



RESEARCH ARTICLE

Optimization of ANN-based models and its EM co-simulation for printed RF devices

Shuai Yang¹  | Ahmad Khusro²  | Weiwei Li¹ | Mohammad Vaseem¹ |
Mohammad Hashmi³ | Atif Shamim¹

¹CEMSE Division, King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia

²Electrical Engineering Department, Jamia Millia Islamia, New Delhi, India

³School of Engineering and Digital Sciences (SEDS), Nazarbayev University, Nur-Sultan, Kazakhstan

Correspondence

Shuai Yang, CEMSE Division, King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia.
Email: shuai.yang@kaust.edu.sa

Funding information

King Abdullah University of Science and Technology

Abstract

Printed VO₂ RF switch finds immense potential in RF reconfigurable applications. However, their generic electrical equivalent model is still intangible that can be further integrated in CAD tools and utilize for simulation, analysis and design of RF/microwave circuits and systems. The artificial neural network (ANN) has been gaining popularity in modeling various types of RF components. However, most of these works merely demonstrate the establishment of the ANN-based RF model in the MATLAB environment without involving significant optimization. Furthermore, the integration of such ANN-based RF models in the EM and circuit simulator as well as the co-simulation between the ANN-based model and conventional models have not been demonstrated or validated. Therefore, the earlier reported models are still one step removed from its real RF applications. In this work, by using the fully printed vanadium dioxide (VO₂) RF switch as the modeling example, a systematic hyperparameter optimization process has been conducted. Compared to the non-optimized ANN model, a dramatic improvement in the model's accuracy has been observed for the ANN model with fully optimized hyperparameters. A correlation coefficient of more than 99.2% for broad frequency range demonstrates the accuracy of the modeling technique. In addition, we have also integrated the Python-backed ANN-based model into Advanced Design System (ADS), where a reconfigurable T-resonator band stop filter is used as an example to demonstrate the co-simulation between the ANN-based model and the conventional lumped-based model.

KEYWORDS

artificial neural network (ANN), hyperparameter optimization, co-simulation, modeling, vanadium dioxide (VO₂) RF switch

1 | INTRODUCTION

The rapid advancement in microwave and millimeter-wave technology has led to smaller components with higher operating frequencies. It is well known that high-frequency devices are quite sensitive to dimensional

variations, and even a small change in the devices' size can result in a significant difference in the overall performance. Typically, many simulation iterations are required during the design stage to obtain the optimum dimensions for high-frequency devices. Therefore, there is a strong need to have highly accurate, dimensional-

dependent, fast-simulation-speed, and simulator-compatible models for high-frequency RF devices. Several modeling techniques have been widely used for RF devices, such as equivalent circuit models,^{1–4} electromagnetic (EM) physical models,^{5,6} and S-parameter models.^{7,8}

The equivalent circuit models are fast in the simulation and are capable to include device size parameters and therefore are ideal for parametric optimization. But the accuracy of the equivalent circuit model degrades exponentially as the operating frequency increases because a large number of parasitic elements become prominent at higher frequency ranges that are not included in the model. The main reason is that such parasitic elements are difficult and time-consuming to be characterized accurately. Comparatively, the EM physical models are generally very accurate even for very high frequency ranges. However, they are very computationally expensive and time-consuming. Therefore, they are not very suitable for the parametric simulations, especially with many dimensional variables needed to be tuned and optimized. For the S-parameter model, while it has very high simulation speed and accuracy, it does not include any dimensional information. It is therefore safe to conclude that the S-parameter model is not always appropriate for the parametric optimization simulation.

In recent years, the concept of machine learning (ML), which often occurs in the form of artificial neural networks (ANN), has drawn increasing attention as an unconventional modeling technique in the high-frequency RF device field.^{9–16} Unlike the above-mentioned RF modeling techniques, the ANN-based models need to be trained to achieve the desired level of accuracy before utilization. During the training process, a series of sufficient training datasets are needed to be fed into the model so that the model can learn the input–output pattern automatically, where the input could be the device's dimensions and output could be the S-parameters. Once it is well trained, the ANN-based model is ready for use at EM-level accuracy and equivalent-circuit-level speed. Moreover, the ANN-based model is also highly scalable. Therefore, more input parameters can easily be introduced into the existing models to make the model more versatile and complex. Therefore, the ANN-based model is an excellent option for high-frequency RF modeling as compared to many other techniques.^{17–20}

Several ANN-based RF modeling works have been reported in the literature for different applications, such as coplanar waveguide (CPW) components,²¹ spiral inductors,²² HEMTs,^{23–28} amplifiers,^{29–31} antennas,³² and filters.^{33–35} However, some issues have yet to be solved to enhance its usefulness in the real applications of high-frequency circuit design. Firstly, the accuracy of the ANN-based RF model is largely affected by its parameters (hyperparameters), such as neuron numbers and

activation functions. However, most of the ANN-based EM modeling works only use randomly selected hyperparameters without systematic optimization, leaving considerable room for improvement in the model's accuracy. Secondly, it has been determined that most literature only demonstrates the establishment of ANN-based RF models, where the compatibility and implementation of ANN models in the EM simulator are never mentioned and validated. While some studies have shown the independent use of ANN models in a MATLAB environment without the need for an EM simulator environment, it is still highly desirable to have the co-simulation compatibility between the ANN models and the conventional model in commercial EM simulators.³³

To address the first issue, a systematical hyperparameter optimization procedure needs to be conducted. In this study, the combination of a variety of the key hyperparameters has been evaluated and the results compared. The optimized hyperparameters can be identified by comparing the results. A non-conventional ANN structure is also used to allow more flexible control of the model. In this paper, a very recent work of fully printed vanadium dioxide (VO₂) based RF switch^{36,37} has been employed as a modeling example, in which five input variables (switch length, switch width, switch thickness, temperature, and frequency) are included in the model. The results demonstrate that the accuracy for the optimized ANN model is approximately two orders higher than that for the non-optimized ANN model. Concerning the second issue, an integration mechanism of the ANN model in the Computer-Aided Design (CAD) tools must be established. In this work, we have demonstrated the co-simulation between the ANN-based VO₂ switch model and a popular EM simulator-Advanced Design System (ADS). As a proof of concept for the usefulness of the co-simulation, an example of a reconfigurable bandstop filter is also provided.

The remainder of the article is organized as follows. A brief introduction of the printed VO₂ RF switch is covered in Section 2. Section 3 introduces the ANN structures and the necessary steps for data preparation. The details of the hyperparameter optimization procedure are provided in Section 4, while the results of the optimized ANN-based model are discussed in Section 5. The co-simulation between the ANN-based model and the conventional model in ADS simulator is covered in Section 6. Lastly, Section 7 provides the conclusion and future works.

2 | VO₂ RF SWITCH

In this work, the modeling of a fully printed vanadium dioxide (VO₂) based shunt RF switch is used as an

example. The structure of the RF switch is shown in Figure 1A,B, where the black color represents the printed VO₂ film, the yellow color represents the substrate, and the red color represents the printed conductive trace. The printing process is illustrated in Figure 2. All the devices are fully printed using the screen printer (AUREL 900PA) with the conductive silver ink (DuPont PE819) and the customized VO₂ ink. The CPW line is first printed using the conductive silver ink on the pre-cleaned Sapphire substrate, as shown in Figure 2A. The thickness of the 1-layer printed silver film is 4 μm , which is approximately five times of the skin depth at 10 GHz. Therefore, one pass of printing is sufficient to ensure the RF transmission on the silver traces. The printed silver is annealed at 200°C in a vacuum oven for 1 h to achieve an electrical conductivity of $1.8 \times 10^7 \text{ S/m}$. Next, the customized VO₂ ink is screen printed on the silver traces at the exact position for VO₂ switches, where the alignment between the mask and the substrate is required, as shown in Figure 2B. In the real mask, several alignment marks on the corners have been designed along with the CPW and VO₂ layout. Note that one printing layer of VO₂ film is typically insufficient due to the inadequate thickness and the poor film quality. Therefore, the printing of VO₂ film is repeated until the desired quality and thickness is achieved, as shown in Figure 2B. In this work, two thicknesses (44 and 88 μm , which corresponds to 6 layers and 12 layers, respectively) of VO₂ switches have been fabricated, as is tabulated in Table 1. An ink drying process between each pass of VO₂ printing is performed using the hot gun. Once all the layers are printed, the sample is transferred into the vacuum oven for 1 h of annealing to achieve a better film quality. Afterward, the devices are ready for measurement. The fabricated sample on a sapphire substrate is shown in Figure 1C.

