IUST
Iran University of Science and Technology

Iranian Journal of Electrical and Electronic Engineering

Journal Homepage: ijeee.iust.ac.ir

Research Paper

Multi-Stage Beamforming Using DNNs

P. Ramezanpour*, M. Aghababaie**, M. R. Mosavi*(C.A.), and D. M. de Andrés***

Abstract: Through beamforming, the desired signal is estimated by calculating the weighted sum of the input signals of an array of antenna elements. In the classical beamforming methods, computing the optimal weight vector requires prior knowledge on the direction of arrival (DoA) of the desired signal sources. However, in practice, the DoA of the signal of interest is unknown. In this paper, we introduce two different deep-neural-network-based beamformers which can estimate the signal of interest while suppressing noise and interferences in two/three stages when the DoAs are unknown. Employing deep neural networks (DNNs) such as convolutional neural networks (CNNs) and bidirectional long short-term memory (bi-LSTM) networks enables the proposed method to have better performance than existing methods. In most cases, the output signal to interference and noise ratio (SINR) of the proposed beamformer is more than 10dB higher than the output SINR of the classical beamformers.

Keywords: Adaptive Digital Beamforming, Bidirectional Long Short-Term Memory, Convolutional Neural Network.

1 Introduction

BEAMFORMERS boost the signal to interference plus noise ratio (SINR) by establishing a directional gain pattern around the antenna array. Applying the optimum weight vector to the received signal vector results in the mentioned gain pattern which is employed for signal estimation. In the conventional methods, the optimal weight vector is calculated based on beamforming criteria such as MMSE and MVDR [1].

In non-blind beamforming algorithms, a known training sequence is exploited to estimate the optimum weight vector [1, 2]. In contrast, in blind algorithms, the optimum weight vector is computed by employing known properties of the desired signal such as cyclic repetition [3, 4].

Iranian Journal of Electrical and Electronic Engineering, 2022.
Paper first received 27 August 2021, revised 27 November 2021, and accepted 17 December 2021.
* The authors are with the Department of Electrical Engineering, Iran University of Science and Technology (IUST), Tehran, Iran.
E-mails: parham.ramezanpour@yahoo.com and m_mosavi@iust.ac.ir.
** The author is with the Department of Electrical Engineering, Imam Khomeini Marine Science University, Mazandaran, Iran.
E-mail: majid_aghababaie@alumni.iust.ac.ir.
*** The author is with the ETSI de Telecomunicación, Universidad Politécnica de Madrid, Madrid, Spain.
E-mail: diego.martin.de.andres@upm.es.
Corresponding Author: M. R. Mosavi.
https://doi.org/10.22068/IJEEE.18.2.2266

Generally, in classical beamforming methods, only, spatial samples of the received signals are employed to estimate the optimal weight vector. However, employing both spatial and temporal samples of the received signals can further improve the capability of the beamformer in suppressing interferences while multipath is present. As a result, space-time adaptive processing (STAP) is a beamforming method introduced for estimating the desired signal by employing both spatial and temporal properties of the signal [5].

In practical applications, the autocorrelation matrix should be estimated with the help of snapshots of the received signals. Employing a finite number of snapshots results in an inaccurate estimation of the autocorrelation matrix. The inaccurate estimation degrades the performance of the beamformers. Thus, another class of beamformers called robust adaptive beamformers (RAB) are introduced which are more robust to the poor estimations in comparison with the classical beamformers. For example, in [6], the authors show higher-order statistics improve the robustness of the beamformer to the autocorrelation matrix and DoA mismatches. In [7], a diagonal matrix with complex constant elements in its main diagonal is added to the received signal autocorrelation matrix to boost the robustness of beamforming to the autocorrelation mismatches.

In an emerging class of beamforming, one or multiple

deep neural networks (DNNs) are exploited to estimate the signal of interest. Function approximation and generalization capabilities of DNNs such as convolutional neural networks (CNNs) and long short-term memory (LSTM) networks enable the DNN-based beamformers to steer the main beam toward the signal of interest and form nulls in directions of interferences when the DoA of the signal of interest is unknown. For example, in [8], two CNNs are fed with channel matrices to compute nearly-optimum weight vector when the DoA of the signal of interest is unknown. In [9], a deep Q-learning algorithm is employed for joint beamforming, power control, and interference coordination in downlink direction for 5G wireless communications. In this method, a DNN is fed with the state vector of the Q-learning algorithm. The optimal state-action value function of the Q-learning algorithm is estimated by the DNN. Finally, the state-action values are employed to choose appropriate transmit power and beamforming code from the candidate transmit powers and beamforming codebook. In [10], a DNN which is fed with the feature vector of the communication channel is exploited to estimate the optimal phase shift matrix of a reconfigurable intelligent surface (RIS). The introduced DNN-based beamformer incurs low computational complexity in comparison with the conventional beamformers. In [11], two CNNs are employed to estimate downlink transmit power, virtual uplink power, and normalized virtual uplink beamforming (VUB) weights by taking in the channel matrix. The estimated values are used for computing downlink beamforming weights. The introduced DNN-based beamformer is capable of estimating beamforming weights without requiring complex operations and iterations.

As mentioned in [12], the LSTM network is exploited to extract temporal features of signals. The LSTM network consists of memory cells that are fed with a data vector contains information of time samples and two state vectors that contains the data processed at the previous or following cells. The architecture of the memory cells enables the LSTM networks to extract the short-term and long-term dependencies of input data. Bidirectional LSTM (bi-LSTM) network is a more complicated version of the conventional LSTMs. Unlike conventional LSTMs which consist of only a forward layer, bi-LSTMs are composed of both forward and backward layers. Outputs of bi-LSTMs are concluded by concatenating the outputs of memory cells placed at backward and forward layers. In other words, the outputs of the network are dependent on information from both previous and following time samples. This property enables the bi-LSTM to detect temporal features of data better than the conventional LSTM [13].

Besides, CNNs are dominant in extracting spatial features of input data. For instance, in [14], CNNs are employed to extract spatial and spectral features in hyperspectral images.

In this paper, the unique capabilities of bi-LSTM and CNN are employed for estimating the signal of interest and suppressing interferences. In the following, two different DNN-based beamformers are introduced for estimating signals whose DoAs are unknown. In the first scheme, at first, similar to [15], multiple overlapped subarrays are formed. Afterward, the autocorrelation matrix of one of the subarrays is calculated and fed into an interference estimator implemented with a CNN. The weights generated by the CNN are used for estimating the interference vector. Subtracting the interference vector from the received signal vector mitigates the effect of interferences. After that, another CNN is fed with the autocorrelation matrix of the interference-free signal vector (concluded in the previous stage) to generate the weights used for estimating the signal of interest. Multiplying the weights to the interference-free signal vector results in a more accurate estimation of the interference-free signal vector. Finally, a bi-LSTM takes in the interference-free signal vector to estimate the time samples of the signal of interest.

Similar to the first proposed neural beamformer, in the second proposed beamforming method, at first, the same grouping scheme is applied to form subarrays. Afterward, the space-time autocorrelation matrix of the signals received at the first subarray is estimated and fed into a CNN to estimate the weight vector. The interference-free signal vector is formed by applying the weight vector to the received signals. The signal snapshots are concluded by employing a bi-LSTM which is fed with the interference-free signal vector.

Unlike the conventional beamforming methods, the proposed beamformers do not require DoAs of signals to estimate the signal of interest. In addition, employing the proposed method results in a higher SINR at the output of the beamformers.

…The rest of this paper is organized as follows. In Section 2, the general structures of the beamformers are specified. In Section 3, the structures of the employed DNNs are described. In Section 4, the capability of the beamformers in improving the SINR is evaluated. Section 5 concludes the paper.

