Predicting Macro Basis Functions for Method of Moments Scattering Problems Using Deep Neural Networks

Cam Key, *Student Member, IEEE*, and Branislav M. Notaroš, *Fellow, IEEE*

***Abstract*—In this letter, we present research on the application of deep neural networks to predicting macro basis functions for complicated computational electromagnetics problems. We provide error statistics and representative examples for networks trained on simple and complicated datasets of method of moments scattering problems. Notably, we demonstrate that the networks learn generalizable knowledge applicable to problem types on which they were not trained. We conclude that the networks produce encouraging results, especially for cross-validation, and larger training datasets will improve reliability for general scattering problems.**

***Index Terms*—method of moments, machine learning, neural networks, macro basis functions, variational methods, computational electromagnetics.**

I. Introduction

PREVIOUSLY, we proposed a robust application of deep neural networks [1] for accelerating variational methods like finite element method (FEM), method of moments (MoM) and finite difference (FD) method for computational electromagnetics (CEM) and computational science and engineering (CSE) problems [2]. Rather than predicting a solution directly, our approach learns a model to predict a set of macro basis functions on which the problem is then re-solved. We demonstrated in [2] that such an approach meaningfully reduces the number of unknowns (and therefore runtime) of simple 1-D FEM scattering problems while producing solutions of comparable accuracy to classical solutions of substantially higher dimensionality. Although the 1-D FEM slab scattering problem set explored in [2] was ideal for proof of concept, we are unsatisfied with drawing conclusions about the usefulness of our method from such a simple case. In this paper, we explore the ability of neural networks to predict macro basis functions for more-complicated sets of 3-D MoM PEC scattering problems: a set of 10,000 warped cylinders, tori, plates, and spheroids; and a comparably simple set of bent plates. These datasets capture a greater variety and complexity of the hypothetical set of all scattering problems of interest to CEM practitioners, advancing toward validation and a generalizable application of our method. This paper is a continuation of the research in [2], so we present only a brief summary of related literature and our method before detailing the key content of this paper.

Few studies have sought to apply machine learning to CEM and CSE, especially in the modern context of deep learning. Most prior work has focused on predicting quantities derived from a numerical solution given a problem description such as material parameters throughout a domain and an excitation [3]-[7]. Others have focused on predicting the numerical solution itself [8]-[9], effectively attempting to replace the existing methodology (variational methods like MoM) with learned models. Our predicted macro basis function approach, in contrast, leverages learned models to reduce the dimensionality of CEM and CSE problems, then solved by classical methods like MoM, by decreasing the total number of basis functions. This is inspired by some significant advancements in application of neural networks to completing challenging tasks with high accuracy. Such advancements, like [10] and [11], have used existing mathematically formal methods guided by the predictive capability of neural networks. We believe this is critical to the application of machine learning in most engineering contexts due to the fragility of deep neural networks when faced with novel inputs [12]-[14]. Although we have found little existing research that couples neural networks with variational methods in a broadly applicable way, [15]-[20] are in a similar vein of reasoning, applying networks to predict bulk material parameters for faster multi-scale FEM structural mechanics simulations.

II. Background Theory

As in [2], we consider a discretized linear integral or differential equation-based problem with solution S, set of basis functions F, and linear system of form $[A]x = b$, where $f_i \in F$ and x_i denote the i^{th} basis function and associated solution weight, respectively. The weak solution to such a problem with N basis functions is of form

$$\tilde{S} = \sum_{F} x_i f_i \approx S \tag{1}$$

We apply a neural network to predict solution weights, x, given the solution, $\check{x}$, to a computationally inexpensive analogue of the problem solved on a reduced basis, $\check{F}$, most simply, a small subset of the complete basis, F. To compensate for errors in predicted solution weights, we use x predicted by the network not as the final solution, but to

Manuscript first submitted January 4, 2021, resubmitted March 16, 2021. This work was supported in part by the National Science Foundation under grant ECCS-1810492.

