IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 29 May 2024, accepted 22 June 2024, date of publication 27 June 2024, date of current version 5 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3420255

RESEARCH ARTICLE

Nonlinear Circuit Macromodeling Using New Heterogeneous-Layered Deep Clockwork Recurrent Neural Network

MAEDEH AZODI[1], SAYED ALIREZA SADROSSADAT[1], (Senior Member, IEEE), AND YVON SAVARIA[2], (Life Fellow, IEEE)

[1]Department of Computer Engineering, Yazd University, Yazd 8915818411, Iran
[2]Department of Electrical Engineering, Polytechnique Montréal, Montreal, QC H3T 1J4, Canada

Corresponding author: Sayed Alireza Sadrossadat (alireza.sadr@yazd.ac.ir)

ABSTRACT Nonlinear circuit modeling is a complex task involving sequential data and time-domain analysis. While Recurrent Neural Networks (RNNs) handle these tasks, their limitations include low test accuracy and extended training times. Deep time-domain networks, like deep recurrent neural networks (DRNNs) and deep long short-term memory (DLSTM), aim to address these limitations but bring challenges such as long training times, the vanishing gradient problem, overfitting, and significant test errors due to their large number of parameters. This paper proposes a novel structure and macromodeling approach for nonlinear circuits based on deep clockwork recurrent neural networks (DCWRNNs). Deep CWRNNs offer better feature extraction capability than their shallow counterparts and can generate more accurate and faster models than conventional DRNN and DLSTM. Its unique structure with deep modules operating at different clock periods facilitates better extraction of high and low-frequency information, resulting in smaller test errors. A Heterogeneous-layered DCWRNN (HL-DCWRNN) is also introduced, adjusting module rates of each layer separately to enhance accuracy and mitigate overfitting. DCWRNN-based models require less computation effort (20-30 speedup reported) than transistor-level circuit simulator models. Validation through modeling two nonlinear high-speed interconnect circuits confirms the method's efficacy compared to DRNN, DLSTM, and shallow CWRNN methods.

INDEX TERMS Computer-aided design, clockwork recurrent neural network, deep neural network, heterogeneous structure, macromodeling, nonlinear circuits, recurrent neural network.

I. INTRODUCTION

Complex circuits, systems, and new technologies require high-accuracy macromodeling. This dynamic and evolving research area often needs manual trial-and-error to obtain improved models [1], [2]. The goal of nonlinear macromodeling is to create a simplified representation of the circuit that can capture the relationship between the input and output signals so that it can be evaluated faster and easier than simulating the actual circuit [3].

Artificial neural networks (ANNs) have revolutionized computer-aided design (CAD) by improving the efficiency and effectiveness of designing and modeling nonlinear, high-frequency devices and circuits, significantly enhancing optimization, simulation, and analysis quality [4], [5]. Transistor-level and behavioral models are the two main means for modeling nonlinear components and circuits. Transistor-level models, like SPICE models, offer superior accuracy at the expense of larger computational efforts, as they require complicated calculations based on the internal structure of the circuit. Behavioral models, such as IBIS models [6], have lower accuracy and computational effort requirements, as they only concentrate on the circuit's input and output signals. Therefore, creating efficient models for nonlinear components and circuits is challenging and significant.

Artificial neural networks (ANNs) are powerful techniques for representing correlations between inputs and

The associate editor coordinating the review of this manuscript and approving it for publication was Wu-Shiung Feng.

outputs, significantly influencing computer-aided design (CAD) advancement for circuits and systems. ANNs can alternatively undergo training using simulated data, negating the necessity to understand intricate circuit details. However, static feedforward ANNs are inefficient for modeling complex nonlinear circuits as their behavior is generally represented in the time domain. So, several time-dependent neural networks have been proposed for nonlinear circuit macromodeling in the literature. These include dynamic neural network (DNN) [7], time-delay neural network (TDNN) [8], recurrent neural network (RNN) [9], [10], state-space dynamic neural network (SSDNN) [11] and adjoint state-space dynamic neural network (ASSDNN) [12]. Among them, RNNs have gained extensive utilization and have been proposed as a promising approach for nonlinear macromodeling. Indeed, RNNs can capture and depict the nonlinear dynamic characteristics in the time domain by leveraging the time domain data of the circuit.

Artificial neural networks can learn and approximate any nonlinear input-output relationship, thanks to their universal approximation property [13]. They can also use measured or simulated data directly, without requiring the knowledge of the inner intricacies of the original circuit. Moreover, they can evaluate the output results more rapidly than the original models within circuit simulators [3], [9], [14]. While RNNs offer promising capabilities for macromodeling, they grapple with limitations. Training them can be time-consuming, demanding iterative backpropagation through time (BPTT) calculations. This process can lead to vanishing gradients, hindering weight updates, and compromising model accuracy.

Many advancements and alternative architectures have emerged in the literature to overcome these drawbacks. One of the first applications of RNNs for nonlinear macromodeling is presented in [3], where an RNN was trained to model the dynamic response of nonlinear circuits using simulated or measured data. A state-space dynamic neural network (SSDNN) was proposed in [11] for modeling the transient behavior of high-speed nonlinear circuits. The SSDNN's state-space approach facilitated parameter reduction and improved training convergence.

An enhanced version of SSDNN was introduced in [12] to model the transient response of nonlinear electronic/photonic components, employing the Adjoint State-Space Dynamic Neural Network (ASSDNN) technique. Augmenting training data with output signal derivatives reduces the required data set volume without compromising accuracy, a key strength of this approach.

Building upon the RNN framework, L. Zhu, et al. [26] introduced a dynamic neuro-space mapping (Neuro-SM) technique for accurate nonlinear device modeling. Aiming to refine existing approximate models, Neuro-SM leverages an RNN to learn the mapping function between the approximate and actual device behavior.

Other types of neural networks, such as echo state network (ESN) [15], long short-term memory (LSTM) [6], and batch normalized recurrent neural network (BN-RNN) [16] have been applied for nonlinear macromodeling.

A possible solution to conventional RNNs subject to the vanishing gradient problem is to use LSTM networks, which use gates to control the information flow. In [6], the authors proposed an LSTM-powered macromodeling approach for capturing the dynamic behavior of nonlinear circuits. However, LSTMs also have limitations, such as overfitting and large computational requirements, which might make them unsuitable for complex nonlinear circuits with limited resources.

Stacking hidden layers helps to increase the depth and complexity of the neural network, which can improve its ability to learn features and patterns from the data [17]. Simple RNNs and LSTM have only one hidden layer, which limits their capacity to model nonlinear circuits. Stacking hidden layers empowers the network to delve deeper into input-output data, unearthing increasingly abstract and intricate features.

In [14], a new deep method for macromodeling nonlinear electronic circuits, such as boost converters, is introduced. The method is called local feedback deep recurrent neural network (LFDRNN). It uses a deep recurrent neural network (RNN) with feedback connections between the same layers in adjacent time steps. The feedback connections allow the hidden layers to remember and use some information from the previous time step. The LFDRNN method can model and analyze the dynamic behavior of the circuits more accurately and efficiently than shallow RNNs. The paper shows how the LFDRNN method can be used to model and implement an active voltage balancing circuit for series-connected supercapacitors in energy harvesting systems. The experimental results show that the LFDRNN method is better than the existing transistor-level and other RNN-based models regarding accuracy and speed. Another paper also presents a dropout regularization technique to improve the generalization performance of the LFDRNN method [18].

In [16], a new method for macromodeling nonlinear and high-speed circuits using batch-normalized deep recurrent neural networks (BN-RNN) was introduced. The method uses batch normalization (BN) to enhance the training and accuracy of deep RNN and significantly decrease the training time. The BN technique modifies each neuron's inputs based on the batches' mean and variance and adds scaling and shifting parameters trained along with the network. The method lowers the internal covariance shift and improves the gradient flow of the network. The paper validates the performance of the BN-RNN method using three nonlinear circuit examples.

In [19], the authors introduce a heterogeneous methodology that integrates recurrent neural network (RNN) and polynomial regression techniques for macromodeling

nonlinear electronic circuits. The authors called this method heterogeneous RNN-polynomial regression (HRPR). The HRPR method involves two stages: firstly, constructing an RNN structure and, subsequently, merging the RNN output with external input(s) of the circuit to perform regression. The study illustrates that the HRPR method can efficiently model and simulate three nonlinear circuits with greater speed and accuracy than conventional RNN and existing models in simulation tools.

The work in [20] introduces a novel macromodeling technique employing a deep gated recurrent unit (Deep GRU) regularized with Gaussian dropout for nonlinear circuits. The method uses Deep GRU as a new RNN that can solve the vanishing gradient problem and learn long-term dependencies. The method also uses Gaussian dropout as a regularization technique to avoid overfitting and enhance generalization performance. The paper verifies the performance of the proposed method using three nonlinear circuit examples.

A new macromodeling method is proposed. This method, called shallow clockwork recurrent neural network (CWRNN), is a recurrent neural network (RNN) that splits the hidden layer into modules operating at different clock rates, allowing it to capture different frequencies from the input signals. CWRNNs perform better than RNNs and LSTMs in tasks like sequence prediction, audio signal classification, and nonlinear circuit macromodeling [21], [22].

This paper proposes new deep clockwork recurrent neural networks (DCWRNNs), offering a novel structure and macromodeling method to better deal with complex dependencies in sequential data than conventional deep RNN and deep LSTM. DCWRNNs have some hidden layers composed of different modules with different clock rates, enabling them to extract fast and slow changes (happening at different rates) from the training signals. DCWRNNs have fewer parameters, higher accuracy than deep RNN and deep LSTM, and a simpler and more robust architecture. They also have better feature extraction capability than their shallow version, compared to conventional deep methods.

Using modules with different clock rates makes the DCWRNN a strong tool for capturing high- and low-frequency information from the data, and using the deep structure leads to better feature extraction and model accuracy. This paper proposes a heterogeneous structure called Heterogeneous-layered DCWRNN (HL-DCWRNN), which uses layers with different numbers of modules. This heterogeneous structure helps to tune the number of modules in each layer to reduce overfitting and increase accuracy.

