3rd URSI AT-AP-RASC, Gran Canaria, 29 May – 3 June 2022

Antenna Array Beamforming Based on Deep Learning Neural Network Architectures

Haya Al Kassir(1), Zaharias D. Zaharis*(1), Pavlos I. Lazaridis(2), Nikolaos V. Kantartzis(1),
Traianos V. Yioultsis(1), Ioannis P. Chochliouros(3), Albena Mihovska(4), and Thomas D. Xenos(1)
(1) School of Electrical and Computer Engineering, Aristotle University of Thessaloniki, Thessaloniki 54124, Greece
(2) School of Computing and Engineering, University of Huddersfield, Huddersfield HD1 3DH, U.K.
(3) Hellenic Telecommunications Organization S.A., Deutsche Telekom Group of Companies, Athens 15122, Greece
(4) Department of Business Development and Technology, Aarhus University, Herning 7400, Denmark

Abstract

The implementation of antenna array beamforming using several neural network (NN) architectures is compared in this paper. Gated recurrent unit, feed-forward NN, convolutional NN, and long short-term memory architectures have been used for the beamforming process. This comparative study is carried out using various metrics, such as the root mean square error, and the computational time for each NN. In addition, the mean absolute divergences of the antenna array main lobe and nulls directions from their respective desired directions have also been used to assess the performance of each beamformer. The neural networks are trained using the simulation results of a 16-element microstrip patch antenna array. It is demonstrated that deep learning-based beamformers are capable of computing optimum antenna array weights in real time and in environments that change with time.

1. Introduction

In wireless communication and radar systems, beamforming is an extremely important process. Beamforming can be defined as a method that concentrates the signal in one desired direction rather than distributing it out in several other directions. Usually, beamforming creates a main lobe that corresponds to the desired signal's direction, as well as many nulls that correspond to the interference signals' directions. Furthermore, beamforming is considered to be a basic tool to deal with the high propagation loss observed in mm-wave communication systems. Null steering beamforming (NSB), and minimum variance distortion-less response (MVDR) method are well-known adaptive beamforming methods. NSB works by suppressing the interference output signals while maintaining the desired output signal undistorted [1], [2]. On the other hand, MVDR minimizes the beamformer output power caused by interference and noise while maintaining the desired output signal undistorted [3].

Despite their advantages in calculating optimum weights, such techniques are very difficult to consistently treat changing environments, where the emitters are moving, because the weights must be computed many times, not to mention the computation complexity due to the large number of antenna array elements. Deep learning (DL) has achieved notable performance results in the beamforming and direction of arrival (DOA) estimation fields, due to its valuable features in changing environments, i.e., quick response, and fast convergence rates [4]. The application of neural networks (NNs) in the beamforming field has attracted the interest of researchers, and a large body of work is available in the literature [5], [6]. In [7], the mismatch problem between the expected and the actual signal steering vectors has been dealt with by a novel robust adaptive beamforming based on NN, while its use for DOA estimation in [8] has achieved the lowest mean square errors. On the other hand, sum-rate maximization (SRM) is a major issue in massive MIMO systems, and an algorithm based on neural networks is able to address it in [9], by choosing users who would be discarded, thus resulting in SRM. A significant part of literature is concerned about DL, because its use is not limited to the calculation of the optimum weights as in [10] and the estimation of the angles of arrivals as shown in [11] and [12]. DL contributed significantly to the improvement of massive MIMO beamforming performance and capacity [13]. In [14], it is used to perform the beam selection and switching for the classification of the beam channel information. On the other hand, DL had a positive effect on ultrasound beamforming, where it has replaced the conventional beamforming process. Moreover, DL is applied directly to radiofrequency channel data to acquire image segmentation for robotic tracking tasks in order to increase speed and accuracy [15], [16].

However, the application of DL in the beamforming field is very challenging and its efficiency depends on many factors. When a massive number of antennas is used, the NNs must be of high complexity, hard to train, and consume much time to provide a result. In this paper, a comparative study between four different NN architectures is performed, thus aiming at finding the most suitable architecture for beamforming applications in terms of accuracy and temporal response.

