(Received 19 January 2026, Accepted 25 February 2026, Scheduled 3 April 2026)

Parametric Model for Coaxial Cavity Filter Using Deep Learning Neural Networks

Nour El H. S. Senasli*, Bouhafs Bouras, Mohammed Chetioui, Lamia Senasli, and Mehdi Damou

Laboratory of Electronics, Advanced Signal Processing & Microwves Department of Telecommunications, Faculty of Technology, University of Saida — Dr. Moulay Tahar Saida, Algeria

ABSTRACT: This study introduces a deep neural network architecture tailored for accurately modeling the parameters of microwave components, with a specific focus on waveguide filters. Unlike simpler neural networks, this architecture is designed to handle the complex relationships that are prevalent in microwave engineering. The model's inputs include the filter's geometric variables and frequency, whereas the S-parameters serve as outputs. To effectively capture these relationships, the Rectified Linear Unit (ReLU) activation function was employed, which is known for its efficiency in managing a significant number of training parameters. This choice allowed the model to better grasp the intricate connection between the S-parameters and geometric variables, and the relationship was found to be more complex than that with frequency. The primary goal is to reduce the overall count of training parameters within the deep neural network while maintaining a level of accuracy similar to that of fully connected neural networks. This study demonstrates the effectiveness of this approach through waveguide filter parametric modeling, highlighting its capacity to accurately model and optimize the electromagnetic response of the filter.

1. INTRODUCTION

Waveguide filters are crucial components of modern wireless communication systems. They enable the precise control of specific frequencies while blocking others. They are crucial for maintaining the signal quality during transmission and reception and ensuring reliable connectivity across devices and networks. Deep learning, a subset of machine learning, involves training artificial neural networks with multiple layers to learn data representations. Techniques such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformers have been successfully applied to tasks such as image recognition [1, 2], natural language processing [3], and autonomous driving. These methods identify patterns in data, thereby enhancing the modeling accuracy for various applications. The integration of deep neural networks into microwave modeling addresses the challenges associated with high-dimensional inputs. These networks excel at handling complex data and improving the modeling accuracy and efficiency of microwave systems. Parametric modeling in microwave engineering predicts component behavior under different conditions, thereby reducing design time and costs. The models utilize geometric variables and frequency as inputs to predict electromagnetic responses, such as S-parameters, offering a comprehensive understanding of component behavior.

Recent advances between 2023 and 2025 have further strengthened the role of artificial intelligence in microwave and terahertz (THz) filter design. Several studies have demonstrated the effectiveness of deep learning-based surrogate models and physics-informed neural networks for electromagnetic device modeling and inverse design [13]. In addition, recent studies have reported improved AI-assisted modeling strategies for microwave filters and waveguide structures, emphasizing enhanced prediction accuracy and computational efficiency [14]. These contributions confirm that AI-based modeling has become a rapidly evolving research direction in high-frequency engineering.

Despite these advances, several limitations remain in existing studies. Many approaches focus primarily on predicting the magnitude of S-parameters while neglecting phase information. Others are restricted to narrowband responses or specific geometrical configurations. Furthermore, limited attention has been given to the broadband prediction of full S-parameters (real and imaginary components) across multiple frequency samples in waveguide filter structures.

To address these limitations, this study proposes a deep learning neural network (DLNN) framework for broadband parametric modeling of a coaxial cavity waveguide filter. The proposed model establishes a nonlinear mapping between five geometrical parameters and the complete complex S-parameters ($\mathrm{Re}(S_{11})$, $\mathrm{Im}(S_{11})$, $\mathrm{Re}(S_{21})$, $\mathrm{Im}(S_{21})$) evaluated over 80 frequency points. Unlike conventional magnitude-based surrogate models, the proposed approach reconstructs the full electromagnetic response, including phase information, thereby providing a more comprehensive characterization of filter behavior.

* Corresponding author: Nour El H. S. Senasli (nourelhoudasara.senasli@univ-saida.dz).

2. DEEP LEARNING NEURAL NETWORK TOPOLOGY FOR WAVEGUIDE FILTER PARAMETRIC MODELING

2.1. Proposed Deep Learning Neural Network Topology

This study proposes a new method using a deep learning neural network (DLNN) as shown in Fig. 1, to construct parametric models of microwave components. These models predict EM responses based on the geometric parameters and frequency inputs. The model architecture incorporates additional hidden layers to capture the intricate relationship between electromagnetic (EM) responses and geometric parameters, while using fewer layers to model the relationship with frequency.