The rest of the paper is organized as follows: Section II presents background information on conventional deep RNN and deep LSTM. Section III explains the proposed DCWRNN macromodeling approach, including its structure, training, and proposed Heterogeneous-layered DCWRNN (HL-DCWRNN). Section IV showcases numerical results for two nonlinear examples. Finally, Section V concludes the paper.

II. BACKGROUND

A. CONVENTIONAL DEEP RECURRENT NEURAL NETWORK (DRNN)

A recurrent neural network (RNN) comprises similar feedforward neural networks connected through time by feedback connections. These connections enable the network to propagate information across consecutive time steps. RNN has been widely used in domains such as image and language processing [21], where complex inputs and outputs can be handled. RNN has three main layers: the input layer, the hidden layers, and the output layer. Recently, deep RNNs, which have multiple hidden layers, have been used to generate macromodels of nonlinear circuits [14], [17]. These models can capture the dynamic behavior of the circuits more accurately and efficiently than conventional shallow RNN-based models. The output of the i-th hidden neuron in the l-th hidden layer at the k-th time step, where k is an integer value, is given by the following equation:

$$Z_i^l(k) = \sigma\Big(\sum_{j=1}^{N_{l-1}} W_{i,j}^{F_{l-1}} Z_j^{l-1}(k) + \sum_{j=1}^{N_l} W_{i,j}^{R_l} Z_j^l(k-1) + b_i^l\Big) \quad (1)$$

$$y(k) = \sigma\left(\sum_{i=1}^{N_L} W_{1,i}^{F_L} Z_i^L(k) + b_y\right) \quad (2)$$

In this context, equations (1) and (2) correspond to the output of the i-th hidden neuron at layer l and the ultimate output of the network, respectively. The structure considered here features a single output for simplicity. The terms $Z_j^{l-1}(k)$, N_{l-1}, σ, $W_{i,j}^{R_l}$, $W_{i,j}^{F_{l-1}}$, and $W_{1,i}^{F_L}$ represent the output of the j-th node in the $(l-1)$-th hidden layer at time step k, the number of nodes in the $(l-1)$-th hidden layer, the sigmoid activation function, the recurrent weight, the feedforward weight, and the weight connecting the DRNN output to the final hidden layer at time step k, respectively. Additionally, b_i^l and b_y denote the biases associated with the hidden neuron i in the l-th hidden layer and the output neuron, respectively.

Training a recurrent neural network (RNN) requires using a technique known as backpropagation through time (BPTT) [14], [23], [24]. This technique allows for the computation of the gradient of the error function for the network weights. The gradient indicates how to adjust the weights to reduce the error. BPTT works by unfolding the RNN for a fixed number of time steps and treating it as a deep feedforward network with shared weights. Then, the chain rule orchestrates the calculation of error partial derivatives for all weights at each time step. When carefully processed, these derivatives guide weight updates, leading the network towards reduced training error.

Deep LSTM is a time-domain neural network that can better deal with sequential data and learn long-term dependencies than deep RNN. It has multiple layers of LSTM units stacked on each other, allowing for more complex and hierarchical data representations. Deep LSTM was first used for speech recognition and later for machine translation and nonlinear circuit modeling [20].

B. DEPTH IN CLOCKWORK RECURRENT NEURAL NETWORKS VERSUS DEPTH IN CONVENTIONAL RECURRENT NEURAL NETWORKS

The main idea of the proposed method is to use the advantage of deep neural networks to improve their modeling performance. Regarding the clockwork network, it should be emphasized that deepening the network is not simply adding or stacking hidden layers, but rather adjusting the number of modules and neurons inside those modules and the clock periods used in each layer. Therefore, 3 hyperparameters in each layer of DCWRNN are to be configured rather than just the number of neurons in conventional deep RNN. For example, one layer could have 7 modules, each with a clock rate of power of 2 containing 5 neurons (2^1, 2^2, 2^3, 2^4, 2^5, 2^6,2^7), another layer could have 5 modules, each with 4 neurons and clock rates of (2^1, 3^1, 2^2, 2^3, 3^2). Hence, each layer in HL-DCWRNN needs to be carefully designed rather than simply stacking multiple similar layers on top of each other. In clockwork RNNs, prime-numbered clock rates allow modules to update independently at prime intervals, minimizing overlap and interference, thus capturing diverse temporal patterns and optimizing learning efficiency.

In Clockwork RNNs, modules within the hidden layer are connected to the immediate past and previous time steps that align with their designated clock rates. For instance, with clock rates set at [1, 2, 4, 8, 16], modules form connections to time steps that are 1, 2, 4, 8, and 16 steps behind, respectively. This design ensures that each module processes information at intervals specific to its clock rate, allowing the network to handle different aspects of temporal data efficiently and with a structured update pattern.

C. DEEP LONG-SHORT TERM MEMORY (DLSTM)

In Fig. 1, the LSTM unit is a fundamental component of an LSTM layer responsible for storing and manipulating information over time. Comprising essential elements such as a cell, an input gate, a forget gate, an output gate, and a new memory cell, the LSTM unit plays a crucial role in processing sequential data. The cell keeps the long-term memory of the unit by remembering information from previous inputs and outputs. The input gate analyzes the current input and the preceding hidden state to choose new information for incorporation into the cell. Simultaneously, operating on comparable criteria, the forget gate identifies information to discard from the cell. At the same time, the output gate determines the data to be transmitted from the cell by considering the present input, the prior hidden state, and the present cell state. Finally, the newly updated memory cell

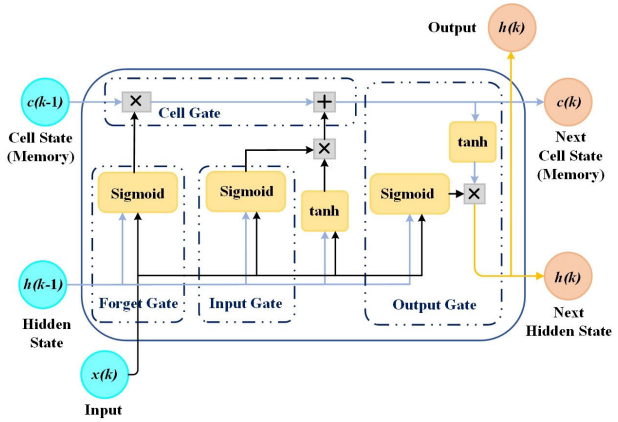

FIGURE 1. The architecture of one LSTM cell.

assesses how the current input impacts the unit's long-term memory. The formulas for each component are given below:

- *Input gate:* $i_k = \sigma(W_i \cdot \boldsymbol{x_k} + U_i \cdot \boldsymbol{h_{k-1}})$.
- *Forget gate:* $f_k = \sigma(W_f \cdot \boldsymbol{x_k} + U_f \cdot \boldsymbol{h_{k-1}})$.
- *Output gate:* $o_k = \sigma(W_o \cdot \boldsymbol{x_k} + U_o \cdot \boldsymbol{h_{k-1}})$.
- *New memory cell:* $\tilde{\boldsymbol{c_k}} = \tanh(W_c \cdot \boldsymbol{x_k} + U_c \cdot \boldsymbol{h_{k-1}})$.
- *Cell state:* $\boldsymbol{c_k} = f_k \odot \boldsymbol{c_{k-1}} + i_k \odot \tilde{\boldsymbol{c}}_k$.
- *Hidden state:* $\boldsymbol{h_k} = o_k \odot \tanh(\boldsymbol{c_k})$.

where the sigmoid function denoted by σ, the hyperbolic tangent function represented by tanh, and the element-wise multiplication operation expressed by $\odot$, $\boldsymbol{x_k}$ is the input vector at time step k, $\boldsymbol{h_k}$ is the hidden state vector at time step k, $\boldsymbol{c_k}$ is the cell state vector at time step k, and W_i, W_f, W_o, W_c, U_i, U_f, U_o, and U_c are weight matrices.

Deep LSTM networks can be used for nonlinear circuit modeling by training them with the original circuit's input and output waveforms as data. These networks can learn the time-domain responses of nonlinear circuits and generate efficient and accurate models for simulation and analysis.

III. PROPOSED DEEP CLOCKWORK RNN MACROMODELING METHOD

A. STRUCTURE OF THE DEEP CLOCKWORK RECURRENT NEURAL NETWORK

The Clockwork RNN splits the hidden layer into separate modules that operate at different temporal granularities and clock rates. The shallow CWRNN has outperformed RNN and LSTM in tasks such as audio signal generation and spoken word classification [21]. The following equations describe the structure of the shallow CWRNN mathematically:

$$Z_{i(a)}(k) = \sigma\Big(\sum_{b=a}^{m}\sum_{j=1}^{h} W^R_{i(a),j(b)} Z_{j(b)}(k-1) + \sum_{j=1}^{N_I} W^I_{i(a),j} I_j(k) + b_{i(a)}\Big) \quad (3)$$