The VO₂ is a metal–insulator-transition (MIT) material. At room temperature, the VO₂ is an insulator, and the RF switch shown in Figure 1 (similar to a simple CPW transmission line) indicates that the RF switch is in ON condition. By increasing the temperature above its transition temperature (68°C), the VO₂ becomes a

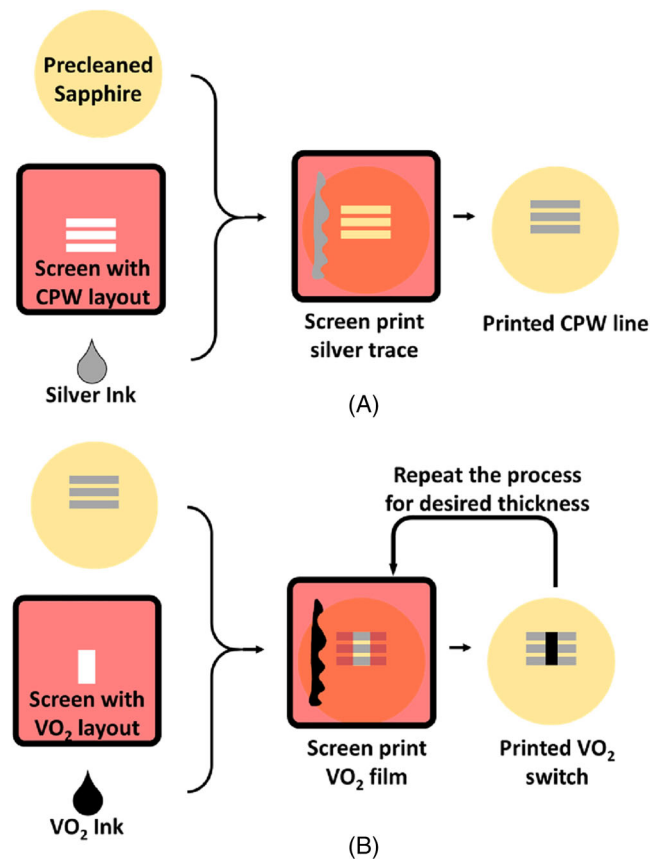


FIGURE 2 The printing process for the fully printed VO₂ switch, including (A) printing CPW trace and (B) printing VO₂ film

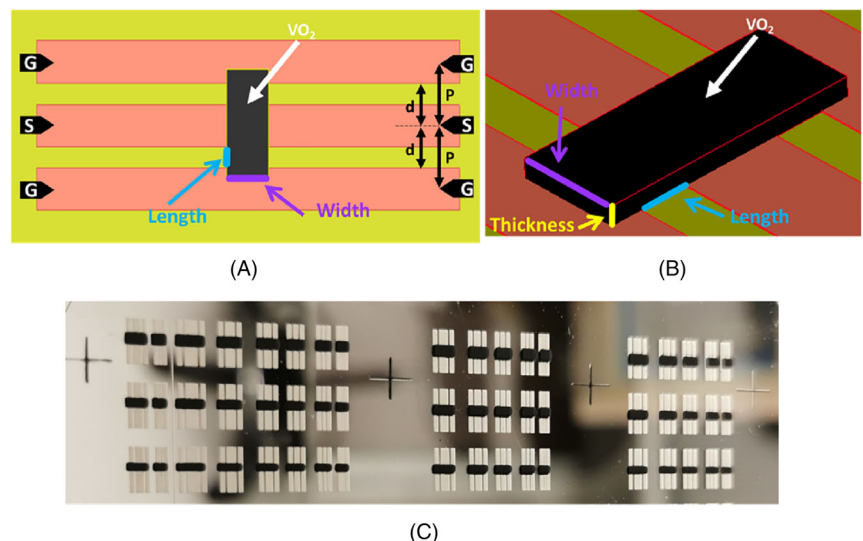


FIGURE 1 (A) 2D (top view) and (B) 3D view of the schematic printed VO₂ switch plot. (C). Fabricated VO₂ switch samples

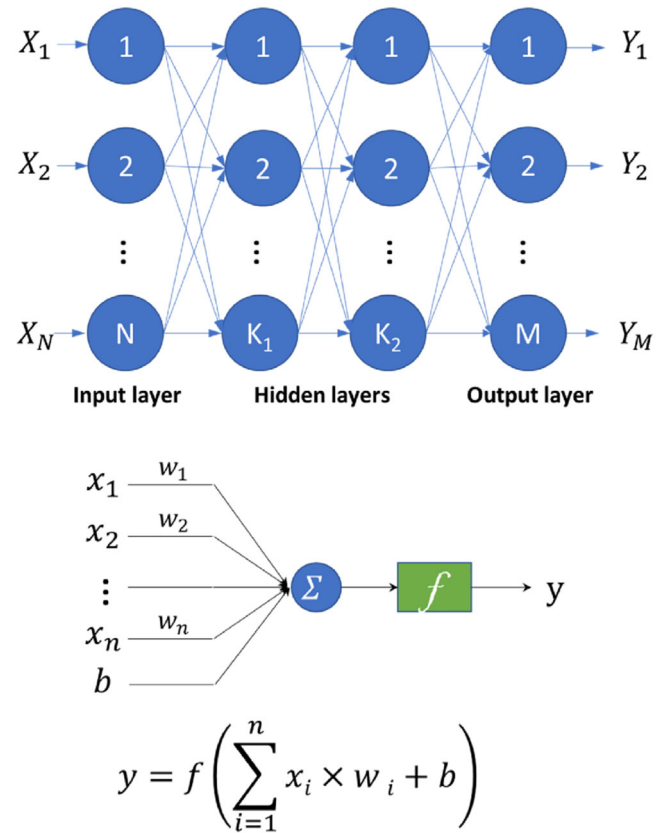
TABLE 1 Input variables ranges (192 combinations - excluding frequency)

Input variables	Values
RF switch width	0.2, 0.4, 0.6, 0.8, 1.0, 1.2 mm
RF switch length	100, 150 μm
RF switch thickness	6 layers (44 μm), 12 layers (88 μm)
Temperature	20°C, 40°C, 50°C, 60°C, 70°C, 80°C, 90°C, 100°C
Frequency	10 MHz to 40 GHz with 5001 points

conductor and it shorts the central signal trace with the ground traces, and thus blocks the RF signal transmission, which indicates the OFF state of the RF switch. The ON-OFF transition is not abrupt because the conductivity of the VO₂ switch is continuously changing with the temperature. Thus, the output S-parameter is a function of the temperature. Similarly, the width, length, and thickness of the RF switch also affect its S-parameter. To achieve a more accurate mimicking of the behavior of the fabricated VO₂ switch, the measurement results are used as the raw data. Due to the constraints of the printer's printability as well as the RF probe's measurability, a total of 192 sets of raw data are collected with different variable value combinations. Table 1 summarizes the input variable ranges.