2 Architectures of the Beamformers

In the following, the signal model for a uniform linear array (ULA) is specified. In the first subsection, the method employed for estimating the signal of interest through space adaptive processing is explained. In the second subsection, a space-time adaptive processor is introduced for estimating the signal snapshots. Also, in the following, the structures of the neural beamformers are explained.

2.1 Space Adaptive Processor

Similar to [15], in an array with M overlapped subarrays of size $N\times1$, $N-1$ antenna elements are shared

between any two adjacent subarrays. Thus, as mentioned in [15], the signal vector of the m-th subarray is calculated as:

$$\boldsymbol{x}_m[k]=\sum_{q=1}^{Q+1}\left(e^{-j(m-1)\Phi_q}\right)\boldsymbol{a}_1(\theta_q)s_{1,q}[k]+\boldsymbol{n}_m[k] \quad (1)$$

In (1), $\boldsymbol{a}_1(\theta_q)$ is the steering vector of the q-th signal received at the first subarray expressed as:

$$\boldsymbol{a}_1(\theta_q)=\left[1,e^{-j\Phi_q},\ldots,e^{-j(N-1)\Phi_q}\right]^T \quad (2)$$

In (2), $\boldsymbol{\Phi}_q = (2\pi/\lambda)d\sin(\theta_q)$ in which d is inter-element space. It is worth noting that, θ_1 is the DoA of the signal of interest and θ_q for $q = 2, \ldots, Q+1$ are the DoAs of the interferences. Also, $s_{1,q}$ denotes the q-th signal received at the first subarray and $\boldsymbol{n}_m$ is the received noise of the m-th subarray. As mentioned in [15], we can assume, for $m = 2, 3, \ldots, M$:

$$s_{m,q}[k]=e^{-j(m-1)\Phi_q}s_{1,q}[k] \quad (3)$$

$$\boldsymbol{a}_m(\theta_q)=\boldsymbol{a}_1(\theta_q) \quad (4)$$

Assuming that K is the number of available snapshots from the time instant k_0 to k_0+K-1, autocorrelation matrix and cross-correlation vector of the subarrays for $m = 1, 2, \ldots, M$ are estimated as:

$$\hat{\boldsymbol{R}}_m \cong \hat{\boldsymbol{R}}_1=\left(\frac{1}{K}\right)\sum_{k=k_0}^{k=k_0+K-1}\boldsymbol{x}_1[k]\boldsymbol{x}_1^H[k] \quad (5)$$

$$\hat{\boldsymbol{r}}_{m,q} \cong \hat{\boldsymbol{r}}_q=\left(\frac{1}{K}\right)\sum_{k=k_0}^{k=k_0+K-1}\left(\boldsymbol{x}_1[k]s_{1,q}^*[k]\right) \quad (6)$$

It can be inferred from (5) and (6) that phase shift $e^{-j(m-1)\Phi_q}$ in (3) does not affect the autocorrelation matrix and cross-correlation vector. The weight vector for estimating all Q interferences is calculated as [15]:

$$\boldsymbol{w}=\sum_{q=2}^{q=Q+1}\boldsymbol{w}_q \quad (7)$$

in which,

$$\boldsymbol{w}_q \cong \hat{\boldsymbol{R}}_1^{-1}\hat{\boldsymbol{r}}_q \quad (8)$$

The interferences for $m = 1, 2, \ldots, M$ are estimated as:

$$\sum_{q=2}^{q=Q+1}s_{m,q}[k]\cong \boldsymbol{w}^H\boldsymbol{x}_m[k]=I_m[k]\cong\sum_{q=2}^{Q+1}e^{-j(m-1)\Phi_q}s_{1,q}[k] \quad (9)$$

The interference vector, i.e. $\boldsymbol{y}^A[k]$ is formed as:

$$\mathbf{y}^A[k]=\left[I_1[k],\ I_2[k],\ \ldots,\ I_M[k]\right]^T\cong\sum_{q=2}^{Q+1}\boldsymbol{a}^A(\theta_q)s_{1,q}[k] \quad (10)$$

where,

$$\boldsymbol{a}^A(\theta_q)=\left[1e^{-j\Phi_q}\ldots e^{-j(M-1)\Phi_q}\right]^T \quad (11)$$

The interference-free signal vector is calculated as [15]:

$$\boldsymbol{x}'[k]=\boldsymbol{x}^A[k]-\boldsymbol{y}^A[k]\cong\boldsymbol{a}^A(\theta_1)s_{1,1}[k]+\boldsymbol{n}^A[k] \quad (12)$$

In (12), $\boldsymbol{x}^A[k]$ and $\boldsymbol{n}^A[k]$ are respectively the signal vector and noise vector received at the first to M-th antenna elements. It can be assumed that the vector of (12) is the received signal vector of a virtual array of size M. Therefore, the same as before, we can group the virtual antenna elements into M' overlapped subarrays each with N' antenna elements. In other words, for $m' = 1, \ldots, M'$, we have:

$$\boldsymbol{x}_{m'}^{IF}[k]=\left[x'_{m'}[k],\ldots,x'_{m'+N'-1}[k]\right]^T \quad (13)$$

In (13), x'_j for $j = m', \ldots, m'+N'-1$ is j-th element of $\boldsymbol{x}'$. The vector of (13) for $m' = 1, \ldots, M'$ can be represented as:

$$\boldsymbol{x}_{m'}^{IF}[k]\cong\left(e^{-j(m'-1)\Phi_1}\right)\boldsymbol{a}_1^{IF}(\theta_1)s_{1,1}[k]+\boldsymbol{n}_{m'}^{IF}[k] \quad (14)$$

In (14), $\boldsymbol{n}_{m'}^{IF}$ is the noise vector received at the m'-th virtual subarray. Also, $\boldsymbol{a}_1^{IF}$ is calculated as:

$$\boldsymbol{a}_1^{IF}(\theta_1)=\left[1,e^{-j\Phi_1},\ldots,e^{-j(N'-1)\Phi_1}\right]^T \quad (15)$$

Similar to (5) and (6), the autocorrelation matrix and cross-correlation vector of the virtual subarrays for $m' = 2, \ldots, M'$ are estimated as:

$$\hat{\boldsymbol{R}}_{m'}^{IF}\cong\hat{\boldsymbol{R}}_1^{IF} \quad (16)$$

$$\hat{\boldsymbol{r}}_{m'}^{IF}\cong\hat{\boldsymbol{r}}_1^{IF} \quad (17)$$

In the m'-th virtual subarray, the desired signal is estimated as:

$$\left(\boldsymbol{w}^{IF}\right)^H\boldsymbol{x}_{m'}^{IF}\approx s_{m'}^{IF} \quad (18)$$

in which, $\boldsymbol{w}^{IF}$ is calculated as:

$$\boldsymbol{w}^{IF}=\left(\hat{\boldsymbol{R}}_1^{IF}\right)^{-1}\hat{\boldsymbol{r}}_1^{IF} \quad (19)$$

By applying the weight vector of (19) to the received signal vector of the virtual subarrays, we have:

$$\boldsymbol{y}^B[k]\approx\left[s_1^{IF},\ldots,s_{M'}^{IF}\right]^T\approx\boldsymbol{a}^B(\theta_1)s_{1,1}[k]+\boldsymbol{n}^B[k] \quad (20)$$

in which, $\boldsymbol{a}^B(\theta_1)$ is computed as:

$$\boldsymbol{a}^B(\theta_1)=\left[1,e^{-j\Phi_1},\ldots,e^{-j(M'-1)\Phi_1}\right]^T \quad (21)$$

Also, $\boldsymbol{n}^B$ represents the effect of noise at the outputs of the beamformers.