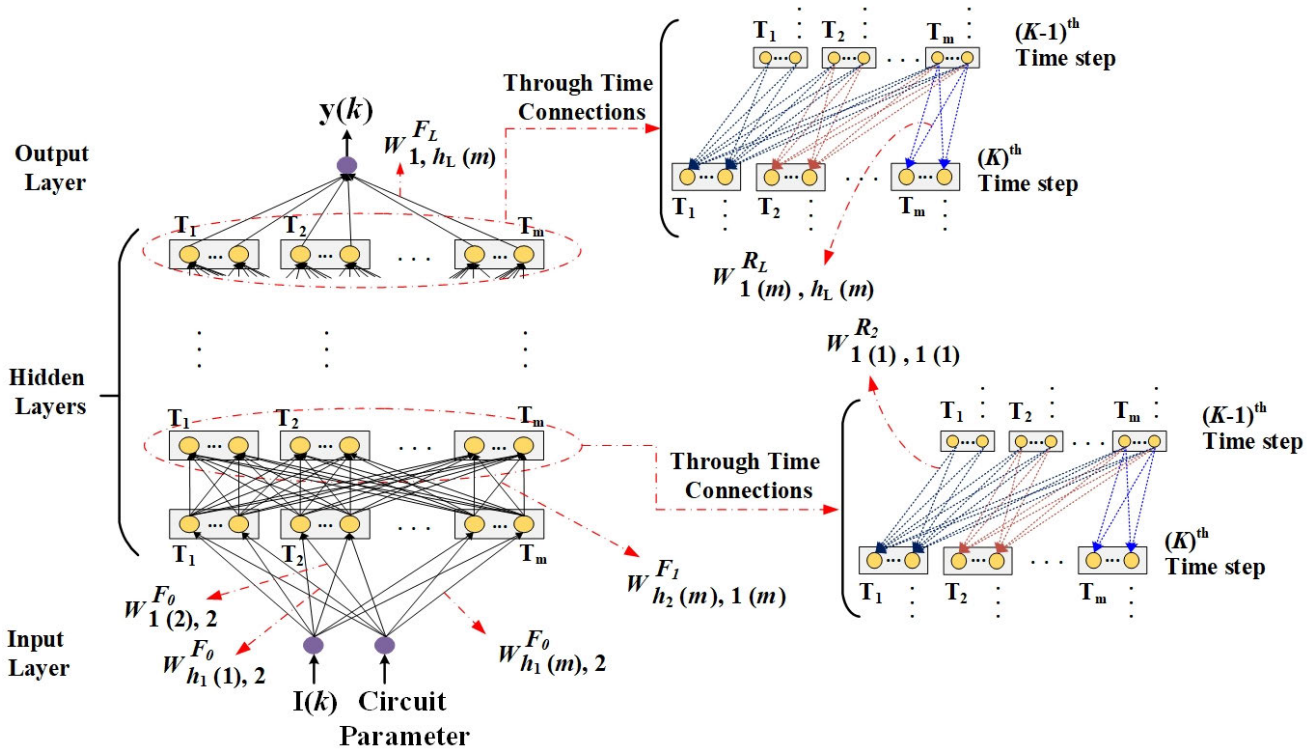

FIGURE 2. Structure of the proposed deep clockwork recurrent neural network (DCWRNN).

$$y(k) = \sigma\left(\sum_{a=1}^{m}\sum_{j=1}^{h} W^{y}_{1,j(a)} Z_{j(a)}(k) + b_y\right) \tag{4}$$

Let's denote (3) and (4) as the output of the i-th hidden neuron within the module a and the ultimate output of the network, correspondingly. In this context, a simplified configuration featuring just one output was considered. The expressions encapsulated in these equations delineate the model's underlying dynamics. In (3), $Z_{j(b)}(k-1)$ signifies the output of the j-th node within module b at time step $(k-1)$, h characterizes the number of nodes within each module in the hidden layer, σ denote the sigmoid activation function, $W^{R}_{i(a),j(b)}$ responsible for linking neuron i in module a at time step k to neuron j in module b at time step $(k-1)$ in the same hidden layer, $W^{I}_{i(a),j}$ signifies the feedforward weight linking neuron j in the input layer to neuron i within a module a in the hidden layer, simultaneously at time step k, $W^{y}_{1,j(a)}$ encapsulates the weight associated with the connection between the shallow CWRNN output and neuron j within a module a in the hidden layer, simultaneously at time step k, $b_{i(a)}$ accounts for the bias affecting hidden neuron i within a module a in the hidden layer. b_y also represents the bias of the output neuron.

The deep clockwork recurrent neural network (DCWRNN) is a type of deep recurrent neural network (DRNN) that has multiple hidden layers, each having a clockwork structure. As shown in Fig. 2, the DCWRNN is delineated into three primary sections: the input layer ($I(k)$ and circuit parameters), the hidden layers, and the output layer ($y(k)$). Additionally, the network assimilates the circuit's input and output signals. The hidden layers are uniformly structured, each comprising a fixed number of modules. This uniformity is a hallmark of the DCWRNN, facilitating synchronized processing across various time scales due to its clockwork architecture. Connections $W^{R_L}_{1(m),h_l(m)}$ link slower module neurons at one time step to faster ones at the next, as depicted in the right side of Fig. 2. This rule is pivotal in differentiating the DCWRNN's information processing capabilities from those of a standard recurrent network.

Each hidden layer in DCWRNN comprises numerous modules, each operating at a different clock period and an equal number of neurons. The clock period of the module i follows the rule $T_i = 2^{i-1}$. The modules are interconnected throughout the layers, with recurrent connections from module a at time step $(k-1)$ to module b at time step k existing only when the clock period of module a, T_a is greater than or equal to T_b, the clock period of module b, $T_a \geq T_b$. The arrangement of the modules in ascending order is designed to enable the smooth transfer of the hidden state from right to left, indicating the transition from slower to faster ones. So, the feedforward weights of a DCWRNN are similar to those of a DRNN, but the recurrent weights are different. In a DCWRNN, the neurons of a slower module T_j with a larger period at time step $(k-1)$ are connected to the neurons of a faster module T_i with a smaller period $(i < j)$ at time step k. This allows a DCWRNN to capture rapid and slow variations in the training signals accurately. A DCWRNN is a simplified DRNN architecture with fewer

connections and parameters than a DRNN. This makes it a less complex network and reduces its computational complexity significantly. The following equations give the mathematical formulation of the DCWRNN architecture:

$$Z^l_{i(a)}(k) = \sigma\Big(\sum_{b=a}^{m}\sum_{j=1}^{h_l} W^{R_l}_{i(a),j(b)} Z^l_{j(b)}(k-1) + \sum_{b=1}^{m}\sum_{j=1}^{h_{l-1}} W^{F_{l-1}}_{i(a),j(b)} Z^{l-1}_{j(b)}(k) + b^l_{i(a)}\Big) \quad (5)$$

$$y(k) = \left(\sum_{a=1}^{m}\sum_{j=1}^{h_L} W^{F_L}_{1,j(a)} Z^L_{j(a)}(k) + b^L_y\right) \quad (6)$$

where (5) and (6) represent the output of the i-th hidden neuron within the module a in layer l and the overall output of the network, both at time step k, correspondingly. For simplicity, we first focus on a configuration featuring a single output in this context. In these equations $Z^{l-1}_{j(b)}(k)$ signifies the output of the j-th node of b-th module in l-th hidden layer, h_l represents the number of nodes in each module of l-th hidden layer, σ represents the sigmoid activation function, the recurrent weight $W^{R_l}_{i(a),j(b)}$ connects neuron i in module a at time step k to neuron j in module b at time step $(k-1)$ in the l-th hidden layer. The feedforward weight $W^{F_{l-1}}_{i(a),j(b)}$ links neuron j in module b of the $(l-1)$-th layer to neuron i in module a of the l-th hidden layer at time step k. The weight $W^{F_L}_{1,j(a)}$ connects the DCWRNN output to the neuron j in module a of the last hidden layer at time step k. The bias term $b^l_{i(a)}$ influences the hidden neuron i in module a within the l-th hidden layer, and b^L_y represents the output neuron bias.

$$\begin{pmatrix} M(1,1)^{R_l}_{h\times h} & M(1,2)^{R_l}_{h\times h} & M(1,3)^{R_l}_{h\times h} & \dots & M(1,m)^{R_l}_{h\times h} \\ 0 & M(2,2)^{R_l}_{h\times h} & M(2,3)^{R_l}_{h\times h} & \dots & M(2,m)^{R_l}_{h\times h} \\ 0 & 0 & M(3,3)^{R_l}_{h\times h} & \dots & M(3,m)^{R_l}_{h\times h} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & M(m,m)^{R_l}_{h\times h} \end{pmatrix} \quad (7)$$

Equation. (7) shows a block-matrix representation of W^{R_l}, the recurrent weight matrix connecting layer l's neurons at time step k to neurons of the same layer at time step $(k-1)$. In this figure, in row i, $M(i,j)^{R_l}_{h\times h}$ represents the $h\times h$ matrix of recurrent connections for the l-th hidden layer between neurons in module i at time step k and neurons in module j at time step $(k-1)$. The modules at the present step establish connections internally with modules of shorter periods at the subsequent time step, resulting in varying connection counts: $h^2 \times m$ in row one, $h^2 \times (m-1)$ in row two, and so forth.

In a DRNN hidden layer, each neuron is transformed at every time step by applying an affine operation to the input at the present step and the hidden state from the prior time step. However, in a DCWRNN at time step k, only the neurons of the i-th module update their values if k is divisible by T_i. Therefore, the modules with large T_i react to "slow" changes in the input, while others (the ones with small T_i) respond to "fast" changes. Furthermore, "fast" modules with short periods get inputs from "slow" modules with long periods from the previous time step, but not vice versa. It means that the "slow" changes influence the "fast" changes, but will get updated without being disturbed by them [25].

The following formula shows the update formula at time step k for the state $Z^l_{i(a)}(k)$ of the i-th feedforward neuron in module a of l-th hidden layer. Assume,

$$A = \Big(\sum_{b=a}^{m}\sum_{j=1}^{h_l} W^{R_l}_{i(a),j(b)} Z^l_{j(b)}(k-1) + \sum_{b=1}^{m}\sum_{j=1}^{h_l} W^{F_{l-1}}_{i(a),j(b)} Z^{l-1}_{i(a),j(b)}(k) + b^l_{i(a)}\Big) \quad (8)$$

Then,

$$Z^l_{i(a)}(k) = \begin{cases} \sigma(A) & if\,(k\ \ MOD\ \ T_a = 0) \\ Z^l_{i(a)}(k-1) & Otherwise \end{cases} \quad (9)$$

For instance, during the time step $k = 8$, only modules with periods $T_1 = 1, T_2 = 2, T_3 = 4$, and $T_4 = 8$ undergo updates, demonstrating that not every module receives updates at each time step. In the equations, the order of the modules satisfies $T_i < T_j$ for $i < j$.