3 | ANN STRUCTURES AND DATA PROCESSING

The structure of a typical feed-forward fully connected ANN is shown in Figure 3 (up),³⁸ where multiple inputs (X_1, X_2, \dots, X_N) and outputs (Y_1, Y_2, \dots, Y_M) are connected to the neural network. All the arrows in Figure 3 (up) indicate the data flow from the inputs to the outputs through the blue circles. Each blue circle represents a neuron whose structure is shown in Figure 3 (below). As can be seen, the output y is a function f of the input values (x_1, x_2, \dots, x_n), the weights (w_1, w_2, \dots, w_n) and the bias value b . The mathematical relationship is also given in Figure 3 (below). The output value y will be the input value for the neurons in the subsequent layers until it propagates to the output layer. There are several drawbacks to the conventional ANN structure shown in Figure 3(up). Firstly, all the outputs are affected by any neuron in the hidden layer, which means that a small adjustment to any neuron in the hidden layers would affect all the output results. This is not desirable for the case of only a few outputs needing improvement during

**FIGURE 3** Fully connected ANN structure with two hidden layers (up), Functional structure diagram of an individual neuron and its mathematical representation (below)

the model training stage because it can undermine the accuracy of the other outputs. Secondly, with the knowledge that the model complexity is proportional to the number of hidden layers and number of neurons, it becomes impossible for such a structure to increase the complexity for only some of the outputs. Instead, the complexity increases for all the outputs provided the number of hidden layers or neurons increases. It results in a significant increase in the overall ANN model size and, thus, dramatically slows down the training process.

To alleviate the above-mentioned drawbacks, a new ANN structure, shown in Figure 4, is proposed and used in this work. As demonstrated in the figure, every output has its own set of neurons in the hidden layers (color-coded), which are more independent from each other. Therefore, it is more convenient to adjust individually for each output on request, such as increasing the training steps to a specific output or increasing the model complexity to a specific output. It can be concluded that the proposed ANN structure provides far more flexibility than the conventional ANN structure does.

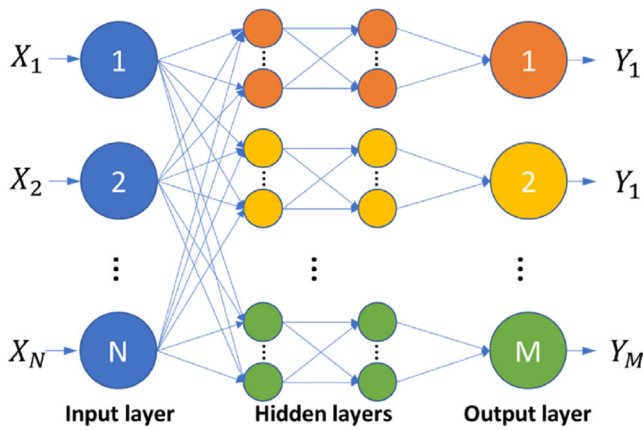


FIGURE 4 Proposed ANN structure with non-shareable neurons for the outputs

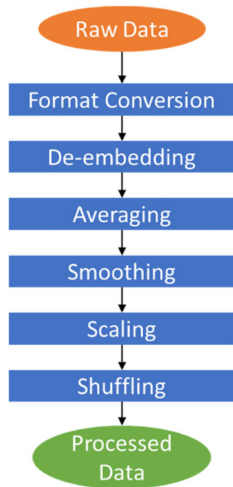


FIGURE 5 Diagrams of preprocessing steps

An ANN model needs to be trained to achieve the desired accuracy. In the training process, all the ANN parameters (i.e., weights and bias) are updated using an optimization algorithm. Sufficient input–output data need to be fed into the ANN model for the model to learn the input–output patterns. Once the training process is completed and the ANN is well trained, the ANN model can use and predict the output result accurately. However, in most cases, the input–output raw data are not suitable for use directly in the training stage. Therefore, some data preprocessing steps are required. In this VO₂ switch example, six steps are required in the preprocessing stage. The data preprocessing flow diagram is shown in Figure 5.

The first step is to do the format conversion of the S-parameter from dB/angle to real/imaginary. The abrupt

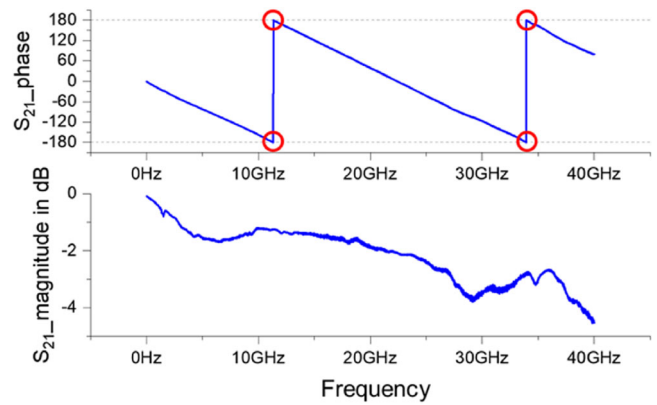


FIGURE 6 S_{21} plot of a randomly picked measured raw data in the form of phase and magnitude dB

value changes in the raw data can cause problems in the ANN training stage. By performing the conversion, the abrupt changes on the phase value are avoided, as illustrated in Figures 6 and 7. Moreover, converting the S-parameter from dB/angle to real/imaginary automatic sets the value bound in the range of $\{1, 1\}$, which is also beneficial as it avoids the potential saturation problem in the model training stage. Following the conversion, the derived data must undergo a de-embedding process. During the de-embedding, the effect of the unnecessary feeding lines shown in Figure 8 is removed to ensure that the resultant S-parameter only represents the RF switch itself, represented by matrix [B]. The de-embedding equation is also provided in Figure 8. The averaging step follows the de-embedding. The VO₂ switch is a symmetrical passive device, whose S-parameter could be reduced by half because S_{11} and S_{22} are identical and S_{12} and S_{21} are identical. However, due to the systematic and random error from the measurement setup, S_{11} (or S_{12}) and S_{22} (or S_{21}) are not identical, but they have a small difference. Therefore, in the averaging step, the arithmetic average of S_{11} and S_{22} will be taken as S_{xx} , and the arithmetic average of S_{21} and S_{12} will be taken as S_{xy} .

Since the raw data is collected by measurement, there is some noise associated with the measured result. Therefore, the smoothing is applied after step 3 on the raw data using the Savitzky–Golay algorithm. After the smoothing, all the raw data undergoes a down-scaling (standardization) procedure to ensure the processed data follows a Gaussian distribution, where the mean value μ is equal to 0 and the standard deviation σ is equal to 1. This step is helpful to avoid the saturation problem in the ANN training stage. The equations of downscaling and upscaling are presented in (1).

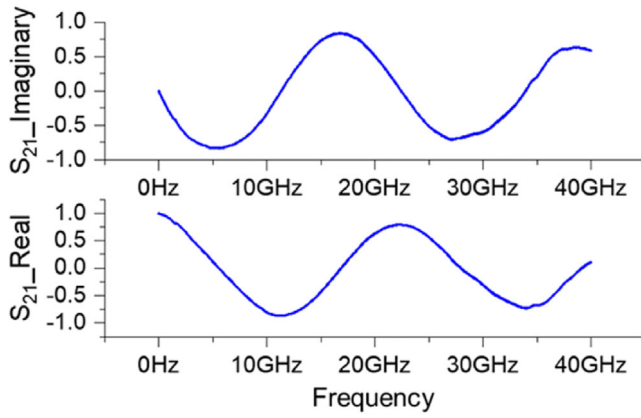


FIGURE 7 Same S_{21} result as in Figure 6 but plotted as real and imaginary

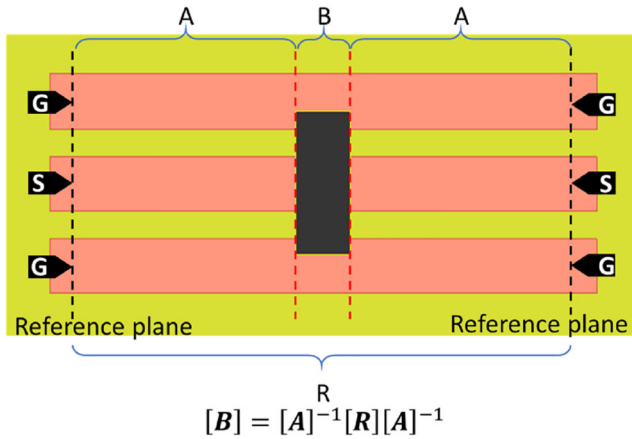


FIGURE 8 2D illustration of the VO_2 switch measurement and de-embedding equation

$$\begin{cases} x_{i_stan} = \frac{x_i - \text{mean}\{\mathbf{x}\}}{\text{sta_dev}\{\mathbf{x}\}}, & \text{standardization} \\ x_i = x_{i_stan} \times \text{sta_dev}\{\mathbf{x}\} + \text{mean}\{\mathbf{x}\}, & \text{destandardization} \end{cases} \quad (1)$$

The last step of preprocessing is data shuffling. Shuffled data helps achieve a more generalized ANN model and also speeds up the convergence. In this work, the training dataset and the test dataset are randomly sampled, meaning that there is no pattern of these datasets.

