classifier outputs corresponding to $\boldsymbol{r}(\theta, \Delta)$ by $\boldsymbol{y}(\theta, \Delta)$ and $\hat{\boldsymbol{y}}(\theta, \Delta)$, respectively, and also denote the reconstruction error by

$$\tilde{\boldsymbol{y}}(\theta, \Delta) = \hat{\boldsymbol{y}}(\theta, \Delta) - \boldsymbol{y}(\theta, \Delta). \tag{33}$$

The loss function for the classifiers is taken as the squared l_2-norm of the spectrum reconstruction error, that is,

$$\boldsymbol{\epsilon}^{(2)}(\theta, \Delta) = \frac{1}{2}\|\tilde{\boldsymbol{y}}(\theta, \Delta)\|_2^2. \tag{34}$$

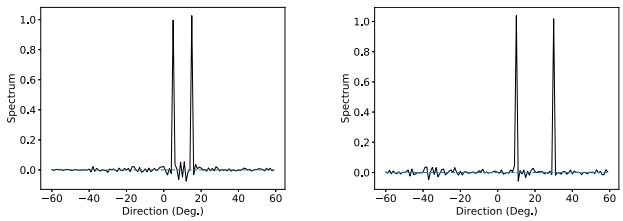
The gradients of the loss function with respect to the classifier variants can be derived via straightforward mathematical analyses. We skip details of the derivations here and refer readers interested in them to previous studies such as [43]. Most deep learning platforms, such as TensorFlow [52], also provide callable instructions for computing the gradients automatically.

The elements of the weight matrices and bias vectors are then updated using their gradients as follows:

$$\alpha_{\text{new}} = \alpha_{\text{old}} + \mu_2 \frac{\partial \boldsymbol{\epsilon}^{(2)}(\theta, \Delta)}{\partial \alpha} \tag{35}$$

where μ_2 is the learning rate.

After training the classifiers with the settings detailed in Section V, we reinput the array covariance vectors $\boldsymbol{r}(\theta = 5^\circ, \Delta = 10^\circ)$ and $\boldsymbol{r}(\theta = 10^\circ, \Delta = 20^\circ)$ associated with Fig. 2(c) and (d) to the whole DNN and get the reconstructed spectra shown in Fig. 3(a) and (b). The two signals in both scenarios are well separated, no matter they impinge from the same or different spatial subregions. There are only slight perturbations on the spectrum grids without incident signals. The directions of the signals can finally be estimated based on the estimated spectrum via linear interpolation within the spectrum peaks.



(a) (b)

Fig. 3. Reconstructed spatial spectrum of two signals. (a) $\theta_1 = 5^\circ$ and $\theta_2 = 15^\circ$. (b) $\theta_1 = 10^\circ$ and $\theta_2 = 30^\circ$.

C. Adaptation to Array Imperfections

As has been discussed in Section II, data-driven DOA estimation methods are expected to have built-in adaptations to typical array imperfections, such as gain and phase inconsistence [27], [28], sensor position error [25], [26], and mutual coupling [22]–[24]. We validate such property of the proposed method in this section.

Suppose that the responding function of the array is perturbed by a particular kind of or combined imperfections with parameters $\boldsymbol{e}$, and the mapping from signal directions to covariance vectors is $\theta \overset{\boldsymbol{e}}{\mapsto} \boldsymbol{r}_{\boldsymbol{e}}(\theta)$. The DNN is assumed to have no prior information about the imperfections and the perturbed array responding function. When the perturbed vector $\boldsymbol{r}_{\boldsymbol{e}}(\vartheta_i)$ with $\lceil i/I_0 \rceil = p$ is inputted to the autoencoder, the associated label vector is

$$\begin{aligned} \boldsymbol{u} &= \left[\boldsymbol{u}_1^T, \ldots, \boldsymbol{u}_P^T\right]^T \\ &= \left[\underbrace{\boldsymbol{0}_{\kappa\times 1}^T, \ldots, \boldsymbol{0}_{\kappa\times 1}^T}_{p-1}, \boldsymbol{r}_{\boldsymbol{e}}^T(\vartheta_i), \underbrace{\boldsymbol{0}_{\kappa\times 1}^T, \ldots, \boldsymbol{0}_{\kappa\times 1}^T}_{P-p}\right]^T. \end{aligned} \tag{36}$$

That is to say, the vector is filtered into the pth filter even in the presence of array imperfections.

Afterward, the decoder outputs are inputted to the classifiers. As the component corresponding to the signal from direction ϑ_i is embedded in the output of the pth decoder, it will be processed by the pth classifier. The associated spectrum label contains a spectrum peak on the one or two grids closest to ϑ_i, which can be interpolated to obtain a DOA estimate of ϑ_i exactly. Therefore, the whole DNN (the autoencoder together with the parallel classifiers) actually forms an inverse mapping of $\boldsymbol{r}_{\boldsymbol{e}}(\theta) \overset{\boldsymbol{e}}{\mapsto} \theta$, no matter which kinds of imperfections are present and how they perturbs the array responding function. The derived inverse mapping from DNN inputs to signal directions with the effect of array

imperfections embedded in it also adapts to test data and is expected to obtain correct DOA estimates in spite of array imperfections.