B. TRAINING OF DEEP CLOCKWORK RECURRENT NEURAL NETWORK

Training the DCWRNN structure is necessary to develop an appropriate model exhibiting a nonlinear component's transient behavior. Computing the error and the gradient are two essential steps in the training process. A collection of input-output waveforms, called training data, is used to train the network [26]. In the optimization process of network training, the subsequent objective function (or error function) is employed for the necessary calculations:

$$E_s = \frac{1}{2}\sum_{j=1}^{N_y}\sum_{k=1}^{N_t}(\hat{y}_{s,j}(k) - y_{s,j}(k))^2 \quad (10)$$

The error associated with the s-th training signal, denoted by E_s, is obtained by (10), N_y and N_t, which are the counts of model outputs and the number of time samples, correspondingly. $\hat{y}_{s,j}(k)$ and $y_{s,j}(k)$ indicate the anticipated and desired values for j-th output of s-th training signals at time step k, respectively. The cumulative error across all training waveforms is expressed by E and defined as:

$$E = \frac{1}{2}\sum_{s=1}^{N_s}\sum_{j=1}^{N_y}\sum_{k=1}^{N_t}(\hat{y}_s(j,k) - y_s(j,k))^2 \quad (11)$$

where N_s stands for the total number of training signals. The training aims to adjust the DCWRNN weights $W^{R_l}_{i(a),j(b)}$ and $W^{F_{l-1}}_{i(a),j(b)}$ in (8) to minimize E [27]. This paper utilized The gradient-based optimization method to train the DCWRNN

weights and generate the model. The derivatives of the error function for each training parameter (weight) are needed for applying an effective gradient-based optimization method in training the macromodel [24], [28]. Furthermore, using the traditional error back-propagation strategy for training is not applicable for the DCWRNN due to the dependence of current output on past outputs [23]. The recursive formula given in (12) can be used to calculate the gradients, for $W^{F_l}_{i(a),j(b)}$ $(0 < l < L)$:

$$\frac{\partial E_s(k)}{\partial W^{F_l}_{i(a),j(b)}} = \frac{\partial E_s(k)}{\partial y(k)} \times \sum_{a'=1}^{m} \sum_{p=1}^{h_l} \frac{\partial y(k)}{\partial Z^{L}_{p(a')}(k)} \times \frac{\partial Z^{L}_{p(a')}(k)}{\partial W^{F_l}_{i(a),j(b)}} \tag{12}$$

The computation of (12) continues for several of its chain-rule steps to demonstrate the recursive process of gradient calculations. Notice that, first, the last fraction in (12) should be expanded to reach the desired weight, as shown below:

$$\frac{\partial Z^{L}_{p(a')}(k)}{\partial W^{F_l}_{i(a),j(b)}} = \sum_{a''=1}^{m} \sum_{p'=1}^{h_{l-1}} \frac{\partial Z^{L}_{p(a')}(k)}{\partial Z^{L-1}_{p'(a'')}(k)} \times \frac{\partial Z^{L-1}_{p'(a'')}(k)}{\partial W^{F_l}_{i(a),j(b)}} \tag{13}$$

Also,

$$\frac{\partial y(k)}{\partial Z^{L}_{p(a')}(k)} = W^{F_L}_{1,p(a')} \tag{14}$$

According to (5), the derived relation is based on the condition that $W^{F_{l-1}}_{p(a'),p'(a'')}$ and $Z^{L-1}_{p'(a'')}(k-1)$ will be present in (15) only when $p = i$ and $a' = a$, respectively. Assume,

$$B = W^{F_{l-1}}_{p(a'),p'(a'')} + \left(\sum_{a''=a'}^{m} \sum_{p'=1}^{h_l} W^{R_{L-1}}_{p(a'),p'(a'')} \times \frac{\partial Z^{L-1}_{p'(a'')}(k-1)}{\partial W^{F_{L-2}}_{i(a),j(b)}} \right)$$

Then,

$$\frac{\partial Z^{L}_{p(a')}(k)}{\partial Z^{L-1}_{p'(a'')}(k)} = Z^{L}_{p(a')}(k) \times (1 - Z^{L}_{p(a')}(k)) \times B \tag{15}$$

Additionally, as per (5), the resulting relation is established under the condition that $Z^{L-2}_{i(a),j(b)}(k)$, $W^{R_{L-1}}_{p'(a''),q(b)}$, and $Z^{L-1}_{q(b)}(k-1)$ will be included in (16) and (17) only when $p = i$ and $a' = a$, respectively. Assume,

$$C = Z^{L-2}_{i(a),j(b)}(k) + \left(\sum_{b=a''}^{m} \sum_{q=1}^{h_l} W^{L-1}_{p'(a''),q(b)} \times \frac{\partial Z^{L-1}_{q(b)}(k-1)}{\partial W^{F_l}_{i(a),j(b)}} \right)$$

Then,

$$\frac{\partial Z^{L-1}_{p'(a'')}(k)}{\partial W^{F_l}_{i(a),j(b)}} = \sigma' \times C \tag{16}$$

Moreover, by substituting the feedforward weight $W^{F_l}_{i(a),j(b)}$ with the recurrent weight $W^{R_l}_{i(a),j(b)}$ in (12), the ensuing formulas can be derived. Suppose,

$$D = \left(\sum_{b=a''}^{m} \sum_{q=1}^{h_l} W^{R_{L-1}}_{p'(a''),q(b)} \times \frac{\partial Z^{L-1}_{q(b)}(k-1)}{\partial W^{R_l}_{i(a),j(b)}} \right) + Z^{L-1}_{i(a),j(b)}(k-1)$$

Then,

$$\frac{\partial Z^{L-1}_{p'(a'')}(k)}{\partial W^{R_l}_{i(a),j(b)}} = \sigma' \times D \tag{17}$$

For performing the recursive operation, (17) is expanded into the following detailed formula, provided that $Z^{L-1}_{q'(b'')}(k-2)$ appears in (18) only if $q = i$ and $a = b'$. Let's consider,

$$E = \left(\sum_{b''=b'}^{m} \sum_{q'=1}^{h_l} W^{R_{L-1}}_{q(b'),q'(b'')} \times \frac{\partial Z^{L-1}_{q'(b'')}(k-2)}{\partial W^{R_{L-2}}_{i(a),j(b)}} \right) + Z^{L-1}_{i(a),j(b)}(k-2)$$

Then,

$$\frac{\partial Z^{L-1}_{q(b')}(k-1)}{\partial W^{R_{L-2}}_{i(a),j(b)}} = \sigma' \times E \tag{18}$$

This iterative procedure persists until the gradients are determined. Simultaneously, it is entirely trainable end-to-end and can be optimized using the standard BPTT [12], [14]. Leveraging the distinctive structure of the DCWRNN, the slow modules handle and convey long-term information from the input signal. In contrast, the fast modules concentrate on local, high-frequency information (leveraging the information from the slow modules).

The error backpropagation follows a procedure similar to that of the DRNN. However, the key difference is that only the updated modules at time step k affect the error. The errors from other modules are retroactively transported in time and combined with the backward-propagated error [25]. Employing a DCWRNN for training and evaluation, with an equivalent count of layers and neurons, consumes less time than a traditional DRNN, as not all modules undergo updates at every time step [29], [30].

C. RUNTIME COMPARISON BETWEEN THE PROPOSED DCWRNN AND THE CONVENTIONAL DRNN MODELING METHODS

A DCWRNN exhibits a reduction in the number of parameters and computations per time step compared to a conventional DRNN. Imagine a DCWRNN like Fig. 2. A neuron in the previous time step, $(k-1)$, can exclusively establish connections with neurons having an identical or shorter time step k. So, it can be demonstrated that the recurrent matrix W^{R_l} (Fig. 7) has almost half of the n^2 parameters in the traditional DRNN recurrent matrix W_h. Since each module is evaluated only when the time step is a multiple of the clock rate of that module, the maximum number of operations needed for W^R (Fig. 7) at each time step for the original shallow CWRNN is $\frac{2}{m}n^2$. At the same time, the conventional RNN requires n^2 recurrent calculations at each time step, which typically constitutes the most time-intensive operations. Increasing the number of modules in the DCWRNN can be expected to speed up faster than the DRNN.

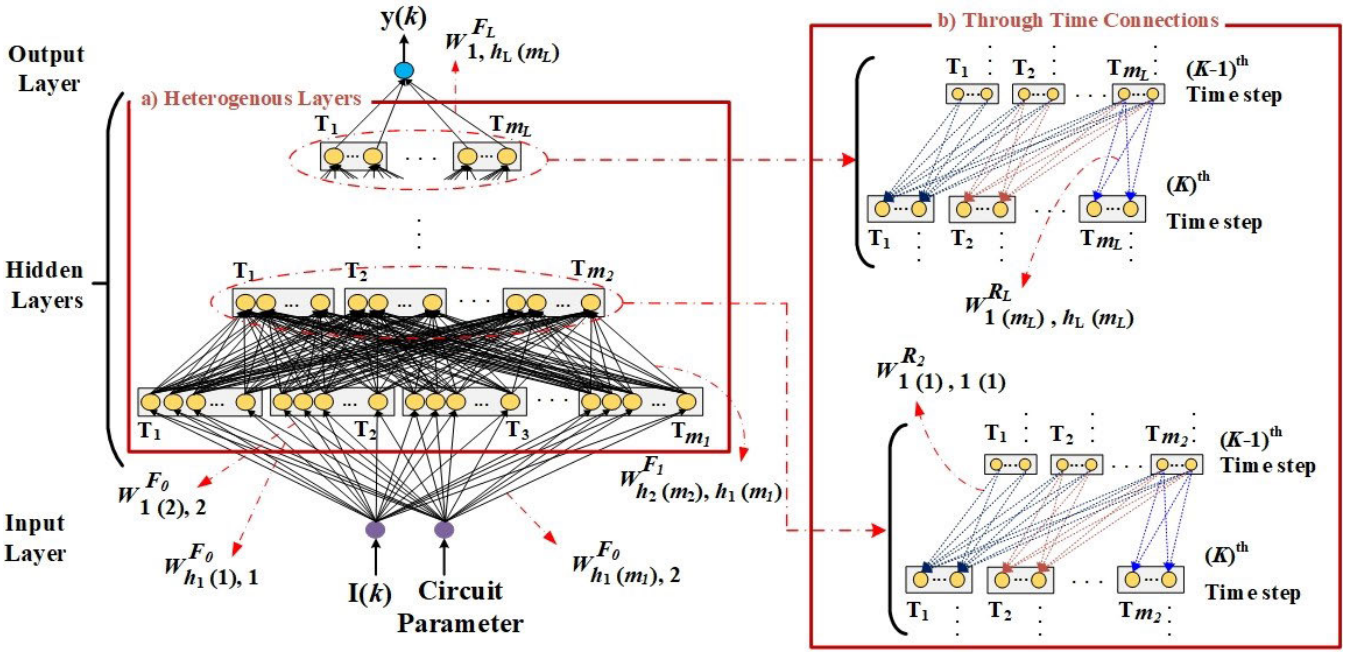

FIGURE 3. Structure of the proposed Heterogeneous-layered DCWRNN (HL-DCWRNN) a) Heterogenous Layers: The heterogeneous approach adds flexibility by varying the clock rates and module sizes, potentially improving the network's ability to model complex data sequences. b) Through Time Connections: connections exist between the current time step and the previous one and among different modules that update at their own clock rates. (These modules are part of the hidden layer and are designed to update only at time steps multiples of their assigned clock rates.)