After finishing all the above-mentioned preprocessing steps, the data is ready to be used for training the ANN model. It is noted that not all the data are used for the training purpose. A small portion of the data should be left for the validation purpose. This study uses 70% of the data for training (training dataset), and the remaining 30% of the data is used to evaluate the trained model (testing dataset).

4 | HYPERPARAMETER OPTIMIZATION

The accuracy of the ANN-based RF model is largely affected by the hyperparameters, including the number of layers and neurons, the activation and loss functions, the optimizer, and the epochs. Most of these hyperparameters have a broad range of selections, which results in a significant number of hyperparameter combinations, and, thus, it is extremely computationally expensive and time-consuming to perform all the combinations. Therefore, a practicable design of experiments (DoE) should be followed to reduce the number of combinations as explained below.

4.1 | Number of layers

Typically, only one layer is sufficient for linear regression problems. But for non-linear regression problems, such as this VO_2 switch modeling work, one layer is insufficient to describe the non-linear relationship between the inputs and outputs. Therefore, more than one layer is needed in this study. The high degree of complexity of ANN should be used for high non-linear input and output. Either the number of layers or the number of neurons can control the complexity of an ANN. Typically, the shallow ANN with two or three hidden layers is widely used for modeling studies, whereas the deep neural networks (with more than three hidden layers) are only used for other applications, such as speech recognition and computer vision. Therefore, two hidden layers will be used in this model and the model's complexity is solely controlled by the number of neurons.

4.2 | Number of neurons

Typically, the number of neurons is larger than the number of input variables. There are five input variables for the VO_2 RF switch modeling, so the number of neurons should be greater than five. In this study, eight different values of neuron numbers are assessed, which are 5, 10, 15, 20, 25, 30, 35, and 40.

4.3 | Activation function (AF)

The AF is the transfer function used in every neuron. Selecting the AF appropriately is crucial because it not only affects the nature of the ANN but it also significantly affects the training speed. For example, linear AF

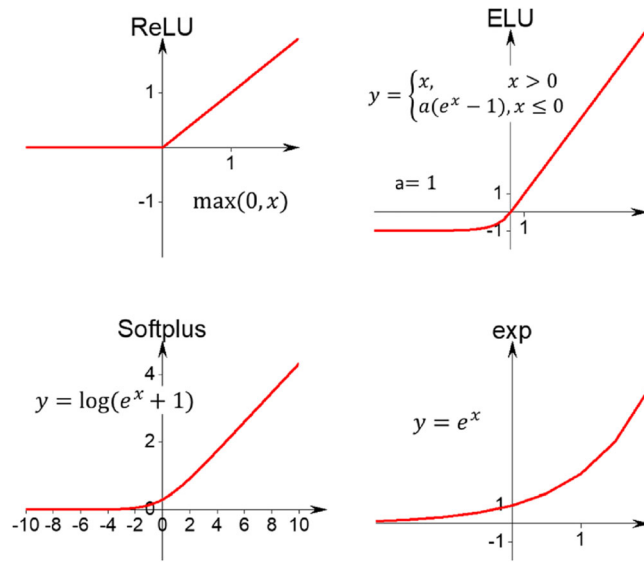


FIGURE 9 The plots and equations of four selected AFs

is only useful for linear problems, and it is incompetent for nonlinear problems. Some of the AFs are more suitable for classification problems, such as sigmoid and tanh, while other AFs are best used for regression problems, such as elu and relu. However, most ANN-based RF modeling studies did not select the AF properly because most of them use sigmoid as the AF in their work. Since most RF modeling works are typical regression problems including this VO₂ RF switch, the regression problem-oriented AFs should be used. In this study, four different AFs are evaluated and the results are compared, which are elu, relu, softplus, and exp. The plots and the equations for these four AFs are provided in Figure 9.

4.4 | Loss function

The loss functions define the method for evaluating the neural network. Typical percentage-based loss functions are the mean absolute percentage error (MAPE) and mean squared logarithmic error (MSLE). However, both MAPE and MSLE are not suitable for our VO₂ switch modeling because the percentage error is not of interest. Instead, the absolute difference between the predicted S-parameter and the actual S-parameter is more important and represents the actual model's accuracy. Therefore, two commonly used loss functions left for selection are the mean absolute error (MAE) and mean squared error (MSE).³⁰ The expression for both loss functions are given in (2)–(3), where y_i is the true value and y_i^p is the predicted value.

$$\text{MSE} = \frac{\sum_{i=1}^n (y_i - y_i^p)^2}{n} \quad (2)$$

$$\text{MAE} = \frac{\sum_{i=1}^n |y_i - y_i^p|}{n} \quad (3)$$

It is noted that MAE is more suitable for noisy data and MSE is more suitable for noise-free data. Since all the necessary steps in the data pre-processing stage are performed, it is believed that the processed data are free from noise and all the outlier data points are valid and equally important as other data points. Therefore, MSE should be used as the loss function for our VO₂ switch modeling work.

4.5 | Optimizer

The optimizer is the training algorithm of the ANN model. Different optimizers have different methods to update the ANN model, that is, the weights and biases. There is a wide range of useable optimizers. However, some second-order methods and their variations suffer from the slow computation because they need to compute the second derivative in every iteration. Therefore, they are not widely used in any practical applications. In this work, four optimizers (i.e., adam, adadelta, adagrad, and rmsprop) are selected for evaluation.

4.6 | Epoch

All the training data will be fed into the ANN model multiple times during the training stage. Every iteration is referred to as one epoch. Typically, more epochs lead to a more accurate model. However, too many epochs can result in the loss of generality of the ANN model; in other words, it becomes very accurate for the training data but become worst for the testing data. In this study, 200 epochs are conducted to ensure enough training iterations during the training stage.

4.7 | Batch size

The batch size defines the number of data points to be fed in every update. The batch size can be set to be equal to the training datapoint, which means that only one parameter (weights and biases) update per epoch. However, it results in a very slow training speed. Typically, a batch size of 50 to 10 000 is used, depending on the

application. A large batch size slows down the update but it is more stable, and a small batch size can speed up the update but it is more likely to become unstable. In this work, five different batch sizes are tested (i.e., 50 010, 5001, 500, 50). To conclude, there are a total of 512 combinations of parameters, which are summarized in Table 2. The aim is to find the optimum value for each hyperparameter by comparing the error values for each combination. In every epoch for a single combination, a total of 10 output error results are generated, which are listed in Table 3.

Note that there are 10 outputs for every combination during the training process, which are summarized in Table 3. The first four results are the errors for training data. The fifth result (marked as “Loss”) is the summed value of these four error results, which is used to monitor overall loss trend. These error results reflect the model's accuracy only for the training data. To assess the model's generalization, we must evaluate it according to the new data that is not “seen” by the model. Therefore, the error results for the testing data are more representative to reflect the accuracy of the model. Therefore, the four output error values (val_Sxx_Re_loss, val_Sxx_Im_loss, val_Sxy_Re_loss, val_Sxy_Im_loss), together with the summed value (val_Loss) shown in Table 3 are used for the comparison. It is known that every combination contains 200 epochs, and every epoch generates 10 output results. It is, therefore, very time consuming to conduct all 512 combinations on a standard computer and compare all the data together.