V. Simulations and Analyses

This section carries out simulations to show the predominance of the proposed method over state-of-the-art machine learning-based DOA estimation method [32], [33] in generalization, and also its predominance over the most common parametric method [6] in imperfection adaptation. The simulations are implemented on TensorFlow [52], and the gradients are computed using its embedded tools directly. The more recently proposed methods [46]–[50] are not chosen as baselines, because some of them work only in single-source scenarios [47], [48], and the others take time-frequency representations of incident signals as inputs [46], [49], [50]. They do not adapt to the considered multisource scenarios with only a few hundreds of snapshots in the simulations. Autocalibration techniques [22]–[28] are not introduced in parametric methods, as no prior information (neither formulation nor value) about array imperfections is assumed to be known beforehand. Such settings help to show the robustness of different methods to uncalibrated arrays and make fairer performance comparisons. Also, this simulation setting prevents usages of calibration techniques designed for certain kinds of imperfections with preassumed formulations [22]–[28].

A. Simulation Settings

In the following simulations, we use a 10-element uniform linear array (ULA) to estimate directions of signals impinging from the spatial scope of $[-60^\circ, 60^\circ)$, i.e., $M = 10$, $\theta^{(0)} = -60^\circ$, and $\theta^{(P)} = 60^\circ$. The interelement spacing of the ULA is half-wavelength, and the potential space is divided into $P = 6$ subregions with equal spatial scopes. The spatial spectrum is constructed with a grid of 1°, thus there are $I = 120$ grids in total with $\vartheta_1 = -60^\circ, \vartheta_2 = -59^\circ, \ldots, \vartheta_I = 59^\circ$, and each spatial subregion has $I_0 = 20$ grids. The covariance vectors $\boldsymbol{r}$ in the training data sets of both the autoencoder and the classifiers, and also, the vectors in the test data sets are obtained from $N = 400$ snapshots.

For the training of the autoencoder, the $[-60^\circ, 60^\circ)$ space is also sampled with an interval of 1° to obtain a direction set with $\vartheta_1 = -60^\circ, \vartheta_2 = -59^\circ, \ldots, \vartheta_I = 59^\circ$ and compute the covariance vectors and associated labels according to (14) and (15). At each of the directional grids, only one group of snapshots is collected to compute one covariance vector. The SNR of the snapshots is 10 dB. The minibatch training strategy [53] is followed with a batch size of 32 and learning rate of $\mu_1 = 0.001$, and 1000 epochs are taken for the training with the data set shuffled in each epoch. The size of the input layer is $\kappa = M(M-1)/2 = 45$, and that of the hidden and output layers are $\lfloor 45/2 \rfloor = 22$ and $\kappa I = 45 \times 6$, respectively.

The autoencoder parameters are fixed after the training process, and another data set is collected in the two-signal scenarios to train the classifiers. The intersignal angle Δ is sampled from the set of $\{2^\circ, 4^\circ, \ldots, 40^\circ\}$, which covers scenarios from very close signals to signals separated by twice the width of a subregion. Then, the direction of the first signal (denoted by θ) is sampled with an interval of 1° from -60° to $60^\circ - \Delta$, and the direction of the second signal is $\theta + \Delta$. The SNR of both signals is 10 dB, and 10 groups of snapshots are collected for each direction setting with random noise. Finally, $(118 + 116 + \cdots + 80) \times 10 = 19800$ covariance vectors are collected in the data set. The vectors are used for training with the minibatch size of 32 and learning rate of $\mu_2 = 0.001$, and the order of the vectors is shuffled during each of the 300 training epochs. The number of the hidden layers is chosen to be $L_2 - 1 = 2$ as a tradeoff between the expressivity power (improves with deeper networks [41]) and undertraining risk (aggravates with more network parameters [51]) of the classifiers, and the sizes of the hidden and output layers in each classifier are $\lfloor 2/3 \times \kappa \rfloor = 30$, $\lfloor 4/9 \times \kappa \rfloor = 20$ and $I_0 = 20$, respectively. All the weights and biases of the DNN are randomly initialized according to a uniform distribution between -0.1 and 0.1.

Three typical kinds of array imperfections are considered in the simulations, including gain and phase inconsistence, sensor position error, and intersensor mutual coupling. The imperfections may be very complicated to be modeled with concise mathematical models, so we use simplified models to facilitate simulations in this paper. The gain biases of the array sensors are

$$\boldsymbol{e}_{\text{gain}} = \rho \times [0, \underbrace{0.2, \ldots, 0.2}_{5}, \underbrace{-0.2, \ldots, -0.2}_{4}]^T \tag{37}$$

where the parameter $\rho \in [0, 1]$ is introduced to control the strength of the imperfections.The phase biases are

$$\boldsymbol{e}_{\text{phase}} = \rho \times [0, \underbrace{-30^\circ, \ldots, -30^\circ}_{5}, \underbrace{30^\circ, \ldots, 30^\circ}_{4}]^T. \tag{38}$$

The position biases are

$$\boldsymbol{e}_{\text{pos}} = \rho \times [0, \underbrace{-0.2, \ldots, -0.2}_{5}, \underbrace{0.2, \ldots, 0.2}_{4}]^T \times d \tag{39}$$

where d is the intersensor spacing of the ULA. The mutual coupling coefficient vector is

$$\boldsymbol{e}_{\text{mc}} = \rho \times [0, \gamma^1, \ldots, \gamma^{M-1}]^T \tag{40}$$

where $\gamma = 0.3e^{j60^\circ}$ is the mutual coupling coefficient between adjacent sensors.

