International Journal of Computer Vision and Signal Processing, 14(1), 1-8(2024) ORIGINAL ARTICLE

Direction-of-arrival estimation for conventional co-prime arrays using probabilistic Bayesian neural networks

Wael Elshennawy*

Abstract

The paper investigates the direction-of-arrival (DOA) estimation of narrow band signals with conventional co-prime arrays by using efficient probabilistic Bayesian neural networks (PBNN). A super resolution DOA estimation method based on Bayesian neural networks and a spatially overcomplete array output formulation overcomes the pre-assumption dependencies of the model-driven DOA estimation methods. The proposed DOA estimation method utilizes a PBNN model to capture both data and model uncertainty. The developed PBNN model is trained to do the mapping from the pseudo-spectrum to the super resolution spectrum. This learning-based method enhances the generalization of untrained scenarios, and it provides robustness to non-ideal conditions, e.g., small angle separation, data scarcity, and imperfect arrays, etc. Simulation results demonstrate the root mean square error (RMSE) and loss curves of the PBNN model in comparison with deterministic model and spatial-smoothing MUSIC (SS-MUSIC) method. The proposed Bayesian estimator improves the DOA estimation performance for the case of low signal-to-noise ratio (SNR) or with a limited number of model trainable variables or spatially adjacent signals.

Keywords: Direction-of-arrival estimation, co-prime arrays, Bayesian neural networks, neural networks

IJCVSP

ISSN: 2186-1390 (Online)
http://cennser.org/IJCVSP

Article History:
Received: 8 Oct. 2023
Revised: 19 Dec. 2023
Accepted: 20 Jan. 2024
Published Online: 30 Jan. 2024

1. INTRODUCTION

The co-prime arrays are class of sparse arrays, which can achieve higher degrees-of-freedom (DOF) that can be exploited in both beamforming and DOA estimation [1]. The coprime arrangement has shown to possess the capability of cancelling spatial aliasing [2], Though the side lobes may still exist in the beampattern that affects the resolution of a DOA estimation algorithm. Therefore, DOA estimation approaches are needed to further explore the advantage of the co-prime arrays. The earlier approaches rely on the subspace-based DOA estimation methods such as multiple signal classification (MUSIC) [3, 4], etc. Meanwhile, these methods require spatial smoothing to restore the rank of the signal covariance matrix [5]. A short and non-exhaustive list of recent works is based on sparse reconstruction so as to use all the unique lags [6, 2]. However, these model-driven methods face great robustness challenges under non-ideal conditions [4].

Another approach provides robust performance against non-ideal conditions include the use of deep convolutional neural networks in [2, 7]. Nevertheless, it is based on the deterministic neural networks. The necessity to develop an approach that exhibits a robustness to the adverse environment. Probabilistic deep learning removes this limitation by quantifying and processing the uncertainty [8]. To further tackle the model and data uncertainty, an off-grid DOA estimation method is proposed from the perspective of variational Bayesian inference [9]. Motivated by the advantages of Bayesian neural networks in [10], this deep probabilistic model is developed based on the normalizing flows for Bayesian neural network to model complex probability distributions [11].

The main contribution of this paper is mainly to consider a probabilistic approach integrated with the deep learning that allows to account for the uncertainty in DOAs estimation of co-prime arrays [11], so that the trained model can assign less levels of confidence to incorrect DOAs pre-

*Corresponding author
Email address: Wael.Elshennawy@Orange.com (Wael Elshennawy)

dictions. The model is developed by balancing between the quality and footprint metrics toward achieving Pareto-optimal model for the purpose of further on-device deployment [12]. The PBNN model is created by using the TensorFlow Probability (TFP) library [13]. Many concepts have been used throughout this paper, including latent variables [14], probabilistic layers [15], bijectors [16], evidence lower bound (ELBO) optimization, and Kullback-Leibeier divergence (KL) divergence regularizers [17] to develop the PBNN model.

The sparsity-inducing DOA estimation methods are generally based on the sparse Bayesian learning (SBL), these methods have been demonstrated to achieve enhanced precision [18, 4]. However, the learning process of those methods converges much slowly when the SNR is relatively low. To overcome this obstacle in this paper, the co-prime arrays is used which provides high SNRs. Moreover, the PBNN model offers an adaptation to various array imperfections and enhanced generalization to unseen scenarios. The Bayesian neural networks (BNN) focuses on marginalization, the estimates would be maximum a posteriori (MAP), and it relies on variational inference and normalization flows to find the optimal values. It quantifies the model and data uncertainty to explain the trustworthiness of the prediction. Thereby avoiding overfitting problem.

The key contributions of this paper are: 1) a probabilistic approach is being integrated with a DL-approach for DOA estimation of coprime arrays. 2) the proposed DOA estimation method enhances the generalization of untrained scenarios, and it also provides robustness to non-ideal conditions, 3) the PBNN model offers a deep learning framework with variational Bayesian networks to directly learn the mapping from the pseudo spectrum to the super resolution spectrum especially at low SNR, and 4) the DOA estimation based PBNN accounts for the modeling of data and model uncertainty.