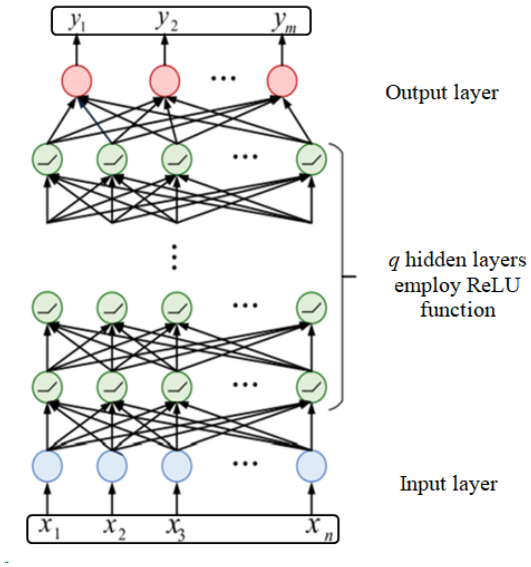

FIGURE 1. The proposed DLNN topology.

Let x represent the external inputs (geometric parameters and frequency) to the DLNN model, where n is the number of inputs to the model. $x = [x_1 x_2 x_3 \dots x_n]^T$ represents the external inputs (geometric parameters and frequency) to the DLNN model with n inputs, and $y = [y_1 y_2 y_3 \dots y_n]^T$ represents a vector of the model's outputs containing the S-parameters of a microwave component with m outputs. The model has hidden layers with the ReLU function, and the weight matrix is denoted as u. The total number of layers in the model was $L = q + 2$, including the hidden, input, and output layers. The deep neural network model is described as follows:

$$y = g(x, u) \tag{1}$$

where g represents the input-output function of the proposed DLNN model.

2.2. Activation Function for the Proposed Deep Neural Network Topology

A deep neural network consists of hidden layers located between the input and output layers. Three or more hidden layers are often recommended for effective learning [4]. The selection of activation functions is critical for the training efficacy of these network models.

For modeling parametric microwave components, where the relationships among the EM responses, geometric parameters, and frequency are complex, using a combination of activation functions can be beneficial. In our proposed method for modeling complex microwave components with deep neural networks, we suggest using Rectified Linear Units as the activation function for hidden neurons.

The ReLU function, illustrated in Fig. 2, is expressed as follows [5, 6]:

$$ReLU\,(x) = Max\,(0, x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

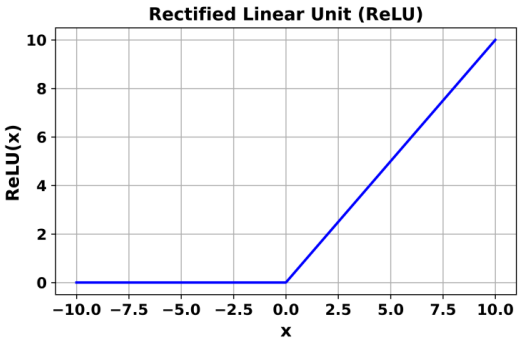

FIGURE 2. ReLU activation function.

The gradient of ReLU is given by:

$$ReLU'\,(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

A ReLU-activated neuron is triggered only when the input is above zero. Its derivative remains constant at one during this activation. This property helps mitigate the issue of vanishing gradients in deep neural networks that use ReLUs. The gradients were calculated through a series of multiplications, each resulting in a value of one. This ensures efficient gradient propagation along the path of active neurons, making ReLUs valuable for facilitating the gradient flow in the network.

2.3. The Feed Forward Computation of the DLNN Model

Let h represent the indices of the layers, where $h = 1, 2, \dots, q$. Let N^h represent the number of hidden neurons in the h^{-th} layer. $Z^{h,i}$ is the output of the i-th neuron in the h^{-th} layer. $Z^{h,i}$ can be calculated using a feed forward process. Let $u_j^{h,i}$ represent the weight between the j^{-th} neuron in the $(h-1)^{-th}$ layer and the i^{-th} neuron in the h^{-th} layer, for $h = 2, 3, \dots, q$, $j = 1, 2, \dots, N^{h-1}$, and $i = 1, 2, \dots, N_1^h$. An additional weight parameter $u_{i_0}^h$ is introduced to represent the bias for the i^{-th} neuron in the h^{-th} layer. Thus, the vector u comprises u_{ij}^h, $h = 1, 2, 3, \dots, q$, $j = 0, 1, \dots, N_1^{h-1}$, and $i = 1, 2, \dots, N_1^h$.