By specializing ρ, the array imperfections will be determined, and the perturbed array responding function is rewritten as follows:

$$\begin{aligned}\boldsymbol{a}(\theta, \boldsymbol{e}) = {} & (\boldsymbol{I}_M + \delta_{\text{mc}}\boldsymbol{E}_{\text{mc}}) \times (\boldsymbol{I}_M + \text{Diag}(\delta_{\text{gain}}\boldsymbol{e}_{\text{gain}})) \\ & \times \text{Diag}(\exp(j\delta_{\text{phase}}\boldsymbol{e}_{\text{phase}})) \times \boldsymbol{a}(\theta, \delta_{\text{pos}}\boldsymbol{e}_{\text{pos}})\end{aligned} \tag{41}$$

where $\delta_{(\bullet)}$ is used to indicate whether a certain kind of imperfection exists, $\boldsymbol{I}_M$ is the $M \times M$ unitary matrix, $\text{Diag}(\bullet)$ forms diagonal matrices with the given vector on the diagonal, $\boldsymbol{E}_{\text{mc}}$ is a toeplitz matrix with parameter vector $\boldsymbol{e}_{\text{mc}}$ [22], and $\boldsymbol{a}(\theta, \delta_{\text{pos}}\boldsymbol{e}_{\text{pos}})$ is the actual array responding vector corresponding to the signal from direction θ when position error $\boldsymbol{e}_{\text{pos}}$ is embedded in the array geometry.

The array responding function given in (41) has been largely simplified when compared with its counterpart in

practical applications, which can be measured more precisely with computational electromagnetic methods, such as [54]–[56]. The array imperfection formulations in (37)–(40) can also be modeled more precisely following previous studies, such as [57]–[60] for mutual coupling. However, we use the simplified formulations mainly to facilitate simulation, and we believe that these simplifications are reasonable for performance comparison. That is because the proposed machine-learning method does not make use of any prior information about the array imperfections and steering vectors. The proposed end-to-end training and testing strategies can be generalized straightforwardly to other array geometries and imperfections, no matter how the antennas are fed and how much the array steering vector has been biased by imperfections.

In Figs. 4–7, we use dashed lines to represent true values of signal directions, and dots with triangular or circle markers to represent their estimates and (statistical) estimation errors.

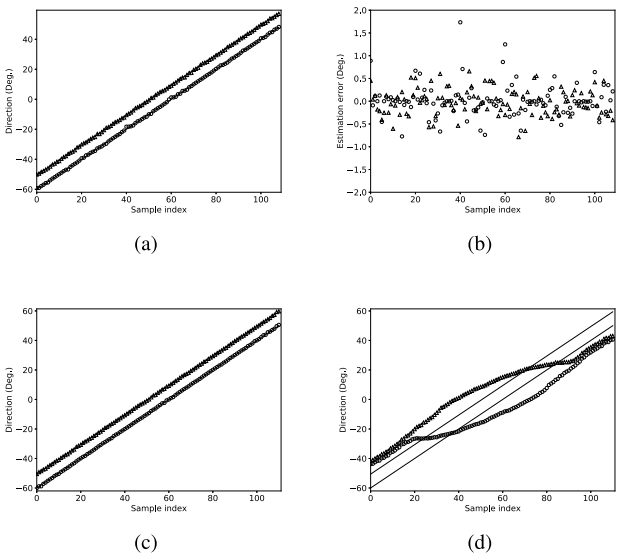


Fig. 4. DOA estimation performance of off-grid signals. (a) DNN-based DOA estimates. (b) DNN-based DOA estimation errors. (c) SVR-based DOA estimates with noise-free training data. (d) SVR-based DOA estimates with training data of SNR = 10 dB.

B. Generalization to Untrained Scenarios

In this section, we compare the proposed method with the state-of-the-art machine learning-based DOA estimation method, i.e., the SVR-based DOA estimator [32], [33], to show how they generalize to scenarios not included in the training data set. No array imperfection is considered in the simulations.

First, two signals with an angular distance of 9.4° and SNR = 10 dB are assumed to impinge onto the array simultaneous, and the direction of the first signal varies from −60° to 50°. This angular distance is not contained in the training set $\mathbf{\Delta}$, and the direction of the second signal deviates from the preseted training directions and the output spectrum grids. The final DOA estimates are obtained via amplitude interpolation within the two most significant peaks of the reconstructed spectra. The estimated directions and the estimation errors of the two signals when the first signal direction increases from −60° with a step of 1° to 50° are shown in Fig. 4(a) and (b), respectively. The DOA estimates well match their true values and most of the estimation errors are smaller than 0.5°. In Fig. 4(c) and (d), we plot the results of the SVR-based DOA estimation results in the same scenarios. In Fig. 4(c), the SVRs are trained with the same training data set as the DNN classifiers, except that the array outputs for training are noise-free, while the testing data are noise contaminated with SNR = 10 dB. In Fig. 4(d), the training data of the SVRs are also polluted by noise with SNR = 10 dB, the same as the proposed method without exceptions. The SVRs also perform well when the training data are noise-free, but their performance aggravates significantly when there are perturbations in the training data. As it is very difficult or even impossible to collect noise-free training data, the proposed method is believed to behave better than the SVR-based method in practice.

Second, two signals with different SNRs impinge onto the array simultaneously, with the SNR of the first signal being 10 dB and that of the second one being 13 dB. The angular distance between them is 16.4° and the direction of the first signal varies from −60° to 43°. In order to improve the ability of the proposed method in adaptation to power divergences between multiple signals, we enlarge the training data set of the classifiers introduced in Sections III-C and IV-B. Intersignal SNR divergences of ±6, ±3, and 0 dB are considered in addition to original training data set settings, and the network training process keeps unchanged. The DOA estimation results and errors of the proposed DNN-based method are shown in Fig. 5(a) and (b), and the performances of the existing SVR-based method are shown in Fig. 5(c) and (d). In order to obtain valid DOA estimates, the SVRs are trained with noise-free data set, while the DNN is trained with array outputs of SNR = 10 dB. The DNN-based method still performs satisfyingly in DOA estimation precision in spite of intersignal SNR divergences. On the contrary, as the different signal SNR introduced diversities to the distributions of the training and testing data, the SVR-based method only obtained biased DOA estimates. The DOA estimation biases of the first signal, which has lower SNR, are as large as 4°∼5° in most of the cases.