The more viable solution is to conduct the experiment by only varying a few hyperparameters while adjusting the values for other hyperparameters. The best hyperparameter value is then identified by comparing the error results, and it will be used in the next comparison phase to identify an optimum value for another hyperparameter. Eventually, the optimum values for all the hyperparameters will be determined. However, the order of hyperparameter optimization is very important. In this work, there are five hyperparameters needed to be optimized, which are the AF, optimizer, number of neurons, epochs, and batch size. The selection of AF has a significant impact on the efficiency of the optimizer and, thus,

TABLE 2 Summary of the hyperparameter combinations

Hyperparameters	Values
Number of neurons	5, 10, 15, 20, 25, 30, 35, 40
Batch size	50 010, 5001, 500, 50
AF	elu, relu, softplus, exponential
Optimizer	Adam, adadelta, adagrad, rmsprop
Total number of combinations: 512	

TABLE 3 Summary of output error results

Name of output	Comments
Sxx_Re_loss	Loss values of Sxx real (training data)
Sxx_Im_loss	Loss values of Sxx imaginary (training data)
Sxy_Re_loss	Loss values of Sxy real (training data)
Sxy_Im_loss	Loss values of Sxy imaginary (training data)
Loss	Summed loss (training data)
val_Sxx_Re_loss	Loss values of Sxx real (testing data)
val_Sxx_Im_loss	Loss values of Sxx imaginary (testing data)
val_Sxy_Re_loss	Loss values of Sxy real (testing data)
val_Sxy_Im_loss	Loss values of Sxy imaginary (testing data)
val_Loss	Summed loss (testing data)

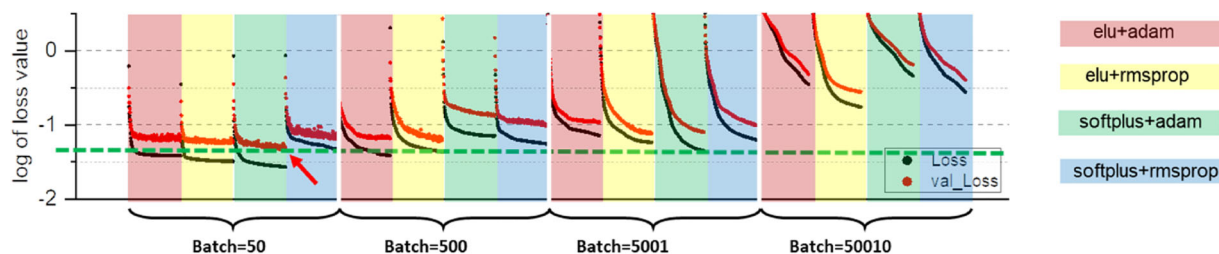
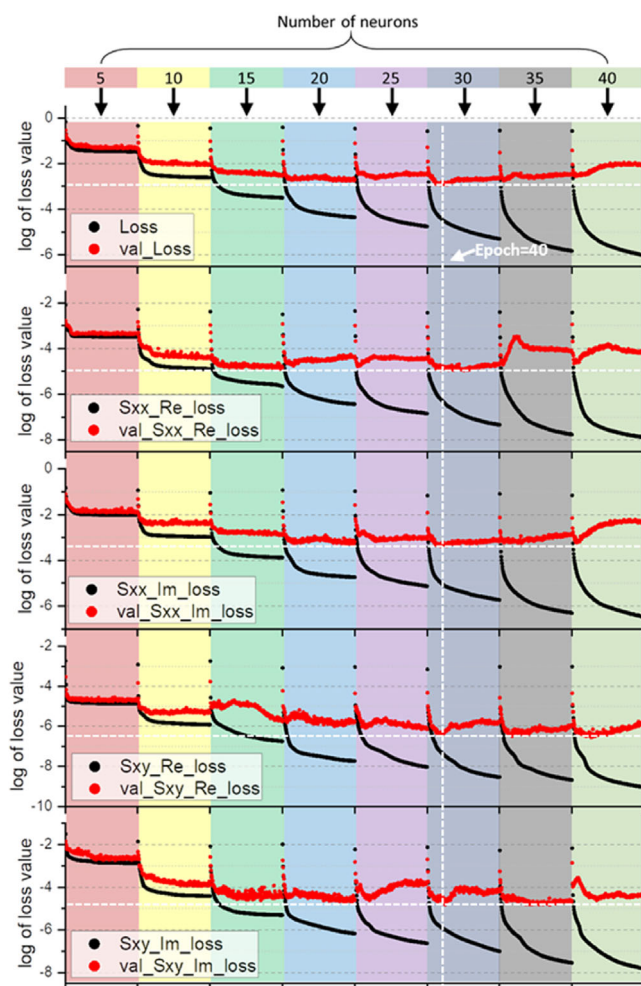
the training efficiency and model accuracy, and other hyperparameters do not affect the AF and optimizer. Therefore, the AF and optimizer should be optimized before other hyperparameters. Once the AF and optimizer are determined, the optimization of batch size should be followed as relates more to AF and optimizer than the other hyperparameters (i.e., number of layers and epoch). The third optimization step is for the number of neurons as it determines the model's complexity. Lastly, the number of epochs are to be optimized.

Since the value assignment of other hyperparameters does not largely affect the determination of the best selection of AF and optimizer, the batch size of 50 and neuron number of 5 are used in the first optimization stage. According to Table 2, there are four optimizers and four AFs, resulting a total of 16 sets of error results, where the 6 best sets of the result are tabulated and shown in Table 4. This table demonstrates that the best combination (softplus + adam) gives the lowest error value of $10^{-1.31}$ (or 0.049), where the worst combination (relu + rmsprop) in Table 4 gives the highest error value ($10^{-0.63}$, or 0.234). However, the other three combinations (elu + adam, elu + rmsprop, elu + softplus) also produce similar levels of error results. Therefore, for reference purposes, all these four combinations are used in the next comparison.

As demonstrated in Table 2, four different batch size values are evaluated following the AF and optimizer optimization step. Figure 10 shows the total error result for the training dataset and testing dataset for these four combinations. Four blocks are identified on the x-axis that correspond to four different batch sizes, where each block contains four segments that represent the four color-coded AF-optimizer combinations. The y-axis

TABLE 4 Comparison between different AFs and optimizers

Log value (linear value)	adam			rmsprop		
	elu	relu	softplus	elu	relu	softplus
val_Loss	-1.18 (0.066)	-0.81 (0.155)	-1.31 (0.049)	-1.22 (0.06)	-0.63 (0.234)	-1.13 (0.074)
val_Sxx_Re_loss	-3.11	-2.59	-3.46	-3.02	-2.61	-3.47
val_Sxx_Im_loss	-1.75	-1.37	-1.82	-1.76	-1.21	-1.62
val_Sxy_Re_loss	-4.34	-4.12	-4.46	-4.77	-3.13	-4.31
val_Sxy_Im_loss	-2.59	-2.39	-2.74	-2.74	-2.14	-2.54

**FIGURE 10** Comparison between different batch sizes for four different AF-optimizer combinations**FIGURE 11** Comparison result between different neuron numbers

represents the loss values of the model. The log scale has been used on the y-axis because the error value becomes very small during the training process, and it becomes indistinguishable and incomparable if they are plotted in a linear scale. The black curves indicate the loss changes for the training dataset, and the red curves indicate the loss changes for the testing dataset. Therefore, the red curve is more indicative of the accuracy of the model. Figure 10 shows that the error value decreases by decreasing the batch size, and it is gradually saturated when the batch size equals 50. If the batch is equal to 50, the AF-optimizer combination of softplus+adam gives the lowest error value among all other combinations, which is also highlighted with the red arrow. This also confirms the result derived from Table 4 and is further proof that the AF-optimizer hyperparameters should be optimized first. Another observation is that the error for the training dataset (black curves) is always higher than that for the testing dataset (red curves). This is also expected and commonly seen in other ANN modeling works.