The remainder of this paper is organized as follows: in section 2, the formulation of the beamforming problem is introduced, followed by a description of the NN structures

in section 3. The results are given in section 4, and finally brief conclusions are provided in the last section.

2. Beamforming problem

In this study, we assume a uniform linear array of M elements placed at equal distances from each other equal to $\lambda/2$ (λ is the free space wavelength). The array receives N+1 monochromatic signals (where N+1<M) at wavelength λ. $s_0(k)$ is the desired incoming signal with angle of arrival θ_0, while the rest of the signals $(s_n(k), n = 1, \dots, N)$ are undesired (interfering) ones with respective angles of arrival $\theta_n, n = 1, \dots, N$, as shown in Fig. 1.

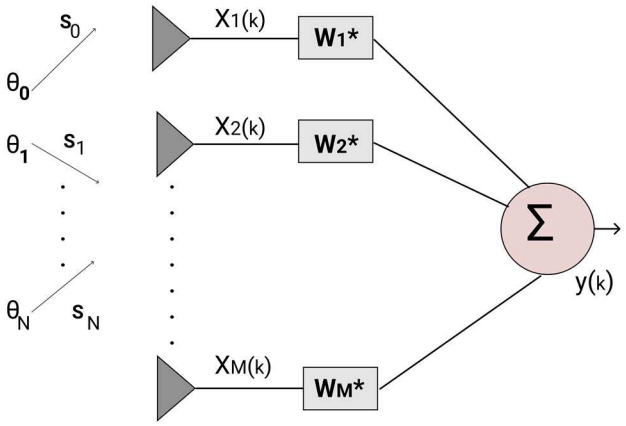

Figure 1. Beamformer structure, where $x_1(k)$, …, $x_M(k)$ are the kth samples of the signals at the inputs of the M array elements, due to incoming signals from various directions.

3. Neural networks description

A NN is typically made up of three layers: the input layer, which contains the data fed to the NN, the hidden layers, which handle all computations, and the output layer, which displays the results. In this paper, we propose four different NN architectures to perform antenna array beamforming: a feed-forward NN (FFNN), a convolutional NN (CNN), a long short-term memory (LSTM) -based NN, and a gated recurrent unit (GRU) -based NN. The FFNN is a conventional NN structure with no feedback connections. The CNN is a major class of NNs that use convolutions rather than multiply-add neurons. LSTM and GRU are advanced recurrent NN (RNN) architectures developed to manage the vanishing gradient problem. The main difference between these two RNN architectures is in the number of gates used. To be able to undertake adequate NN training, we will need a large training data set containing an adequate number of records representing random inputs with their corresponding outputs.

In this paper, a case with three incoming signals is examined, i.e., one desired incoming signal (DIS) and two undesired incoming signals (UIS1 and UIS2). As a result, all the NNs in our study have three inputs, which represent the angles of arrivals of the three incoming signals and 32 neurons in the output layer (see Fig 2), which represent the real and imaginary parts of the excitation weights that feed a 16-element array (M=16). The training dataset is obtained by employing the NSB method. The inputs and outputs of every NN are all appropriately normalized in order to enhance the training, while the number of hidden layers and the activation function types in these simulations are variable and under investigation. The performance evaluation results of the NNs are displayed here for each choice by using a variety of metrics, which will be specified later in the paper.

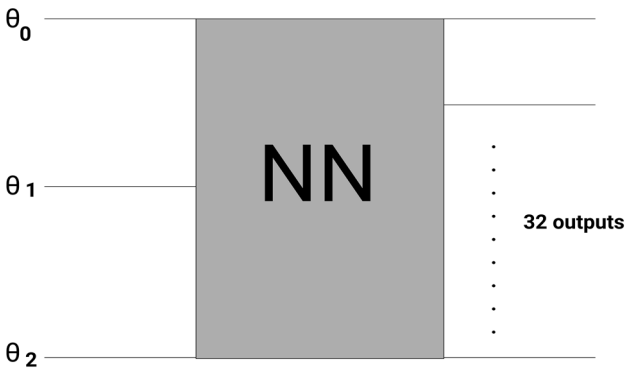

Figure 2. General neural network structure that accepts the angles of arrival of one desired and two undesired incoming signals as input and produces 32 numbers that compose the feeding weights of the antenna array.