D. THE PROPOSED HETEROGENEOUS-LAYERED DCWRNN (HL-DCWRNN)

In the Heterogeneous-layered DCWRNN, modules with varying neuron counts are strategically employed based on prime numbers (2,3,...) and their powers, as depicted in Fig. 3. This diverges from the conventional uniform module usage, leveraging the unique timing advantages of prime numbers and their powers to enhance network dynamics across each hidden layer. This method improves the DCWRNN network's efficiency by minimizing parameter count and maximizing module usage efficiency throughout the hidden layers.

The Heterogeneous Deep Clockwork Recurrent Neural Network (HL-DCWRNN), while maintaining the core structure of input (I(k) and circuit parameters), hidden layers, and output (y(k)), introduces a variation in the hidden layer architecture. Unlike the DCWRNN's consistent number of modules, the HL-DCWRNN allows a variable number of modules and neurons across different layers, as shown in Fig. 3. This heterogeneity endows the network with greater flexibility, enabling it to adapt more dynamically to the circuit's behavior by varying the clock rates and module numbers in accordance with the specific demands of each layer. This feature not only enhances the HL-DCWRNN's ability to handle complex temporal patterns, setting it apart from the DCWRNN but also augments its processing capabilities, offering a more nuanced approach compared to its DCWRNN counterpart.

In deep neural networks, hidden layers are crucial in feature extraction from the input signal. Each hidden layer typically extracts distinct features, which can have varying impacts on the final model's effectiveness. Our proposed method selectively activates modules multiples of k in different hidden layers at time step k. This results in diverse clock rates within the Heterogeneous-layered DCWRNN, extracting different features in each layer. For example, the top layers not only possess their own information but also integrate data from the active modules in the lower layers. This heterogeneous approach leads to more efficient training, enabling the detection of a broader range of input signal frequencies and enhancing nonlinear circuit modeling over the original DCWRNN.

The heterogeneous-layered technique consistently outperforms the DCWRNN in accuracy and efficiency while maintaining fewer network parameters. As a result, it doesn't unnecessarily complicate the network's architecture. It is crucial to highlight that the training procedure for the Heterogeneous-layered DCWRNN remains consistent with the original DCWRNN as shown in Fig. 4.

This novel architecture promises enhanced performance and efficiency without introducing unnecessary complexity into the network, making it a valuable advancement in deep learning and nonlinear circuit macromodeling.

IV. NUMERICAL RESULTS

In this section, two nonlinear examples have been used as evidence to verify the validity of the proposed DCWRNN and the Heterogeneous-layered DCWRNN macromodeling techniques.

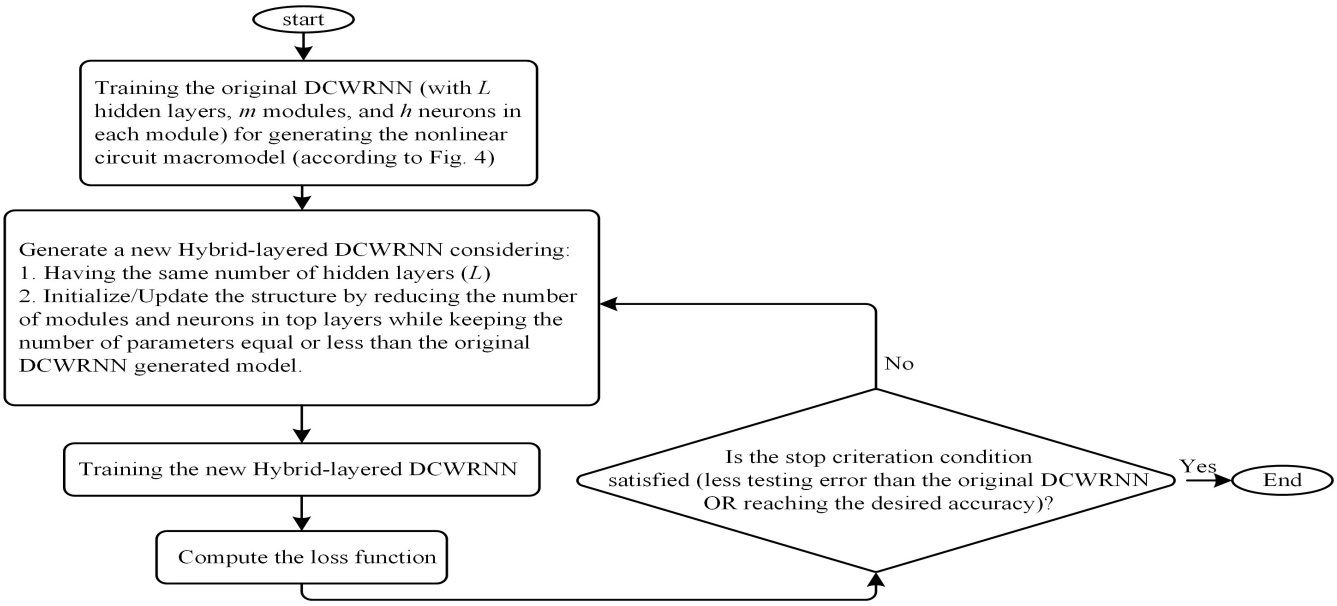

FIGURE 4. The diagram of the macromodeling process according to the proposed Heterogeneous-layered DCWRNN (HL-DCWRNN) method.

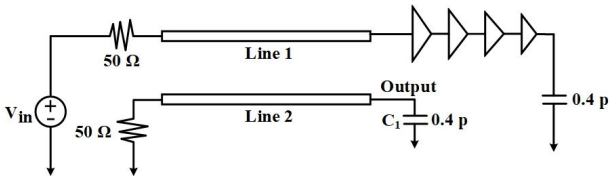

FIGURE 5. 2-coupled line interconnect driven by 4-stage receiver circuit.

A. 2-COUPLED LINE INTERCONNECT TERMINATED BY 4-STAGE RECEIVER

Fig. 5 shows a 2-coupled high-speed interconnected circuit. The input signal is V_{in} (1.2V amplitude, 1ns period). It is connected to a 10 cm lossy transmission line terminated by four CMOS receivers with different sizes of PMOS and NMOS transistors (130nm channel length). The receiver sizes are (PMOS/NMOS widths in nm): 2500/1000, 7850/3140, 24670/9869, and 77515/31006.

Some square waveforms with a 1ns period were generated to train the model for the circuit in Fig. 5 using the DCWRNN approach, and these waveforms' rise and fall times ranged from 80 to 120 ps with 10 ps steps. The static parameter (load capacitance) values varied from 200 to 800 fF with 200fF steps. Some other data not used in the training process were generated as test signals, and these test signals' rise and fall times ranged from 85 to 115ps with 10 ps steps. The test data were produced with 200, 400, 600, and 800 fF load capacitances.

The block diagram in Fig.6 illustrates the DCWRNN-based model employed for this particular circuit. In this diagram, the error value, denoted E, is the average disparity between the DCWRNN-based model's output and the transistor-level model outputs. Additionally, the diagram features key elements, including the load capacitance C_1, the input signal represented as $V_{in}(t)$, and the output signal denoted as $V_{out}(t)$. This representation is a visual overview of the DCWRNN-based modeling approach applied to the specific circuit under examination.

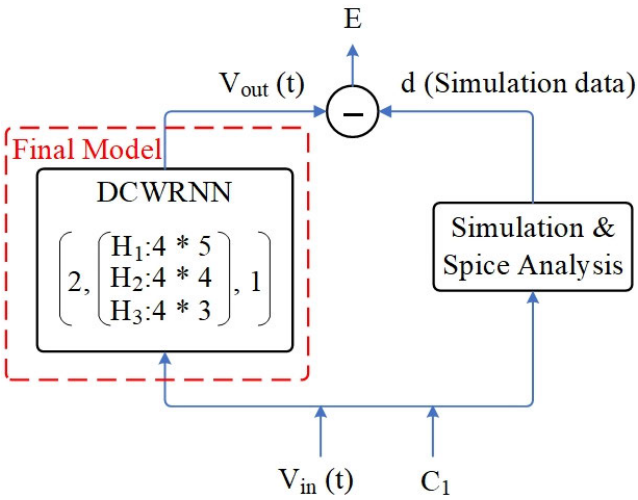

FIGURE 6. Structure of the deep clockwork recurrent neural network (DCWRNN) for modeling the circuit of Fig. 5.

In the proposed DCWRNN framework, the network architecture for the circuit in Fig. 5 is characterized by two inputs, a three-layered hidden structure, each layer including four modules that module sizes are 5, 4, and 3 from the first hidden layer to the last one respectively, and a single output. This configuration is succinctly encapsulated as $(2, [4 \times 5], [4 \times 4], [4 \times 3], 1)$.

Table 1 shows the modeling structure and the number of model parameters. It compares the training and testing errors for the circuit of Fig. 5 using the DRNN, the DLSTM, the shallow CWRNN, and the proposed DCWRNN. Moreover, the proposed method achieves a much lower testing error

TABLE 1. Comparison among the DRNN, the DLSTM, the shallow CWRNN, and the proposed DCWRNN for modeling the circuit of Fig.5.

Model Type	No. of Parameters	Structure	Training error	Testing error
DRNN	1,413	(2, [20, 16, 12], 1)	4.5e-04	9.1e-04
DLSTM	5,613	(2, [20, 16, 12], 1)	3.51e-04	7.4e-04
Shallow CWRNN	1,387	(2, [4 × 11], 1)	3.32e-04	5.72e-04
Proposed DCWRNN	1,113	(2, [4 × 5], [4 × 4], [4 × 3], 1)	3.19e-04	4.3e-04

TABLE 2. Five models' CPU time speedup for Fig.5: DRNN, DLSTM, shallow CWRNN, proposed DCWRNN, and transistor-level.