Fig. 1 shows our proposed neural space processor. Our proposed neural space processor is composed of three separate DNNs. In this scheme, two CNNs and a bi-LSTM are exploited to estimate the signal of interest. In the proposed beamforming method, at first, a

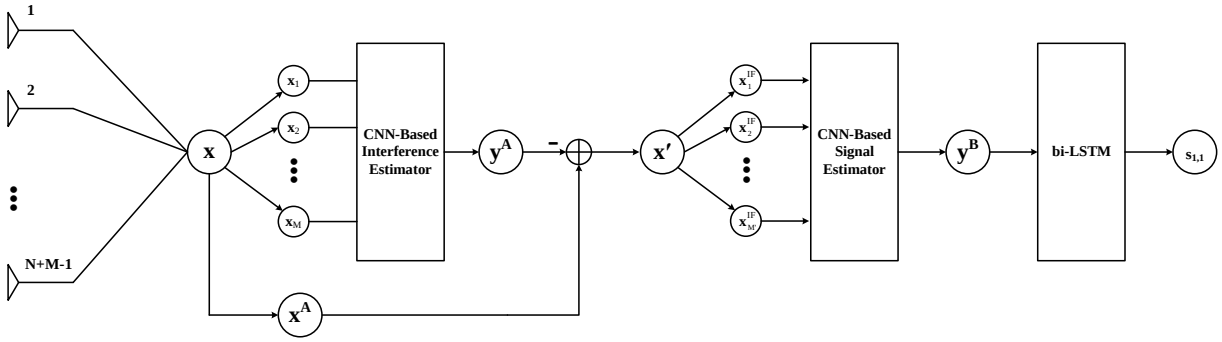


Fig. 1 Architecture of the proposed neural space adaptive processor.

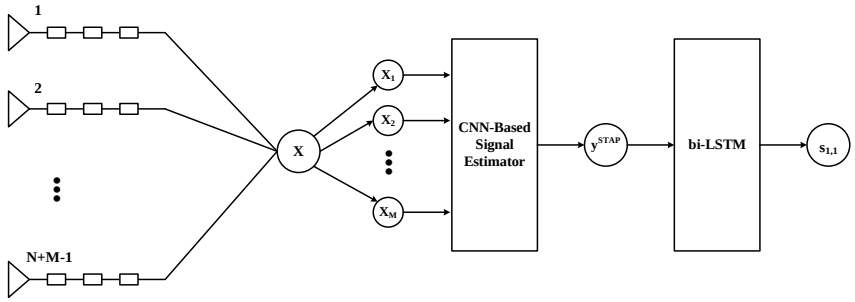


Fig. 2 Architecture of the proposed neural space-time adaptive processor.

CNN-based interference estimator takes in the autocorrelation matrix of the first subarray, i.e. $\hat{\boldsymbol{R}}_1$ to estimate the weight vector of (7), i.e. $\boldsymbol{w}$. Afterward, the estimated weight vector is applied to the received signal vectors of subarrays and the interference vector of (10), i.e. $\boldsymbol{y}^A$ is generated. After that, the interference-free signal vector of (12), i.e. $\boldsymbol{x}'$ is calculated. Afterward, the autocorrelation matrix of (16), i.e. $\hat{\boldsymbol{R}}_1^{IF}$ is calculated and fed into the signal estimator CNN to estimate the weight vector of (19), i.e. $\boldsymbol{w}^{IF}$. After weight vector application, the interference-free signal vector of (20), i.e. $\boldsymbol{y}^B$ is generated. Finally, the snapshots of the signal of interest i.e. $s_{1,1}$ are estimated by the bi-LSTM which is fed with $\boldsymbol{y}^B$.

In comparison with the beamformer introduced in [15], the proposed beamformer produces higher SINR at its output due to employing virtual subarrays and additional CNN. However, the additional CNN increases the computational complexity.

2.2 Space-Time Adaptive Processor

The same as before, by employing M overlapped subarrays each with N antenna elements, we can estimate the desired signal through space-time processing. For a space-time processor, the received signal of an antenna element is processed by an L taps filter. Therefore, the signal vector received at the m-th subarray is represented as:

$$\boldsymbol{X}_m[k]=\left[x_{m,1}[k],\ldots,x_{m,1}\left[k-(L-1)\right],\ldots,x_{m,N}[k],\ldots,x_{m,N}\left[k-(L-1)\right]\right]^T \quad (22)$$

In (22), $x_{m,i}$ for i = 1, 2, …, N is the i-th element of vector of (1), i.e. $\boldsymbol{x}_m$ computed as:

$$x_{m,i}[k]=\sum_{q=1}^{Q+1}\left(e^{-j(m-1)\Phi_q}\right)a_{1,i}\left(\theta_q\right)s_{1,q}[k]+n_{m,i}[k] \quad (23)$$

In (23), $a_{1,i}$ and $n_{m,i}$ for i = 1, 2, …, N are respectively i-th element of $\boldsymbol{a}_1$ and $\boldsymbol{n}_m$ which are defined previously. The space-time autocorrelation matrix and cross-correlation vector between the received signal vector and the desired signal are defined as:

$$\hat{\boldsymbol{R}}_m^{STAP}=\hat{\boldsymbol{R}}_1^{STAP} \quad (24)$$

$$\hat{\boldsymbol{r}}_m^{STAP}=\hat{\boldsymbol{r}}_1^{STAP} \quad (25)$$

By employing the MMSE criterion, the weight vector is computed as:

$$\boldsymbol{w}^{STAP}\cong\left(\hat{\boldsymbol{R}}_1^{STAP}\right)^{-1}\hat{\boldsymbol{r}}_1^{STAP} \quad (26)$$

Afterward, the desired signals for m = 1, 2, …, M are concluded as:

$$s_{m,1}\approx\left(\boldsymbol{w}^{STAP}\right)^H\boldsymbol{X}_m \quad (27)$$

By aggregating the outputs of all beamformers, we conclude:

$$\boldsymbol{y}^{STAP}[k]\approx\left[s_{1,1}[k],\ldots,s_{M,1}[k]\right]^T \approx\boldsymbol{a}^A\left(\theta_1\right)s_{1,1}[k]+\boldsymbol{n}^A[k] \quad (28)$$

We have proposed the architecture of Fig. 2 based on the explained analysis. At first, $\hat{\boldsymbol{R}}_1^{STAP}$ is estimated and

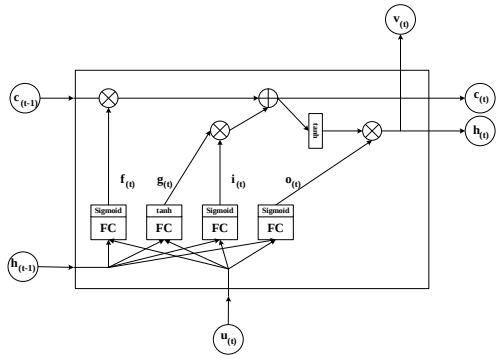


Fig. 3 Structure of LSTM cell.

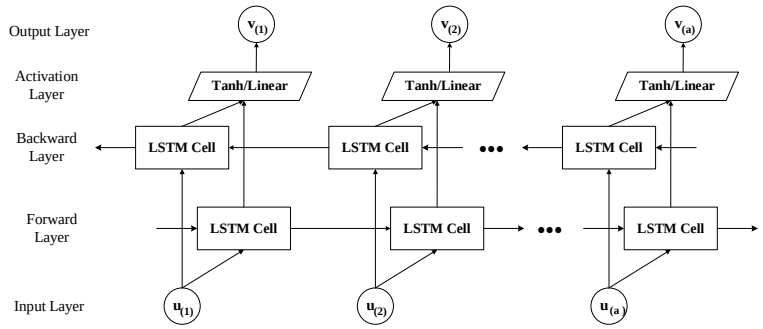


Fig. 4 Structure of bi-LSTM.