Using direct computation, $Z_i^h Z^{h,i}$ can be calculated as follows [7, 8]:

$$Z^{h,i} = \begin{cases} ReLU\left(\sum\limits_{j=1}^{n} u_{ij}^h x_j + u_{i_0}^h\right) & \text{if } h = 1 \\ ReLU\left(\sum\limits_{j=1}^{N^{h-1}} u_{ij}^h z^{h-1,i} + u_{i_0}^h\right) & \text{if } h = 2, \dots, q \end{cases} \tag{4}$$

After the direct computation, we can retrieve the outputs of the DLNN model from the output layer as follows, where ReLU (·) represents the activation function utilized in the hidden neurons, specifically, the Rectified Linear Unit function [1–8]:

$$y_i = \sum_{j=1}^{N^{q-1}} u_i^q z^{q-1,j} + u_{i0}^q \tag{5}$$

2.4. Learning Algorithm for the DLNN Training

The DLNN model is trained using input samples paired with the corresponding desired model outputs. These samples consist of pairs of geometric configurations, with N representing the total number of frequency samples in the frequency band of interest. The evaluation of the DLNN model's performance during training relies on the error function, which is defined as follows [9, 10]:

$$E(u) = \frac{1}{2mN_gN_f} \sum_{k=1}^{N_g} \sum_{t=1}^{N_f} \sum_{j=1}^{m} \left(y_j(x_k, f_t, u) - d_{kt}^j \right)^2 \tag{6}$$

where y_j is the j^{-th} output of the proposed deep neural network model corresponding to x_k and f_t, and d_{kt}^j is the j^{-th} element of d_{kt}. We harness the widely recognized effectiveness of gradient-based training in neural networks [8]. This method adjusts weights by considering derivatives of the error function (Equation (6)). To tailor this approach to our specific network architecture, we extended the backpropagation concept from multilayer perceptrons (MLP) [11]. This extension involves introducing new formulas to compute the derivatives of the error function with respect to the training parameters, which guide the gradient-based training of our network. Equation (6) defines the error function, which computes the average errors of the model across all geometric samples and frequencies. It calculates the total error of all model outputs for a specific geometric sample (k) at a particular sampled frequency (t), as described in [11].

$$E(\mathbf{u}) = \frac{1}{2} \sum_{j=1}^{m} \left(y_j(x_k, f_t, \mathbf{u}) - d_{kt}^j \right)^2 \tag{7}$$

for the k-th data sample. Let $\delta^{h,i}$ represent the error between the i-th neural-network output and the i-th output in the training data, that is,

$$\delta^{h,i} = y_i(x_k, f_t, \mathbf{u}) - d_{kt}^i \quad h = q+1 \tag{8}$$

Starting from the output layer, this error can be propagated backward to the hidden layers as follows:

$$\begin{aligned} \delta^{h,i} &= \left(\sum_{j=1}^{N^{h+1}} \delta^{h+1,i} u_{ij}^{h+1} \right) z^{h,i} \left(1 - z^{h,i}\right), h \\ &= q-1 \ldots 2, 1 \end{aligned} \tag{9}$$

The symbol $\delta^{h,i}$ signifies the local error at the i-th neuron in the h layer. The derivative of the per-sample error in Equation (7) with respect to a specific neural network weight parameter is expressed as follows:

$$\frac{\partial E(\mathbf{u})}{\partial u_{ij}^h} = \begin{cases} \delta^{h,i} z^{h-1,i} & h = q, q-1, \ldots, 3, 2 \\ \delta^{h,i} x_j & h = 1 \end{cases} \tag{10}$$

The value $\delta^{h,i}$ is computed using Equation (9), and $z^{h-1,i}$ is determined using Equation (4). Equations (8)–(10) are instrumental in calculating the derivative of the error function with respect to each weight parameter. These derivatives guide the adjustment of the weight parameters, as explained in detail in [12].

$$\delta^{h,i} = \begin{cases} \left(\sum_{j=1}^{N^{h+1}} \delta^{h+1,i} u_{ij}^{h+1} \right) & \text{if } y_i > 0 \\ 0 & \text{if } y_i \le 0 \end{cases} \tag{11}$$

The flowchart outlines the process of developing a deep neural network (DNN) for a parametric model of a waveguide filter. It focuses on achieving an optimal network structure and training for accurate predictions.

Parameters:

- E_{be}: Training error before adding the layer
- E_{tr}: Training error after adding a layer
- E_{re}: Required error threshold
- p: Number of hidden layers in the DNN

Process:

1. **Data Generation:**
   - Electromagnetic (EM) simulation was used to generate training and test data with random sampling.
   - Fix the number of neurons in each hidden layer (for now).
   - Initialize $p = 1$ (one hidden layer).
2. **Initial Training:**
   - Train the DNN with p hidden layers.
   - Calculate E_{tr} and test error (E_{te}).
3. **Layer Addition Check ($p > 1$):**
   - If $p = 1$, skip to Step 4 (first training iteration).
   - Otherwise, proceed to Step 3.
4. **Error Improvement Check:**
   - Compare E_{tr} and E_{be}:
     - If $E_{tr} < E_{be}$ (adding layer reduced error), go to Step
     - If $E_{tr} >= E_{be}$ (adding layer did not improve error), go to Step 7 (layer removal).
5. **Error Threshold Check:**
   - Compare Etr and Ere:
     - If $E_{tr} <= E_{re}$ (reached or surpassed threshold), proceed to Step 5 (test error evaluation).

