IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 25, 2019, accepted May 18, 2019, date of publication May 22, 2019, date of current version June 19, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2918268

Automated Neural-Based Modeling of Microwave Devices Using Parallel Computation and Interpolation Approaches

WEICONG NA[1], WANRONG ZHANG[1], SHUXIA YAN[2], AND GAOHUA LIU[3]

[1]Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China
[2]School of Electronics and Information Engineering, Tianjin Polytechnic University, Tianjin 300387, China
[3]School of Electronics and Information Engineering, Tianjin University, Tianjin 300072, China

Corresponding author: Weicong Na (weicongna@bjut.edu.cn)

This work was supported in part by the Project funded by the China Postdoctoral Science Foundation under Grant 2019M650404, in part by the National Natural Science Foundation of China under Grant 61774012 and Grant 61601323, and in part by the Beijing Municipal Natural Science Foundation under Grant 4192014.

ABSTRACT Automated model generation (AMG) is an automated artificial neural network (ANN) modeling algorithm, which integrates all the subtasks (including adaptive sampling/data generation, model structure adaptation, training, and testing) in neural model development into one unified framework. In existing AMG, most of the time is spent on data sampling and model structure adaptation due to the iterative neural network training and the sequential computation mechanism. In this paper, we propose an advanced AMG algorithm using parallel computation and interpolation approaches to speed up the neural modeling of microwave devices. Efficient interpolation approaches are incorporated to avoid repetitive training of the intermediate neural networks during adaptive sampling process in AMG. Parallel computation formulation based on a multi-processor environment is proposed to further save time during interpolation calculation, data generation, and model structure adaptation process. Examples of automated modeling of two microwave filters are presented to show the advantage of this paper.

INDEX TERMS Design automation, modeling, neural networks, parallel computation, interpolation approaches.

I. INTRODUCTION

Artificial neural networks (ANNs) have been recognized as powerful tools in microwave modeling and design [1]–[3], such as nonlinear microwave device modeling [4]–[6], parametric modeling [7]–[9], electromagnetic (EM) optimization [10], [11], multiphysics modeling [12], and inverse modeling [13]. ANNs are trained to learn EM/physics data which represent the behavior of microwave devices. The trained ANNs can then be used for high-level circuit design to provide fast and accurate solutions to the task they have learned.

Automated model generation (AMG) with training-driven adaptive sampling algorithm was firstly introduced in [14] to automate the neural network model development process. It integrates all the subtasks involved in neural model development like adaptive sampling/data generation, model structure adaptation, training and testing into one unified framework, thus reducing the intensive human effort demanded by conventional manual modeling approaches. In AMG of [14], training data are dynamically generated during ANN training by driving EM/physical/circuit simulators. Neural model structure is automatically adjusted by training ANNs with different numbers of hidden neurons to obtain the most compact model with best accuracy. Therefore, ANN training is needed to be performed both in adaptive sampling process and model structure adaptation process in conventional AMG algorithm. Recently, a modified AMG algorithm with interpolation approaches [15] was presented as an improvement of [14] to avoid repetitively training intermediate neural networks during adaptive sampling process. Different localized interpolation functions are produced as local models in different subregions of the modeling input space to assess the adequacy of training data during adaptive

The associate editor coordinating the review of this manuscript and approving it for publication was Haiquan Zhao.

sampling process in AMG. The computationally intensive and time-consuming ANN training during adaptive sampling process in [14] is replaced by simple and efficient interpolation computation, thus making AMG faster than that in [14]. This modified AMG algorithm [15] has been applied in microwave power amplifier modeling [16] and interconnect reliability analysis [17]. However, the existing AMG with interpolation approaches in [15] is based on a sequential computation mechanism, involving sequential interpolation calculation for all subregions during adaptive data sampling, sequential data generation by repetitively driving detailed EM/physics/circuit simulators, and batch ANN model training through iterative training stages.

With increased complexity in microwave modeling problems, the efficiency of existing AMG becomes important. Since the detailed EM/physics/circuit simulations are usually CPU expensive and the ANN training is an iterative process, they are the most computationally intensive and time-consuming processes in existing sequential AMG. When the complexity of microwave modeling problem increases, the efficiency of existing AMG decreases. Motivated by this, the purpose of this paper is to incorporate parallel computation mechanism into the AMG with interpolation approaches. Parallel computation is a type of computation in which many calculations or the execution of processes are carried out simultaneously [18]. In the field of neural network, development of parallel processing methods for ANN has become an active research topic. Several techniques have been reported for the parallel architecture study [19], [20] and parallel backpropagation training of ANN [21]. Recently, parallel computation has also been studied and utilized in several areas of microwave applications. A parallel space mapping approach to EM optimization is presented in [22], using parallel coarse/fine model evaluations to reduce the number of space mapping iterations and speed up the optimization process. In [23], distributed parallel computing technique is incorporated to EM data generation process in EM modeling. This technique runs multiple EM simulations in parallel to speed up the computationally expensive EM data generation process. A parallel ANN training technique based on parallel matrix formulation [24] is proposed for dynamic nonlinear transistor modeling with large datasets. The ANN feedforward and derivatives used in ANN training are calculated in parallel to reduce total time for training a neural network model.

In this paper, we propose an advanced AMG algorithm using parallel computation and interpolation approaches. A new parallel mechanism of interpolation calculation is proposed to generate multiple local models in parallel during adaptive sampling. These local models are utilized to determine the amount of training data and their distribution in the entire model input space. Parallel data generation and parallel neural network training techniques are incorporated in the proposed AMG algorithm to further improve the neural modeling efficiency. Parallel data generation is achieved by driving multiple EM/physical/circuit simulators in multiple processors simultaneously. Parallel neural network training during model structure adaptation is achieved by distributing the workload of error feedforward and derivatives calculation to multiple processors. The proposed AMG algorithm takes advantage of both parallel computation and interpolation approaches, further improving the efficiency of neural-based AMG process.

II. PROPOSED AMG ALGORITHM USING PARALLEL COMPUTATION AND INTERPOLATION APPROACHES

The proposed AMG algorithm proceeds in a stage-wise manner. In each stage, the proposed AMG algorithm performs either parallel interpolation for adaptive sampling, or parallel data generation, or parallel neural network training. The workload in each stage is distributed into multiple processors for parallel processing, thus speeding up the AMG process. Finally, a compact neural network model with good accuracy is obtained in a shorter time than existing AMG methods.