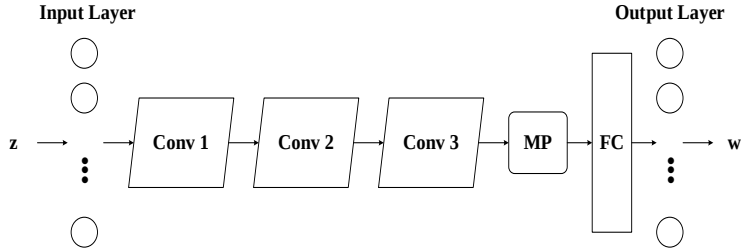


Fig. 5 Structure of CNN.

fed into a CNN. Then, $\boldsymbol{w}^{STAP}$ is estimated by the CNN. After applying the weights to the received signals, the vector of (28) is concluded. Finally, the bi-LSTM is employed to estimate $s_{1,1}$ by taking the vector of (28).

3 Structure of the DNNs

Employing CNN and bi-LSTM in the architecture enables the proposed beamformers to extract temporal and spatial features of the signals. In the following, the detailed structures of the employed CNN and bi-LSTM are described.

As explained in [14], CNNs are used for detecting spatial features of 2-D data. On the other hand, LSTM networks are dominant in detecting temporal features of data and mitigating the effects of noise in the input signal [16, 17]. The properties of CNNs and LSTMs enable our proposed beamformers to estimate the signal of interest and mitigate noise and interferences. In our proposed beamformers, CNNs are employed to estimate the optimum weight vectors by taking in autocorrelation matrix. In addition, the bi-LSTM is exploited to estimate snapshots of the signal of interest by taking in the interference-free signal vector.

As mentioned previously, bi-LSTM networks consist of connected memory cells. As shown in Fig. 3, $\boldsymbol{u}_{(t)}$ and $\boldsymbol{v}_{(t)}$ are respectively input and output data vectors. Also, $\boldsymbol{h}_{(t-1)}$ and $\boldsymbol{c}_{(t-1)}$ are the input state vectors and $\boldsymbol{h}_{(t)}$ and $\boldsymbol{c}_{(t)}$ are the output state vector. Internal nodes of the cells are updated as:

$$\boldsymbol{i}_{(t)} = \sigma\left(\boldsymbol{W}_{ui}^{T}.\boldsymbol{u}_{(t)} + \boldsymbol{W}_{hi}^{T}.\boldsymbol{h}_{(t-1)} + \boldsymbol{b}_{i}\right) \tag{29}$$

$$\boldsymbol{f}_{(t)} = \sigma\left(\boldsymbol{W}_{uf}^{T}.\boldsymbol{u}_{(t)} + \boldsymbol{W}_{hf}^{T}.\boldsymbol{h}_{(t-1)} + \boldsymbol{b}_{f}\right) \tag{30}$$

$$\boldsymbol{o}_{(t)} = \sigma\left(\boldsymbol{W}_{uo}^{T}.\boldsymbol{u}_{(t)} + \boldsymbol{W}_{ho}^{T}.\boldsymbol{h}_{(t-1)} + \boldsymbol{b}_{o}\right) \tag{31}$$

$$\boldsymbol{g}_{(t)} = \tanh\left(\boldsymbol{W}_{ug}^{T}.\boldsymbol{u}_{(t)} + \boldsymbol{W}_{hg}^{T}.\boldsymbol{h}_{(t-1)} + \boldsymbol{b}_{g}\right) \tag{32}$$

$$\boldsymbol{c}_{(t)} = \boldsymbol{f}_{(t)} \otimes \boldsymbol{c}_{(t-1)} + \boldsymbol{i}_{(t)} \otimes \boldsymbol{g}_{(t)} \tag{33}$$

$$\boldsymbol{v}_{(t)} = \boldsymbol{h}_{(t)} = \boldsymbol{o}_{(t)} \otimes \tanh\left(\boldsymbol{c}_{(t)}\right) \tag{34}$$

In (29) to (34), $\boldsymbol{W}_{ui}$, $\boldsymbol{W}_{uf}$, $\boldsymbol{W}_{uo}$, and $\boldsymbol{W}_{ug}$ are the values of the weights of the connections between the fully-connected (FC) layers of the cell and the input data vector. Also, $\boldsymbol{W}_{hi}$, $\boldsymbol{W}_{hf}$, $\boldsymbol{W}_{ho}$, and $\boldsymbol{W}_{hg}$ are the values of the weights of the connections between the FC layers and the short-term state vector. $\boldsymbol{b}_{i}$, $\boldsymbol{b}_{f}$, $\boldsymbol{b}_{o}$, and $\boldsymbol{b}_{g}$ denotes the biases of the FC layers. Fig. 4 shows that the bi-LSTM is composed of two layers, namely, backward and forward layers.

In the following, the architectures of the employed DNNs are described.

3.1 Neural Space Processor

As shown in Fig. 5, in the proposed space processor, the interference estimator CNN consists of three convolutional layers with respectively 16, 32, and 64 kernels whose sizes are equivalent to 3×3. Also, their strides are equivalent to 1. All convolutional layers benefit from zero padding and ELU activation function. After these layers, a max-pooling (MP) layer is designed with kernels whose size and stride are respectively equivalent to 2×2 and 2. In addition, an FC layer with 200 neurons is designed before the output layer of the CNN. Input and output layers are composed of respectively $N(N–1)$ and $2N$ neurons in which N is the number of antenna elements in each subarray.

The architecture of the signal estimator CNN is almost the same as the architecture of the interference estimator CNN except that the numbers of kernels from the first to

third convolutional layers are equivalent to 4, 8, and 16, respectively. Also, the hidden FC layer consists of 50 neurons. Input and output layers of the CNN are composed of respectively $N'(N'–1)$ and $2N'$ neurons in which N' is the number of antenna elements in each virtual subarray.

In the employed bi-LSTM, 100 cells are placed at both backward and forward layers. The FC layers of the cells are composed of 100 neurons whose activation function is hyperbolic tangent. In the proposed scheme, the bi-LSTM is fed with 100 vectors of size $2M'\times1$ in which M' is the number of virtual subarrays. Also, the bi-LSTM outputs 100 vectors of size 2×1. The activation layer of the bi-LSTM benefits from the linear activation function.

Similar to [18], input vectors of the interference estimator CNN is formed of elements located at the upper or lower triangular part of $\hat{\boldsymbol{R}}_1$ which is denoted as $\boldsymbol{b}$:

$$\boldsymbol{b}=\left[R_{12},\ldots,R_{1(N)},R_{23},\ldots,R_{2(N)},\ldots,R_{(N-1)(N)}\right] \tag{35}$$

Inputs of the CNN are generated as:

$$\boldsymbol{z}_1=\frac{\boldsymbol{b}}{\|\boldsymbol{b}\|} \tag{36}$$

In the training phase, ($\boldsymbol{z}_1$, $\boldsymbol{w}$) forms the input/output pair of the interference estimator CNN.

In addition, the signal estimator CNN is trained with ($\boldsymbol{z}_2$, $\boldsymbol{w}^{IF}$) in which $\boldsymbol{z}_2$ is a normalized vector which is formed of elements of the upper or lower triangular part of $\hat{\boldsymbol{R}}_1^{IF}$.