4. Simulation results

All the proposed NN structures are created using 3 or 4 hidden layers, while the number of neurons per hidden layer is chosen to be 32, 64 or 128. Also, we choose either the hyperbolic tangent (tanh) or the rectified linear unit (ReLU) as activation function. The data set in this research contains 1,100,000 records, of which 1,000,000 are used to train the NNs and the remaining 100,000 records are utilized to evaluate the NN performance. All the proposed NN structures are trained using 1,000 epochs and the Adam optimizer with a learning rate equal to 0.0001. The metric used for the training process is the root mean squared error (RMSE) of the 32 values at the output layer with respect to the corresponding values given in the training data set of 1,000,000 records.

Figs. 3 and 4 depict the variation of the RMSE versus the number of epochs (iterations) during training of the most complex GRU-NN and CNN structures studied in this paper (these are structures containing 4 hidden layers with 128 nodes per hidden layer and using the tanh activation function). The final RMSE values achieved at the end of the training phase are given in Table I. It seems that the best NN structure is a GRU-NN that contains 4 hidden layers with 128 nodes per hidden layer and uses the tanh activation function. This structure achieves the lowest final RMSE equal to 0.049. As also shown in Table I, a better RMSE is obtained at the end of the training process, when a NN structure becomes more complex (i.e., contains more hidden layers and more nodes per hidden layer). By

comparing all NN structures composed of 3 hidden layers with 64 nodes per hidden layer, it seems that the tanh activation function provides better RMSE values compared to the ReLU function.

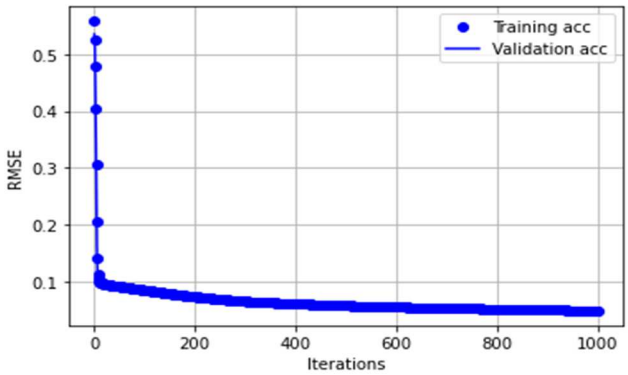

Figure 3. RMSE vs. the number of epochs (iterations) during training of a GRU-NN that contains 4 hidden layers with 128 nodes per hidden layer and uses the tanh activation function.

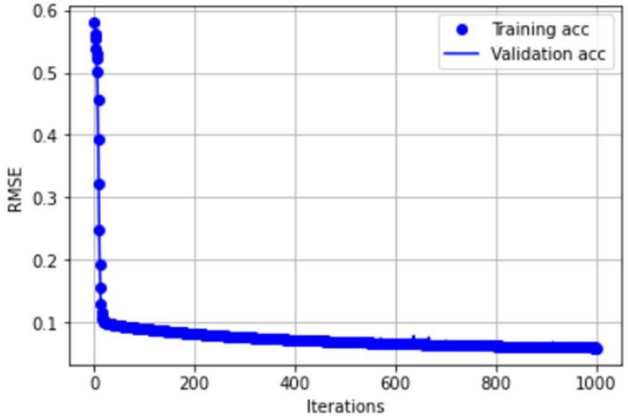

Figure 4. RMSE vs. the number of epochs (iterations) during training of a CNN that contains 4 hidden layers with 128 nodes per hidden layer and uses the tanh activation function.