The third optimization is for the number of neurons. According to Table 2, the error values of ANN with eight different neuron numbers are compared and Figure 11 shows the results. There are five sub-plots in the Figure 11, where the first sub-plot is the total error result, and the remaining four sub-plots are the error results of the four individual outputs. The error results are color-coded in eight different segments, which represent eight different numbers of neurons. By observing the red curve that represents the errors for testing dataset, a clear trend

TABLE 5 Optimized hyperparameters value

Parameters/Metrics	Value
Number of layers	2
Number of neurons in each layer	30
Batch size	50
AF	Softplus
optimizer	Adam
Loss function	Mse
epochs	40
Percentage of training data	70%
Percentage of testing data	30%

TABLE 6 Model metrics with optimized hyperparameters

Item	Log of error value	r value
Total	-2.908	N.A.
Sxx_Re	-4.906	0.994
Sxx_Im	-3.348	0.978
Sxy_Re	-6.433	0.998
Sxy_Im	-4.566	0.992

of the total loss reduction can be identified from neuron numbers increasing from 5 to 20. With neuron numbers above 25, the performances start fluctuating and even deteriorating, which can also be observed in the four individual sub-plots. By observing the first sub-plot, the lowest total error value is identified as -3 when the number of neurons is 30. Interestingly, all the lowest values for the four individual results shown in Figure 11 are with the neuron number of 30, which are all marked with the white dashed lines in the figure. It indicates that the optimized neuron number for all four individual outputs is 30, which is just a coincidence. Although the optimum numbers of neurons are confirmed as 30, Figure 11 indicates that the loss value does not decrease monotonically. The loss values reach the minimum after several epochs and then deflect upward. The critical number of epochs is 40 because the value starts to increase after 40 epochs, which indicates the beginning of overfitting. Therefore, the training should stop at 40 epochs. In other words, the optimum number of epochs is 40.

The optimization of the parameters is thus completed. The optimized parameters are summarized in Table 5 and some error results metrics are tabulated in Table 6. In Table 6, the Pearson correlation coefficient r is also calculated as another measure of how well the model

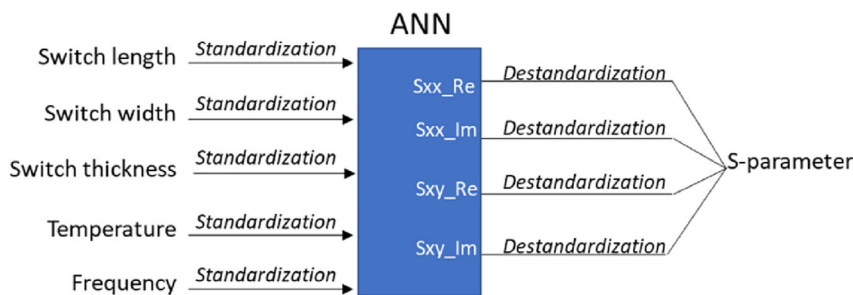


FIGURE 12 The standardization-destandardization diagram

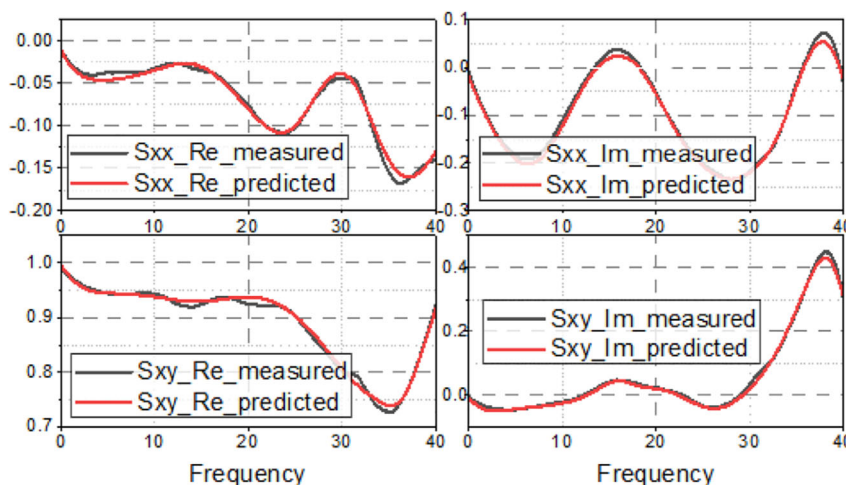


FIGURE 13 The measured and predicted S-parameters with Width = 0.8 mm, Length = 0.1 mm, Thickness = 6 layers, Temperature = 50°C

performs. An r value that is close enough to 1 indicates an accurate model. From Table 6, we know that the summed error value is $10^{-2.908}$ after the optimization. When compared with the error values (worst case $10^{-0.63}$) for un-optimized cases shown in Table 4, it is readily observed that there is a more than 100-fold improvement in the model's accuracy.

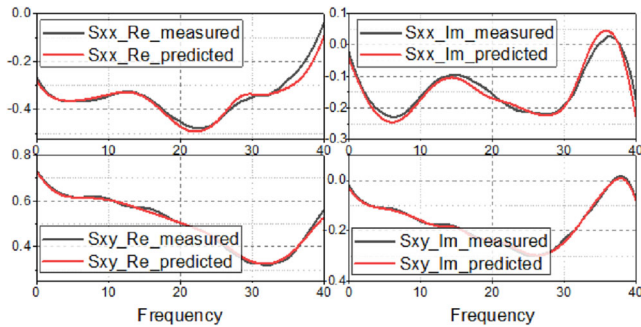


FIGURE 14 The measured and predicted S-parameters with Width = 1 mm, Length = 0.15 mm, Thickness = 6 layers, Temperature = 80°C

5 | DEVELOPED ANN MODEL EVALUATION

In this section, the well-trained ANN model is used to predict some results that will be compared with the measured results. To accurately evaluate the model, all the input combinations used for the training purpose will not be used again for the comparison in this section. When using the ANN model, the input values need to perform a de-standardization using the same scaling factor before feeding into the model. Moreover, the output S-parameter results also need to be de-standardized before use. The diagram is shown in Figure 12.

Three different input combinations have been tested and the comparison results are plotted in Figures 13–15. It is noted that all these plots are in real-imaginary format because the S-parameter outputs of the ANN model are in real-imaginary format. As can be seen from these figures, the predicted results almost overlap the actual measurement results at all frequency points in various VO₂-based RF switch conditions. This is expected because the total error value is only 0.0012 ($10^{-2.908}$), as listed in Table 6. While there are some discrepancies at

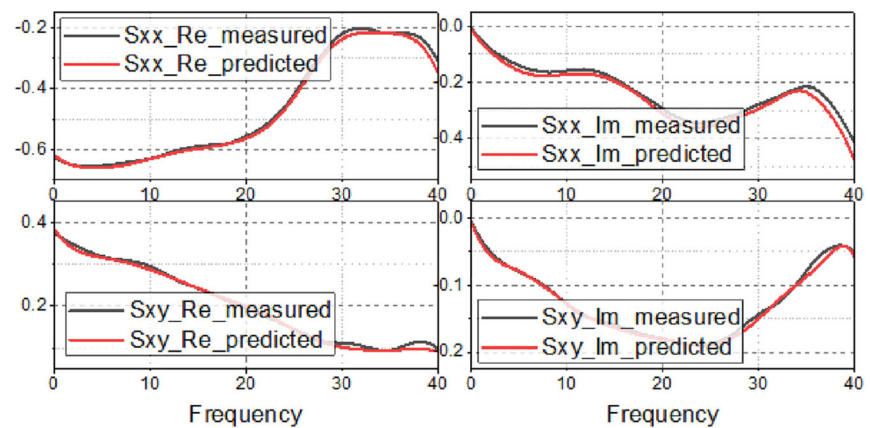


FIGURE 15 The measured and predicted S-parameters with Width = 0.8 mm, Length = 0.1 mm, Thickness = 12 layers, Temperature = 90°C

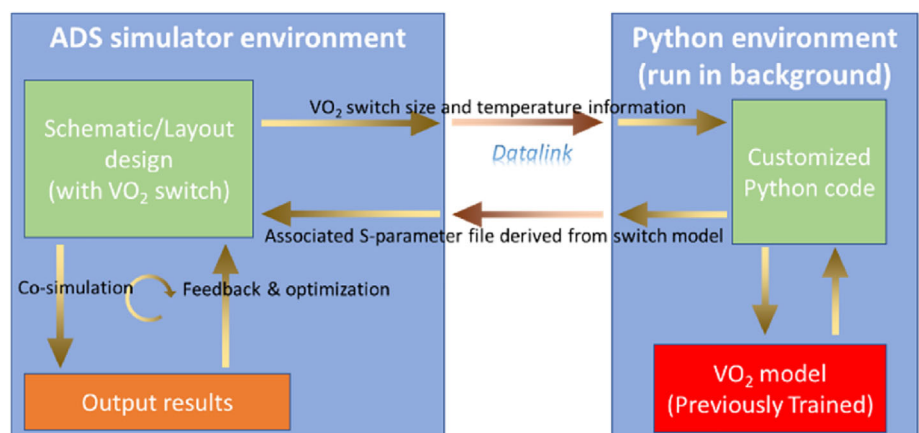


FIGURE 16 The co-simulation design flow

some frequency points, the difference are very small and, thus, negligible.