We then keep the SNR of the two signals fixed at 10 dB and enlarge their angular distance to 60°, which deviates from Δs in the training set largely. When the first signal direction varies from −60° to −1°, the DOA estimates of the DNN-based method and the SVR-based method are shown in Fig. 6. The proposed DNN-based method again shows much better adaptation to such a previously unseen scenario during training, and the SVR-method fails to obtain valid DOA estimates for the signals.

Finally, we show how the proposed method behaves when the testing data contain different number of signals as the training data. The DNN and SVR have been trained with array outputs in two-signal scenarios, and the SVR-method forms two regression machines for processing test data and

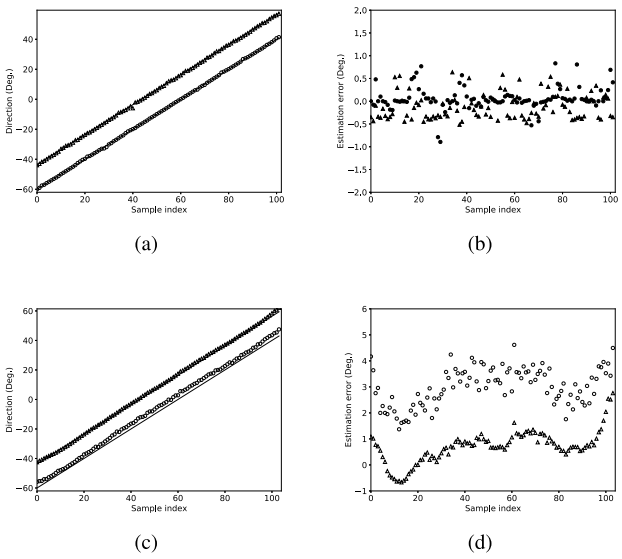


Fig. 5. DOA estimation performance of two unequally powered signals with SNR of 10 and 13 dB. (a) DNN-based DOA estimates. (b) DNN-based DOA estimation errors. (c) SVR-based DOA estimates. (d) SVR-based DOA estimation errors.

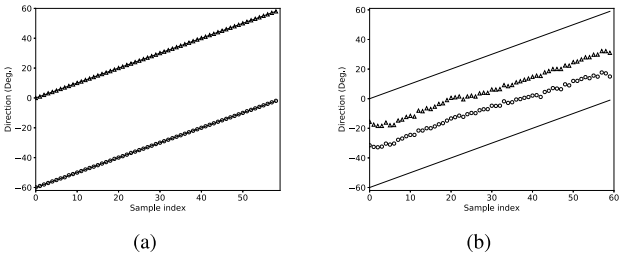


Fig. 6. DOA estimation results of two signals separated by 60°, much larger than the separations in the training data set. (a) DNN-based method. (b) SVR-based method.

outputs two DOA estimates for each given data [32], [33]. If the input covariance vector contains more or fewer signal components, the SVR outputs make no sense. However, the proposed DNN-based method still performs satisfyingly in the one-signal and three-signal scenarios, as are shown in Fig. 7. The SNR of the signals are all 10 dB, and the angular distances between adjacent signals in the three-signal scenarios are both 14°. There is also a similar phenomenon in Fig. 7(b) as that in Fig. 5(b), i.e., when some of the incident signals are located at the edge of the filter subregions, the corresponding DOA estimates aggravate in precision or even disappear in the estimated spectrum. The strategy of spatially overlapping filters is again suggested to improve the behavior of the proposed method in such scenarios.

C. Array Imperfection Adaptation

In this section, we carry out simulations to demonstrate the adaptation of the proposed DNN-based method to various kinds of array imperfections and compare it with the most widely studied parametric DOA estimation method MUltiple SIgnal Classification (MUSIC) [6]. There are many state-of-the-art parametric methods with improved DOA

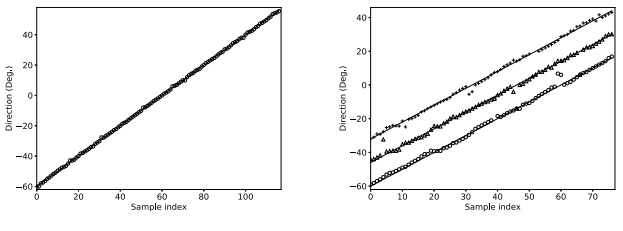


Fig. 7. DOA estimation results in one-signal and three-signal scenarios with DNN trained with two-signal data set. (a) One-signal case. (b) Three-signal case with adjacent angular distances of 14° and 14°.

estimation precision when compared with MUSIC, such as the sparsity-inducing ones [9]–[13]. However, in the presence of significant array imperfections, the differences in DOA estimation precision between different parametric methods are much minor. Among the existing parametric methods, we have chosen MUSIC as it is a widely accepted baseline method. The SVR method is also excluded here because it lacks robustness to noisy training data sets, and SVR models trained with noise-free data sets will make the comparisons unfair.