The NN architectures are also compared between each other as well as with the NSB method in terms of their temporal response (it refers to the mean time required by a NN or the NSB to calculate the optimal feeding weights for every set of incoming signals, i.e., DIS, UIS1 and UIS2, received by the antenna array). As shown in Table II, the FFNN has the fastest response regardless of its structure (i.e., regardless of the number of hidden layers or the number of nodes per hidden layer or the activation function). Its mean computational time is found to be less than 1 msec. The LSTM-NN and GRU-NN are close to each other in terms of computational time, with both not exceeding 2 msecs when using three hidden layers. However, this time becomes almost double for both when using four hidden layers. Despite its worse temporal response compared to the FFNN, the GRU-NN seems to be more promising, because it has better performance compared to all the other NN architectures (see Table I). The CNN needs the most computational time compared to other NNs that have a similar structure. This time is over 5 msecs, unless the ReLU activation function is used. In this case, the computational time is less than 3 msecs, but the CNN has the worst performance, as shown in Table I. Nevertheless, it is worth noting that all NN architectures have a better temporal response compared to the NSB, since the mean computational time of the NSB is approximately 960 msecs.

Finally, we run two scenarios with respective SNR values equal to 0 and 10 dB, in order to compare the GRU-NN-based beamformer with the NSB method. For each set of incoming signals, i.e., DIS, UIS1 and UIS2, received by the antenna array, we estimate the divergence of the main lobe from the direction of DIS, the divergences of the nulls from the directions of UIS1 and UIS2, respectively, as well as the SINR value.

Table I. RMSE values achieved by various NN architectures at the end of their training.

No. hidden layers / neurons per hidden layer / activation	GRU-NN	LSTM-NN	FFNN	CNN
3 / 32 / tanh	0.0829	0.0951	0.0839	0.0973
3 / 64 / ReLU	0.0745	0.0836	0.1044	0.1194
3 / 64 / tanh	0.0661	0.0759	0.0810	0.0819
4 / 128 / tanh	0.0490	0.0502	0.0527	0.0585

Table II. Mean computational time in milliseconds for various NN architectures.

No. hidden layers / neurons per hidden layer / activation	GRU-NN	LSTM-NN	FFNN	CNN
3 / 32 / tanh	1.998	1.997	0.999	4.996
3 / 64 / ReLU	1.997	1.998	0.998	2.998
3 / 64 / tanh	1.998	1.998	0.999	4.995
4 / 128 / tanh	3.999	3.996	0.999	5.995

Table III. The mean value and standard deviation of the main lobe divergence, the nulls divergence, and the SINR.

Method	SNR (dB)	Main lobe divergence (deg) [mean/std]	Nulls divergence (deg) [mean/std]	SINR (dB) [mean/std]
NSB	0	0.436/0.322	0/0	27.196/2.960
GRU-NN	0	0.439/0.328	0.084/0.12	27.145/2.973
NSB	10	0.436/0.322	0/0	37.196/2.973
GRU-NN	10	0.439/0.328	0.084/0.12	36.756/3.109

A statistical analysis of these divergences and SINR values is given in Table III. The mean and standard deviation values of the main lobe divergence and nulls divergence remain the same when the SNR increases from 0 to 10 dB, and only the SINR is affected by the SNR value. This

applies to both the NSB and the GRU-NN-based beamformer. We can also notice that the NSB ensures fair accuracy in the main lobe direction and very high accuracy in the nulls positions regardless of the SNR value. The accuracy of the GRU-NN-based beamformer with respect to the main lobe and nulls positions is inferior but very close to the accuracy provided by the NSB. Nevertheless, what makes the GRU-NN-based beamformer be superior to the NSB is its fast temporal response, which translates to less calculation complexity.

5. Conclusion

In this paper, we have presented a comparison of antenna array beamformers based on four different NN architectures that enable deep learning and provide accurate results together with a very fast response, which is much faster than the well-known NSB method. The NNs have been designed and trained for a 16-element antenna array. The GRU-NN with 4 hidden layers, 128 neurons per hidden layer, and tanh activation is found to have the best performance among the NN architectures studied.