A. NOTATION

Let $\boldsymbol{x}$ represent a vector containing n physical parameters of a microwave component, and d ($\boldsymbol{x}$) represent the response of the component under consideration. Therefore, the training data can be denoted by input-output sample pair ($\boldsymbol{x}$, $d(\boldsymbol{x})$). We define the input-output relationship of the ANN model as

$$y = y(\boldsymbol{x}, \boldsymbol{w}) \tag{1}$$

where y is the ANN output and $\boldsymbol{w}$ is a vector containing the weights of the ANN model. Let k represent the number of stages during AMG process. We define E^k_{train} and E^k_{test} to represent the training error and testing error of the ANN model in the kth stage of AMG, respectively. We also define E_d to represent the user-desired neural model error. In our proposed AMG algorithm using parallel computation and interpolation approaches, N_p is denoted as the number of parallel processors.

B. PARALLEL INTERPOLATION APPROACH

In adaptive sampling process, the original model input space is regarded as one region at first. If the testing error of this region is large, the region would be divided into 2^n subregions and new data are generated in this region. In every stage, the algorithm compares the testing error of every subregion and chooses the subregion with largest testing error as the worst region. The worst region would be divided into 2^n new subregions again in the next stage by generating new data in this region. In this way, the algorithm can automatically distinguish nonlinear and smooth regions of the model and generate training data accordingly. More data are generated in the nonlinear region while fewer data are generated in the linear region. Suppose the original model input space is a two-dimensional (2-D) space. Fig. 1 shows an example of the distribution pattern of training data in the 2-D input space during the first 4 stages of adaptive sampling process.

We propose parallel interpolation approaches to obtain the testing error of every subregion during every adaptive

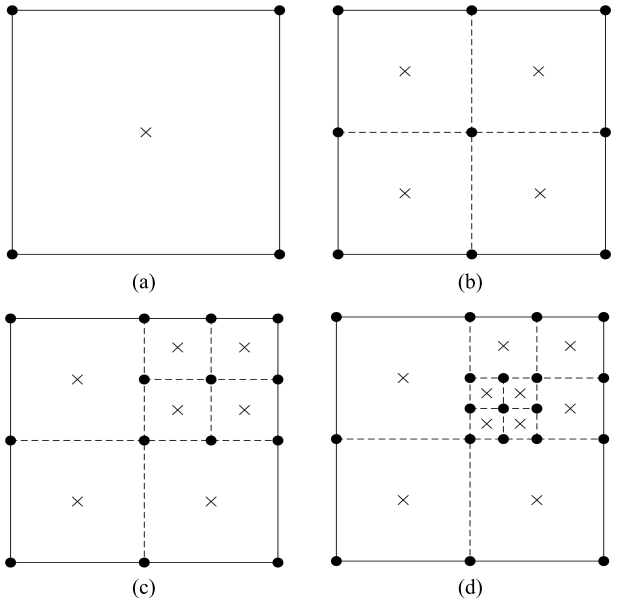


FIGURE 1. The distribution patterns of the training data in the 2-D input space during the first 4 stages of adaptive sampling. "•" represents the locations of training data and "×" represents the locations where to measure the testing error of each subregion. (a) 1st stage. (b) 2nd stage. (c) 3rd stage. (d) 4th stage.

sampling stage. Different interpolation functions are created as local models of different subregions to calculate the testing error and to determine where to add new training data in the next stage. While the neural network model represents the nonlinear input and output relationship in the entire input space, simpler interpolation functions are sufficient to represent the model in local subregions. Therefore, these interpolation functions can replace the intermediate neural network models in [14] as the local models of subregions, avoiding the repetitive and time-consuming neural network training during the first few stages of adaptive sampling in [14].

For each subregion, we formulate the interpolation function $f(\boldsymbol{x})$ as

$$\begin{aligned} f(\boldsymbol{x}) &= \alpha_0 + \sum_{j=1}^{n} \alpha_j x_j + \alpha_{n+1} x_1 x_2 + \alpha_{n+2} x_1 x_3 + \cdots \\ &\quad + \alpha_K x_1 x_2 \cdots x_n + \sum_{j=1}^{n} \alpha_{K+j} x_j^2 \\ &= \boldsymbol{\alpha}^T \cdot \boldsymbol{\phi} \end{aligned} \tag{2}$$

where $\boldsymbol{\alpha}$ is a vector containing the coefficients needed to be determined of the interpolation function, and $\boldsymbol{\phi}$ is a vector containing the products of various combinations of $x_1, x_2, \cdots$, and x_n.

To compute the values of coefficients $\boldsymbol{\alpha}$, the training data in and around the interpolation region is selected to form a matrix $\boldsymbol{X}$ which contains the location information of these training data in the input space, and a vector $\boldsymbol{d}$ which contains the output values of these training data. The details of matrix $\boldsymbol{X}$ and vector $\boldsymbol{d}$ are provided in Appendix. Then we obtain

$$\boldsymbol{X}\boldsymbol{\alpha} = \boldsymbol{d} \tag{3}$$

The least square solution of (3) is

$$\boldsymbol{\alpha} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{d} \tag{4}$$

In this way, we obtain the interpolation function $f(\boldsymbol{x})$ for each subregion. Since the interpolation functions are completely independent between different subregions, they can be computed concurrently and asynchronously by the N_p parallel processors to increase the interpolation efficiency. We define R^k to represent the number of subregions in the kth stage of adaptive sampling. Since the number of subregions in different stages is different, R^k is a function of stage k, i.e.,

$$R^k = \begin{cases} 1, & k = 1 \\ 2^{n+k-2} - k + 2, & \text{otherwise} \end{cases} \tag{5}$$

The workload of interpolation calculation in these R^k subregions is divided into N_p sections and distributed to N_p processors to implement in parallel. The ith processor ($i = 1, 2, \cdots, N_p$) creates interpolation functions and calculates testing errors for $N_{f,i}^k$ subregions, where

$$N_{f,i}^k = \begin{cases} \left\lfloor \dfrac{R^k}{N_p} \right\rfloor, & \text{if } 1 \leqslant i \leqslant N_p - 1 \\ R^k \bmod N_p, & \text{otherwise} \end{cases} \tag{6}$$

We define f_j and e_j as the interpolation function and the testing error of the jth subregion ($j = 1, 2, \cdots, R^k$) in the kth stage respectively, i.e.,

$$e_j = f_j(\boldsymbol{x}_{mid,j}) \tag{7}$$

where $\boldsymbol{x}_{mid,j}$ represent the location of the middle point of the jth subregion in the input space. Fig. 1 shows an example of the locations where to measure the testing error of each subregion in a 2-D input space. The testing error of the entire input space in the kth stage E_{test}^k is calculated by

$$E_{test}^k = \max_{1 \leqslant j \leqslant R^k} e_j \tag{8}$$

If $E_{test}^k > E_d$ (same as over-learning phenomena in ANN training process), the proposed AMG algorithm divides the worst region by generating new training data in the next stage. Once $E_{test}^k \leqslant E_d$ (same as good-learning phenomena in ANN training process), the adaptive sampling process stops.