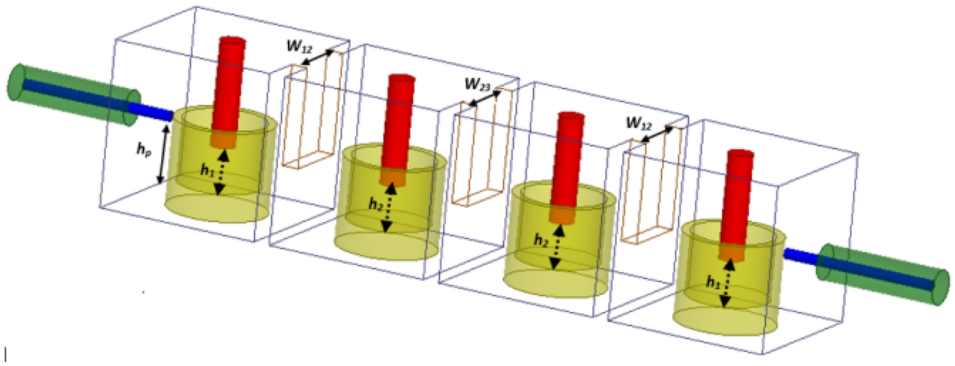

FIGURE 3. 3D waveguide filter.

– If $E_{tr} > E_{re}$ (still above threshold), update $E_{be} = E_{tr}$, add a layer $(p = p + 1)$, and return to Step 2 (retrain with more layers).

6. **Test Error Evaluation:**

- Compare E_{te} and E_{re}:
  - If $E_{te} > E_{re}$ (test error above the threshold), proceed to Step 6 (increase the training data).
  - If $E_{te} <= E_{re}$ (test error acceptable), stop training (optimal model achieved).

7. **Layer Removal (Underfitting):**

- If adding layers did not improve the error (Step 4), the last added layer was removed ($p = p - 1$).
- This aims to reduce model complexity and potentially improve training.

8. **Refined Training:**

- Train the DNN with the optimized p hidden layers.
- Calculate E_{tr} and E_{te}.

9. **Final Error Threshold Check:**

- Compare E_{tr} and E_{re}:
  - If $E_{tr} <= E_{re}$ (reached or surpassed threshold), proceed to Step 10 (test error evaluation).
  - If $E_{tr} > E_{re}$ (still above the threshold), add a layer $(p = p + 1)$ and return to Step 8 (retrain with more layers).

10. **Final Test Error Evaluation:**

- Compare E_{te} and E_{re}:
  - If $E_{te} > E_{re}$ (test error above threshold), increase the training data and return to Step 8 (retrain with more data).
  - If $E_{te} <= E_{re}$ (test error acceptable), stop training (optimal model achieved).

This flowchart emphasizes iterative training, layer addition/removal, and error threshold checks to achieve a DNN model with an optimal structure and performance for waveguide filter modeling.

2.5. Parametric Modeling of the Waveguide Filter

This study aims to develop a parametric model for a waveguide filter, whose structure is illustrated in Fig. 3, utilizing the proposed DLNN topology. The resonant frequency and five geometric variables are used as input variables, namely $x = [f, h_1, h_2, h_p, w_{12}, w_{23}]^T$. However, the real and imaginary parts of S_{11} and S_{21} are used as outputs, namely $y = [\mathrm{Im}(S_{11}), \mathrm{Re}(S_{11}), \mathrm{Im}(S_{21}), \mathrm{Re}(S_{21})]^T$.

During the parametric model development, the proposed DLNN topology was applied as defined in Table 1. The geometric parameters varied within a narrow range. For this example, the frequency range was from 1.75 to 1.95 GHz.

TABLE 1. Training/testing data for the waveguide filter.

Physical dimensions (mm)	min	Max	Count
h_1	*26.3*	*26.6*	*4*
h_2	*28.1*	*28.4*	*4*
w_{12}	*38.4*	*38.7*	*4*
w_{23}	*32*	*32.3*	*4*
h_p	*20.1*	*20.4*	*4*

Different EM simulations were performed using random sampling to generate training and testing data. Geometric samples are distributed within a specified range, where for each set of geometric parameters, the filter is simulated across the selected frequency samples. Assuming that there are 80 frequency samples $(N_f = 80)$, the training/test data sets are obtained for each set of geometric parameters, each corresponding to a different frequency sample.