Cam Key and Branislav M. Notaroš are with Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523-1373 USA (e-mail: camkey@rams.colostate.edu, notaros@colostate.edu).

generate a set of macro basis functions that can be used to re-solve the problem. To satisfy boundary conditions, we require careful definition of the macro basis functions. Denoting by $F_{boundary} \subset F$ the set of basis functions in the original basis that are nonzero wherever a boundary condition is imposed in the original problem and by $F_{remainder} = F - F_{boundary}$ the remaining basis functions in the original basis, we impose a simple constraint on macro basis functions:

$$f_{macro} = \sum_{F_a} \alpha_i f_i, \quad F_a = F_{remainder} \; or \; F_a = F_{boundary} \tag{2}$$

Note that this constraint is relaxed from that in [2], allowing macro basis functions to include $f_i \in F_{boundary}$ so long as they do not include $f_i \in F_{remainder}$. We then solve the problem for solution weights $\bar{x}$ using the modified basis $\bar{F} \subset F_{macro}$, a chosen subset of all macro basis functions using the network's prediction of x_i as α_i for each f_i present in a macro basis function. In summary, the envisioned method occurs in three steps, where MoM_Solve rigorously computes solution weights on the given basis and excitation, and DNN estimates all solution weights given those for a known, reduced problem:

1 $\check{x} \leftarrow MoM\_Solve(\check{F}, b)$
2 $\leftarrow DNN(\check{x})$
3 $\bar{x} \leftarrow MoM\_Solve(\bar{F}, b)$

Our simple approach to forming macro basis functions is therefore limited by the network's ability to predict x accurately from $\check{x}$, so we seek to evaluate that ability (step 2 above) in this paper.

III. Datasets and Networks

Meshes used for general MoM problems are highly varied both geometrically and topologically, describing a broad range of surfaces and consisting of a variety of surface element types in a wide range of configurations. Our goal here is only to explore the ability of neural networks to predict x for complicated MoM scattering problems, so we chose to constrain all models in our datasets to a single mesh topology: a 20x20 grid of rectangular surface elements. Each element was geometrically 2nd order and had basis function order of 5, with geometric order and basis function order as described in [21] and [22], respectively. Each model in our datasets therefore had the same number of elements and basis functions, all with the same relative ordering in two parametric dimensions. This allows us to leverage common network architectures used for image processing and circumvent challenges associated with mapping surface currents on arbitrary meshes to network inputs for the time being, although we note methods like [23] exist. Such a 20x20 grid of rectangular elements is surprisingly versatile and can be warped and glued to describe a variety of surface types using parametric mappings.

Fig. 1 shows representative objects from the complicated dataset of 10,000 plates, cylinders, tori, and spheroids. These objects vary in length scale from 0.1 to 10 λ and have a wide distribution of aspect ratios and deformations on multiple scales. Fig. 2 shows representative objects from the simple dataset of 1,000 bent plates. These objects have simple deformation patterns and length scales on the order of a wavelength. All objects in the simple dataset shared the same orientation with respect to the incident field and the same aspect ratio (square) in the non-vertical directions. Note that objects in Fig. 1 and Fig. 2 are rendered at the same scale, and