The remainder of this paper is organized as follows: Section 2 reviews the signal model of the conventional co-prime arrays. Section 3 presents the proposed approach in DOA estimating of narrowband signals based on spatially overcomplete array output formulation and preprocessing and feature extraction. Section 4 introduces efficient PBNN model and its implementation with a coarse-refinement procedure. Simulations results of the proposed DOA estimation method are presented in Section 5 along with evaluating the results and comparing it with a deterministic model and a co-prime spatial smoothing MUSIC (SS-MUSIC) method [19]. The conclusions are drawn in Section 6. The presented deep learning approach tends to bring more reliable DOAs estimation, and it has the potential to be applied in real-world environments [20].

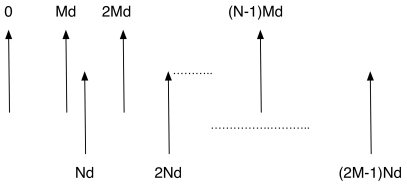


Figure 1: Geometry of co-prime arrays. Adapted from [3].

2. SIGNAL MODEL OF CO-PRIME ARRAYS

The co-prime arrays are the union of two uniform linear sub-arrays as illustrated in Fig. 1. One sub-array consists of $2M$-elements with a spacing of N units. The other composed of N-elements with a spacing of M units. The positions are given by the set $\mathbb{P}$ in [6] as

$$\mathbb{P} = \{Mnd, 0 \leq n \leq N-1\} \cup \{Nmd, 0 \leq m \leq 2M-1\}. \tag{1}$$

Where M and N are co-prime, and it is assumed that $M < N$. The zeroth sensor positions are collocated, so the co-prime arrays consist of $N + 2M - 1$ elements. The fundamental spacing d usually sets to a half-wavelength to avoid the spatial aliasing. K independent narrow band sources $\mathbf{s}(t) = [s_1(t)s_2(t)...s_K(t)]$ are impinging on the co-prime arrays from the directions $\{\theta_1, ..., \theta_K\}$. The array output is formulated in [5] as

$$\mathbf{x}(t) = \sum_{k=1}^{K} \mathbf{a}(\theta_k)s_k(t) + \mathbf{n}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{n}(t), \tag{2}$$

where $\mathbf{A} = [\mathbf{a}(\theta_1), \mathbf{a}(\theta_2), ..., \mathbf{a}(\theta_K)]$ denotes the array manifold matrix, and

$$\mathbf{a}(\theta_k) = [e^{-j2\pi d_1/\lambda \sin\theta_k}, ..., e^{-j2\pi d_{N+2M-1}/\lambda \sin\theta_k}]^T \tag{3}$$

is the steering vector corresponding to θ_k. The $d_1, d_2, ...,$ d_{N+2M-1} hold the information of the sparse elements positions. Whereas $[.]^T$ denotes the transpose of a matrix. $\mathbf{s}(t)$ represents the source signals vector with $s_k(t)$ distributed as $\mathcal{CN}(0, \sigma_k^2)$. The source signals are assumed to be temporally uncorrelated. The entries of the noise vector $\mathbf{n}(t)$ are assumed to be independent and identically distributed (i.i.d) random variables. Also, $\mathbf{n}(t)$ follows a complex Gaussian distributed $\mathcal{CN}(0, \sigma_n^2)$, and their entries are not correlated with source signals.

3. PROPOSED APPROACH

The proposed DOA estimation method for co-prime arrays is illustrated in Fig. 2. The array output is preprocessed to be used by a Bayesian neural network-based model for classification. The pseudo spectrum is calculated from the observation vector and the extended array manifold matrix of a virtual array. This pseudo spectrum is

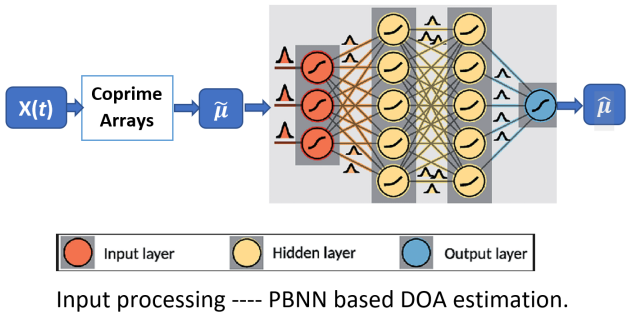


Figure 2: Architecture of the proposed DOA estimation method.

used as the input vector of the PBNN model, and the corresponding super resolution spectrum will be recovered in the output. Thus, this allows to integrate the probabilistic deep learning into a super-resolution DOA estimation method. In addition to, this processing fully maintains the virtual array and effectively improves the original SNR. However, this implies many numbers of parameters, latency, resources required to train, etc. which are increased significantly. Consequently, it has become important to reduce these footprint metrics of a model as well, not just its quality. While this model can perform well on the classification task it is trained on, it might not necessarily be efficient enough for direct deployment in the real world. Therefore, the common theme around the model would be efficiency in terms of inference and training phases.

Here, the PBNN is developed based on Pareto-optimality approach discussed in [12]. To this end, compression technique helps trade off some quality for a better model footprint such as model size, latency, training resources etc. It worth to note that many machine learning models, like deep neural networks, can extract the necessary features from large inputs automatically. However, there are two main problems with this approach i.e., computational complexity and they require lots of training dataset. So, this takes a lot of processing power to automatically learn what features are useful for the classifiers. Moreover, it is required to store, at least, that amount of data in memory, and be able to perform mathematical operations on each of those values. Thus, the objective is to keep the model being efficient as small and fast as possible. Feature extraction fulfills this requirement, where it builds features from raw data by reformatting, combining, transforming primary features into more valuable features as outlined in the next subsection.