3. RESULTS AND DISCUSSION

In this study, a new DLNN is proposed to model two different geometrical samples of the developed waveguide filter, as presented in Table 2, setting a training/testing error threshold of 0.001. This customizable threshold indicates the accuracy of the proposed model, which is tailored to meet specific accuracy requirements across different applications. For the proposed filter design, the threshold was selected as it adequately met the required precision.

Figures 4 and 5 compare the predicted S-parameters (S_{11}, S_{21}) from the DLNN model with the desired S-parameters.

TABLE 2. Comparison of training/ test errors in the proposed DLNN model for the two geometrical design samples.

Geometrical samples	h_1	h_2	h_p	w_{12}	w_{23}
A	*26.6*	*28.1*	*20.4*	*38.7*	*32.0*
B	*26.6*	*28.3*	*20.3*	*38.7*	*32.3*

Neural network topology		**Details of hidden layers**	**Total number of weight parameters**	**Training error**	**Testing error**
Proposed DLNN	**Sample A**	*5 hidden layers with* 100 neurons per layer	*1200*	*1.40 *10^{-3}*	*1.393 *10^{-3}*
	Sample B		*1200*	*1.5129 *10^{-3}*	*1.5137 *10^{-3}*

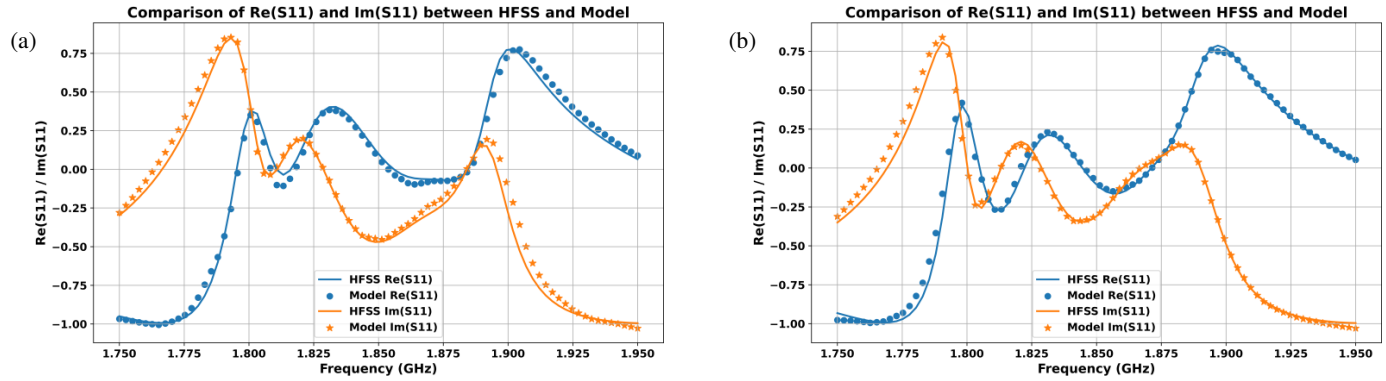

FIGURE 4. S_{11} modeling results for the two selected geometrical waveguide filter design samples, (a) : $x = [26.6, 28.1, 20.4, 38.7, 32.0]^T$ (mm), (b) : $x = [26.6, 28.3, 20.3, 38.7, 32.0]^T$ (mm).

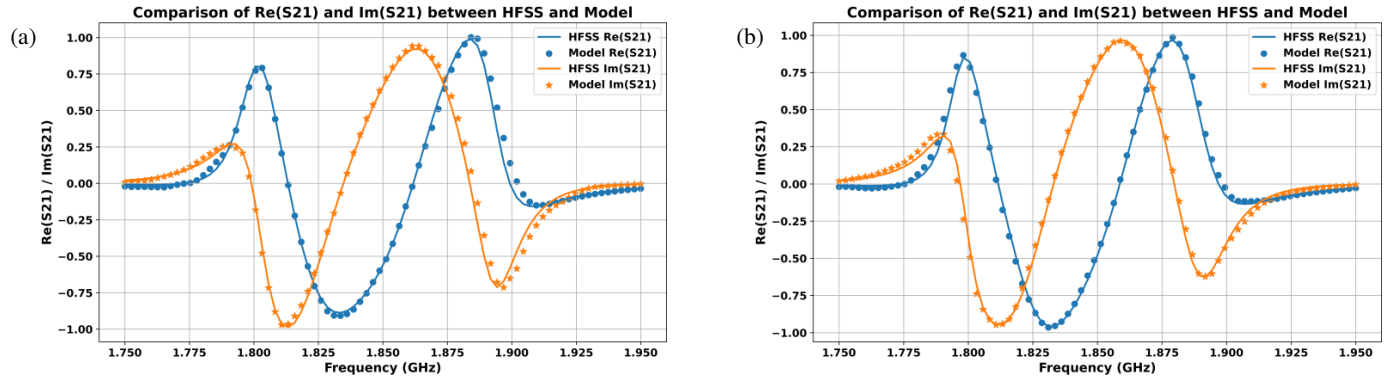

FIGURE 5. S_{21} modeling results for the two selected geometrical waveguide filter design samples, (a) : $x = [26.6, 28.1, 20.4, 38.7, 32.0]^T$ (mm), (b) : $x = [26.6, 28.3, 20.3, 38.7, 32.0]^T$ (mm).