6. Acknowledgments

This research was supported by the European Union, partially through the Horizon 2020 Marie Skłodowska-Curie Innovative Training Networks Programme "Mobility and Training for beyond 5G Ecosystems (MOTOR5G)" under grant agreement no. 861219, and partially through the Horizon 2020 Marie Skłodowska-Curie Research and Innovation Staff Exchange Programme "Research Collaboration and Mobility for Beyond 5G Future Wireless Networks (RECOMBINE)" under grant agreement no. 872857.

References

[1] L. Godara, *Smart Antennas*. Boca Raton, FL, USA: CRC Press, 2004.

[2] F. B. Gross, *Smart Antennas for Wireless Communications With MATLAB*. New York, NY, USA: McGraw-Hill, 2005.

[3] D. D. Feldman and L. J. Griffths, ``A projection approach for robust adaptive beamforming," *IEEE Trans. Signal Process*., vol. 42, no. 4, pp. 867-876, Apr. 1994.

[4] A. H. Sallomi and S. Ahmed, "Multi-layer feed forward neural network application in adaptive beamforming of smart antenna system," *2016 Al-Sadeq Int. Conf. Multidiscipl. IT Commun. Sci. Appl. (AIC-MITCSA)*, 2016, pp. 1-6.

[5] Z. D. Zaharis *et al*., "Implementation of antenna array beamforming by using a novel neural network structure," *2016 Int. Conf. Telecommun. Multimedia (TEMU),* 2016, pp. 1-5.

[6] P. Ramezanpour and M.-R. Mosavi, "Two-stage beamforming for rejecting interferences using deep neural networks," *IEEE Syst. J.,* vol. 15, no. 3, pp. 4439-4447, Sept. 2021.

[7] J. Papari, H. R. Dalili Oskouei, and A. Keshavarz, "Robust adaptive beamforming algorithm based on sampling function neural network*," 2nd Int. Conf. Control, Instrum. Autom*., 2011, pp. 1187-1192.

[8] M. F. Ünlerşen and E. Yaldiz, "Direction of arrival estimation by using artificial neural networks," *Europ. Model. Symp*., 2016, pp. 242-245.

[9] A. Farsaei, A. Sheikh, U. Gustavsson, A. Alvarado, and F. M. J. Willems, "DropNet: An improved dropping algorithm based on neural networks for line-of-sight massive MIMO," *IEEE Access*, vol. 9, pp.29441-29448, 2021.

[10] R. Lovato and X. Gong, "Phased antenna array beamforming using convolutional neural networks," *2019 IEEE Int. Symp. Antennas Propag. & USNC-URSI Radio Sci. Meeting*, 2019, pp. 1247-1248.

[11] J. Hu, Q. Mo, Z. Liu, and Lihan, "Multi-source classification: A DOA-based deep learning approach," *2020 Int. Conf. Computer Engr. Appl. (ICCEA),* 2020, pp. 463-467.

[12] M. Chen, Y. Gong and X. Mao, "Deep neural network for estimation of direction of arrival with antenna array," *IEEE Access*, vol. 8, pp.140688-140698, 2020.

[13] T. Maksymyuk, J. Gazda, O. Yaremko, and D. Nevinskiy, "Deep learning based massive MIMO beamforming for 5G mobile network," *IEEE 4th Int. Symp. Wireless Syst. – Int. Conf. Intel. Data Acquisition Adv. Computing Syst. (IDAACS-SWS),* 2018, pp. 241-244.

[14] H. P. Tauqir and A. Habib, "Deep learning based beam allocation in switched-beam multiuser massive MIMO systems," *2nd Int. Conf. Latest Trends Electr. Engr. Computing Technol*., 2019, pp. 1-5.

[15] A. A. Nair, T. D. Tran, A. Reiter, and M. A. Lediju Bell, "A deep learning based alternative to beamforming ultrasound images," *2018 IEEE Int. Conf. Acoustics, Speech, Signal Process. (ICASSP*), 2018, pp. 3359-3363.

[16] A. A. Nair, M. R. Gubbi, T. Duy Tran, A. Reiter, and M. A. Lediju Bell, "A fully convolutional neural network for beamforming ultrasound images," *2018 IEEE Int. Ultras. Symp*., 2018, pp. 1-4.