3.1. INPUT PROCESSING AND FEATURE EXTACTOR

The covariance matrix $\mathbf{R}$ is given by [1]

$$\mathbf{R} = \mathbb{E}[\mathbf{x}(t)\mathbf{x}^H(t)] = \sum_{k=1}^{K} \mu_k \mathbf{a}(\theta_k)\mathbf{a}^H(\theta_k) + \sigma_n^2 \mathbf{I}, \quad (4)$$

where $\mathbf{R}$ can only be estimated using Q snapshots in practical applications, i.e.

$$\hat{\mathbf{R}} = \frac{1}{Q}\sum_{q=1}^{Q} \mathbf{x}(t_q)\mathbf{x}^H(t_q) = \mathbf{R} + \Delta\mathbf{R}, \quad (5)$$

where $\hat{\mathbf{R}}$ is the maximum likelihood estimator of $\mathbf{R}$ and $\Delta\mathbf{R}$ is the estimation error of $\mathbf{R}$ [2]. By vectorizing $\hat{\mathbf{R}}$, the observation vector of the virtual array is given in [3] as

$$\begin{aligned} \mathbf{y} &= \mathbf{vec}(\hat{\mathbf{R}}) = \mathbf{vec}(\mathbf{R}) + \mathbf{vec}(\Delta\mathbf{R}) \\ &= \tilde{\mathbf{A}}\boldsymbol{\mu} + \sigma_n^2 \mathbf{vec}(\mathbf{I}) + \Delta\mathbf{y}, \end{aligned} \quad (6)$$

where

$$\begin{aligned} \tilde{\mathbf{A}} &= [\mathbf{a}^*(\theta_1) \otimes \mathbf{a}(\theta_1), \mathbf{a}^*(\theta_2) \otimes \mathbf{a}(\theta_2), ..., \mathbf{a}^*(\theta_K) \otimes \mathbf{a}(\theta_K)] \\ &= [\tilde{\mathbf{a}}(\theta_1), \tilde{\mathbf{a}}(\theta_2), ..., \tilde{\mathbf{a}}(\theta_K)], \end{aligned} \quad (7)$$

$\otimes$ represents the Kroncher product and $(.)^*$ is the conjugate operation. The signal of interest becomes $\boldsymbol{\mu} = [\mu_1, \mu_2, ..., \mu_K]^{\mathbf{T}}$, μ_k denotes the input signal power of the kth sources and $\Delta\mathbf{y} = \mathbf{vec}(\Delta\mathbf{R})$, where $\Delta\mathbf{y}$ becomes negligible as the number of snapshots $Q \to \infty$ under stationary and ergodic assumptions. Note that $\mathbf{y}$ amounts to the received data from a virtual array with a much larger aperture defined by the virtual steering matrix $\tilde{\mathbf{A}}$ having the co-array lag locations [5]. Therefore $\tilde{\mathbf{A}}$ behaves like the manifold of a longer equivalent virtual array [6].

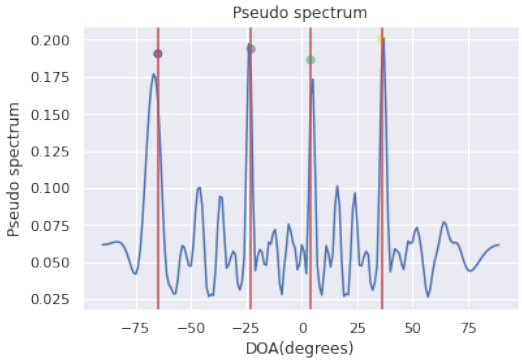
Next, by removing the repeated elements of $\mathbf{y}$ and sorting the remaining in an increasing order from $-(MN+M-1)$ to $(MN+M-1)$, the output $\tilde{\mathbf{y}}$ is extracted without redundancy for a linear model [3]. By extending the corresponding steering vector, the output of the virtual array can be reconstructed in [1] as

$$\begin{aligned} &\tilde{\mathbf{y}} = \mathbf{B}\boldsymbol{\mu} + \sigma_n^2 \mathbf{vec}(\mathbf{I}), \\ &\mathbf{B} = [\mathbf{b}^*(\theta_1) \otimes \mathbf{b}(\theta_1), \mathbf{b}^*(\theta_2) \otimes \mathbf{b}(\theta_2), ..., \mathbf{b}^*(\theta_W) \otimes \mathbf{b}(\theta_W)], \end{aligned} \quad (8)$$