The proposed parallel interpolation approach is summarized in the following steps:

Step 1) Initialize the multi-processor environment.

Step 2) In the kth stage of proposed AMG, we first calculate R^k and $N_{f,i}^k$, then divides and distributes the total workload of interpolation calculation to N_p processors.

Step 3) Each processor completes the interpolation tasks assigned to it. The ith processor ($i = 1, 2, \cdots, N_p$) creates interpolation functions for $N_{f,i}^k$ subregions and calculates the testing errors of these subregions using (2) to (7).

Step 4) We collect all the results from all processors to compare the testing errors of all subregions in the

kth stage. The subregion with the worst testing error E^k_{test} is selected as the worst region.

Step 5) If $E^k_{test} > E_d$, the worst region will be divided by generating new training data in this worst region, and proposed AMG algorithm enters into the parallel data generation process; otherwise, the interpolation calculation and data generation process stop.

We use the speedup factor S_f and the parallel efficiency η_f to measure the performance of our parallel interpolation approach. Let S_f be the ratio between the interpolation time on one single processor and that on N_p processors, and let η_f be equal to the speedup factor divided by the number of processors, i.e.,

$$S_f = \frac{T_1 + T_f \cdot \sum_{k=1}^{m} R^k}{T_1 + T_f \cdot \sum_{k=1}^{m} \left(\max_{1 \leqslant i \leqslant N_p} N^k_{f,i} \right)} \tag{9}$$

and

$$\eta_f = \frac{S_f}{N_p} \times 100\% \tag{10}$$

where m is the total number of stages performing interpolation calculation, T_1 represents the overhead time during interpolation, and T_f represents the individual time of each interpolation calculation on a single processor. We suppose T_f on every processor is equal to each other. It is observed that large speedup and high parallel efficiency of interpolation calculation can be achieved if T_1 is smaller than T_f.

C. PARALLEL DATA GENERATION METHOD

In adaptive sampling/data generation process of AMG, multiple new training data are generated in the worst region of the input space in every stage. Since repetitively driving detailed EM/physical/circuit simulators to generate training data is a time-consuming process in conventional AMG algorithm, we propose to use parallel data generation method by driving N_p EM/physical/circuit simulators on N_p parallel processors simultaneously to improve the data generation efficiency. We define D to represent the number of new training data generated in every stage, which only relates to the dimensionality n of the modeling problem, i.e.,

$$D = \sum_{j=1}^{n} \left(\binom{n}{j} \cdot 2^{(n-j)} \right) \tag{11}$$

Let $N_{d,i}$ represent the number of simulations that the ith processor ($i = 1, 2, \cdots, N_p$) is performed in every stage of data generation process, and $N_{d,i}$ is formulated as

$$N_{d,i} = \begin{cases} \left\lfloor \frac{D}{N_p} \right\rfloor, & \text{if } 1 \leqslant i \leqslant N_p - 1 \\ D \bmod N_p, & \text{otherwise} \end{cases} \tag{12}$$

The proposed parallel data generation method is summarized in the following steps:

Step 1) Initialize the multi-processor environment.

Step 2) We first calculate $N_{d,i}$ and divide the D new input samples $\boldsymbol{x}_j, j = 1, 2, \cdots, D$ into N_p subsets. The ith subset ($i = 1, 2, \cdots, N_p$) contains $N_{d,i}$ samples. These subsets are written into N_p files and distributed to N_p processors, respectively.

Step 3) Each processor first reads the corresponding file containing new input samples, then drives EM/physical/circuit simulators to obtain the responses of these samples $d(\boldsymbol{x}_j), j = 1, 2, \cdots, N_{d,i}$. After data generation, these $N_{d,i}$ input samples and their responses is written in a file.

Step 4) All files containing the newly generated training data from all processors are collected. Parallel data generation in the kth stage of AMG stops.

We use the speedup factor S_d and the parallel efficiency η_d to measure the performance of the parallel data generation method. The definitions of S_d and η_d are similar to those of S_f in (9) and η_f in (10), i.e.,

$$S_d = \frac{T_2 + m \cdot D \cdot T_d}{T_2 + m \cdot \left(\max_{1 \leqslant i \leqslant N_p} N_{d,i} \right) \cdot T_d} \tag{13}$$

and

$$\eta_d = \frac{S_d}{N_p} \times 100\% \tag{14}$$

where m is the total number of stages performing data generation (same as the total number of stages performing interpolation calculation), T_2 represents the overhead time during data generation process, and T_d represents the simulation time for each data generation on a single processor. Since T_2 is much smaller than T_d, a large speedup and high parallel efficiency of data generation can be achieved.

D. PARALLEL ANN TRAINING

After data generation, the proposed AMG algorithm proceeds to neural model structure adaptation stages to determine the suitable number of hidden neurons in the ANN model. The initial guess of the number of hidden neurons in the ANN model can be flexible. According to the training and testing errors in every stage, the proposed AMG algorithm automatically detects the ANN learning phenomenon (i.e., under-learning, over-learning, good-learning), and decides whether to add or delete hidden neurons of ANN in the next stage. If the ANN with H^k hidden neurons is under-learning (i.e., $E^k_{train} > E_d$) in the kth stage, an ANN with $H^k + \delta$ hidden neurons is trained by the algorithm in the $(k+1)$th stage; if the ANN with H^k hidden neurons is over-learning (i.e., $E^k_{train} \leqslant E_d$ and $E^k_{test} > E_d$) in the kth stage, an ANN with $H^k - \delta$ hidden neurons is trained by the algorithm in the $(k+1)$th stage; otherwise, the ANN is good-learning (i.e., $E^k_{test} \leqslant E_d$) and the model structure adaptation process is finished.