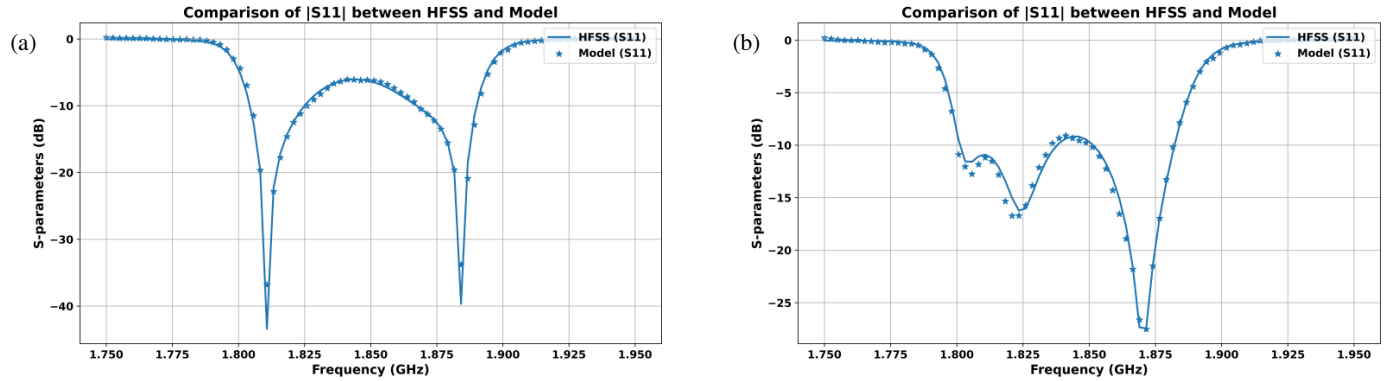

FIGURE 6. The magnitude of the complex quantity S_{11} in dB plotted as function of frequency (GHz) for both the true values and the predicted values of our subdata : (a) : $x = [26.6, 28.1, 20.4, 38.7, 32.0]^T$ (mm), (b) : $x = [26.6, 28.3, 20.3, 38.7, 32.0]^T$ (mm).

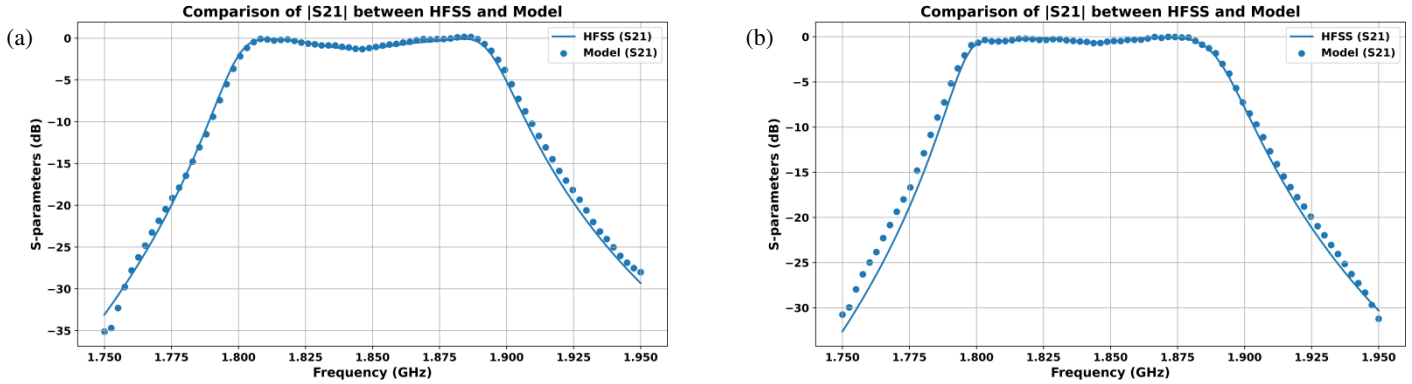

FIGURE 7. The magnitude of the complex quantity S_{21} in dB plotted as function of frequency (GHz) for both the true values and the predicted values of our subdata : (a) : $x = [26.6, 28.1, 20.4, 38.7, 32.0]^T$ (mm), (b) : $x = [26.6, 28.3, 20.3, 38.7, 32.0]^T$ (mm).