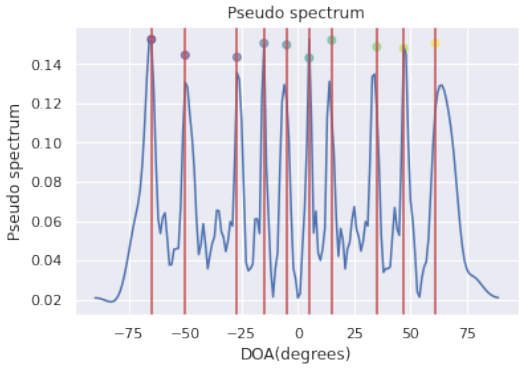
where $\mathbf{B} \in \mathbb{C}^{(N+2M-1)^2 \times W}$. $\boldsymbol{\mu} = [\mu_1, \mu_2, ..., \mu_W]^{\mathbf{T}}$, $W \geq K$. $[\theta_1, \theta_2, ..., \theta_W]$ is sampled from the spatial spectrum of incident signals with an interval of $\Delta\theta$. The spatial spectrum $\boldsymbol{\mu}$ is constructed with W grids, which has nonzero values at the true signal directions. The pseudo-spectrum is given by [1]

$$\tilde{\boldsymbol{\mu}} = \mathbf{B}^{\mathbf{H}}\tilde{\mathbf{y}}, \quad (9)$$

as the input of the Bayesian neural network. This strategy maintains the virtual array generating from co-prime arrays, and it effectively improves the original SNR [7]. To demonstrate the resolution of the pseudo-spectrum $\tilde{\boldsymbol{\mu}}$ of co-prime arrays and shows how it helps in training of the Bayesian neural network. Consider co-prime arrays consisting of 10 physical antenna elements, which is designed by assuming $M = 3$ and $N = 5$. Suppose two different signal sources are impinging on the array from



(a) Scenario A_s, 4 signal sources with 0dB.



(b) Scenario B_s, 10 signal sources with 0dB.

Figure 3: Resolution predominance of the pseudo spectrum $\tilde{\boldsymbol{\mu}}$ of a coprime arrays a) 4 signals with 0dB, (b) 10 signals with 0dB.

two directions sets $A_s = \{-65^o, -23^o, 4^o, 36^o\}$ and $B_s = \{-65^o, -50^o, -27^o, -15^o, -5^o, 5^o, 15^o, 35^o, 47^o, 61^o\}$. As shown in Fig. 3, coprime arrays achieve higher DOF and resolution.

4. DOA ESTIMATION BASED ON PBNN MODEL

The idea is to create a neural network with a weight uncertainty by combining the neural network with Bayesian inference. Usually, there are two categories of uncertainty; aleatoric and epistemic [11], so there is a necessity to introduce a method for designing a deep learning model that accounts for the uncertainty. In practice, and especially considering the dataset as being finite, there will most likely be many possible parameters values that can do a good job of modeling the relationship between the dataset inputs and the targets values. If more dataset is being collected, then the model would have more information about that relationship, and the likely sets of model parameters would probably narrow down. This likely set of parameters values given a dataset is represented as a distribution over all possible parameter values and is called the posterior distribution [16]. Conventionally the term weights will be used to refer to weights and biases for the remainder sections of the paper. Here, the PBNN model is developed based on the use of probabilistic neural network [16] and the probabilistic layers are implemented by employing TFP library [17].

4.1. BAYESIAN INFERENCE AND POSTERIOR PROBABILITY

The Bayesian approach is usually implemented by using Backprop algorithm [14], that uses variational inference to give an approximation of the posterior distribution over the model weights[9]. Concisely, the true labels and the likelihood function are used to find the best weights of the Bayesian neural network [13]. For instance, the neural network is a function that maps a pseudo-spectrum data point $\tilde{\mu}_i$ to the proper parameters of some distribution. The PBNN model with weights $\mathbf{W}$ is developed to classify data points $\tilde{\mu}_i$. Hence, the neural network prediction (the feed-forward value) $\hat{\mu}_i$ is defined in [16] as

$$\hat{\mu}_i = \text{BNN}(\tilde{\mu}_i|\mathbf{W}). \tag{10}$$

Determining $\mathbf{W}$ implies that training a model and assuming that the prediction $\hat{\mu}_i$ forms a part of a distribution that the true label is drawn from. Let the data be $\tilde{\mu}_i$ and the true labels μ_i for $i = 1, ..., N_s$, where N_s is the number of training samples. Then the training dataset is given as

$$\boldsymbol{D} = \{(\tilde{\mu}_i, \mu_i), ..., ((\tilde{\mu}_{N_s}, \mu_{N_s})\}. \tag{11}$$

For each point $\tilde{\mu}_i$ has the corresponding prediction $\hat{\mu}_i$, where it assumes specifying a distribution in addition to the true label μ_i. The weights of the trained neural network are then those that minimize the negative log-likelihood loss function in [10] as

$$\begin{aligned}\mathbf{W}^* &= \arg\min_{\mathbf{W}}(-\sum_{i}^{N_s} \log L(\mu_i|\hat{\mu}_i)),\\ &= \arg\min_{\mathbf{W}}(-\sum_{i}^{N_s} \log L(\mu_i|\text{BNN}(\tilde{\mu}_i|\mathbf{W}))).\end{aligned} \tag{12}$$

In practice, determining the true optimum $\mathbf{W}^*$ is not always possible. Instead, an approximated value is sought using optimization algorithms such as root mean squared propagation (RMSProp) or adaptive moment estimation (adam) [11].