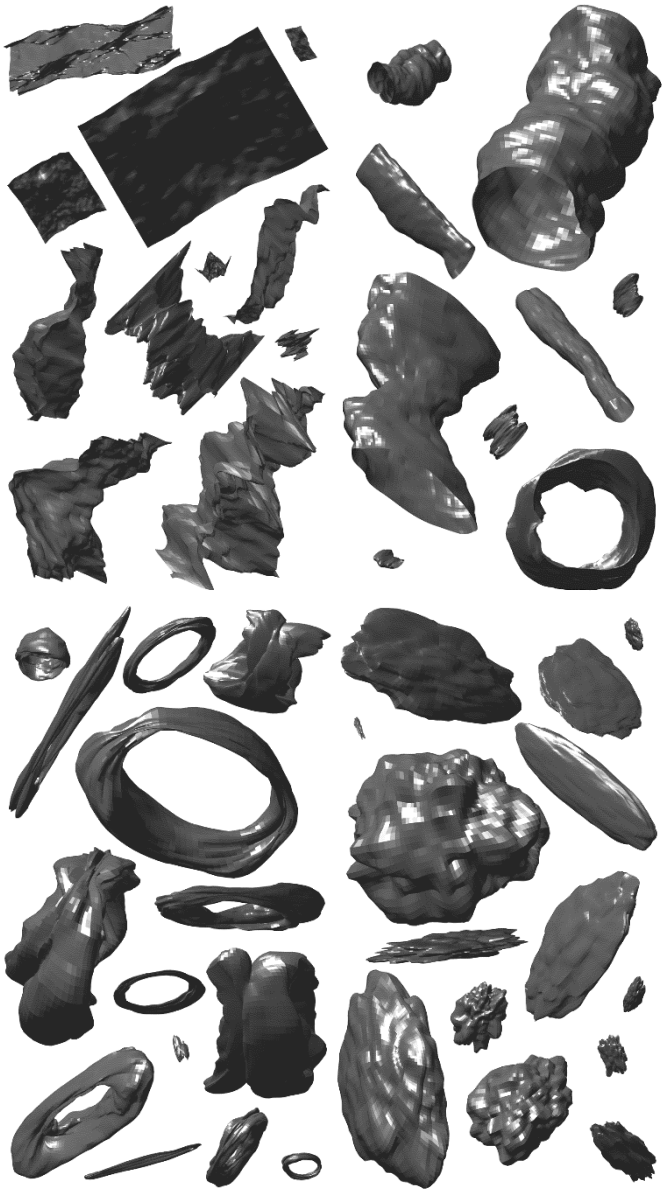

Fig. 1. Representative objects from the complicated dataset. Each object is viewed from the direction of the incident excitation. Planes shown top left; cylinders top right; tori bottom left; spheroids bottom right.

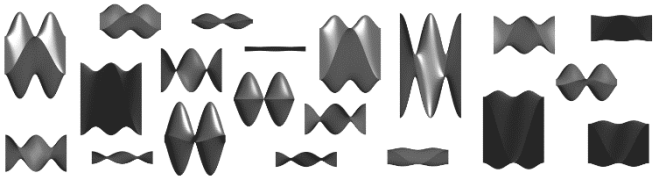

Fig. 2. Representative objects from the simple dataset. Each object is viewed from the direction of the incident excitation.

all objects are shown from the direction of the incident plane wave used as excitation.

Maximally orthogonal divergence-conforming quadrilateral surface current basis functions from [22] were used to discretize surface currents for simulation of scattering from objects in Figs. 1 and 2. We note that, while [22] uses p and s to denote element parametric coordinates in which basis functions are defined, we instead use u and v in the convention of [21]. For instance, using the *P*-expansion and *S*-expansion functions from [22], the u-oriented current expansion over a quadrilateral is given by

$$\mathbf{J}_u(u,v) = \frac{\mathbf{a}_u}{J_2}\sum_{j=0}^{n_v-1}\sum_{k=0}^{n_u}\alpha_{jk}P_j(v)S_k(u) \tag{3}$$

where $\mathbf{a}_u$ is the u-directed unitary vector from [22] and J_2 is the surface Jacobian. The current expansion orders $n_u = n_v = 5$ were chosen for all elements. Note that the expansion is one order lower in the parametric direction perpendicular to its orientation. An analogous form exists for the v-directed expansion. The basis function weights for each element can be displayed intuitively in a 6x5 grid. For symmetry, we present weights in a 6x6 grid for both u- and v-directed current expansions, with the rightmost column left empty (or set to zero) to represent the column of missing basis functions in the perpendicular direction due to the lower sum limit. The complete basis F for each element therefore contains 60 basis functions (30 for u and 30 for v). Per-element basis function grid patterns are shown in Fig. 3. Note that each element will have two such grids of complex-valued basis functions weights associated with it: one for u- and one for v- directed current expansions. Gray cells represent weights in $\check{F}$. Green, orange, and blue cells represent weights to be predicted. Weights in the first two rows in Fig. 3 constitute $F_{boundary}$, so either the first or second row of weights are zero for elements without neighbors (present on plates and cylinders).

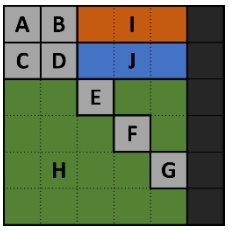

Fig. 3. Basis function weight grids. Function weights in A-G were given to network as input. Weights in regions H, I, and J were predicted by the network.