6 | CO-SIMULATION EXAMPLE

The ANN-based RF model has proved accurate in the previous section. However, to make it more usable for the real RF applications, the model must be compatible with simulators. To the authors' best knowledge, the integration and co-simulation of the ANN-based RF model into a simulator has never been demonstrated. This is mainly because the ANN-based models are inherently incompatible with most simulators. In this study, we have demonstrated the co-simulation of our ANN model in a popular simulator Advanced Design System (ADS) with the aid of Datalink (ADS programming interface protocol). It is noted that our ANN-based RF model is built and executed only under Python programming

language, which is different from the ADS simulator environment. The Datalink connects the ADS environment and the Python environment. The detailed design flow of the working principle is shown in Figure 16. As the first step, the ANN-based model is represented as a block in the ADS, which is also a part of the system. The variable values, such as length and width, which are associated with the ANN model are then sent to the Python environment through the Datalink protocol. The Python environment will plug the variable values into the previously trained ANN model and then generate the corresponding S-parameters. The ADS environment then receives the S-parameter to complete the co-simulation. Based on the quality of the simulation, new variable values may be assigned to the ANN model and, thus, a new cycle of co-simulation may continue.

A design example is also provided to demonstrate the co-simulation. A T-resonator-based bandstop filter model is simulated together with the ANN-based VO₂ RF switch model in the ADS environment. The schematic view of the bandstop filter together with the ANN-based RF switch model is shown in Figure 17. By turning the switch on or off, the operating frequency should change accordingly. We have attempted to assign two different sets of input variables into the ANN-based model, and the co-simulation results are plotted in Figure 18. In this particular example, the width, length, and thickness are identical for these two sets of input variables, which are 1.2, 0.1 mm, and 14 layers, respectively. The only difference is the temperature: 20°C in the one set and 80°C in the other set. These two sets of input variables represent two switch conditions (i.e., ON condition and OFF condition), which are plotted as the solid black curve and solid red curve, respectively.

Apparently, the stopband frequencies are different for these two sets of input variables. While some difference is noted between the co-simulated result and the result of ideal cases, it is expected because the ANN switch model is not a model for an ideal RF switch. Nevertheless, the results shown in Figure 18 proves the success of the co-simulation of the ANN-based model in a commercial simulator.

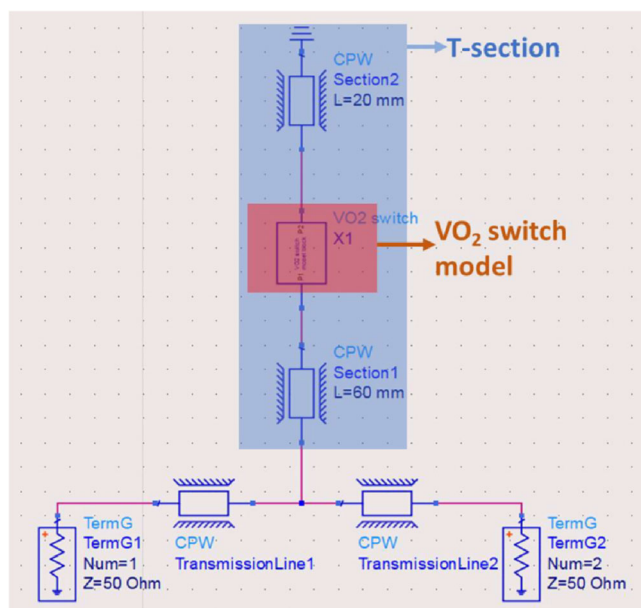


FIGURE 17 Schematic view of filter employing ANN-based switch model

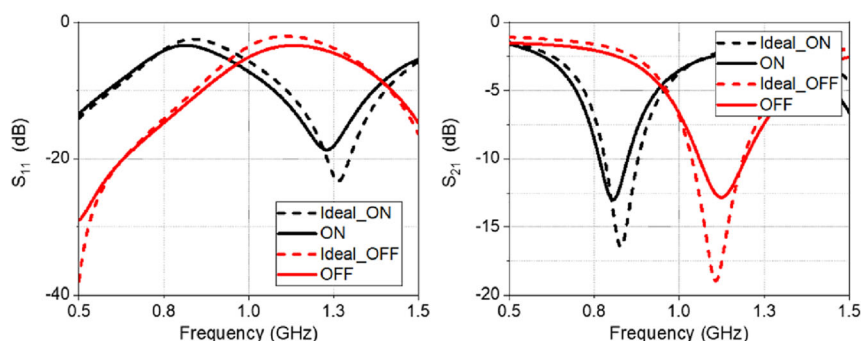


FIGURE 18 The co-simulation results for two sets of input variables. 1). width = 1.2 mm, length = 0.1 mm, thickness = 14 layers, temperature = 20°C. Representing “OFF”. 2). width = 1.2 mm, length = 0.1 mm, thickness = 14 layers, temperature = 80°C. Representing “ON”

7 | CONCLUSION

In this article, a more versatile ANN structure has been introduced for the development of RF device modeling. With the help of a fully printed VO₂ based RF switch as an example, a series of systematic hyperparameter optimization processes have been explored, which has never been presented in other literatures. It has been shown that the use of the optimum hyperparameter on ANN-based model can improve the model's accuracy by more than two orders. Furthermore, for the first time, this work also demonstrated the co-simulation between the ANN model and the commercial circuit and EM simulator. The practical utilization of this ANN-based model in ADS has been verified through a band stop filter design example. In general, the hyperparameter optimization technique together with the co-simulation platform has the potential to bring a paradigm shift in the modeling of high-frequency devices and hence in the design and development of RF circuits, components and systems. In addition, although the printed VO₂ switches have been used as an example, other types of printed RF devices, such as printed diodes, printed antennas, and printed transistors, could be modeled using the same optimization procedure. Furthermore, the co-simulation between the commercial simulator and a number of optimized ANN models representing various RF devices could be very interesting to develop, which is left for future works.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