Two signals with SNR $=$ 10 dB are assumed to impinge onto the array from directions of 31.5° and 41.5°, both off the training and output spectrum grids. The adjusting parameter ρ in (37)–(40) varies from 0 to 1. When $\rho = 0$, no imperfection is contained in the array responding functions. Four cases with different array imperfections are considered by setting the $\delta_{(\bullet)}$'s to different values in (41) . We first set $\delta_{\text{gain}} = \delta_{\text{phase}} = 1$ and $\delta_{\text{pos}} = \delta_{\text{mc}} = 0$, i.e., only gain and phase imperfections are added to the array. The root-mean-square errors (RMSEs) of the two-signal direction estimates in 100 simulations when ρ varies from 0 to 1, are shown in Fig. 8(a). Then, we set $\delta_{\text{gain}} = \delta_{\text{phase}} = \delta_{\text{mc}} = 0$ and $\delta_{\text{pos}} = 1$ to consider the sensor position error only, and the DOA estimation RMSEs are shown in Fig. 8(b). Afterward, we set $\delta_{\text{gain}} = \delta_{\text{phase}} = \delta_{\text{pos}} = 0$ and $\delta_{\text{mc}} = 1$ to retain the mutual coupling effect only, and the corresponding DOA estimation RMSEs are shown in Fig. 8(c). Finally, we set $\delta_{\text{gain}} = \delta_{\text{phase}} = \delta_{\text{pos}} = \delta_{\text{mc}} = 1$ to consider a combination of all the three kinds of array imperfections at the same time and obtain the DOA estimation RMSEs shown in Fig. 8(d).

We have also carried out similar simulations on uniform circular arrays with mutual coupling. In the results, the proposed method again surpasses MUSIC largely in adaptation to array imperfections. We do not include the results here to avoid redundancy.

When $\rho = 0$, the array responding function is ideal and consistent with its counterpart preassumed in the parametric methods, thus MUSIC obtains DOA estimates with very high precisions. However, as the array imperfections become more and more significant, its DOA estimation error increases almost linearly. They have to be modified with various calibration methods to seek for precision improvements. On the contrary, the proposed DNN-based method performs slightly worse than MUSIC when no imperfections are present. However, as it does not rely on any preassumed information about the array geometry or array responding function, it is very

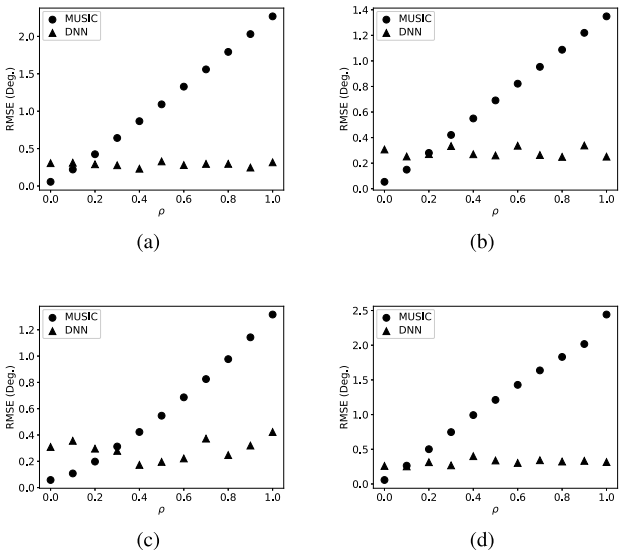


Fig. 8. DOA estimation RMSE of MUSIC and the proposed method for two signals from directions of 31.5° and 41.5° in the presence of different array imperfections. (a) Gain and phase inconsistence. (b) Sensor position error. (c) Mutual coupling. (d) Combined imperfection.

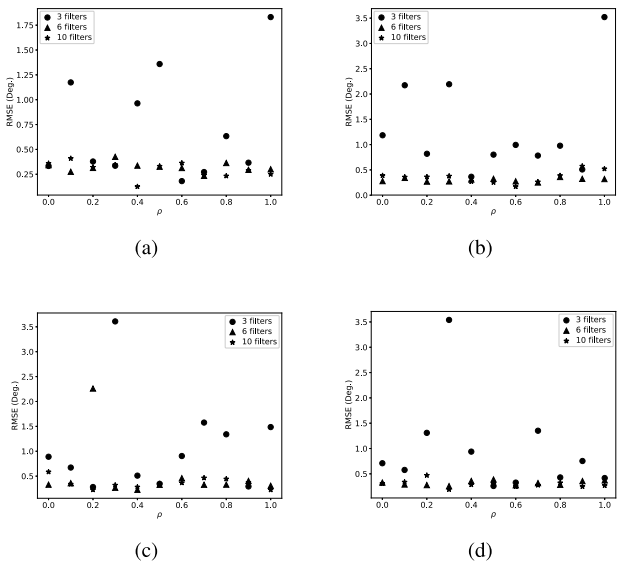


Fig. 9. DOA estimation RMSE of the proposed method with different spatial filters in the presence of array imperfections. (a) Gain and phase inconsistence. (b) Sensor position error. (c) Mutual coupling. (d) Combined imperfection.

robust to different and even combined array imperfections, and its DOA estimation precision seldom changes with the amplitudes of the imperfections. When ρ is as small as 0.1∼0.3, the DNN-based method performs comparably as MUSIC, and when ρ becomes larger and the array responding function deviates farther away from its ideal counterpart, the DNN-based method performs much better than MUSIC.