For such elements, zero-valued weights in the A and B or C and D regions were still input to the network, but network predictions for I or J, appropriately, were ignored when computing error statistics (as they are necessarily set to zero when re-solving the system).

Functions in cells H, I, and J are those intended to be merged into MBFs weighted by network predictions. These regions were chosen for simplicity. If we were attempting to build a solution directly from the weights predicted by the network, we would desire a network that predicted these values on some common scale. However, we intend to use these weights to form MBFs of form (2) for re-solving of the problem on $\bar{F}$, so, ignoring numerical constraints, we do not care about the relative value of weights not in the same region. For this purpose, we normalized all weights within regions H, I, and J such that the value of highest magnitude in each region has unit magnitude.

The same network architecture was used for both the simple and complicated datasets. The architecture consisted of a simple 16-block residual network [24] with no batch normalization and a hyperbolic tangent nonlinearity after the final convolutional layer. We used 128 filters per layer. This architecture was not optimized for the problem but proved sufficient for proof of concept. One version of the network was trained exclusively on 800 examples from the simple dataset, the other 200 held-out for testing. We denote this Network A. Another version, Network B, was trained exclusively on 9,800 examples from the complicated dataset, evenly partitioned between the four subcategories. The remaining 200 objects were held-out as a test set. Both networks were trained using the Adam optimization algorithm [25] with a batch size of 10 and learning rate of 1e-4 for 150 and 500 epochs, respectively. Interestingly, we noted substantial overfitting for longer training runs, indicating these networks will benefit from larger training datasets.

Input to each network was 20x20x28, the 28 channels corresponding to real and imaginary values for both u- and v-directed basis functions in cells A-G. The 20x20 spatial grid corresponds to the 20x20 grid of quadrilateral elements comprising each mesh. Similarly, the output was 20x20x92, the 92 channels corresponding to real and imaginary values for both u- and v- directed basis functions in H, I, and J. Network loss was computed as mean squared error (MSE) between predicted and actual values.

IV. Numerical Results and Discussion

To accurately assess network performance, both networks were evaluated on their test sets after training. Both networks were also tested on the other test set. To present results, we split the 20x20x92 output for each test problem into 400 1x1x92 vectors of element-wise weights. We then further subdivided each vector into four 1x1x23 vectors, for real u, imaginary u, real v, and imaginary v, respectively. Each vector was then rearranged and padded into a 6x6 grid corresponding to weight assignments for H, I, and J in Fig. 3. Values in these grids corresponding to input weights A-G and values in the right-most column were set to zero for presentation of representative examples. Error statistics were computed between these grids and corresponding grids constructed from the true solution weights (normalized in the same way). Fig. 4 shows 15-bin root mean square error (RMSE) histograms. Weight boundaries are the same for all four sub-plots. Fig. 4(a) shows application of Network A to the simple test set, while Fig. 4(b) shows application of Network A to the complicated test set. Similarly, Figs. (c) and (d) show application of Network B to the complicated and simple test sets, respectively.

Figs 5 and 6 show representative examples for each bin, corresponding to histograms in Fig. 4. Fig. 5 shows bin

examples for Figs. 4(a) and (b), while Fig. 6 shows bin examples for Figs. 4(c) and (d). For each, the bin of highest probability (mode) is highlighted green.