5. SIMULATION RESULTS

In this section, the implementation of both deterministic model and PBNN model are presented. DOAs prediction is modeled as a multi-label classification task [21]. The training, validation, and testing datasets are generated by using Keras generator [21]. The simulations are computed by using Python 3 Google compute engine backend enabling graphics processing unit (GPU) in Google colaboratory notebooks with a mounted drive of a size 12.7 GB. The training step covers different scenarios including

changing of the angle of separation, number of DoAs, number of snapshots, SNR etc.

Model uncertainty due to insufficient data availability for the model to learn effectively, this is already alleviated by increasing the size of the training data generated by using Keras data generator, which is based on data augmentation method for better model regularization. Bayesian neural network (BNN) places a probability distribution on network weights and gives a built-in regularization effect making the proposed PBNN model can learn well from small datasets without overfitting. By introducing a prior, and posterior probabilities, so it preserves the uncertainty that reflects the instability of statistical inference of a small number of instances of evidence dataset. The two properties sparsity of the recovery method and stability are at odds of each other, but the variational Bayesian interference introduces algorithmically stable model.

5.1. SIMULATION SETTINGS AND NETWORK TRAINING

Consider co-prime arrays consisting of 10 physical antenna elements, which are designed by taking $M = 3$, $N = 5$. The unit spacing d is chosen to be a half-wavelength. The covariance matrices are computed by using 256 snapshots. The spectrum grid of interest $[-15^o, 15^o]$ is sampled using 1^o intervals to form 31 spectrum grid units. The PBNN model and deterministic model are trained using two-signal sources. The simulated signal sources satisfy the far-field narrow-band plane wave conditions. The SNRs of signal sources are generated from a range $[-10, 10]$dB with an 1dB interval.

The models are trained for 10-epochs with a mini-batch size of 32, and the samples set is shuffled at every epoch. The models are fine-tuned using the RMSProp optimizer [21] with a learning rate of 0.05. The total number of training dataset and testing dataset samples is 500 and 100 respectively. With 20% validation dataset off training samples set aside to evaluate the models after each epoch. The architecture of the deterministic model is illustrated in Fig. 4. The deterministic model is stacked by the following Keras layers: Conv1D. BatchNormalization, AveragePooling1D, Flatten, Dropout, and Dense [11].

To build PBNN model, the deterministic model is transformed into a probabilistic model as an intermediate step, by setting the output of the model final layer to a distribution instead of a deterministic tensor. Then this probabilistic model can capture the aleatoric uncertainty on the target DOAs. This is implemented by an addition of a probabilistic layer as a final model layer [15]. Next, turning this probabilistic model into a PBNN model that is designed to capture aleatoric and epistemic uncertainty by changing model layers into reparametrization layers [13] as illustrated in Fig. 4. To further embed an epistemic uncertainty into the model weights by replacing the Conv1D and Dense layers of the deterministic model with Convolution1DReparameterization and DenseVariational layers [13] respectively. A visualizations of the deterministic model and the PBNN model is depicted in Fig. 5, where it helps to explore the models properties in great detail for each Keras layer.

These models are trained using the same conditions for comparison purpose. The loss functions, negative log-likelihood and RMSE, are used to measure the DOA estimation performance for each model as illustrated in Fig. 6. The PBNN model provides faster convergence at the early stages and lower training loss values throughout the whole training procedure. Considering that the number of trainable variables, and the training time of PBNN model is smaller than the deterministic model as tabulated in 1. On the other hand, the floating-point operations (FLOPS) count is larger for the BPNN model than the DM model. The training and validation loss curves of PBNN model are almost very close which reveal that the PBNN model is well generalized. Clearly, the validation loss curves level off before 10-epochs. Thus, there is no overfitting in the training phase of the PBNN model.

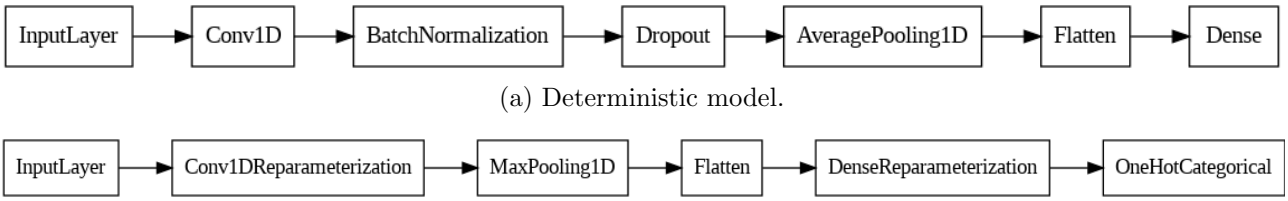
The PBNN model consists of only 2-hidden network layers as illustrated in Fig. 5. Naturally, there exists a trade-off between quality and Footprint metrics. A higher-capacity deeper model is more likely to achieve a better accuracy, but at the cost of model size, latency, etc. On the other hand, a model with lesser capacity/shallower, while possibly suitable for deployment, is also likely to be worse in accuracy. Though, the objective is to develop an efficient PBNN model for further deployment on spintronic devices [22].