Model Type	CPU Time(ms)	Speed-up
DRNN	15.9	17
DLSTM	35.2	7.69
Shallow CWRNN	9.8	27.6
Proposed DCWRNN	8.3	32.6
Transistor Level	271	1

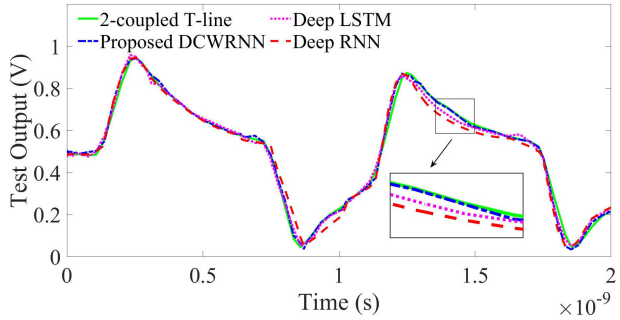

FIGURE 7. Comparison of the test data outputs generated by four different models: The conventional DRNN-based, the conventional DLSTM-based, the proposed DCWRNN-based, and the transistor-level models. The models are applied to the circuit shown in Fig. 5.

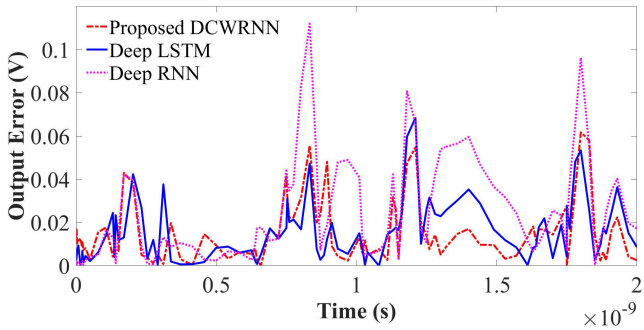

FIGURE 8. Comparison of the absolute test errors of three models: The conventional DRNN-based, the conventional DLSTM-based, and the proposed DCWRNN-based for the circuit shown in Fig. 5.

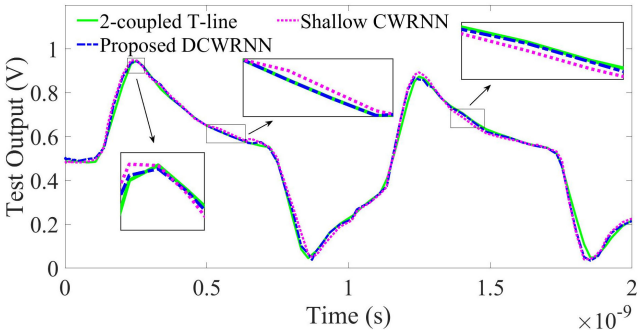

FIGURE 9. Comparison of the output test data generated by four different models: The DRNN-based, the DLSTM-based, the proposed DCWRNN-based, and the transistor-level models for the circuit shown in Fig. 5.

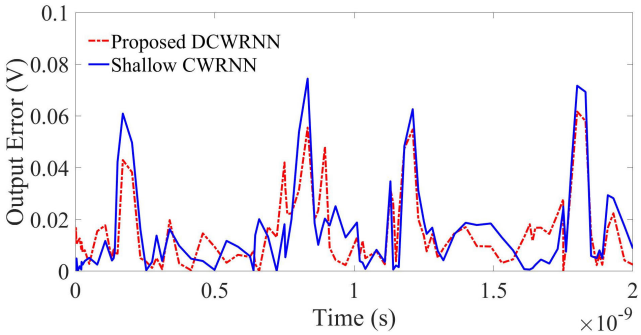

FIGURE 10. Comparison of the absolute test errors of two models: The shallow CWRNN-based and the proposed DCWRNN-based for the circuit shown in Fig. 5.

in the table, which makes the DCWRNN-based model a suitable option for large-scale modeling problems such as high-frequency and complex nonlinear circuits.

The CPU time and the speed-up of the DRNN, the DLSTM, the shallow CWRNN, the proposed DCWRNN, and the transistor-level models for the circuit of Fig. 5 are compared in Table 2. The table reveals that the output of the DCWRNN model is generated much quicker than the transistor-level model (and other models). Thus, the proposed method generates a much faster model suitable for nonlinear applications.

Fig. 7 displays the simulated waveforms for the circuit of Fig. 5. It contrasts the test output signals generated by the DCWRNN, the DRNN, the DLSTM, and transistor-level models. The results in Fig. 7 demonstrate that the DCWRNN model resembles the original circuit output more closely, while the DRNN, and DLSTM perform inadequately in this case. Therefore, the DCWRNN method offers a better model. Furthermore, Fig. 8 presents the output error (absolute error) resulting from comparing the outputs of the DRNN, the DLSTM, and the DCWRNN with that of the original transistor-level model for the circuit in Fig. 5. The figure shows that the DCWRNN performs better according to its smaller output error.

Fig. 9 presents the simulated waveforms of the circuit depicted in Fig. 5, providing a comparative analysis of test output signals generated by the shallow CWRNN, the proposed DCWRNN, and the transistor-level models. The results depicted in Fig. 9 unambiguously establish that the DCWRNN model closely approximates the output characteristics of the original circuit. This observation underscores its capacity to faithfully emulate the circuit's operational behavior. In contrast, the shallow CWRNN

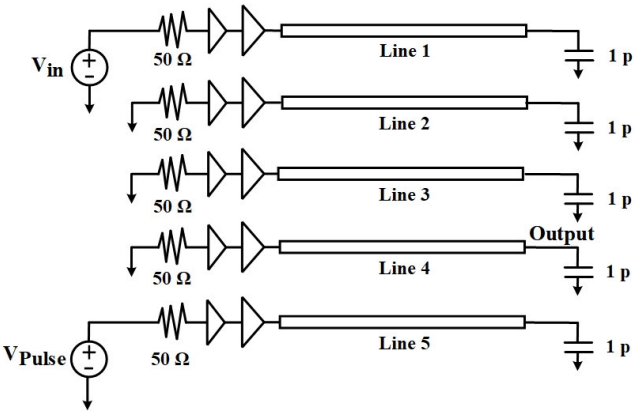

FIGURE 11. 5-coupled transmission line driven by 2-stage driver circuit.

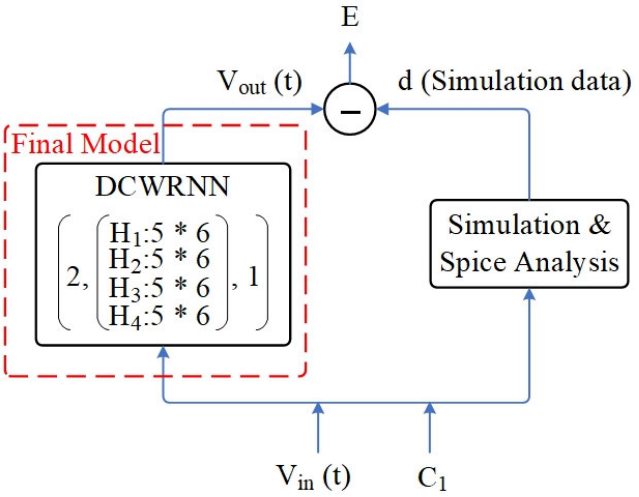

FIGURE 12. Structure of the deep clockwork recurrent neural network (DCWRNN) for modeling the circuit of Fig. 11.

exhibits suboptimal performance within this context, thus reaffirming the preeminence of the DCWRNN model.

Furthermore, Fig. 10 provides a rigorous examination of output error, precisely the absolute error, ensuing from the comparative evaluation of the shallow CWRNN and the DCWRNN outputs with the output of the original transistor-level model associated with the circuit presented in Fig. 5. This graphical representation substantiates that the DCWRNN surpasses the shallow CWRNN, as shown by its diminished output error.

B. 5-COUPLED LINE INTERCONNECT WITH 2-STAGE DRIVER

Demonstrating the DCWRNN method's effectiveness in modeling nonlinear circuits, our second example involves a 2-stage driver connected to a 5-coupled line interconnect. Fig. 11, illustrates the circuit schematic, with V_{in} as the input signal and V_{pulse} as a square wave with an amplitude of 1.2 V and a period of 4.5 ns. Each line functions as a lossy transmission line, spanning a length of 25 cm. The CMOS drivers, featuring a channel length of 130nm, exhibit varied PMOS and NMOS widths. Specifically, the smaller driver

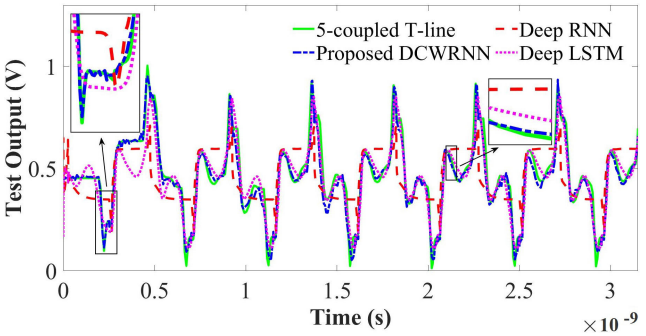

FIGURE 13. Comparison of the test data outputs generated by four different models: The conventional DRNN-based, the conventional DLSTM-based, the proposed DCWRNN-based, and the transistor-level models. The models are applied to the circuit shown in Fig. 11.

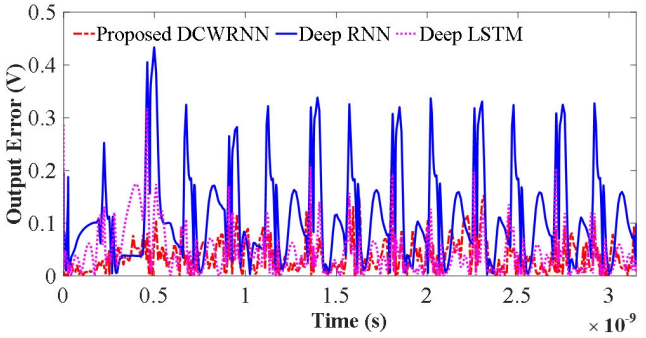

FIGURE 14. Comparison of the absolute test errors of three models: The conventional DRNN-based, the conventional DLSTM-based, and the proposed DCWRNN-based for the circuit shown in Fig. 11.

has 2500nm and 1000nm widths, while the larger driver boasts PMOS and NMOS widths of 7850nm and 3140nm, respectively.