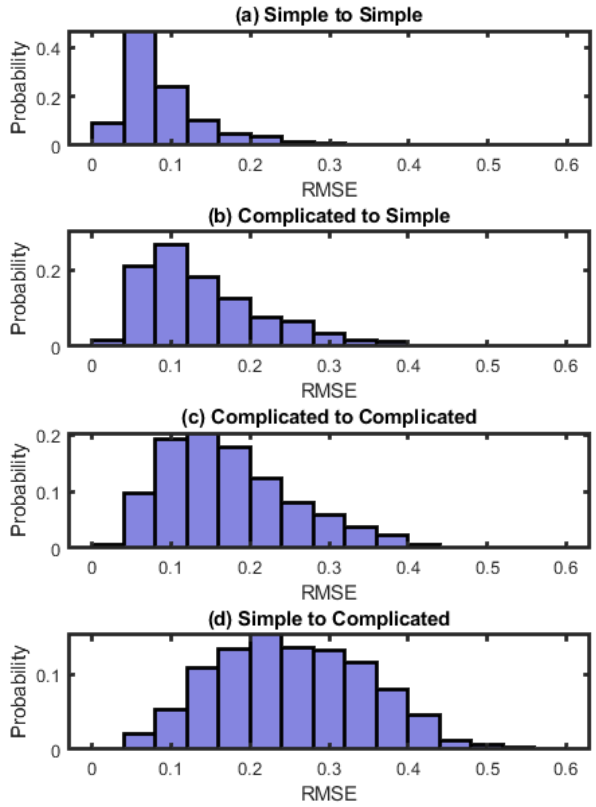

Fig. 4. RMS error histograms for both networks on both validation datasets. (a) Network A on simple set; (b) Network B on simple set; (c) Network B on complicated set; (d) Network A on complicated set.

	Network A		Network B	
Bin#	Predicted/Actual	Error	Predicted/Actual	Error
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				

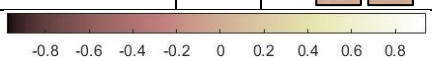

Fig. 5. Representative bin examples for histograms in Fig. 4(a), left, and Fig. 4(b), right. Entries were left blank for Network A on the simple dataset, as no examples fell in bins 10-11 for this case. Neither case produced examples that fell in bins 12 or higher, so these bins were omitted from this plot.

Network A, trained only on the simple dataset, was able to predict weights for objects in the simple test set accurately. This is especially apparent in the representative mode example (green) for Network A in Fig. 5. Network B, trained on the complicated dataset, performed similarly well for the complicated test set, although with somewhat higher error overall. Both results are expected, but we are pleased with the accuracy with which the network's predicted basis function weights on their corresponding test sets. More interestingly, however, were the cross-validation results. Despite the complicated dataset containing no objects similar to those found in the simple dataset (evident by comparison of the plates in Fig. 1 to those in Fig. 2), Network B performed better on the simple test set than it did on its own (the complicated) test set. This indicates that Network B learned generalizable knowledge applicable beyond the complicated dataset on which it was trained. Network A clearly also learned generalizable knowledge, demonstrated in Figs. 4(d) and 6, but at far higher error.

	Network B		Network A	
Bin#	Predicted/Actual	Error	Predicted/Actual	Error
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				

Fig. 6. Representative bin examples for histograms in Fig. 4(c), left, and Fig. 4(d), right. Entries were left blank for Network B on the complicated dataset, as no examples fell in bins 13-15 for this case. Color scale reused from Fig. 5.

V. Conclusion

This letter has presented research on the application of deep neural networks to macro basis function prediction. Using simple and complicated datasets of MoM PEC scattering problems, we showed that deep neural networks are a promising approach to macro basis function prediction for a variety of CEM problems.

REFERENCES

[1] J. Schmidhuber, "Deep Learning in Neural Networks: An Overview," *Neural Networks,* vol. 61, pp. 85-117, Jan 2015.

[2] C. Key and B. M. Notaros, "Data-Enabled Advancement of Computation in Engineering: A Robust Machine Learning Approach to Accelerating Variational Methods in Electromagnetics and Other Disciplines," *IEEE Antennas and Wireless Propagation Letters,* Vol. 19, No. 4, pp. 626-630, April 2020.

[3] R. Mishra, "An overview of neural network methods in computational electromagnetics," *International Journal of RF and Microwave Computer-Aided Engineering,* vol. 12, no. 1, Dec 2001.

[4] T. Nguyen-Thien and T. Tran-Cong, "Approximation of functions and their derivatives: A neural network implementation with applications," *Applied Mathematical Modelling*, vol. 23, no. 9, pp. 687-704, 1999.

[5] L. Lanzi, C. Bisagni, and S. Ricci, "Neural network systems to reproduce crash behavior of structural components*," Computers & Structures*, vol. 82, no. 1, pp. 93–108, 2004.

[6] J. Ghaboussi, J. H. Garrett, and X. Wu, "Knowledge-based modeling of material behavior with neural networks," *Journal of Engineering Mechanics*, vol. 117, no. 1, pp. 132–153, 1991.