Next, testing the angular resolution of the trained PBNN model by incrementally changing the angular separation between two closely spaced signal sources for an angular range varying between 1^o and 10^o per step size 1^o is illustrated in Fig.7, It is obvious that the PBNN model indeed learned to predict the DOAs, and the PBNN model shows robustness. As the separation between the sources increases, the RMSE value increases because the effect of the radiation pattern especially for the case of a low SNR. On the other hand, the RMSE at high SNR decreases since the sources fall within the main beam of the antenna radiation patterns.

Table 1: Performance comparison between deterministic model and PBNN model.

Parameter	**Deterministic**	**PBNN**
Training Time (s)	1130.0	804.1
Trainable Variables	1,695	1,382
Total Parameter	1,727	1,382
FLOPS	6,455	23,256

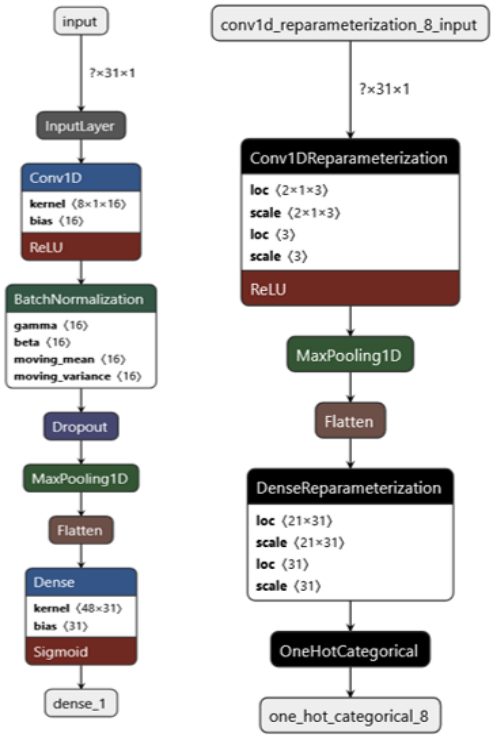
Finally, the robustness of the PBNN model, deterministic model and SS-MUSIC method is compared through Monte Carlo (MC) simulations under different SNRs, and angle separation. First, two 0dB sources with a set of angular separation $[1^o, ..., 10^o]$ are assumed to impinging onto the coprime arrays simultaneously. The RMSE as a per-



(a) Deterministic model.

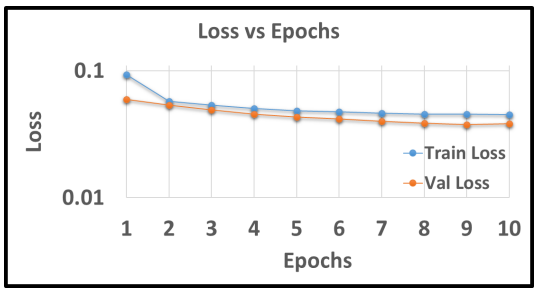
(b) PBNN model.

Figure 4: Deterministic model and PBNN model architectures.

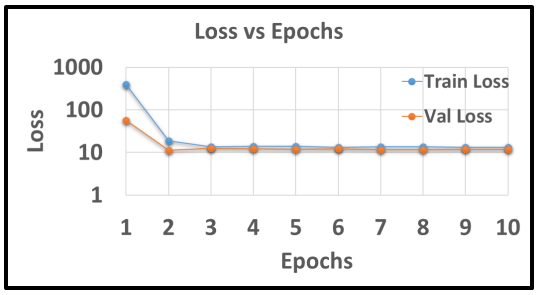


(a) Deterministic model, (b) PBNN model.

Figure 5: Visualization of the models.



(a) Deterministic model.



(b) PBNN model.

Figure 6: Training and validationlosses versus epoch.

formance metric is used to evaluate the DOA estimation precision of these three DOA estimation methods as illustrated in Fig. 8(a). Then, the two-sources DOAs are fixed and the snapshot sets to 1024. The RMSEs of these three DOA estimation methods with respect to varying input signal SNR = [-15dB, 9dB] are shown in Fig. 8(b). The RMSEs of the all three DOA estimations methods with varying number of snapshots = [16, 32, ... ,1024] are illustrated in Fig. 8c. The RMSE is expressed as [2]

$$RMSE = \sqrt{\frac{1}{KQ}\sum_{k}^{K}\sum_{q}^{Q}||(\hat{\theta}_{k,q} - \theta_k)^2||} \qquad (13)$$

where $\hat{\theta}_{k,q}$ represents the estimated DOA of the k-th signal source in the q-th MC round, and Q is the total number of MC simulation rounds. For each simulation scenario, 200 rounds of MC simulation are conducted.

The RMSE performance of the DOA estimation methods as a function of the input angle separation, SNR, and snapshots is compared in Fig. 8 . It is evident that DOA estimation performance is improved with the increase of the input SNR and snapshots. The performance of PBNN model and deterministic model is better than the SS-MUSIC method counterpart for the three simulations. With an efficient PBNN model having lower number of trainable variables compared to the deterministic model as tabulated in 1, The performance of the DOA estimation of the PBNN model is better than deterministic model. It is worth to note that the PBNN method achieves almost the same DOA estimation performance as deterministic method whilst varying of the snapshot number below 1024. Though, the number of snapshots becomes low and it would result in perturbation of the covariance matrix, the resulting DOA estimation performance is still satisfactory.