ANN training is an iterative and a computationally intensive process in AMG, in which the neural network weights

$\boldsymbol{w}$ are iteratively updated based on the training error and its derivatives w.r.t weights to minimize the overall training error E_{train}. Because the feedforward training error calculation and the backpropagation derivative computation are completely independent between different training data, we propose to use parallel ANN training method by performing training error calculation and error backpropagation in N_p processors parallely. We divide the total number of training data into N_p subsets and distribute to N_p processors. Each processor computes the training error and its derivatives using its corresponding training data subsets. Then all the results are added together to obtain the total training error and derivatives. The neural network weights are synchronously updated among all the processors. Let L represent the total number of training data obtained from data generation process, and $N_{l,i}$ represent the number of training data sent to the ith processor ($i = 1, 2, \cdots, N_p$) for training error and derivative calculation. The formulation of $N_{l,i}$ is

$$N_{l,i} = \begin{cases} \left\lfloor \dfrac{L}{N_p} \right\rfloor, & \text{if } 1 \leqslant i \leqslant N_p - 1 \\ L \bmod N_p, & \text{otherwise} \end{cases} \tag{15}$$

The training error on the ith processor $E^k_{train,i}$ and its derivative $\frac{\partial E^k_{train,i}}{\partial \boldsymbol{w}}$ in the kth stage of AMG are defined as

$$E^k_{train,i} = \frac{1}{2} \sum_{j=1}^{N_{l,i}} \left\| y^k(\boldsymbol{x}_j, \boldsymbol{w}) - d(\boldsymbol{x}_j) \right\|^2 \tag{16}$$

and

$$\frac{\partial E^k_{train,i}}{\partial \boldsymbol{w}} = \sum_{j=1}^{N_{l,i}} \left(y^k(\boldsymbol{x}_j, \boldsymbol{w}) - d(\boldsymbol{x}_j) \right) \cdot \frac{\partial y^k(\boldsymbol{x}_j, \boldsymbol{w})}{\partial \boldsymbol{w}} \tag{17}$$

respectively, where $y^k(\boldsymbol{x}_j, \boldsymbol{w})$ represent the response of the ANN model at the input sample $\boldsymbol{x}_j$ in the kth stage of AMG. The total training error E^k_{train} and its derivative $\frac{\partial E^k_{train}}{\partial \boldsymbol{w}}$ for each training iteration are defined as

$$E^k_{train} = \frac{1}{2} \sum_{i=1}^{N_p} \sum_{j=1}^{N_{l,i}} \left\| y^k(\boldsymbol{x}_j, \boldsymbol{w}) - d(\boldsymbol{x}_j) \right\|^2 \tag{18}$$

and

$$\frac{\partial E^k_{train}}{\partial \boldsymbol{w}} = \sum_{i=1}^{N_p} \sum_{j=1}^{N_{l,i}} \left(y^k(\boldsymbol{x}_j, \boldsymbol{w}) - d(\boldsymbol{x}_j) \right) \cdot \frac{\partial y^k(\boldsymbol{x}_j, \boldsymbol{w})}{\partial \boldsymbol{w}} \tag{19}$$

respectively.

One iteration of the proposed parallel ANN training is summarized in the following steps:

Step 1) Initialize the multi-processor environment.

Step 2) We calculate $N_{l,i}$ and divide the L training data into N_p subsets. The ith subset ($i = 1, 2, \cdots, N_p$) contains $N_{l,i}$ samples. These subsets are written into N_p files, then distributed to N_p processors, respectively.

Step 3) Each processor reads the corresponding file, i.e., the ith processor reads the file containing $N_{l,i}$ training data, then computes $E^k_{train,i}$ and $\frac{\partial E^k_{train,i}}{\partial \boldsymbol{w}}$ using (16) and (17).

Step 4) We gather the results of $E^k_{train,i}$ and $\frac{\partial E^k_{train,i}}{\partial \boldsymbol{w}}$ from all processors, then calculate the overall training error E^k_{train} and its derivative $\frac{\partial E^k_{train}}{\partial \boldsymbol{w}}$ using (18) and (19).

Step 5) All processors update the weight vector using quasi-Newton method. One iteration of ANN training is finished.

We use the speedup factor S_l and the parallel efficiency η_l to measure the performance of the parallel ANN training method. The definitions of S_l and η_l are similar to those of S_f in (9) and η_f in (10), i.e.,

$$S_l = \frac{T_3 + q \cdot L \cdot T_l}{T_3 + q \cdot \left(\max\limits_{1 \leqslant i \leqslant N_p} N_{l,i} \right) \cdot T_l} \tag{20}$$

and

$$\eta_l = \frac{S_l}{N_p} \times 100\% \tag{21}$$

where q represents the number of iterations for one ANN training, T_3 represents the overhead time during ANN training, and T_l represents the computation time of training error and its derivative per training data on a single processor. In (20), T_3 and T_l are in the same order of magnitude, and the number of training iterations q is usually more than thousands. Using the proposed parallel training method, a large speedup and high parallel efficiency of data generation can be achieved.

The flowchart of the proposed AMG algorithm with using parallel computation and interpolation approaches is shown in Fig. 2. Using the proposed AMG algorithm, the model development time can be greatly shortened with the increase of the number of parallel processors. In this way, the proposed AMG algorithm is more efficient in automated neural-based model development than existing AMG methods.

III. EXAMPLES

A. AUTOMATED MODEL DEVELOPMENT OF TWO-SECTION LOW-PASS ELLIPTIC MICROSTRIP FILTER

In this example, we develop a parametric model for a two-section low-pass elliptic microstrip filter [9], [22], [25], as shown in Fig. 3 (a). The thickness of the alumina substrate h equals to 0.508 mm and the dielectric constant of the substrate ϵ_r equals to 9.4. The geometrical input parameters of the ANN model shown in Fig. 3 (b) are $\boldsymbol{x} = [L_1\ L_2\ L_{c1}\ L_{c2}\ W_c\ G_c]^T$, chosen based on the sensitivity data of the above low-pass filter. The model output is the magnitude of S_{21}.