In order to demonstrate the contribution of the multitask autoencoder in DOA estimation, we select different decoder numbers to show how the DOA estimation precision changes with the spatial filter (also the decoder) number. Three filter numbers of 3, 6, and 10 are selected, and the corresponding DNNs are trained and tested in scenarios of different kinds of array imperfections and different ρs. Parameters of the training and testing data sets and processes are set the same as that in the simulations corresponding to Fig. 8. The DOA estimation RMSE is shown in Fig. 9. The results show that when the filter number is as small as 3, each of the filter covers a wide spatial subregion and the filter outputs are still much divergent in distribution. Thus, the trained DNN does not work well in some of the testing scenarios, and the DOA estimation RMSEs have very large variances. When the filter number increases to 6, the DOA estimation RMSE becomes very small and keeps stable in different scenarios. Afterward, when we further increase the filter number to 10, the RMSE does not decrease largely any more. It can be concluded from this group of simulation results that decoder numbers smaller than a certain threshold lead to worse DOA estimation performance in the proposed DNN framework, while decoder numbers larger than the threshold do not lead to significant performance improvements. Therefore, we have set the decoder number to be 6 in previous simulations empirically. Another special value of the decoder number is 1, which is equivalent to removing the autoencoder in the proposed DNN framework in Fig. 1. In this case, a single classifier will be trained for DOA estimation directly. The results in Fig. 9 indicate that the DOA estimation performance of the single classifier will be much worse than that of a DNN with a three-task autoencoder. That is why we have introduced a multitask autoencoder in the DNN framework to spatially filter the inputs to concentrate their distributions, so as to reduce the burden in the generalization of the subsequent DOA estimation classifiers.

VI. Conclusion

This paper proposes a DNN framework to deal with the problem of DOA estimation, so as to make up for the drawbacks of previous parametric and data-driven methods in terms of array imperfection adaptation and generalization. The proposed DNN-based framework consists of a multitask autoencoder and a series of parallel multilayer classifiers. The two parts are trained separately with different data sets, and the data set for training the refined classifiers is generated only in two-signal scenarios. In spite of the simplicity of the training data set, the proposed method is demonstrated via simulations to own much-enhanced generalizations when compared with the SVR-based method in the machine learning community. It adapts well to noisy training data, off-grid signals, unequally powered signals, much large angular distances, and even different numbers of signals as that in the training data set. The proposed method has also been shown to adapt well to various kinds of array imperfections. It obtains DOA estimates with much higher precisions than the most widely studied parametric method of MUSIC when the imperfections are significant.

The proposed method also has an obvious drawback when compared with existing parametric counterparts, such as MUSIC. It requires a large amount of labeled data to train the DNN framework for DOA estimation, which may be very demanding in practical applications when it is difficult to collect such data. Therefore, a potential future work of our

research is reducing the size of the training data set or using other substitutional training processes, e.g., training the DNN with simulation data first and then adjust it for practical usage via transfer learning with a small amount of practical data collected with arrays.

ACKNOWLEDGMENT

The Source codes for this paper in Python can be found at https://github.com/LiuzmNUDT/DNN-DOA.

REFERENCES

[1] D. H. Johnson and D. E. Dudgeon, *Array Signal Processing: Concepts and Techniques*. New York, NY, USA: Simon & Schuster, 1992.

[2] H. Krim and M. Viberg, "Two decades of array signal processing research: The parametric approach," *IEEE Signal Process. Mag.*, vol. 13, no. 4, pp. 67–94, Jul. 1996.

[3] B. D. Van Veen and K. M. Buckley, "Beamforming: A versatile approach to spatial filtering," *IEEE ASSP Mag.*, vol. 5, no. 2, pp. 4–24, Apr. 1988.

[4] J. Litva and T. K. Lo, *Digital Beamforming in Wireless Communications*. Norwood, MA, USA: Artech House, 1996.

[5] J. Li and P. Stoica, *Robust Adaptive Beamforming*, vol. 88. Hoboken, NJ, USA: Wiley, 2005.

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

[7] 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.

[8] E. Gonen and J. M. Mendel, "Subspace-based direction finding methods," in *The Digital Signal Processing Handbook*, vol. 62, V. K. Madisetti and D. B. Williams, Eds. Boca Raton, FL, USA: CRC Press, 1999.

[9] D. Malioutov, M. Çetin, and A. S. Willsky, "A sparse signal reconstruction perspective for source localization with sensor arrays," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 3010–3022, Aug. 2005.

[10] Z. M. Liu, Z. T. Huang, and Y. Y. Zhou, "Direction-of-arrival estimation of wideband signals via covariance matrix sparse representation," *IEEE Trans. Signal Process.*, vol. 59, no. 9, pp. 4256–4270, Sep. 2011.

[11] Z.-M. Liu, Z.-T. Huang, and Y.-Y. Zhou, "An efficient maximum likelihood method for direction-of-arrival estimation via sparse Bayesian learning," *IEEE Trans. Wireless Commun.*, vol. 11, no. 10, pp. 1–11, Oct. 2012.

[12] Z. Yang, L. Xie, and C. Zhang, "Off-grid direction of arrival estimation using sparse Bayesian inference," *IEEE Trans. Signal Process.*, vol. 61, no. 1, pp. 38–43, Jan. 2013.

[13] Z. M. Liu and F. C. Guo, "Azimuth and elevation estimation with rotating long-baseline interferometers," *IEEE Trans. Signal Process.*, vol. 63, no. 9, pp. 2405–2419, May 2015.

[14] A. G. Jaffer, "Maximum likelihood direction finding of stochastic sources: A separable solution," in *Proc. Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Apr. 1988, pp. 2893–2896.

[15] P. Stoica and N. Arye, "MUSIC, maximum likelihood, and Cramer-Rao bound," *IEEE Trans. Acoust., Speech Signal Process.*, vol. 37, no. 5, pp. 720–741, May 1989.

[16] M. I. Miller and D. R. Fuhrmann, "Maximum-likelihood narrow-band direction finding and the EM algorithm," *IEEE Trans. Acoust., Speech Signal Process.*, vol. 38, no. 9, pp. 1560–1577, Sep. 1990.

[17] B. Allen and M. Ghavami, *Adaptive Array Systems: Fundamentals and Applications*. Hoboken, NJ, USA: Wiley, 2006.