The proposed PBNN model is a statistical model that

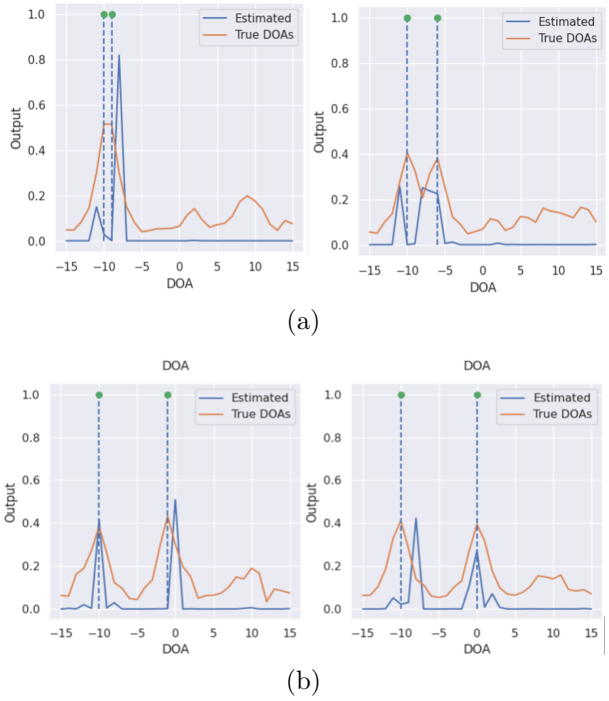
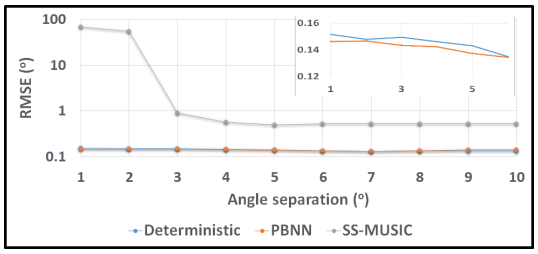
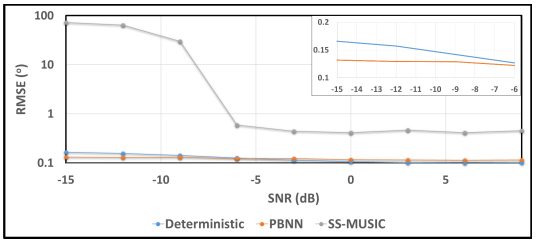


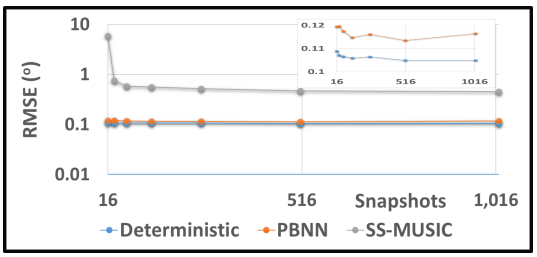
Figure 7: Testing PBNN model by changing the angular separation $\theta_k = [1^o, 4^o, 9^o, 10^o]$ between two DOAs. There corresponding RMSEs = [1.0^o,0.70^o,0.70^o,0.88^o] with SNR = 0dB



(a) RMSE performance versus angle separation with SNR=0dB.



(b) RMSE performance versus SNR with snapshots = 1024.



(b) RMSE performance versus snapshots with SNR=0dB.

Figure 8: RMSE performance comparison with two incident sources.

provides a way to update our beliefs or hypotheses about the DOA estimation based on the new evidence by collecting more snapshots. The RMSE of the PBNN model and deterministic model has a similar trend and exhibits the typical saturation behavior, where the RMSE converges to a constant value when the SNR value is larger than -6dB rather than keeps decreasing. Thus, the uncertainty of DOA estimation methods is quantified by using MC method where MC is categorized as a sampling-based approach that has widely used for quantification and propagation of uncertainties [23].

Most importantly, probabilistic neural networks give a built-in regularization effect making the PBNN model is able to learn well from small datasets without overfitting. Though, Bayesian estimation is computationally very expensive since it greatly widens the parameter space [15]. The pros and cons must be weighed by the user to determine whether the choice of this neural network type is appropriate for the application used. Since the weights of the network are distributions instead of single values, more data is required to accurately estimate the weights.

6. CONCLUSIONS

The paper presents an efficient PBNN-based sparse signal recovery method for DOA estimation with co-prime arrays. The PBNN-DOA follows a coarse-refinement procedure, it first reconstructs the array output to recover the signal components on a discrete spatial grid without any prior signal information about their number or DOA preestimates, and the peaks in the reconstructed spectrum indicate the coarse signal locations. Then refined DOA estimates are obtained based on the reconstruction result via sparse Bayesian learning, which is more robust and efficient. The DOA estimation based PBNN accounts for the modeling of data and model uncertainty. A convolutional neural network (CNN) is combined with probabilistic layers to learn the mapping from the pseudo-spectrum to the spatial spectrum. The input processing fully maintains the DOF and resolution of the virtual array. The PBNN model can achieve faster convergence at the early training stages and lower training losses. Moreover, the PBNN model adapts well to small angular separation, Simulation results demonstrate that the performance advantages of the PBNN model over deterministic model and SS-MUSIC method according to multiple evaluation metrics. DOAs coarse refinement is obtained by balancing the accuracy and efficiency of parameter estimation using the variational Bayesian-based DOA estimation method. With this PBNN model, the possibility of misclassification is minimized. Thus, this proposed DOA estimation method can achieve spectrum autocalibration under non-ideal conditions for the co-prime arrays. In the future, the goal is

to develop the PBNN model for real-time scenario with limited computational resources such as embedded ML deployed on a hardware accelerator.