Fig. 12 outlines the application of the DCWRNN-based model to simulate the circuit depicted in Fig. 11. This representation visually depicts the error metric, E, which quantifies the average deviation between the output of the DCWRNN-based model and the transistor-level model outputs. Additionally, it highlights key components such as the load capacitance C_1, the input signal $V_{in}(t)$, and the output signal $V_{out}(t)$. This graphical depiction offers insights into the specific implementation of the DCWRNN-based modeling approach for the analyzed circuit.

The characteristics of this circuit necessitate a higher number of sampled time steps, making conventional RNN models inadequate for accurately modeling the circuit. Physical source swiping is used to generate different voltage signals for modeling the circuit of Fig. 11 with the proposed DCWRNN, the conventional DRNN, and the DLSTM methods. Variations in rise/fall time are introduced incrementally in steps of 10ps, ranging from 100ps to 180ps, to generate square wave signals for the training procedure using the simulation tool. Additionally, for testing purposes, square wave signals with rise/fall times ranging from 105 to 165ps in 10ps increments and amplitudes of 1.12V, 1.15V, and 1.17V are generated. However, they are not utilized in the training process.

TABLE 3. Comparison between the DRNN, the DLSTM, the shallow CWRNN, and the proposed DCWRNN for modeling the circuit of Fig.11.

Model Type	No. of Parameters	Structure	Training error	Testing error
DRNN	6,481	(1, [30, 30, 30, 30], 1)	63e-04	162e-04
DLSTM	25,831	(1, [30, 30, 30, 30], 1)	9.66e-04	47e-04
Shallow CWRNN	5,041	(1, [5 × 20], 1)	9.63e-04	36e-04
Proposed DCWRNN	5,041	(1, [5 × 6], [5 × 6], [5 × 6], [5 × 6], 1)	9.54e-04	29e-04
Proposed HL-DCWRNN	5,016	(1, [6 × 6], [5 × 6], [4 × 7], [3 × 8], 1)	9.4e-04	24e-04

TABLE 4. Five models' CPU time speedup for Fig.11: DRNN, DLSTM, shallow CWRNN, proposed DCWRNN, and transistor-level.

Model Type	CPU Time(ms)	Speed-up
DRNN	268	11.51
DLSTM	542	5.69
Shallow CWRNN	163	18.93
Proposed DCWRNN	154	20.04
Proposed HL-DCWRNN	142	21.71
Transistor Level	3087	1

To model the interconnect circuit, the signal situated at the end of line 4 in Fig. 11 is used as the desired output signal. Table 3 compares the models' structure and accuracy obtained from the proposed DCWRNN, the conventional DRNN, and DLSTM methods. The table reveals that the proposed DCWRNN method has a suitable training error, while the traditional methods perform poorly. This is because the vanishing gradient problem is mitigated more efficiently by the proposed DCWRNN method than by the traditional DRNN method.

The proposed methods architecture for the circuit in Fig. 11 is presented by a DCWRNN model with a single input and output, and four hidden layers, each consisting of five modules with six neurons, represented as (1, [5 × 6], [5 × 6], [5×6], [5×6], 1). The HL-DCWRNN variant for the same circuit introduces heterogeneity within its hidden layers, featuring a variable number of modules and neurons across four layers, detailed as (1, [6×6], [5×6], [4×7], [3×8], 1).

Table 4 depicts the assessment time for the current transistor-level model, the proposed DCWRNN, the proposed HL-DCWRNN, the DRNN, and the DLSTM methods while simulating the interconnect circuit in Fig. 11. The DCWRNN-based models, compared to the transistor-level model, provide a shortened evaluation time, leading to a significantly accelerated model with an approximately 20-fold speed improvement. Therefore, the proposed method also outperforms the DRNN and DLSTM methods in speed because it has fewer recurrent connections.

In Fig. 13, the output from the transistor-level model for test data is shown with the predicted signals. These predictions stem from various methods, including the proposed DCWRNN, alongside conventional DRNN and DLSTM approaches, all applied to the interconnect circuit showcased in Fig. 11. This figure shows that the DCWRNN method can more accurately model the target waveform. In contrast, the DRNN and DLSTM methods fail to capture the circuit's dynamics due to a long data sequence and overfitting, respectively. Fig. 14 also confirms this by showing the absolute error of each method.

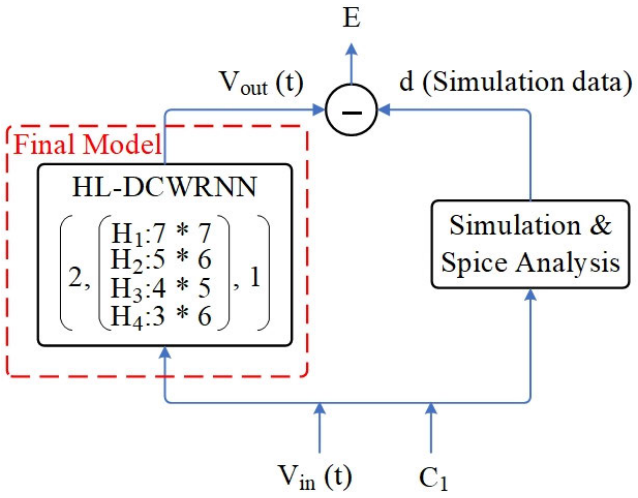

FIGURE 15. Structure of the Heterogeneous-layered DCWRNN (HL-DCWRNN) for modeling the circuit of Fig. 11.

The structural representation of the HL-DCWRNN modeling approach for the circuit of Fig. 11 is illustrated in the block diagram of Fig. 15. This method is characterized by a four-hidden-layer architecture, wherein each hidden layer accommodates a variable number of modules. The diagram depicts key components such as the input signal denoted as $V_{in}(t)$, the output signal represented as $V_{out}(t)$, the load capacitance identified as C_1, and the error value denoted as E. The latter signifies the average disparity between the outcomes derived from the HL-DCWRNN-based and transistor-level models. The block diagram is an illustrative overview of the HL-DCWRNN methodology applied to the circuit under consideration.

Leveraging the waveform at line 4 in the circuit of Fig. 11 as the target, the interconnect circuit is modeled using HL-DCWRNN and shallow CWRNN methods. Table 3 then contrasts their structure and accuracy, highlighting the strengths of the proposed HL-DCWRNN and DCWRNN compared to the shallow CWRNN. The table reveals that the proposed HL-DCWRNN method has a lower error and a higher efficiency than the proposed DCWRNN and the shallow CWRNN methods. This is because the HL-DCWRNN method can learn different frequencies from the data and more accurately model the complex features of nonlinear and high-frequency circuits.

Fig. 16 demonstrates the predictive power of HL-DCWRNN, DCWRNN, and shallow CWRNN for the circuit in Fig. 11 circuit against the actual transistor-level output for a specific test case. This figure shows that the

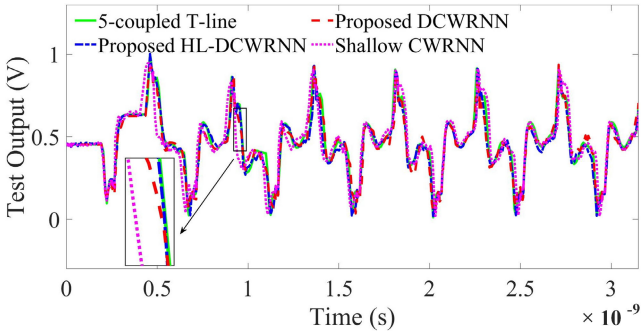

FIGURE 16. Comparison of the test data outputs generated by four different models: The shallow CWRNN-based, the proposed DCWRNN-based, the proposed HL-DCWRNN-based, and the transistor-level models. The models are applied to the circuit shown in Fig. 11.

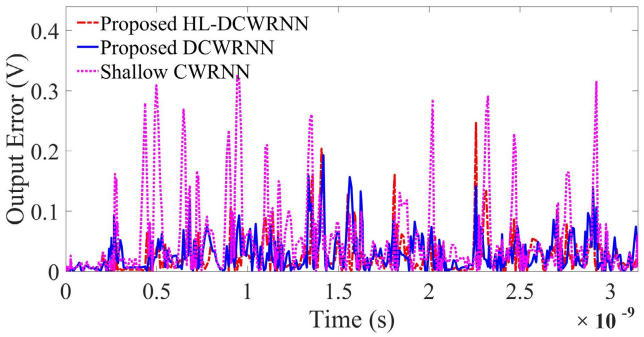

FIGURE 17. Comparison of the absolute test errors of three proposed models: The shallow CWRNN-based, the proposed DCWRNN-based, and the proposed HL-DCWRNN-based for the circuit shown in Fig. 11.

HL-DCWRNN method can more accurately model the target waveform than the DCWRNN method, while the shallow CWRNN method fails to capture some nonlinear dynamics of the circuit. Fig. 17 also confirms this by showing the absolute error of each method.

Table 4 showcases evaluation times for various models tackling the same circuit. Compared to the proposed DCWRNN and the shallow CWRNN methods, the proposed HL-DCWRNN achieves a 22x speedup over the traditional transistor-level model. Its fewer recurrent connections also give it an edge over DCWRNN.

V. CONCLUSION

This paper introduced an innovative macromodeling technique called deep clockwork recurrent neural networks (DCWRNN) that allows creating accurate and computationally light macromodels. The DCWRNN technique is based on a simple deep recurrent neural network (DRNN) structure that operates on multiple time scales. It can also capture complex dependencies without requiring many parameters, which reduces the computational cost. The DCWRNN technique is easy to understand and implement and allows for flexible architectural design. This can significantly speed up the training process of deep neural networks. Two test cases were developed in this paper. These circuits demonstrate that the DCWRNN technique outperforms conventional DRNN, DLSTM, and shallow CWRNN techniques regarding accuracy and speed. DCWRNN-based models exhibit substantially faster execution times compared to simulation tools. Another new modeling technique, called Heterogeneous-layered DCWRNN (HL-DCWRNN), is also proposed in this paper. To boost model accuracy compared to the original DCWRNN, this work introduces a novel architecture incorporating heterogeneous layers with different modules. The reported fast training and promising results demonstrate the suitability of these macromodeling techniques for complex nonlinear circuits.