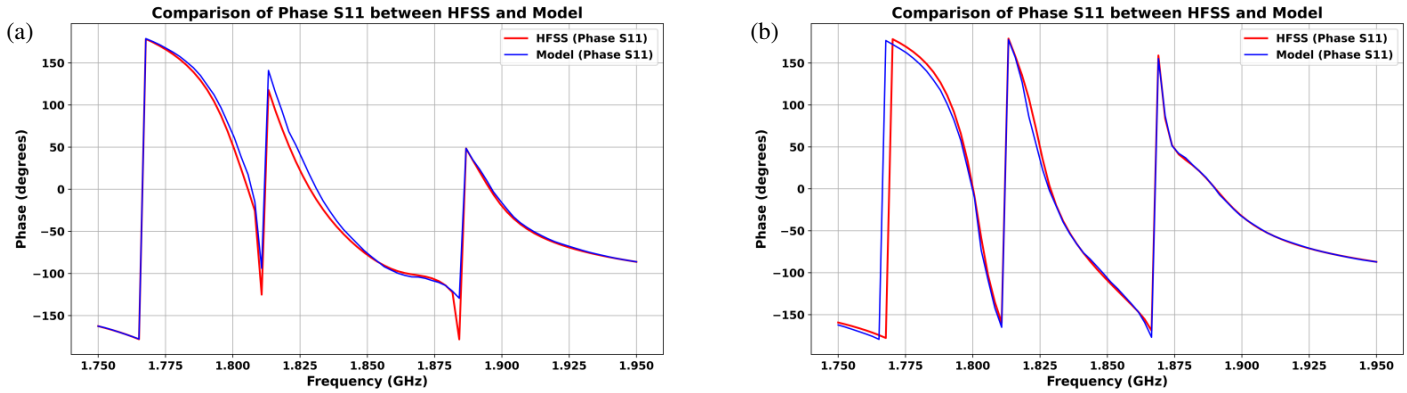

FIGURE 8. The phase of the complex quantity S_{11} plotted as function of frequency (GHz) for both the true values and the predicted values : (a) : $x = [26.6, 28.1, 20.4, 38.7, 32.0]^T$ (mm), (b) : $x = [26.6, 28.3, 20.3, 38.7, 32.0]^T$ (mm).

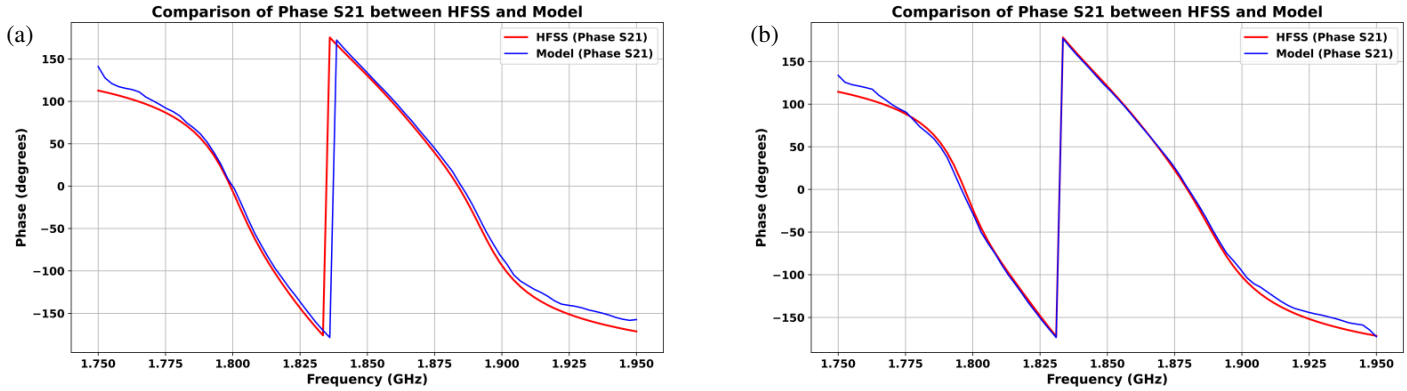

FIGURE 9. The phase of the complex quantity S_{21} plotted as function of frequency (GHz) for both the true values and the predicted values : (a) : $x = [26.6, 28.1, 20.4, 38.7, 32.0]^T$ (mm), (b) : $x = [26.6, 28.3, 20.3, 38.7, 32.0]^T$ (mm).

Two test samples, randomly chosen from the overall test data, were included in the comparison. The figure shows a close alignment between the predicted and expected S-parameter values, indicating a high level of accuracy in the model predictions.