The bi-LSTM is trained with 100 consecutive time samples of $\boldsymbol{y}^B$, and 100 consecutive time samples of $s_{1,1}$. In other words, ($\boldsymbol{y}^B$, $s_{1,1}$) forms the input/output pair of the cells of the bi-LSTM network in the training phase. The loss function of the CNN is computed as follows:

$$\mathcal{L}^{CNN}\left(\boldsymbol{c},\boldsymbol{c}^L\right)=\frac{1}{N_c}\left\|\boldsymbol{c}-\boldsymbol{c}^L\right\|_2^2 \tag{37}$$

In (37), while $\boldsymbol{c}$ is the vector produced at the output of the CNN, $\boldsymbol{c}^L$ is the label of the CNN in the training phase. Besides, N_c is the number of output nodes of the CNN. Similarly, the loss function of the bi-LSTM is denoted as:

$$\mathcal{L}^{bi-LSTM}\left(\boldsymbol{b},\boldsymbol{b}^L\right)=\frac{1}{TN_b}\sum_{t=1}^{T}\mathcal{L}_{(t)}^{bi-LSTM}\left(\boldsymbol{b}_{(t)},\boldsymbol{b}_{(t)}^L\right) \tag{38}$$

In (38), $\boldsymbol{b}_{(t)}$ is the t-th actual output of the bi-LSTM and $\boldsymbol{b}^L{}_{(t)}$ is the t-th desired output of the bi-LSTM and N_b is the size of output vectors of the bi-LSTM. In addition, $\mathcal{L}_{(t)}^{bi-LSTM}\left(\boldsymbol{b}_{(t)},\boldsymbol{b}_{(t)}^L\right)$ is calculated as:

$$\mathcal{L}_{(t)}^{bi-LSTM}\left(\boldsymbol{b}_{(t)},\boldsymbol{b}_{(t)}^L\right)=\left\|\boldsymbol{b}_{(t)}-\boldsymbol{b}_{(t)}^L\right\|_2^2 \tag{39}$$

In (39), T is the number of memory cells in the bi-LSTM. As mentioned previously, in our proposed beamformer, in the interference estimator CNN and signal estimator CNN, $\boldsymbol{c}^L$ is respectively equivalent to the weight vector of (7) and (19). Also, $\boldsymbol{b}^L{}_{(t)}$ is equivalent to $s_{1,1}$.

It is worth noting that, in the training phase, the ADAM optimizer [19] is used for training the networks.

In the test phase, the proposed beamformer is tested with inputs that are not in the training set.

3.2 Neural Space-Time Processor

As mentioned before, in the architecture of the neural space-time processor a CNN-based signal estimator and a bi-LSTM are employed. The architecture of the CNN is almost the same as the architecture of the CNN employed for estimating interferences in the space processor except that its input and output layers consist of respectively $LN(LN–1)$ and $2LN$ neurons. Also, the values of the hyperparameters of the bi-LSTM are almost the same as the values of the hyperparameters of the bi-LSTM employed for space processing except that its activation layer benefits from the Tanh activation function. The inputs of the bi-LSTM are vectors of size $2M\times1$ and its outputs are vectors of size 2×1.

For testing the CNN, the elements of triangular part of $\hat{\boldsymbol{R}}_1^{STAP}$ are aggregated in a vector. Afterward, the concluded vector is normalized and used as inputs of the CNN. The optimal weights of the array i.e. $\boldsymbol{w}^{STAP}$ are estimated by the CNN. Also, in this beamformer, the bi-LSTM takes in 100 consecutive time samples of $\boldsymbol{y}^{STAP}$ to produce 100 consecutive time samples of i.e. $s_{1,1}$.

In comparison with the proposed space processor and also the processor proposed in [15], employing the proposed space-time processor for estimating the desired signal has its own pros and cons. As mentioned previously, there is no need to train the neural space-time processor with weight vectors whose computation requires prior knowledge on DoAs and powers of interferences (such as the weight vector of (7)) to make sure that the beamformer presents a promising performance in the test phase. This property can be considered an advantage for the proposed space-time processor over the proposed space processor. However, one of the disadvantages is that the proposed neural space-time processor has promising performance only for low values of input interference to signal ratio (ISR) compared to the proposed neural space processor. In addition, it can estimate the signals with less complicated structures. For example, while the space processor is capable of estimating the QAM modulated signal, the space-time processor does not have acceptable performance in estimating the signal. However, it has a good performance in estimating signals with less complicated structures such as QPSK modulated signals. It is worth noting that although the proposed space processor produces higher output SINR

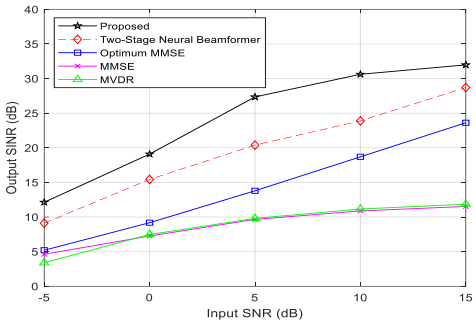


Fig. 6 Output SINR versus input SNR of the space adaptive processors in the first scenario.

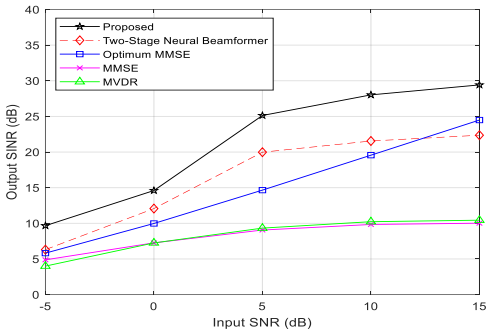


Fig. 7 Output SINR versus input SNR of the space adaptive processors in the second scenario.

than the processor introduced in [15], it incures larger computational complexity due to the additional CNN.

In the following, the performance of the proposed beamformers is presented in detail.

4 Performance Analysis

In this section, the beamformers are analyzed in terms of output SINR for varying input SNR and input ISR. The inter-element distance is equivalent to $\lambda/2$. In the following, at first, the performance of the neural space processor is evaluated. Then, the performance of the neural space-time processor is analyzed.

4.1 Performance of Space Processors

For the neural space adaptive processor, the whole antenna array is grouped into $M = 9$ subarrays each with $N = 10$ antenna elements. Thus, the total number of antenna elements is $N + M - 1 = 18$. Also, the number of virtual subarrays is $M' = 5$ and each virtual subarray consists of $N' = 5$ virtual antenna elements. In the simulations, the signal of interest is a QAM-16 [6] modulated signal which is estimated in presence of chirp interferences [20].

In the following, the scenarios of simulations are explained. In the simulations, the number of available time samples is equivalent to $K = 100$. In these scenarios, the bi-LSTM and signal estimator CNN are trained with the signal source whose DoA differs respectively from –90° to 90° with steps of 2° and from –90° to 90° with steps of 0.1°. The batch size, mini-batch size, and the number of epochs are respectively equivalent to 9100, 1000, and 80 for the bi-LSTM in the training phase. These values are respectively equivalent to 1801, 1800, and 300 for the CNN. The learning rate is equivalent to 0.001 and 0.05 for respectively the CNNs and bi-LSTM in the training phase.

The interference estimator CNN is evaluated in three different scenarios explained in the following.

In the first scenario, the CNN-based interference estimator is trained with multiple interferences whose DoAs are equivalent to 20°, –20°, 40°, –40°, and 60° in training and inference phase. While the DoA of the signal of interest differs from –90° to 90° with steps of 5° in the training phase, its DoA is equivalent to –30° in the test phase. In the test phase, the input ISR is 36.20dB. The output SINR of five different beamformers including the proposed one, optimum MMSE [15], MMSE [1], MVDR [1], and two-stage neural beamformer [15] versus input SNR is shown in Fig. 6. It can be inferred from the figure that in most cases, the output SINR of the proposed beamformer is more than the output SINR of the conventional beamformers and the two-stage beamformer introduced in [15]. The batch size, mini-batch size, and the number of epochs are respectively equivalent to 3600, 3600, and 300 in the training phase.