ORCID

Shuai Yang  <https://orcid.org/0000-0002-3577-5565>

Ahmad Khusro  <https://orcid.org/0000-0003-2025-3040>

REFERENCES

- Zhu C, Li K, Chen Y, et al. Equivalent circuit models for the analysis of electric response and magnetic response of compact triangular electromagnetic resonator. *J Appl Phys*. 2013;113(4):044905.
- Elwi TA, Jassim DA, Mohammed HH. Novel miniaturized folded UWB microstrip antenna-based metamaterial for RF energy harvesting. *Int J Commun Syst*. 2020;33(6):e4305.
- Elwi TA. Printed microwave metamaterial-antenna circuitries on nickel oxide polymerized palm fiber substrates. *Sci Rep*. 2019;9(1):2174.
- Elwi TA, Abdul Hassain ZA, Tawfeeq OA. Hilbert metamaterial printed antenna based on organic substrates for energy harvesting. *IET Microwaves Antennas Propag*. 2019;13(12):2185-2192.
- Kevenaar TA, Ter Maten E, Janssen H, Onneweer S. Methods and approaches for RF circuit simulation and electromagnetic modelling. *Scientific Computing in Electrical Engineering*. Springer; 2004:29-45.
- Nalli A, Raffo A, Crupi G, et al. GaN HEMT noise model based on electromagnetic simulations. *IEEE Trans Microwave Theory Tech*. 2015;63(8):2498-2508.
- Sischka F. Device modeling and measurement for RF systems. *VLSI: Systems on a Chip*. Springer; 2000:569-582.
- Khusro A, Hashmi MS, Ansari AQ, Mishra A, Tarique M. An accurate and simplified small signal parameter extraction method for GaN HEMT. *Int J Circuit Theory Appl*. 2019;47(6):941-953.
- Lee Y, Filipovic d S. ANN basegn of RF MEMS switches. *IEEE Microwave Wireless Compon Lett*. 2005;15(11):823-825.
- Khusro A, Hashmi MS, Ansari AQ. Enabling the development of accurate intrinsic parameter extraction model for GaN HEMT using support vector regression (SVR). *IET Microwaves Antennas Propag*. 2019;13(9):1457-1466.
- Marinković Z, Crupi G, Caddemi A, Marković V, Schreurs DMP. A review on the artificial neural network applications for small-signal modeling of microwave FETs. *Int J Numer Modell Electron Networks Devices Fields*. 2020;33(3):e2668.
- Leszczynska N, Couckuyt I, Dhaene T, Mrozowski M. Low-cost surrogate models for microwave filters. *IEEE Microwave Wireless Compon Lett*. 2016;26(12):969-971.
- Feng F, Zhang C, Ma J, Zhang Q-J. Parametric modeling of EM behavior of microwave components using combined neural networks and pole-residue-based transfer functions. *IEEE Trans Microwave Theory Tech*. 2015;64(1):60-77.
- Koziel S, Mahouti P, Calik N, Belen MA, Szczepanski S. Improved modeling of microwave structures using performance-driven fully-connected regression surrogate. *IEEE Access*. 2021;9:71470-71481.
- Calik N, Belen MA, Mahouti P, Koziel S. Accurate modeling of frequency selective surfaces using fully-connected regression model with automated architecture determination and parameter selection based on Bayesian optimization. *IEEE Access*. 2021;9:38396-38410.
- Liu B, Aliakbarian H, Ma Z, Vandenbosch GA, Gielen G, Excell P. An efficient method for antenna design optimization based on evolutionary computation and machine learning techniques. *IEEE Trans Antennas Propag*. 2013;62(1):7-18.
- Natarajan SK, Caro MA. Particle swarm based hyper-parameter optimization for machine learned interatomic potentials, *arXiv Preprint arXiv:2101.00049*, 2020.
- Yu T, Zhu H. Hyper-parameter optimization: a review of algorithms and applications, *arXiv Preprint arXiv:2003.05689*, 2020.
- Kouhalvandi L, Ceylan O, Ozoguz S. Automated deep neural learning-based optimization for high performance high power amplifier designs. *IEEE trans Circuits Syst I Regul Pap*. 2020;67(12):4420-4433.
- Jin J, Feng F, Na W, et al. Recent advances in neural network-based inverse modeling techniques for microwave applications. *Int J Numer Modell Electron Networks Devices Fields*. 2020;33(6):e2732.
- Watson PM, Gupta KC. Design and optimization of CPW circuits using EM-ANN models for CPW components. *IEEE Trans Microwave Theory Tech*. 1997;45(12):2515-2523.

22. Creech GL, Paul BJ, Lesniak CD, Jenkins TJ, Calcaterra MC. Artificial neural networks for fast and accurate EM-CAD of microwave circuits. *IEEE Trans Microwave Theory Tech.* 1997; 45(5):794-802.
23. Long Y, Zhong Z, Guo Y-X. A novel 4-D artificial-neural-network-based hybrid large-signal model of GaAs pHEMTs. *IEEE Trans Microwave Theory Tech.* 2016;64(6):1752-1762.
24. Du X, Helaoui M, Jarndal A, Liu T, Hu B, Hu X, Ghannouchi FM. ANN-Based Large-Signal Model of AlGaIn/GaN HEMTs With Accurate Buffer-Related Trapping Effects Characterization. *IEEE Trans Microwave Theory Tech.* 2020;68(7):3090-3099.
25. Huang A-D, Zhong Z, Wu W, Guo Y-X. An artificial neural network-based electrothermal model for GaN HEMTs with dynamic trapping effects consideration. *IEEE Trans Microwave Theory Tech.* 2016;64(8):2519-2528.
26. Khusro A, Husain S, Hashmi MS, Ansari AQ. Small signal behavioral modeling technique of GaN high electron mobility transistor using artificial neural network: an accurate, fast, and reliable approach. *Int J RF Microwave Comput-Aided Eng.* 2020; 30(4):e22112.
27. Jarndal A, Husain S, Hashmi M. *On Temperature-Dependent Small-Signal Modelling of GaN HEMTs Using Artificial Neural Networks and Support Vector Regression.* IET Microwaves, Antennas & Propagation; 2021 On temperature-dependent small-signal modelling of GaN HEMTs using artificial neural networks and support vector regression.
28. Jarndal A, Husain S, Hashmi M, Ghannouchi FM. Large-signal modeling of GaN HEMTs using hybrid GA-ANN, PSO-SVR, and GPR-based approaches. *IEEE J Electron Devices Soc.* 2020; 9:195-208.
29. Mkadem F, Boumaiza S. Physically inspired neural network model for RF power amplifier behavioral modeling and digital predistortion. *IEEE Trans Microwave Theory Tech.* 2011;59(4): 913-923.
30. Li M, Liu J, Jiang Y, Feng W. Complex-Chebyshev functional link neural network behavioral model for broadband wireless power amplifiers. *IEEE Trans Microwave Theory Tech.* 2012; 60(6):1979-1989.
31. Hongyo R, Egashira Y, Hone TM, Yamaguchi K. Deep neural network-based digital predistorter for Doherty power amplifiers. *IEEE Microwave Wireless Compon Lett.* 2019;29(2):146-148.
32. Xiao L-Y, Shao W, Jin F-L, Wang B-Z. Multiparameter modeling with ANN for antenna design. *IEEE Trans Antennas Propag.* 2018;66(7):3718-3723.
33. Zhao P, Wu K. Homotopy optimization of microwave and millimeter-wave filters based on neural network model. *IEEE Trans Microwave Theory Tech.* 2020;68(4):1390-1400.
34. Zhang C, Jin J, Na W, Zhang Q-J, Yu M. Multivalued neural network inverse modeling and applications to microwave filters. *IEEE Trans Microwave Theory Tech.* 2018;66(8): 3781-3797.
35. Li S, Wang Y, Yu M, Panariello A. Efficient modeling of Ku-band high power dielectric resonator filter with applications of neural networks. *IEEE Trans Microwave Theory Tech.* 2019; 67(8):3427-3435.
36. Yang S, Khusro A, Li W, Vaseem M, Hashmi M, Shamim A. A machine learning-based microwave device model for fully printed VO₂ RF switches. *2020 50th Eur Microwave Conf (EuMC).* 2021: IEEE;662-665.
37. Yang S, Vaseem M, Shamim A. Fully inkjet-printed VO₂-based radio-frequency switches for flexible reconfigurable components. *Adv Mater Technol.* 2019;4(1):1800276.
38. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Louppe G, Prettenhofer P, Weiss R, Weiss RJ, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E. Scikit-learn: machine learning in python. *J Mach Learn Res.* 2011;12:2825-2830.

How to cite this article: Yang S, Khusro A, Li W, Vaseem M, Hashmi M, Shamim A. Optimization of ANN-based models and its EM co-simulation for printed RF devices. *Int J RF Microw Comput Aided Eng.* 2022;32(3):e23012. doi:10.1002/mmce.23012