[18] B. Porat and B. Friedlander, "Accuracy requirements in off-line array calibration," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 33, no. 2, pp. 545–556, Apr. 1997.

[19] G. R. Hopkinson, T. M. Goodman, and S. R. Prince, *A Guide to the Use and Calibration of Detector Array Equipment*, vol. 142. Bellingham, WA, USA: SPIE, 2004.

[20] M. Viberg and A. L. Swindlehurst, "Analysis of the combined effects of finite samples and model errors on array processing performance," *IEEE Trans. Signal Process.*, vol. 42, no. 11, pp. 3073–3083, Nov. 1994.

[21] Z. M. Liu, Z. T. Huang, and Y. Y. Zhou, "Bias analysis of MUSIC in the presence of mutual coupling," *IET Signal Process.*, vol. 3, no. 1, pp. 74–84, Jan. 2009.

[22] B. Friedlander and A. J. Weiss, "Direction finding in the presence of mutual coupling," *IEEE Trans. Antennas Propag.*, vol. 39, no. 3, pp. 273–284, Mar. 1991.

[23] T. Svantesson, "Modeling and estimation of mutual coupling in a uniform linear array of dipoles," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, vol. 5, Mar. 1999, pp. 2961–2964.

[24] M. Lin and L. Yang, "Blind calibration and DOA estimation with uniform circular arrays in the presence of mutual coupling," *IEEE Antennas Wireless Propag. Lett.*, vol. 5, no. 1, pp. 315–318, Dec. 2006.

[25] A. J. Weiss and B. Friedlander, "Array shape calibration using eigenstructure methods," *Signal Process.*, vol. 22, no. 3, pp. 251–258, Mar. 1991.

[26] B. P. Flanagan and K. L. Bell, "Array self-calibration with large sensor position errors," *Signal Process.*, vol. 81, no. 10, pp. 2201–2214, Oct. 2001.

[27] A. Paulraj and T. Kailath, "Direction of arrival estimation by eigenstructure methods with unknown sensor gain and phase," in *Proc. Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, vol. 10, Apr. 1985, pp. 640–643.

[28] Y. Li and M. H. Er, "Theoretical analyses of gain and phase error calibration with optimal implementation for linear equispaced array," *IEEE Trans. Signal Process.*, vol. 54, no. 2, pp. 712–723, Feb. 2006.

[29] B. C. Ng and C. M. S. See, "Sensor-array calibration using a maximum-likelihood approach," *IEEE Trans. Antennas Propag.*, vol. 44, no. 6, pp. 827–835, Jun. 1996.

[30] K. V. Stavropoulos and A. Manikas, "Array calibration in the presence of unknown sensor characteristics and mutual coupling," in *Proc. 10th Eur. Signal Process. Conf.*, Sep. 2000, pp. 1–4.

[31] Z.-M. Liu and Y.-Y. Zhou, "A unified framework and sparse Bayesian perspective for direction-of-arrival estimation in the presence of array imperfections," *IEEE Trans. Signal Process.*, vol. 61, no. 15, pp. 3786–3798, Aug. 2013.

[32] M. Pastorino and A. Randazzo, "A smart antenna system for direction of arrival estimation based on a support vector regression," *IEEE Trans. Antennas Propag.*, vol. 53, no. 7, pp. 2161–2168, Jul. 2005.

[33] A. Randazzo, M. A. Abou-Khousa, M. Pastorino, and R. Zoughi, "Direction of arrival estimation based on support vector regression: Experimental validation and comparison with MUSIC," *IEEE Antennas Wireless Propag. Lett.*, vol. 6, pp. 379–382, 2007.

[34] A. Rawat, R. N. Yadav, and S. Shrivastava, "Neural network applications in smart antenna arrays: A review," *AEU-Int. J. Electron. Commun.*, vol. 66, no. 11, pp. 903–912, Nov. 2012.

[35] K. Terabayashi, R. Natsuaki, and A. Hirose, "Ultrawideband direction-of-arrival estimation using complex-valued spatiotemporal neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 9, pp. 1727–1732, Sep. 2014.

[36] Y. Gao, D. Hu, Y. Chen, and Y. Ma, "Gridless 1-b DOA estimation exploiting SVM approach," *IEEE Commun. Lett.*, vol. 21, no. 10, pp. 2210–2213, Oct. 2017.

[37] S. Jha and T. Durrani, "Direction of arrival estimation using artificial neural networks," *IEEE Trans. Syst., Man, Cybern.*, vol. 21, no. 5, pp. 1192–1201, Sep./Oct. 1991.

[38] H. L. Southall, J. A. Simmers, and T. H. O'Donnell, "Direction finding in phased arrays with a neural network beamformer," *IEEE Trans. Antennas Propag.*, vol. 43, no. 12, pp. 1369–1374, Dec. 1995.

[39] 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.

[40] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[41] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.

[42] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[43] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[44] J. Quioñero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset Shift in Machine Learning*. Cambridge, MA, USA: MIT Press, 2009.

[45] M. Sugiyama and M. Kawanabe, *Machine Learning in Non-Stationary Environments: Introduction to Covariate Shift Adaptation* . Cambridge, MA, USA: MIT Press, 2012.

[46] R. Takeda and K. Komatani, "Discriminative multiple sound source localization based on deep neural networks using independent location model," in *Proc. IEEE Spoken Lang. Technol. Workshop (SLT)*, Dec. 2016, pp. 603–609.