In the second scenario, two interference sources are employed to train the interference estimator CNN whose DoAs differs respectively from –90° to –30° with steps of 3° and from 30° to 90° with steps of 3°. The DoA of the signal of interest varies from –30° to 30° with steps of 3° in the training phase. In the test phase, the DoA of the signal of interest and interference sources are respectively equivalent to 25°, –70°, and 35°. Input ISR is 20.85dB, in both the training and inference phases. Fig. 7 shows the results for this scenario. The batch size, mini-batch size, and the number of epochs are respectively equivalent to 9261, 4000, and 300 in the training phase.

In the third scenario, besides the desired signal source, an interference source is employed to train the interference estimator CNN. The DoA of the signal of interest differs from 0° to 90° with steps of 3° and DoA of the interference source differs from –90° to 0° with steps of 3°. In the test phase, the DoA of the desired signal source and the interference source are 5° and –5°, respectively. In both the training and inference phases, the input SNR is 5dB. Fig. 8 shows the results of the simulations. The batch size, mini-batch size, and the number of epochs are respectively equivalent to 961, 900, and 500 in the training phase.

To show the robustness of the neural beamformer in rejecting the interferences other than a chirp, we trained the neural beamformer with chirp interferences and evaluate its performance in presence of SCWI [21]. Fig. 9 shows the output SINR of the beamformers in presence of five interferences. The figure confirms the

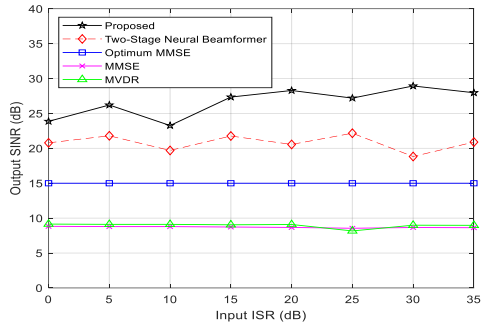


Fig. 8 Output SINR versus input ISR of the space adaptive processors in the third scenario.

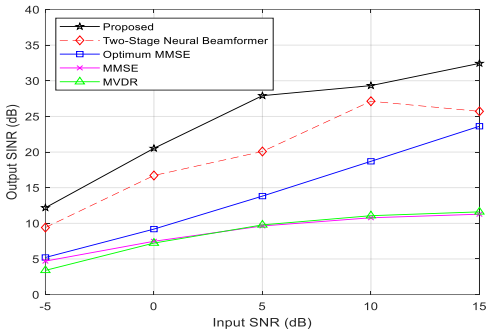


Fig. 9 Output SINR versus input SNR of the space adaptive processors in the first scenario in presence of SCWI.

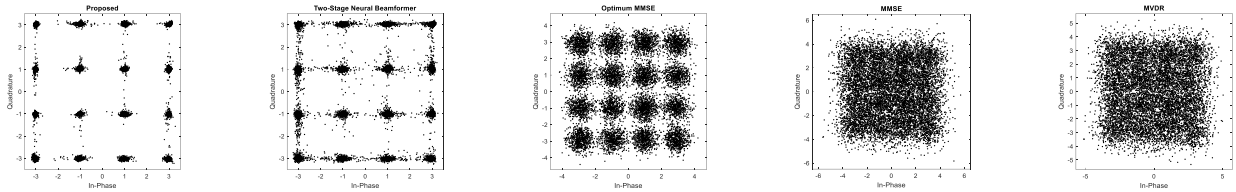


Fig. 10 Scatter plot of constellations of the beamformers' outputs.

acceptable performance of the proposed beamformer in presence of SCWI.

The scatter plot of constellations of the outputs of the beamformers for the third scenario is shown in Fig 10.

4.2 Performance of Space-Time Processors

In the following, the performance of the proposed neural space-time processor in rejecting the interferences is scrutinized in three different scenarios. The simulated ULA consists of 7 antenna elements each with 4 taps. The ULA is grouped into 4 subarrays. The signal of interest is a QPSK modulated signal which is estimated in presence of SCWI or MCWI [21]. In this section, the output SINR of the neural beamformer is compared with the output SINR of MMSE, optimum MVDR, and MVDR beamformers. In the MVDR beamforming method, the optimal weight vector is defined as [22]:

$$\boldsymbol{w}_{MVDR}^{STAP} = \frac{\left(\hat{\boldsymbol{R}}_1^{STAP}\right)^{-1}\boldsymbol{b}_1(\theta_1)}{\boldsymbol{b}_1^H(\theta_1)\left(\hat{\boldsymbol{R}}_1^{STAP}\right)^{-1}\boldsymbol{b}_1(\theta_1)} \tag{40}$$

In (40), $\boldsymbol{b}_1(\theta_1)$ is calculated as:

$$\boldsymbol{b}_1(\theta_1) = \boldsymbol{a}_1(\theta_1) \otimes \boldsymbol{\beta} \tag{41}$$

In (41), $\boldsymbol{\beta}$ is a vector of size $L\times1$ and all of its elements are equivalent to one. When the interference plus noise space-time autocorrelation matrix is available, the optimum MVDR weight vector is computed as:

$$\boldsymbol{w}_{Optimum\ MVDR}^{STAP} = \frac{\left(\boldsymbol{C}_1^{STAP}\right)^{-1}\boldsymbol{b}_1(\theta_1)}{\boldsymbol{b}_1^H(\theta_1)\left(\boldsymbol{C}_1^{STAP}\right)^{-1}\boldsymbol{b}_1(\theta_1)} \tag{42}$$

In (42), $\boldsymbol{C}_1{}^{STAP}$ is the interference plus noise space-time autocorrelation matrix of the first subarray.

In the following, three different scenarios are explained for evaluating the performance of the processors. In these scenarios, the bi-LSTM is trained in the same way mentioned in the previous subsection. Also, the learning rate of the CNN and the bi-LSTM are equivalent to 0.001 and 0.05 in the training phase. The batch size, mini-batch size, and the number of epochs are respectively equivalent to 9100, 1000, and 400 when the bi-LSTM is trained.

In the first scenario, the CNN is trained with three MCWIs whose DoAs are equivalent to 45°, –45°, and 0° in both the training and inference phases. The signal DoA varies from –90° to 90° with steps of 0.1° in the training phase. In the test phase, the DoA of the signal of interest is equivalent to 30°. In this scenario, input ISR is equivalent to 20dB. The performance of four different methods is presented in Fig. 11 when K = 400. The batch size, mini-batch size, and the number of epochs are respectively equivalent to 1801, 1800, and 300 in the training phase.

In the second scenario, in the training phase, the CNN is trained with an SCWI whose frequency varies from 0.5MHz to 2.4MHz with steps of 0.1MHz and its DoA is equivalent to 30° in both the training and inference phases. In the training phase, the CNN is trained with the signal of interest whose DoA varies from –90° to 90° with steps of 2°. In the test phase, the DoA of the signal of interest is equivalent to 50°. In this scenario, input ISR is equivalent to 15dB. Fig. 12 shows the performance of different beamformers. The batch size, mini-batch size, and the number of epochs are respectively equivalent to 1820, 1800, and 300 in the training phase.

In the third scenario, two MCWIs are employed to train the CNN. While, DoA of one of the interference

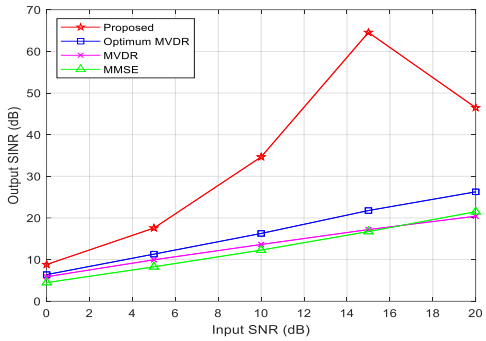


Fig. 11 Output SINR versus input SNR of the space-time adaptive processors in the first scenario.