We perform the proposed AMG algorithm with parallel computation and interpolation approaches to develop a neural

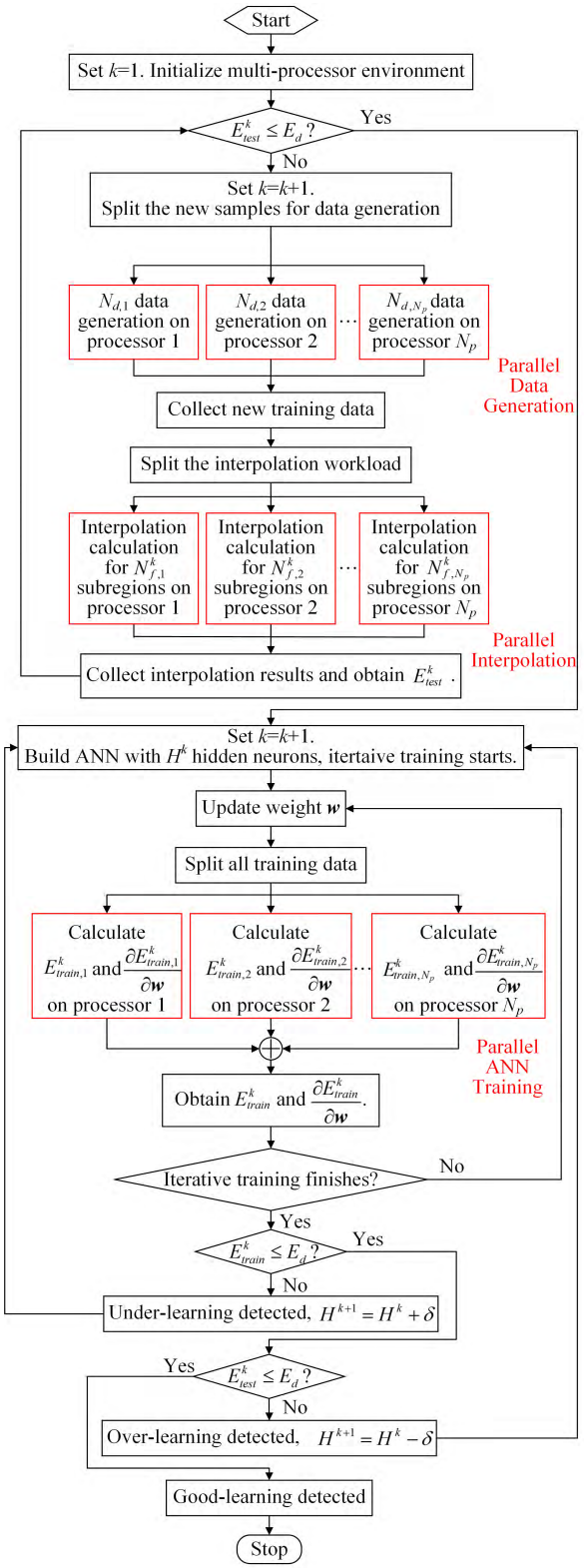


FIGURE 2. Flowchart of the proposed AMG algorithm with parallel interpolation, parallel data generation and parallel ANN training.

network model with 1% testing error for the low-pass filter. The training data are generated by *CST Microwave Studio* EM simulator at 101 frequency points between 1 and 4 GHz.

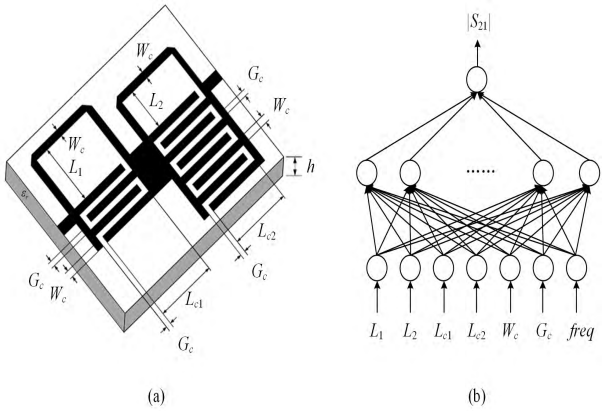


FIGURE 3. Low-pass elliptic microstrip filter example.

We use C language to program the proposed AMG algorithm, including interpolation calculation, driving EM simulator for data generation, driving *NeuroModelerPlus* software for ANN training and testing. Parallel processing is implemented using OpenMP in C language on Intel Xeon E5-2440 processors. In this example, $N_p = 4$.

For comparison purpose, we also perform the AMG algorithm without interpolation and parallel computation [14] and the AMG algorithm with interpolation but without parallel computation [15] to develop models with 1% testing error for this low-pass filter. The initial guess of the number of hidden neurons is 30, i.e., $H^1 = 30$. The comparison of the CPU time used in different AMG algorithms is listed in Table 1. With interpolation techniques, the proposed AMG algorithm avoids 3 stages of intermediate ANN training during adaptive sampling process and the CPU time is saved compared to existing AMG without interpolation in [14]. With parallel computation, the proposed AMG algorithm is nearly 4 times faster than the existing sequential AMG in [14], and 3 times faster than that in [15], leading to improved efficiency in automated neural-based model development for microwave applications. Table 2 shows the detailed speedup and parallel efficiency of parallel interpolation, parallel data generation and parallel ANN training in the proposed AMG algorithm for the low-pass filter modeling example. The time for interpolation calculation in parallel is 0.022 h, and that for the sequential interpolation calculation is 0.053 h, which results in a speedup (S_f) of 2.41 and a parallel efficiency (η_f) of about 60.2%. The time for 1394 training data generation in parallel is 5.60 h, and that for the sequential data generation is 18.54 h, which results in a speedup (S_d) of 3.31 and a parallel efficiency (η_f) of about 82.7%. The time for parallel ANN training is 0.71 h, and that for sequential ANN training is 2.32 h, which results in a speedup (S_l) of 3.27 and a parallel efficiency (η_l) of about 81.7%. It is observed that the proposed AMG algorithm is more efficient in automated neural-based model development than existing AMG algorithms. The total CPU time can be further reduced if more parallel processors are used.