References

[1] A. Raza, W. Liu, Q. Shen, Thinned coprime arrays for doa estimation, in: 2017 25th European Signal Processing Conference (EUSIPCO), IEEE, 2017, pp. 395–399.

[2] Y. Chen, K.-L. Xiong, Z.-T. Huang, Robust direction-of-arrival estimation via sparse representation and deep residual convolutional network for co-prime arrays, in: 2020 IEEE 3rd International Conference on Electronic Information and Communication Technology (ICEICT), IEEE, 2020, pp. 514–519.

[3] Z. Tan, Y. C. Eldar, A. Nehorai, Direction of arrival estimation using co-prime arrays: A super resolution viewpoint, IEEE Transactions on Signal Processing 62 (21) (2014) 5565–5576.

[4] Y. D. Zhang, M. G. Amin, B. Himed, Sparsity-based doa estimation using co-prime arrays, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, pp. 3967–3971.

[5] A. Ahmed, Y. D. Zhang, J.-K. Zhang, Coprime array design with minimum lag redundancy, in: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2019, pp. 4125–4129.

[6] Z. Shi, C. Zhou, Y. Gu, N. A. Goodman, F. Qu, Source estimation using coprime array: A sparse reconstruction perspective, IEEE Sensors Journal 17 (3) (2016) 755–765.

[7] L. Wu, Z.-M. Liu, Z.-T. Huang, Deep convolution network for direction of arrival estimation with sparse prior, IEEE Signal Processing Letters 26 (11) (2019) 1688–1692.

[8] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, et al., A review of uncertainty quantification in deep learning: Techniques, applications and challenges, Information Fusion 76 (2021) 243–297.

[9] J. Yang, Y. Yang, J. Lu, A variational bayesian strategy for solving the doa estimation problem in sparse array, Digital Signal Processing 90 (2019) 28–35.

[10] X. Feng, X. Zhang, R. Song, J. Wang, H. Sun, H. Esmaiel, Direction of arrival estimation under class a modelled noise in shallow water using variational bayesian inference method, IET Radar, Sonar & Navigation 16 (9) (2022) 1503–1515.

[11] S. Depeweg, Modeling epistemic and aleatoric uncertainty with bayesian neural networks and latent variables, Ph.D. thesis, Technische Universität München (2019).

[12] G. Menghani, Efficient deep learning: A survey on making deep learning models smaller, faster, and better, ACM Computing Surveys 55 (12) (2023) 1–37.

[13] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., Tensorflow: a system for large-scale machine learning., in: Osdi, Vol. 16, Savannah, GA, USA, 2016, pp. 265–283.

[14] C. Zhang, J. Bütepage, H. Kjellström, S. Mandt, Advances in variational inference, IEEE transactions on pattern analysis and machine intelligence 41 (8) (2018) 2008–2026.

[15] M. N. Bajwa, S. Khurram, M. Munir, S. A. Siddiqui, M. I. Malik, A. Dengel, S. Ahmed, Confident classification using a hybrid between deterministic and probabilistic convolutional neural networks, IEEE Access 8 (2020) 115476–115485.

[16] O. Dürr, B. Sick, E. Murina, Probabilistic deep learning: With python, keras and tensorflow probability, Manning Publications, 2020.

[17] C. Blundell, J. Cornebise, K. Kavukcuoglu, D. Wierstra, Weight uncertainty in neural network, in: International conference on machine learning, PMLR, 2015, pp. 1613–1622.

[18] Z. Yang, L. Xie, C. Zhang, Off-grid direction of arrival estimation using sparse bayesian inference, IEEE Transactions on Signal Processing 61 (1) (2013) 38–43. doi:10.1109/TSP.2012.2222378.

[19] C.-L. Liu, P. P. Vaidyanathan, Remarks on the spatial smoothing step in coarray music, IEEE Signal Processing Letters 22 (9) (2015) 1438–1442. doi:10.1109/LSP.2015.2409153.

[20] A. Lu, Y. Luo, S. Yu, An algorithm-hardware co-design for bayesian neural network utilizing sot-mram's inherent stochasticity, IEEE Journal on Exploratory Solid-State Computational Devices and Circuits 8 (1) (2022) 27–34.

[21] O. G. Yalçın, T. Istanbul, Applied Neural Networks with TensorFlow 2: API Oriented Deep Learning with Python, Springer, 2021.

[22] P. Debashis, V. Ostwal, R. Faria, S. Datta, J. Appenzeller, Z. Chen, Hardware implementation of bayesian network building blocks with stochastic spintronic devices, Scientific reports 10 (1) (2020) 16002.

[23] J. Zhang, Modern monte carlo methods for efficient uncertainty quantification and propagation: A survey, Wiley Interdisciplinary Reviews: Computational Statistics 13 (5) (2021) e1539.