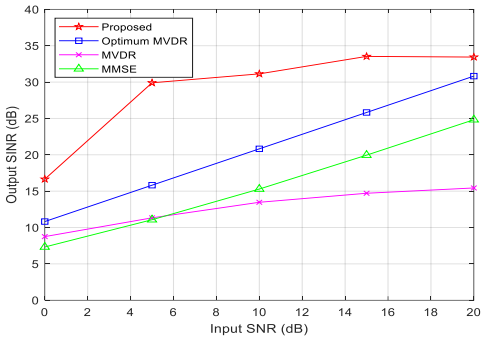


Fig. 12 Output SINR versus input SNR of the space-time adaptive processors in the second scenario.

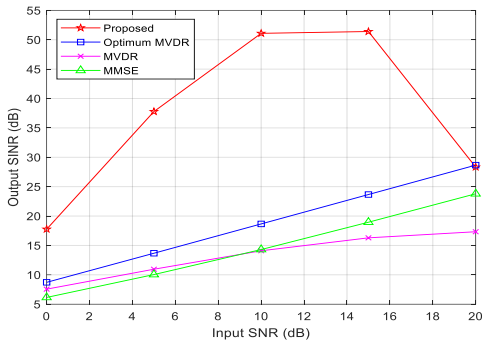


Fig. 13 Output SINR versus input SNR of the space-time adaptive processors in the third scenario.

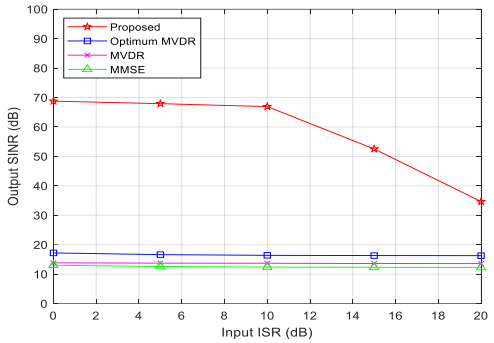
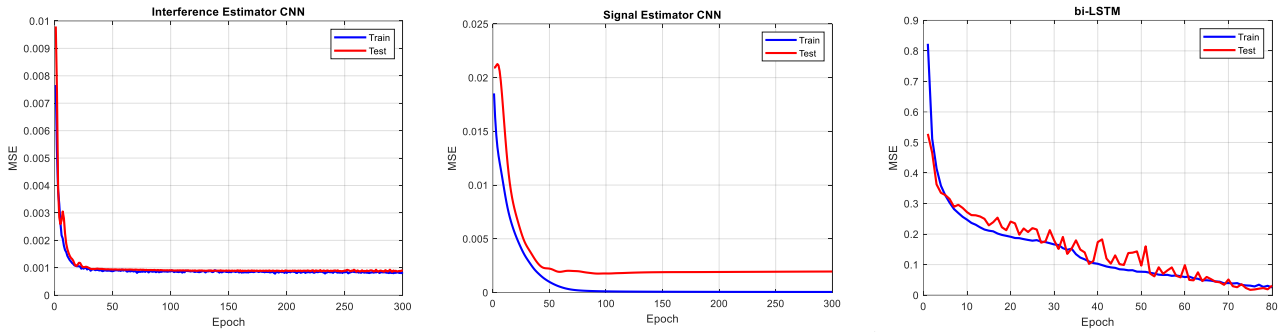


Fig. 14 Output SINR versus input ISR of the space-time adaptive processors in the first scenario.



(a) (b) (c)

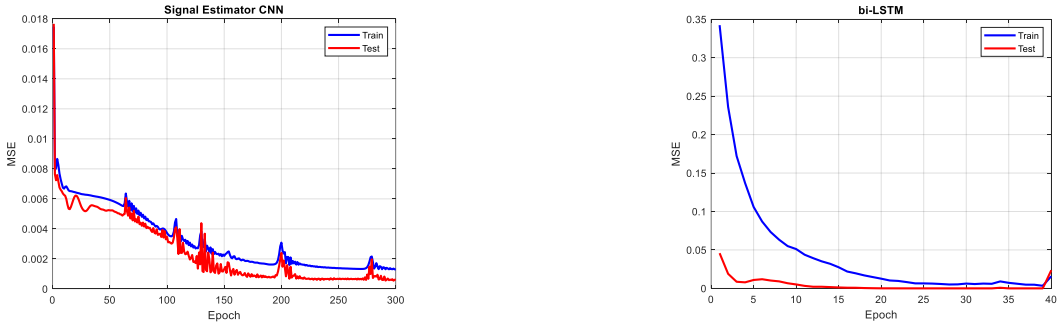
Fig. 15 Output MSE versus epoch of components of the proposed space processor in the training and test phase of the second scenario: a) Interference Estimator CNN, b) Signal Estimator CNN, and c) bi-LSTM.

sources is equivalent to 30° in both the training and inference phases, the DoA of another interference source varies from 0° to 90° with steps of 1° in the training phase. Also, in this phase, the DoA of the signal of interest varies from –90° to 0° with steps of 1°. The neural beamformer is tested when the signal DoA is equivalent to –10° and the DoA of interferences are equivalent to 10° and 30°. Fig. 13 shows output SINR versus input SNR for this scenario in which the input ISR is 18dB. The batch size, mini-batch size, and the number of epochs are respectively equivalent to 8281, 100, and 100 in the training phase.

Fig. 14 presents output SINR versus input ISR for the first scenario. It can be inferred from the figure that the proposed beamformer has effective performance for varying ISR from 0dB to 20dB.

As mentioned in [23], when the mean squared error (MSE) of the model over the test set is not sufficiently low, an underfitted model is obtained. On the other hand, when the gap between the MSE of the model over the training and test set is large, an overfitted model is concluded which is not desirable. To make sure that the training process results in a proper model (neither underfitted nor overfitted), the MSE of the model versus epochs should be analyzed. In Figs. 15 and 16, the final MSE of the model in the training phase and the gap between the MSE of the model in the training and test phase show that training process results in a fitted model.

While using a CPU (Intel Core i7 with 4GB dedicated memory), for the space processing, the proposed beamformer, MMSE beamformer, and MVDR beamformer respectively require 1.0480 s, 0.0049 s, and 0.0049 s to produce 10000 snapshots of the desired



(a) (b)

Fig. 16 Output MSE versus epoch of components of the proposed space-time processor in the training and test phase of the second scenario: a) Signal estimator CNN and b) Bi-LSTM.

signal. For space-time processing, these values are equivalent to 0.8874 s, 0.0059 s, and 0.0019 s, respectively.

5 Conclusion

In this paper, two different DNN-based beamformers are introduced for estimating the signal of interest in presence of noise and interferences. The proposed neural beamformers estimate the signal of interest in either two or three stages.

In the proposed beamforming methods, at first, the whole antenna array is grouped into multiple overlapped subarrays. Afterward, the interference-free signal vector is estimated with the help of either one or two CNNs. Then, a bi-LSTM is employed to estimate the samples of the signal of interest by taking in the interference-free signal vector. The proposed beamformers employ capabilities of CNNs and LSTMs such as their dominance in detecting spatial and temporal features to find the most prominent features of signals. These capabilities significantly improves the performance of beamforming in presence of strong interferences.

While classical beamformers such as MMSE and MVDR are not capable of estimating the snapshots of the desired signal in absence of prior knowledge on DoA of signals, our proposed neural beamformers can estimate the signal of interest when the DoA of the signals is unknown.