TABLE 1. Comparisons of Interpolation Time, Data Generation Time and ANN Training Time Between Proposed AMG Algorithm and Existing AMG Algorithms for the Low-pass Filter Example.

AMG algorithms	AMG without interpolation and parallel computation [14]	AMG with interpolation but without parallel computation [15]	Proposed AMG with interpolation and parallel computation
No. of processors	1	1	4
No. of interpolation stages	/	3	3
Interpolation time (h)	/	0.053	0.022
No. of training data	1394	1394	1394
Data generation time (h)	18.54	18.54	5.60
No. of ANN training stages	6	3	3
ANN training time (h)	4.15	2.32	0.71
Total CPU time (h)	22.69	20.91	5.90

TABLE 2. Speedup and Parallel Efficiency of Parallel Interpolation, Parallel Data Generation and Parallel ANN Training in Proposed AMG Algorithm (Using 4 Parallel Processors) for the Low-pass Filter Modeling Example.

	Speedup	Parallel efficiency
Interpolation calculation	2.41	60.2%
Data generation	3.31	82.7%
ANN training	3.27	81.7%

B. AUTOMATED MODEL DEVELOPMENT OF BANDPASS HTS MICROSTRIP FILTER

In this example, we consider the parametric modeling of an HTS quarter-wave parallel coupled-line microstrip filter [22], [26]–[28], as illustrated in Fig. 4 (a). The geometrical input parameters of the ANN model shown in Fig. 4 (b) are $\boldsymbol{x} = [L_1\ L_2\ L_3\ S_1\ S_2\ S_3]^T$, where L_1, L_2 and L_3 are the lengths of the parallel coupled-line sections, and S_1, S_2 and S_3 are the gaps between the sections. In this bandpass filter structure, L_0 is the length of the input and output transmission line feeding the coupled line filter. We assume the same width $W = 0.635$ mm for all the sections. The thickness of the lanthanum-aluminate substrate h equals to 0.508 mm and the dielectric constant of the substrate ϵ_r equals to 23.425.

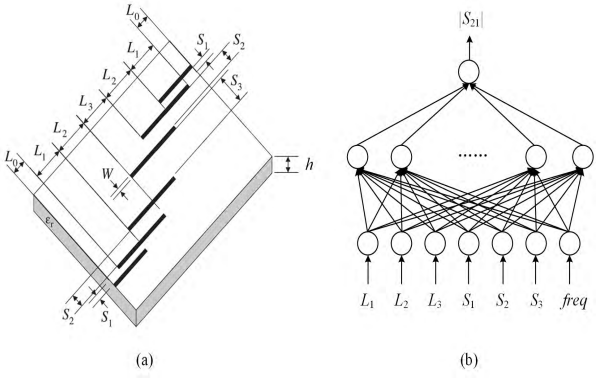


FIGURE 4. Bandpass HTS microstrip filter example.

TABLE 3. Comparisons of Interpolation Time, Data Generation Time and ANN Training Time Between Proposed AMG Algorithm and Existing AMG Algorithms for the Bandpass Filter Example.

AMG algorithms	AMG without interpolation and parallel computation [14]	AMG with interpolation without parallel computation [15]	Proposed AMG with interpolation and parallel computation
No. of processors	1	1	8
No. of interpolation stages	/	3	3
Interpolation time (h)	/	0.062	0.013
No. of training data	1394	1394	1394
Data generation time (h)	19.36	19.36	4.23
No. of ANN training stages	6	3	3
ANN training time (h)	4.80	2.51	0.52
Total CPU time (h)	24.16	21.93	4.76

For comparison purpose, we perform the existing sequential AMG algorithms with or without interpolation techniques [14], [15] and our proposed AMG algorithm with interpolation and parallel computation to develop ANN models for the bandpass filter. The user-desired testing error of the model is 1%. In this example, 8 processors are used for parallel processing of the proposed AMG algorithm, i.e., $N_p = 8$. Table 3 shows the comparison of the model development time

using three AMG algorithms. Because of the incorporation approaches, the proposed AMG algorithm uses less time for ANN training than the AMG algorithm in [14]. Furthermore, the proposed AMG algorithm uses parallel processing in interpolation calculation and ANN training process, leading to further reduction in interpolation time and ANN training time over the existing AMG algorithms in [14] and [15]. During the AMG process, 1394 training data are needed to develop the neural network model. Using parallel data generation method, the data generation time is decreased from 19.36 h to 4.23 h. In this way, the total CPU time of the model development using proposed AMG algorithm is less than that using the previously published AMG algorithms.

TABLE 4. Speedup and Parallel Efficiency of Parallel Interpolation, Parallel Data Generation and Parallel ANN Training in Proposed AMG Algorithm (Using 8 Parallel Processors) for the Bandpass Filter Modeling Example.

	Speedup	Parallel efficiency
Interpolation calculation	4.77	59.6%
Data generation	4.58	57.2%
ANN training	4.46	55.8%

The speedup and parallel efficiency of the proposed AMG algorithm is shown in Table 4. A speedup (S_f) of 4.77 and a parallel efficiency (η_f) of 59.6% is achieved for parallel interpolation calculation; a speedup (S_d) of 4.58 and a parallel efficiency (η_d) of 57.2% is achieved for parallel data generation; a speedup (S_l) of 4.46 and a parallel efficiency (η_l) of 55.8% is achieved for parallel ANN training. Therefore, the entire AMG process is benefited from the computation speedup of the proposed parallelization including parallel interpolation calculation, parallel data generation and parallel ANN training. From Table 3 and Table 4, it is observed the proposed AMG algorithm is more efficient for automated neural model development than existing AMG algorithms.

IV. CONCLUSION

In this paper, an advanced automated neural-based modeling algorithm of microwave devices using parallel computation and interpolation approaches has been proposed to improve the neural model development efficiency. Parallel interpolation approaches has been proposed to assess the adequacy of training data and to determine the distribution of training data in the modeling input space. Parallel data generation has been achieved by automatically driving multiple EM/physical/circuit simulators in parallel on multiple processors. Parallel neural network training has been incorporated by performing training/testing error computation and error backpropagation in parallel on multiple processors. The proposed algorithm has been illustrated by automated modeling of two microwave filter examples.