[47] X. Xiao, S. Zhao, X. Zhong, D. L. Jones, E. S. Chng, and H. Li, "A learning-based approach to direction of arrival estimation in noisy and reverberant environments," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2015, pp. 2814–2818.
[48] F. Vesperini, P. Vecchiotti, E. Principi, S. Squartini, and F. Piazza, "A neural network based algorithm for speaker localization in a multi-room environment," in *Proc. IEEE 26th Int. Workshop Mach. Learn. Signal Process. (MLSP)*, Sep. 2016, pp. 1–6.
[49] S. Chakrabarty and E. A. P. Habets. (2017). "Broadband DOA estimation using Convolutional neural networks trained with noise signals." [Online]. Available: https://arxiv.org/abs/1705.00919
[50] S. Adavanne, A. Politis, and T. Virtanen. (2017). "Direction of arrival estimation for multiple sound sources using convolutional recurrent neural network." [Online]. Available: https://arxiv.org/abs/1710.10059
[51] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.
[52] M. Abadi *et al.* (2016). "TensorFlow: Large-scale machine learning on heterogeneous distributed systems." [Online]. Available: https://arxiv.org/abs/1603.04467
[53] A. Cotter, O. Shamir, N. Srebro, and K. Sridharan, "Better mini-batch algorithms via accelerated gradient methods," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 1647–1655.
[54] R. W. Kindt, K. Sertel, and J. L. Volakis, "A review of finite array modeling via finite-element and integral-equation-based decomposition methods," *Radio Sci. Bull.*, vol. 2011, no. 336, pp. 12–22, 2011.
[55] J. Hu, W. Lu, H. Shao, and Z. Nie, "Electromagnetic analysis of large scale periodic arrays using a two-level CBFs method accelerated with FMM-FFT," *IEEE Trans. Antennas Propag.*, vol. 60, no. 12, pp. 5709–5716, Dec. 2012.
[56] D. J. Ludick, M. M. Botha, R. Maaskant, and D. B. Davidson, "The CBFM-enhanced Jacobi method for efficient finite antenna array analysis," *IEEE Antennas Wireless Propag. Lett.*, vol. 16, pp. 2700–2703, 2017.
[57] K. M. Pasala and E. M. Friel, "Mutual coupling effects and their reduction in wideband direction of arrival estimation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 30, no. 4, pp. 1116–1122, Oct. 1994.
[58] R. S. Adve and T. K. Sarkar, "Compensation for the effects of mutual coupling on direct data domain adaptive algorithms," *IEEE Trans. Antennas Propag.*, vol. 48, no. 1, pp. 86–94, Jan. 2000.
[59] C. K. E. Lau, R. S. Adve, and T. K. Sarkar, "Minimum norm mutual coupling compensation with applications in direction of arrival estimation," *IEEE Trans. Antennas Propag.*, vol. 52, no. 8, pp. 2034–2041, Aug. 2004.
[60] H.-S. Lui and H. T. Hui, "Direction-of-arrival estimation: Measurement using compact antenna arrays under the influence of mutual coupling," *IEEE Antennas Propag. Mag.*, vol. 57, no. 6, pp. 62–68, Dec. 2015.

Zhang-Meng Liu received the Ph.D. degree in statistical signal processing from the National University of Defense Technology (NUDT), Changsha, China, in 2012.

From 2017 to 2018, he was a Visiting Scholar with the Big Data and Social Computing Laboratory, led by Dr. P. S. Yu, The University of Illinois at Chicago, Chicago, IL, USA. He is currently an Associate Professor with NUDT, where he is involved in the interdiscipline of electronics engineering and computer science, especially electronic data mining. He has authored or co-authored more than 20 papers on signal processing, Bayesian learning, and data mining.

Chenwei Zhang received the B.S. degree in computer science and technology from Southwest University, Chongqing, China, in 2014. He is currently pursuing the Ph.D. degree with the Department of Computer Science, The University of Illinois at Chicago, Chicago, IL, USA.

His current research interests include deep learning, natural language processing, and applications in text and web mining.

Philip S. Yu (F'93) received the B.S. degree in electrical engineering (E.E.) from National Taiwan University, Taipei, Taiwan, in 1972, the M.S. and Ph.D. degrees in E.E. from Stanford University, Stanford, CA, USA, in 1976 and 1978, respectively, and the M.B.A. degree from New York University, New York, NY, USA, in 1982.

He was with the IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA, where he was the Manager of the Software Tools and Techniques Department. He is currently a Distinguished Professor with the Department of Computer Science, The University of Illinois at Chicago, Chicago, IL, USA, also holds the Wexler Chair in information and technology. He has authored or co-authored more than 970 papers in refereed journals and conferences with more than 74 500 citations and the value of h-index (127). He holds or has applied for more than 300 U.S. patents. His current research interests include big data, data mining (especially on graph/network mining), social network, privacy-preserving data publishing, data stream, database systems, and Internet applications and technologies.

Dr. Yu is a Fellow of the ACM. He is on the steering committee of the ACM Conference on Information and Knowledge Management and was a Steering Committee Member of the IEEE Conference on Data Mining and the IEEE Conference on Data Engineering. He was a recipient of the ACM SIGKDD 2016 Innovation Award for his influential research and scientific contributions on mining, fusion, and anonymization of big data, the IEEE Computer Society's 2013 Technical Achievement Award for pioneering and fundamentally innovative contributions to scalable indexing, querying, searching, mining, and anonymization of big data, the Research Contributions Award from the IEEE International Conference on Data Mining in 2003 for his pioneering contributions to the field of data mining, and the IEEE Region 1 Award for promoting and perpetuating numerous new electrical engineering concepts in 1999. He is the Editor-in-Chief of the *ACM Transactions on Knowledge Discovery from Data*.