Robustness to autocorrelation matrix mismatches is another superiority of the proposed methods over the conventional methods. It can be inferred from the simulation results that output SINR of the proposed methods is more than 10dB higher than the existing methods even when the number of accessible time samples is too low.

Intellectual Property

The authors confirm that they have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property.

Funding

No funding was received for this work.

CRediT Authorship Contribution Statement

P. Ramezanpour: Idea & conceptualization, Research & investigation, methodology, Software and simulation, Analysis, Original draft preparation. **M. R. Mosavi:** Supervision, Revise and editing. **M. de Andrés:** Supervision.

Declaration of Competing Interest

The authors hereby confirm that the submitted manuscript is an original work and has not been published so far, is not under consideration for publication by any other journal and will not be submitted to any other journal until the decision will be made by this journal. All authors have approved the manuscript and agree with its submission to "Iranian Journal of Electrical and Electronic Engineering".

References

[1] C. Fernandez-Prades, J. Arribas, and P. Closas, "Robust GNSS receivers by array signal processing: theory and implementation," *Proceedings of the IEEE*, Vol. 104, No. 6, pp. 1207–1220, 2016.

[2] X. Jiang, J. Chen, H. C. So, and X. Liu, "Large-scale robust beamforming via ℓ_∞-minimization," *IEEE Transactions on Signal Processing*, Vol. 66, No. 14, pp. 3824–3837, 2018.

[3] I. Joo, S. Choi, J. Chun, and C. Sin, "Robust cross-SCORE algorithm against bit transitions for GPS interference suppression," *IEEE Communications Letters*, Vol. 21, No. 6, pp. 1325–1328, 2017.

[4] X. Wu, Y. Cai, R. C. de Lamare, B. Champagne, and M. Zhao, "Adaptive widely-linear constrained constant modulus reduced-rank beamforming," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 53, No. 1, pp.477–492, 2017.

[5] X. Dai, J. Nie, F. Chen, and G. Ou, "Distortionless space-time adaptive processor based on MVDR beamformer for GNSS receiver," *IET Radar, Sonar, and Navigation*, Vol. 11, No. 10, pp. 1488–1494, 2017.

[6] X. Jiang, W. J. Zeng, A. Yasotharan, H. C. So, and T. Kirubarajan, "Quadratically constrained minimum dispersion beamforming via gradient projection," *IEEE Transactions on Signal Processing*, Vol. 63, No. 1, pp. 192–205, 2015.

[7] X. Mestre and M. A. Lagunas, "Finite sample size effect on minimum variance beamformers: optimum diagonal factor for large arrays," *IEEE Transactions on Signal Processing*, Vol. 54, No. 1, pp. 69–82, 2006.

[8] A. M. Elbir, "CNN-based precoder and combiner design in mmWave MIMO systems," *IEEE Communications Letters*, Vol. 23, No. 7, pp. 1240–1243, 2019.

[9] F. B. Mismar, B. L. Evans, and A. Alkhateeb, "Deep reinforcement learning for 5G networks: joint beamforming, power control, and interference coordination," *IEEE Transactions on Communications*, Vol. 68, No. 3, pp. 1581–1592, 2019.

[10] J. Gao, C. Zhong, X. Chen, H. Lin, and Z. Zhang, "Unsupervised learning for passive beamforming," *IEEE Communications Letters*, Vol. 24, No. 5, pp. 1052–1056, 2020.

[11] H. Huang, Y. Peng, J. Yang, W. Xia, and G. Gui, "Fast beamforming design via deep learning," *IEEE Transactions on Vehicular Technology*, Vol. 69, No. 1, pp. 1065–1069, 2019.

[12] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Fifteenth Annual Conference of the International Speech Communication Association*, Singapore, pp. 1–5, 2014.

[13] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM networks," in *IEEE International Joint Conference on Neural Networks*, Montreal, Quebec, Canada, pp. 602–610, 2005.

[14] L. Lin and X. Song. "Using CNN to classify hyperspectral data based on spatial-spectral information," *Advances in Intelligent Information Hiding and Multimedia Signal Processing*, Vol. 64, pp. 61–68, 2017.

[15] P. Ramezanpour and M. R. Mosavi, "Two-stage beamforming for rejecting interferences using deep neural networks," *IEEE Systems Journal*, Vol. 15, No. 3, pp. 4439–4447, 2021.

[16] M. Coto-Jimenez, J. Goddard-Close, L. Di Persia, and H. L. Rufiner, "Hybrid speech enhancement with Wiener filters and deep LSTM denoising autoencoders," in *IEEE International Work Conference on Bioinspired Intelligence (IWOBI)*, San Carlos, Costa Rica, pp. 1–8, 2018.

[17] M. Strake, B. Defraene, K. Fluyt, W. Tirry, and T. Fingscheidt, "Seperated noise suppression and speech restoration: LSTM-based speech enhancement in two-stages," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, NY, USA, pp. 239–243, 2019.

[18] T. Sallam, A. B. Abdel-Rahman, M. Alghoniemy, Z. Kawasaki, and T. Ushio, "A neural-network-based beamformer for phased array weather radar," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 54, No. 9, pp. 5095–5104, 2016.

[19] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, pp. 1–5, 2014.

[20] M. J. Rezaei, M. Abedi, and M. R. Mosavi, "New GPS anti-jamming system based on multiple short-time Fourier transform," *IET Radar, Sonar and Navigation*, Vol. 10, No. 4, pp. 807–815, 2016.

[21] M. R. Mosavi and F. Shafiee "Narrowband interference suppression for gps navigation using neural networks," *GPS Solutions*, Vol. 20, No. 3, pp. 341–351, 2016.

[22] W. L. Myrick, J. S. Goldstein, and M. D. Zoltowski, "Low complexity anti-jam space-time processing for GPS," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Salt Lake City, UT, USA, Vol. 4, pp. 2233–2236, 2001.

[23] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. The MIT Press, Cambridge, Masachussets, 2016.

P. Ramezanpour received his B.Sc. and M.Sc. degrees in Electrical Engineering from K. N. Toosi University of Technology (KNTU) and Iran University of Science and Technology (IUST) in 2016 and 2020, respectively. His research interests include artificial intelligence and adaptive beamforming.

M. Aghababaei was born on 1971, in Tehran, Iran. He received his B.Sc., M.Sc. and Ph.D. degrees in Electrical and Electronic Engineering from Iran University of Science and Technology (IUST), Sharif University of Technology (SUT) and Sharif University of Technology (SUT), Tehran, Iran in 1992, 1994, and 2011 respectively. He is currently faculty member (Assistant Professor) of the Department of Electrical and Electronic Engineering of Imam Khomeini Maritime Sciences University (IKMSU). He is the author of more than 150 scientific publications in journals and conferences. His research interests include circuits and systems design.

M. R. Mosavi received his B.Sc., M.Sc., and Ph.D. degrees in Electronic Engineering from Iran University of Science and Technology (IUST), Tehran, Iran in 1997, 1998, and 2004, respectively. He is currently a faculty member (Full Professor) of the Department of Electrical Engineering of IUST. He is the author of more than 450 scientific publications in journals and international conferences in addition to 12 academic books. His research interests include circuits and systems design. He is also editor-in-chief of "Iranian Journal of Marine Technology" and editorial board member of "Iranian Journal of Electrical and Electronic Engineering".

D. M. de Andrés received the B.Sc. degree in Computer Engineering and the M.Sc. degree in Computer Science from the Department of Informatics, Carlos III University of Madrid, Spain, where he received his Ph.D. degree in 2012. Now, he is a lecturer at the Department of Telematics of the Technical University of Madrid (UPM). His main research subjects, within the GISAI groups at UPM, are internet of things, cyber-physical systems, physically unclonable functions (PUFs), blockchain, knowledge management, information retrieval, and research methods.