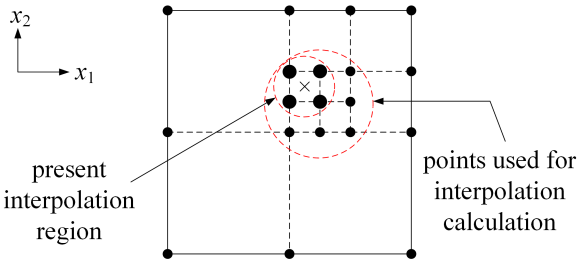


FIGURE 5. Automated sampling distribution in the 4th stage by the proposed AMG for a 2-D example. A total of 9 points are selected in and around the present interpolation region to calculate interpolation function. A factor of 10 is applied to 4 vertices of the interpolation region when formulating matrix X and vector d in (3).

APPENDIX

In this appendix, we provide detailed information of the matrix $\boldsymbol{X}$ and the vector $\boldsymbol{d}$ defined in (3) following the existing literature [15]. When calculating the interpolation in one region, the region is called the interpolation region. In the proposed interpolation formulation of one subregion, (3) is the key formula to compute the values of coefficients $\boldsymbol{\alpha}$ in the interpolation function. We select M training data $(\boldsymbol{x}^{(i)}, d^{(i)}), i = 1, 2, \ldots, M$ in and around the interpolation region to form matrix $\boldsymbol{X}$ and vector $\boldsymbol{d}$. These M points are the available training data nearest to the center of the interpolation region. The elements in $\boldsymbol{d}$ are output values $d^{(i)}$ of these M training data, and the elements in $\boldsymbol{X}$ are the location information of these training data. For models with 2 inputs ($n = 2$) as an example, 9 training data are selected to form $\boldsymbol{X}$ and $\boldsymbol{d}$, including 4 data points (vertices) of the interpolation region, and 5 more data points in the neighboring regions $M = 9$. A weighting factor of 10 is applied to the 4 vertices of the interpolation subregion when forming matrix $\boldsymbol{X}$ and $\boldsymbol{d}$. Fig. 5 shows a 2-dimensional example of the interpolation and points used during the interpolation process.

REFERENCES

[1] Q.-J. Zhang, K. C. Gupta, and V. K. Devabhaktuni, "Artificial neural networks for RF and microwave design—From theory to practice," *IEEE Trans. Microw. Theory Techn.*, vol. 51, no. 4, pp. 1339–1350, Apr. 2003.
[2] J. E. Rayas-Sanchez, "EM-based optimization of microwave circuits using artificial neural networks: The state-of-the-art," *IEEE Trans. Microw. Theory Techn.*, vol. 52, no. 1, pp. 420–435, Jan. 2004.
[3] P. Sen, W. H. Woods, S. Sarkar, R. J. Pratap, B. M. Dufrene, R. Mukhopadhyay, C.-H. Lee, E. F. Mina, and J. Laskar, "Neural-network-based parasitic modeling and extraction verification for RF/millimeter-wave integrated circuit design," *IEEE Trans. Microw. Theory Techn.*, vol. 54, no. 6, pp. 2604–2614, Jun. 2006.
[4] D. E. Root, "Future device modeling trends," *IEEE Microw. Mag.*, vol. 13, no. 7, pp. 45–59, Nov./Dec. 2012.
[5] L. Zhu, Q. Zhang, K. Liu, Y. Ma, B. Peng, and S. Yan, "A novel dynamic neuro-space mapping approach for nonlinear microwave device modeling," *IEEE Microw. Wireless Compon. Lett.*, vol. 26, no. 2, pp. 131–133, Feb. 2016.
[6] W. Liu, 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 Trans. Microw. Theory Techn.*, vol. 65, no. 6, pp. 2043–2062, Feb. 2017.
[7] H. Kabir, L. Zhang, and K. Kim, "Automatic parametric model development technique for RFIC inductors with large modeling space," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Honolulu, HI, USA, Jun. 2017, pp. 551–554.

[8] F. Feng, 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 Trans. Microw. Theory Techn.*, vol. 65, no. 6, pp. 1955–1975, Jun. 2017.

[9] W. Na, F. Feng, C. Zhang, and Q.-J. Zhang, "A unified automated parametric modeling algorithm using knowledge-based neural network and l_1 optimization," *IEEE Trans. Microw. Theory Techn.*, vol. 65, no. 3, pp. 729–745, Mar. 2017.

[10] S. Koziel, J. W. Bandler, and K. Madsen, "Space mapping with adaptive response correction for microwave design optimization," *IEEE Trans. Microw. Theory Techn.*, vol. 57, no. 2, pp. 478–486, Feb. 2009.

[11] R. Ben-Ayed, J. Gong, S. Brisset, F. Gillon, and P. Brochet, "Three-level output space mapping strategy for electromagnetic design optimization," *IEEE Trans. Magn.*, vol. 48, no. 2, pp. 671–674, Feb. 2012.

[12] W. Zhang, F. Feng, V.-M.-R. Gongal-Reddy, J. Zhang, S. Yan, J. Ma, and Q.-J. Zhang, "Space mapping approach to electromagnetic centric multiphysics parametric modeling of microwave components," *IEEE Trans. Microw. Theory Techn.*, vol. 66, no. 7, pp. 3169–3185, Jul. 2018.

[13] C. Zhang, J. Jin, W. Na, Q. J. Zhang, and M. Yu, "Multivalued neural network inverse modeling and applications to microwave filters," *IEEE Trans. Microw. Theory Techn.*, vol. 66, no. 8, pp. 3781–3797, Aug. 2018.

[14] V. K. Devabhaktuni, M. C. E. Yagoub, and Q.-J. Zhang, "A robust algorithm for automatic development of neural-network models for microwave applications," *IEEE Trans. Microw. Theory Techn.*, vol. 49, no. 12, pp. 2282–2291, Dec. 2001.

[15] W. Na and Q.-J. Zhang, "Automated parametric modeling of microwave components using combined neural network and interpolation techniques," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Seattle, WA, USA, Jun. 2013, pp. 1–3.

[16] Q.-F. Cheng, H.-P. Fu, and J.-G. Ma, "Reliability modeling and testing techniques for RF/microwave power amplifiers," in *Proc. IEEE MTT-S Int. Conf. Numer. Electromagn. Multiphys. Modeling Optim. (NEMO)*, Beijing, China, Jul. 2016, pp. 1–3.