[7] J. D. Martín-Guerrero, M. J. Rupérez-Moreno, F. Martinez-Martínez, D. Lorente-Garrido, A. J. Serrano-López, C. Monserrat, S. Martínez-Sanchis, and M. Martínez-Sober, "Machine learning for modeling the biomechanical behavior of human soft tissue," *IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, Barcelona, 2016, pp. 247-253, 2016.

[8] I. Lagaris, A. Likas, D. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," *IEEE Transactions on Neural Networks,* vol. 9, no. 5, pp. 987-1000, Sep 1998.

[9] E. Soliman, M. Bakr, N. Nikolova, "Neural networks-method of moments (NN-MoM) for the efficient filling of the coupling matrix," *IEEE Transactions on Antennas and Propagation,* vol. 52, no. 6, Jun 2004.

[10] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabisal, ''Mastering the game of go without human knowledge,'' *Nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[11] R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, S. Nelson, A. Bridgland, H. Penedones, S. Petersen, K. Simonyan, S. Crossan, D. Jones, D. Silver, K. Kavukcuoglu, D. Hassabis, and A. Senior, "De novo structure prediction with deep-learning based scoring," *Thirteenth Critical Assessment of Techniques for Protein Structure Prediction,* pp. 1-4, Dec 2018.

[12] A. Rozsa, M. Günther, and T. E. Boult, ''Are accuracy and robustness correlated,'' *Proc. IEEE Int. Conf. Mach. Learn. Appl.,* pp. 227–232, 2016.

[13] A. Nguyen, J. Yosinski, and J. Clune, ''Deep neural networks are easily fooled: High confidence predictions for unrecognizable images,'' *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 427–436, 2015.

[14] S. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, ''DeepFool: A simple and accurate method to fool deep neural networks,'' *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 2574–2582, 2016.

[15] A. R. Shahani, S. Setayeshi, S. A. Nodamaie, M. A. Asadi, and S. Rezaie, "Prediction of influence parameters on the hot rolling process using finite element method and neural network," *Journal of Materials Processing Technology*, vol. 209, no. 4, pp. 1920–1935, 2009.

[16] X. Wu and J. Ghaboussi, "Neural network-based material modeling," Thesis, University of Illinois, 1991.

[17] W. Ruijter, R. Spallino, L. Warnet, and A. de Boer, "Optimization of composite panels using neural networks and genetic algorithms," *Second MIT Conference on Compurational Fluid and Solid Mechanics*, K. J. Bathe, Ed. Elsevier Science Ltd, pp. 2359–2363, 2003.

[18] A. A. Javadi, T. P. Tan, and M. Zhang. "Neural network for constitutive modelling in finite element analysis", *Computer Assisted Mechanics and Engineering Sciences*, Vol. 10, pp. 523-5299, 2003.

[19] R. I. Levin and N. A. J. Lieven, "Dynamic finite element model updating using neural networks," *Journal of Sound and Vibration*, vol. 210, no. 5, pp. 593–607, 1998.

[20] Y. Hashash, S. Jung, and J. Ghaboussi, "Numerical implementation of a neural network based material model in finite element analysis," *International Journal for numerical methods in engineering*, vol. 59, no. 7, pp. 989–1005, 2004.

[21] B. M. Notaros, "Higher Order Frequency-Domain Computational Electromagnetics," *invited review paper*, Special Issue on Large and Multiscale Computational Electromagnetics, *IEEE Transactions on Antennas and Propagation*, Vol. 56, No. 8, August 2008, pp. 2251-2276.

[22] M. M. Kostic and B. M. Kolundzija, "Maximally Orthogonalized Higher Order Bases Over Generalized Wired, Quadrilaterals, and Hexahedra," *IEEE Transactions on Antennas and Propagation,* Vol. 61, No. 6, pp. 3135-3148, June 2013.

[23] Z. Sun, E. Rooke, J. Charton, Y. He, J. Lu, and S. Baek, "ZerNet: Convolutional Neural Networks on Arbitrary Surfaces Via Zernike Local Tangent Space Estimation," *Computer Graphics Forum,* pp. 1-13, 2020.

[24] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *International Conference on Learning Representations,* 2014.

[25] K. He, X. Zhang, S. Ren, J. Sun, "Deep Residual Learning for Image Recognition," *IEEE Computer Vision and Pattern Recognition Proceedings,* pp. 770-778, 2016.