REFERENCES

[1] N. Mirzaie and R. Rohrer, "A macromodeling approach for analog behavior of digital integrated circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 12, pp. 5025–5031, Dec. 2020.

[2] J. K. Eshraghian, Q. Lin, X. Wang, H. H. C. Iu, Q. Hu, and H. Tong, "A behavioral model of digital resistive switching for systems level DNN acceleration," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 67, no. 5, pp. 956–960, May 2020.

[3] Y. Fang, M. C. E. Yagoub, F. Wang, and Q.-J. Zhang, "A new macromodeling approach for nonlinear microwave circuits based on recurrent neural networks," *IEEE Trans. Microw. Theory Techn.*, vol. 48, no. 12, pp. 2335–2344, Dec. 2000.

[4] 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.

[5] H. Zhang, Y. Jing, and P. Zhou, "Machine learning-based device modeling and performance optimization for FinFETs," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 70, no. 4, pp. 1585–1589, Apr. 2023.

[6] M. Moradi A., S. A. Sadrossadat, and V. Derhami, "Long short-term memory neural networks for modeling nonlinear electronic components," *IEEE Trans. Compon., Packag., Manuf. Technol.*, vol. 11, no. 5, pp. 840–847, May 2021.

[7] J. Xu, M. C. E. Yagoub, R. Ding, and Q. J. Zhang, "Neural based dynamic modeling of nonlinear microwave circuits," in *IEEE MTT-S Int. Microw. Symp. Dig.*, vol. 2, Jun. 2002, pp. 1101–1104.

[8] Z. Naghibi, S. A. Sadrossadat, and S. Safari, "Dynamic behavioral modeling of nonlinear circuits using a novel recurrent neural network technique," *Int. J. Circuit Theory Appl.*, vol. 47, no. 7, pp. 1071–1085, Jul. 2019.

[9] Y. Cao and Q.-J. Zhang, "A new training approach for robust recurrent neural-network modeling of nonlinear circuits," *IEEE Trans. Microw. Theory Techn.*, vol. 57, no. 6, pp. 1539–1553, Jun. 2009.

[10] S. A. Sadrossadat and Z. Naghibi, "Parallelizing time-delay recurrent neural network modeling technique on multi-core architectures," in *Proc. 16th Int. Microsyst., Packag., Assem. Circuits Technol. Conf. (IMPACT)*, Dec. 2021, pp. 93–96.

[11] Y. Cao, R. Ding, and Q.-J. Zhang, "State-space dynamic neural network technique for high-speed IC applications: Modeling and stability analysis," *IEEE Trans. Microw. Theory Techn.*, vol. 54, no. 6, pp. 2398–2409, Jun. 2006.

[12] S. A. Sadrossadat, P. Gunupudi, and Q.-J. Zhang, "Nonlinear electronic/photonic component modeling using adjoint state-space dynamic neural network technique," *IEEE Trans. Compon., Packag., Manuf. Technol.*, vol. 5, no. 11, pp. 1679–1693, Nov. 2015.

[13] A. M. Schäfer and H. G. Zimmermann, "Recurrent neural networks are universal approximators," in *Proc. Int. Conf. Artif. Neural Netw.*, Athens, Greece. Berlin, Heidelberg: Springer, Sep. 2006, pp. 632–640.

[14] M. Noohi, A. Mirvakili, and S. A. Sadrossadat, "Modeling and implementation of nonlinear boost converter using local feedback deep recurrent neural network for voltage balancing in energy harvesting applications," *Int. J. Circuit Theory Appl.*, vol. 49, no. 12, pp. 4231–4247, Dec. 2021.

[15] M. Magerl, V. Ceperic, and A. Baric, "Echo state networks for black-box modeling of integrated circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 35, no. 8, pp. 1309–1317, Aug. 2016.

[16] A. Faraji, M. Noohi, S. A. Sadrossadat, A. Mirvakili, W. Na, and F. Feng, "Batch-normalized deep recurrent neural network for high-speed nonlinear circuit macromodeling," *IEEE Trans. Microw. Theory Techn.*, vol. 70, no. 11, pp. 4857–4868, Nov. 2022.

[17] H. M. R. Ugalde, J.-C. Carmona, J. Reyes-Reyes, V. M. Alvarado, and J. Mantilla, "Computational cost improvement of neural network models in black box nonlinear system identification," *Neurocomputing*, vol. 166, pp. 96–108, Oct. 2015.
[18] M. Noohi, A. Faraji, S. A. Sadrossadat, A. Mirvakili, and A. Moftakharzadeh, "Modeling and implementation of a novel active voltage balancing circuit using deep recurrent neural network with dropout regularization," *Int. J. Circuit Theory Appl.*, vol. 51, no. 5, pp. 2351–2374, May 2023.
[19] A. Faraji, S. A. Sadrossadat, M. Yazdian-Dehkordi, M. Nabavi, and Y. Savaria, "A hybrid approach based on recurrent neural network for macromodeling of nonlinear electronic circuits," *IEEE Access*, vol. 10, pp. 127996–128006, 2022.
[20] A. Faraji, S. A. Sadrossadat, W. Na, F. Feng, and Q.-J. Zhang, "A new macromodeling method based on deep gated recurrent unit regularized with Gaussian dropout for nonlinear circuits," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 70, no. 7, pp. 2904–2915, Apr. 2023.
[21] J. Koutnik, K. Greff, and F. Gomez, "A clockwork RNN," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1863–1871.
[22] F. Charoosaei, A. Faraji, S. A. Sadrossadat, A. Mirvakili, W. Na, F. Feng, and Q.-J. Zhang, "High-speed nonlinear circuit macromodeling using hybrid-module clockwork recurrent neural network," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 71, no. 2, pp. 767–780, Feb. 2024.
[23] S. Gu, S. Levine, I. Sutskever, and A. Mnih, "MuProp: Unbiased back-propagation for stochastic neural networks," 2015, *arXiv:1511.05176*.
[24] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[25] D. Georgiadou, V. Diakoloukas, V. Tsiaras, and V. Digalakis, "ClockWork-RNN based architectures for slot filling," in *Proc. Interspeech*, Aug. 2017, pp. 2481–2485.
[26] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," 2015, *arXiv:1511.06732*.
[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
[28] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990.
[29] S. Chang, Y. Zhang, W. Han, M. Yu, X. Guo, W. Tan, X. Cui, M. Witbrock, M. A. Hasegawa-Johnson, and T. S. Huang, "Dilated recurrent neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 77–87. [Online]. Available: https://experts.illinois.edu/en/publications/dilated-recurrent-neural-networks
[30] F. M. Bianchi, M. Kampffmeyer, E. Maiorino, and R. Jenssen, "Temporal overdrive recurrent neural network," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 4275–4282.

MAEDEH AZODI received the B.Eng. degree in computer engineering from Sirjan University of Technology, Sirjan, Iran, in 2019, and the master's degree in artificial intelligence from the Department of Computer Engineering, Yazd University, Yazd, Iran. Her research interests include computer-aided design, deep learning, neural networks, and their applications in modeling and simulation of circuits and components.

SAYED ALIREZA SADROSSADAT (Senior Member, IEEE) received the B.Sc. degree from the University of Tehran, Tehran, Iran, in 2007, the master's degree from the University of Waterloo, Waterloo, ON, Canada, in 2010, and the Ph.D. degree from Carleton University, Ottawa, ON, Canada, in 2015. He is currently the Chair of the Artificial Intelligence Group, Computer Engineering Department, Yazd University, Yazd, Iran. His current research interests include optimization and neural network-based modeling of linear/nonlinear components and circuits, computer-aided design, deep learning, very large-scale integration design, probabilistic design, and yield maximization. He has been a Technical Reviewer for several IEEE/IET journals, such as IEEE TRANSACTIONS ON NEURAL NETWORK AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON MICROWAVE THEORY AND TECHNIQUE, IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION SYSTEMS, *IET Microwave and Propagation*, *IET Electronic Letters*, and IEEE CANADIAN JOURNAL OF ELECTRICAL AND COMPUTER ENGINEERING.

YVON SAVARIA (Life Fellow, IEEE) received the B.Ing. and M.Sc.A. degrees in electrical engineering from École Polytechnique Montréal, Canada, in 1980 and 1982, respectively, and the Ph.D. degree in electrical engineering from McGill University, in 1985. Since 1985, he has been with Polytechnique Montréal, where he is currently a Professor with the Department of Electrical Engineering. He is currently involved in several projects related to embedded systems in aircraft, wireless sensor networks, virtual networks, software-defined networks, machine learning (ML), embedded ML, computational efficiency, and application-specific architecture design. He has carried out work in several areas related to microelectronic circuits and microsystems, such as testing, verification, validation, clocking methods, defect and fault tolerance, effects of radiation on electronics, high-speed interconnects, and circuit design techniques, CAD methods, reconfigurable computing and applications of microelectronics to telecommunications, aerospace, image processing, video processing, radar signal processing, and the acceleration of digital signal processing. He holds 16 patents, has published 211 journal articles and 495 conference papers, and was the Thesis Advisor of 190 graduate students who completed their studies. He is a fellow of the Canadian Academy of Engineering and a member of the Ordre des Ingénieurs du Québec (OIQ). In 2001, he was awarded a Tier 1 Canada Research Chair (www.chairs.gc.ca) on the design and architecture of advanced microelectronic systems in June 2015. He also received the Synergy Award from the Natural Sciences and Engineering Research Council of Canada in 2006. Since June 2019, he has been the NSERC-Kaloom-Intel-Noviflow (KIN) as the Chair Professor. He was the Program Co-Chairperson of NEWCAS'2018 and the General Chairperson of NEWCAS'2020. He has been working as a Consultant or was sponsored for carrying out research with Bombardier, Buspass, CNRC, Design Workshop, Dolphin, DREO, Ericsson, Genesis, Gennum, Huawei, Hyperchip, Intel, ISR, Kaloom, LTRIM, Miranda, MiroTech, Nortel, Octasic, PMC-Sierra, Space Codesign, Technocap, Thales, Tundra, and Wavelite. He is the Co-Director of the Regroupement Stratégique en Microélectronique du Québec (RESMIQ).