[17] Q. Lin, H. Fu, W. Na, F. He, X. Li, Q. Cheng, and Y. Zhu, "Interconnect reliability analysis of ULSI using automated model generation algorithm," *Int. J. RF Microw. Comput.-Aided Eng.*, vol. 26, no. 6, pp. 481–488, Aug. 2016.

[18] R. Trobec, M. Vajtersic, and P. Zinterhof, *Parallel Computing: Numerics, Applications, and Trends*. New York, NY, USA: Springer, 2009.

[19] J. Bilski and J. Smolag, "Parallel architectures for learning the RTRN and Elman dynamic neural networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 9, pp. 2561–2570, Sep. 2015.

[20] F. Naveros, N. R. Luque, J. A. Garrido, R. R. Carrillo, M. Anguita, and E. Ros, "A spiking neural simulator integrating event-driven and time-driven computation schemes using parallel CPU-GPU co-processing: A case study," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 7, pp. 1567–1574, Jul. 2015.

[21] J. A. Cruz-López, V. Boyer, and D. Ei-Baz, "Training many neural networks in parallel via back-propagation," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. Workshops*, Lake Buena Vista, FL, USA, May/Jun. 2017, pp. 501–509.

[22] F. Feng, C. Zhang, V.-M.-R. Gongal-Reddy, Q.-J. Zhang, and J. Ma, "Parallel space-mapping approach to EM optimization," *IEEE Trans. Microw. Theory Techn.*, vol. 62, no. 5, pp. 1135–1148, May 2014.

[23] J. Zhang, K. Ma, F. Feng, Z. Zhao, W. Zhang, and Q. Zhang, "Distributed parallel computing technique for EM modeling," in *Proc. IEEE MTT-S Int. Conf. Numer. Electromagn. Multiphys. Modeling Optim. (NEMO)*, Aug. 2015, pp. 1–3.

[24] S. Zhang, J. Xu, Q.-J. Zhang, and D. E. Root, "Parallel matrix neural network training on cluster systems for dynamic FET modeling from large datasets," in *IEEE MTT-S Int. Microw. Symp. Dig.*, San Francisco, CA, USA, May 2016, pp. 1–3.

[25] W. H. Tu and K. Chang, "Microstrip elliptic-function low-pass filters using distributed elements or slotted ground structure," *IEEE Trans. Microw. Theory Techn.*, vol. 54, no. 10, pp. 3786–3792, Oct. 2006.

[26] M. H. Bakr, J. W. Bandler, M. A. Ismail, J. E. Rayas-Sánchez, and Q.-J. Zhang, "Neural space-mapping optimization for EM-based design," *IEEE Trans. Microw. Theory Techn.*, vol. 48, no. 12, pp. 2307–2315, Dec. 2000.

[27] J. W. Bandler, Q. S. Cheng, S. A. Dakroury, A. S. Mohamed, M. H. Bakr, K. Madsen, and J. Sondergaard, "Space mapping: The state of the art," *IEEE Trans. Microw. Theory Techn.*, vol. 52, no. 1, pp. 337–361, Jan. 2004.

[28] S. Koziel, Q. S. Cheng, and J. W. Bandler, "Space mapping," *IEEE Microw. Mag.*, vol. 9, no. 6, pp. 105–122, Dec. 2008.

WEICONG NA received the B.Eng. degree from Tianjin University, Tianjin, China, in 2012, and the Ph.D. degree from the School of Microelectronics, Tianjin University, Tianjin, China, and the Department of Electronics, Carleton University, Ottawa, ON, Canada, in 2018.

She is currently a Lecturer with the Faculty of Information Technology, Beijing University of Technology, Beijing, China. Her research interests include microwave circuit modeling and design, automated neural network model generation algorithm, neural network extrapolation algorithm, and EM field knowledge-based modeling and optimization.

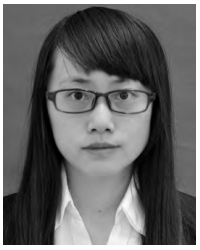
WANRONG ZHANG was born in Hebei, China, in 1964. He received the B.S. and M.S. degrees in microelectronics and solid-state electronics from Lanzhou University, Lanzhou, China, in 1987 and 1990, respectively, and the Ph.D. degree in microelectronics and solid-state electronics from Xi'an Jiaotong University, Xi'an, China, in 1996.

He is currently a Professor and a Ph.D. Supervisor with the Faculty of Information Technology, Beijing University of Technology, Beijing, China. He has authored or coauthored more than 130 publications. He coauthored *Semiconductor Device Electronics* (Beijing, China: Publishing House of Electronics Industry, 2005), *Low Cost Flip Chip Technologies for DCA, WLCSP, and PBGA Assemblies* (Beijing, China: Chemical Industry House, 2006), and *Power Semiconductor Device Theory and Applications* (Beijing, China: Chemical Industry House, 2005). His current research interests include RF/microwave/millimeter-wave devices and circuits, mixed-signal circuits, cryogenic electronics, device-to-circuit interactions, noise and linearity, reliability physics, device-level simulation, and compact circuit modeling.

Dr. Zhang was selected into Beijing Municipal Trans-Century Talent Plan, in 1999, Beijing Municipal Science and Technology New Star Plan, in 1997, China, and Beijing Municipal Pillar Teacher Plan, in 2005, China. He is currently serving as a Member for the Editorial Board of the *Journal of Semiconductor* and *Power Electronics*.

SHUXIA YAN received the B.Eng. degree in communication engineering from Tianjin Polytechnic University, Tianjin, China, in 2010, and the M.E. and Ph.D. degrees in electromagnetic field and microwave technology from Tianjin University, Tianjin, in 2012 and 2015, respectively.

Since 2015, she has been with the School of Electronics and Information Engineering, Tianjin Polytechnic University. Her current research interests include neural-network-based methods for microwave device modeling and circuit design, and the development of a neural-network-based circuit simulator.



GAOHUA LIU received the B.Eng. degree in communication engineering from Qingdao Science and Technology University, Shandong, China, in 2010, and the M.E. degree in electromagnetic field and microwave technology from Tianjin University, Tianjin, China, in 2013, where she has been with the School of Electronics and Information Engineering, since 2013. Her current research interests include deep learning-based behavior analysis and UAV interaction system development methods.