Based on the observations from Fig. 6 and Fig. 7, there is a comparison between the magnitude values (in decibels). The graph indicates a strong similarity between the predicted magnitude values and the actual dataset. The predicted magnitude values (represented by the green line) demonstrate a significant alignment with the actual magnitude values of the dataset (depicted by the blue line). The closeness between the two lines suggests a high level of accuracy in the model predictions.

By observing the alignment of the green line with the blue line, it is clear that the model's predictions effectively capture the magnitude values within the dataset. The closer the green line is to the blue line, the more accurate the model predictions

are in representing the dataset magnitude values. This alignment indicates the reliability and accuracy of the model output in this context.

The code generates plots comparing the magnitude and phase of S_{11} between the true values and model predictions for specific data samples (a) and (b). This comparison assessed the accuracy of the model in capturing the complex behavior of S_{11} at varying frequencies.

In Fig. 8 and Fig. 9, the focus is on comparing the phase values of the dataset with the predicted phase values from the model. The graph shows a strong alignment between the predicted and actual phase values, indicating the high accuracy of the model in predicting the phase values. This alignment suggests that the model effectively learned and represented the phase characteristics of the dataset. This demonstrates the model's ability to accurately predict phase shifts across frequencies, reinforcing its credibility in accurately representing the phase behavior of the data.

4. CONCLUSION

This study proposes a new DLNN topology-based ReLU activation function for designing microwave components, such as waveguide filters. The developed model allocates hidden layers to optimize the training parameters to compute the pathway from inputs to outputs by introducing innovative methods for detecting the error derivative function for the training parameters.

The proposed model demonstrated minimal training/testing error rates, indicating a high ability to understand data training and generalize new ones for overall performance. The proposed deep learning neural network model achieves comparable model accuracy while using fewer training parameters than a fully connected neural network. This efficient parameter utilization represents a streamlined and resource-effective approach that does not compromise the model predictive capacity.

REFERENCES

[1] Farabet, C., C. Couprie, L. Najman, and Y. Le Cun, "Learning hierarchical features for scene labeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 35, No. 8, 1915–1929, Aug. 2013.

[2] Chen, D. and B. K.-W. Mak, "Multitask learning of deep neural networks for low-resource speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, Vol. 23, No. 7, 1172–1183, Jul. 2015.

[3] Collobert, R. and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, 160–167, Jul. 2008.

[4] Bengio, Y., "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, Vol. 2, No. 1, 1–127, 2009.

[5] Huang, A.-D., Z. Zhong, W. Wu, and Y.-X. Guo, "An artificial neural network-based electrothermal model for GaN HEMTs with dynamic trapping effects consideration," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 64, No. 8, 2519–2528, Aug. 2016.

[6] Cho, K., B. V. Merriënboer, C. Gulçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1724–1734, 2014.

[7] Liu, W., W. Na, L. Zhu, J. Ma, and Q.-J. Zhang, "A wiener-type dynamic neural network approach to the modeling of nonlinear microwave devices," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 65, No. 6, 2043–2062, Jun. 2017.

[8] Jin, J., C. Zhang, F. Feng, W. Na, J. Ma, and Q.-J. Zhang, "Deep neural network technique for high-dimensional microwave modeling and applications to parameter extraction of microwave filters," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 67, No. 10, 4140–4155, Oct. 2019.

[9] Zhao, P. and K. Wu, "Homotopy optimization of microwave and millimeter-wave filters based on neural network model," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 68, No. 4, 1390–1400, Apr. 2020.

[10] Feng, F., V.-M.-R. Gongal-Reddy, C. Zhang, J. Ma, and Q.-J. Zhang, "Parametric modeling of microwave components using adjoint neural networks and pole-residue transfer functions with EM sensitivity analysis," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 65, No. 6, 1955–1975, Jun. 2017.

[11] Zhang, Q. J. and K. C. Gupta, *Neural Networks for RF and Microwave Design*, Artech House, 2000.

[12] Zhang, Q.-J., K. C. Gupta, and V. K. Devabhaktuni, "Artificial neural networks for RF and microwave design-from theory to practice," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 51, No. 4, 1339–1350, Apr. 2003.

[13] Karahan, E. A., Z. Liu, A. Gupta, Z. Shao, J. Zhou, U. Khankhoje, and K. Sengupta, "Deep-learning enabled generalized inverse design of multi-port radio-frequency and subterahertz passives and integrated circuits," *Nature Communications*, Vol. 15, No. 1, 10734, 2024.

[14] Javadi, S., B. Rezaee, S. S. Nabavi, M. E. Gadringer, and W. Bösch, "Machine learning-driven approaches for advanced microwave filter design," *Electronics*, Vol. 14, No. 2, 367, 2025.