Received 8 July 2024; accepted 16 August 2024. Date of publication 21 August 2024; date of current version 28 August 2024.
The review of this article was arranged by Editor L. Wei.

Digital Object Identifier 10.1109/JEDS.2024.3447032

Device Modeling Based on Cost-Sensitive Densely Connected Deep Neural Networks

XIAOYING TANG 1,2,3, ZHIQIANG LI 1,2,3 (Member, IEEE), LANG ZENG 4,5 (Senior Member, IEEE), HONGWEI ZHOU 4,5, XIAOXU CHENG 6, AND ZHENJIE YAO 1,2,3

1 Key Laboratory of Fabrication Technologies for Integrated Circuits, Chinese Academy of Sciences, Beijing 100045, China
2 Institute of Microelectronics Chinese Academy of Sciences, Beijing 100029, China
3 School of Integrated Circuits, University of Chinese Academy of Sciences, Beijing 100049, China
4 MIIT Key Laboratory of Spintronics, School of Integrated Circuit Science and Engineering, Fert Beijing Institute, Beihang University, Beijing 100191, China
5 National Key Laboratory of Spintronics, Hangzhou International Innovation Institute, Beihang University, Hangzhou 311115, China
6 Primarius Technologies Company Ltd., Shanghai 201306, China

CORRESPONDING AUTHOR: Z. YAO (e-mail: yaozhenjie@ime.ac.cn)

This work was supported in part by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDA0330401; in part by the CAS Youth Interdisciplinary Team under Grant JCTD-2022-07; and in part by the National Natural Science Foundation of China under Grant 62171009, Grant T2293703, and Grant T2293700.
This paper was produced by the IEEE Publication Technology Group. They are in Piscataway, NJ.

ABSTRACT Engineers used TCAD tools for semiconductor devices modeling. However, it is computationally expensive and time-consuming for advanced devices with smaller dimensions. Therefore, this work proposes a machine learning-based device modeling algorithm to capture the complex nonlinear relationship between parameters and electrical characteristics of gate-all-around (GAA) nanowire field-effect transistors (NWFETs) from technology computer-aided design (TCAD) simulation results. This method utilizes a densely connected deep neural networks (DenseDNN), which establishes direct connections between layers in the neural networks, provides stronger feature extraction and information transmission capabilities. By incorporating cost-sensitive learning methods, the proposed model focus more on the critical data that determines device characteristics, leading to accurate prediction of key device characteristics under various parameters. Experimental results on a test dataset of 116 NWFETs demonstrate the effectiveness of this method. The DenseDNN model with cost-sensitive learning exhibits better performance than traditional deep neural networks (DNN) with various widths and depths, with a prediction error below 1.62%. Moreover, compared to TCAD simulation results, the model can speedup $10^6\times$.

INDEX TERMS Machine learning, device modeling, technology computer-aided design (TCAD) simulation, cost-sensitive densely connected DNN.

I. INTRODUCTION

IN THE past decades, CMOS has consistently followed Moore's Law [1], with the number of transistors per unit area doubling every 18-24 months. CMOS devices continue to be miniaturized, from planar to FinFETs. However, as the transistor sizes continue to shrink into the nanometer scale, traditional device theories and process technologies are confronted with challenges arising from physical limitations. The electrostatic performance of FinFETs becomes worse, the short channel effect (SCE) is aggravated, and the issues of parasitic capacitance and resistance become increasingly severe, which leads to a significant degradation in device performance. The structure of FinFETs is no longer able to meet the requirements. Therefore, gate-all-around (GAA) devices have been proposed, including NWFETs and nanosheet field-effect transistors (NSFETs) [2], [3], [4], [5], [6] to address these challenges. In the GAA structure, the channel is fully enveloped by the gate, enhancing the gate control over the channel and providing better electrostatic characteristics than FinFETs. In order to study GAA devices more effectively, this work conducts experiments using GAA NWFETs to learn the complex nonlinear relationship between device parameters and electrical characteristics.

Generally, engineers rely on TCAD tools for simulating and modeling semiconductor devices to predict and tackle complex physical phenomena in the devices [7]. Nonetheless, TCAD-based device modeling encounters two major challenges: 1) physics- based model equations require a lot of

© 2024 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License.
For more information, see https://creativecommons.org/licenses/by/4.0/

time and professional knowledge [8]; 2) TCAD tools have some limitations when dealing with devices with relatively small feature sizes, and have high computational costs and long simulation time.

Machine learning-based model has emerged as an alternative solution to address these challenges. Artificial neural networks (ANN) were first proposed by Litovski in the early 1990s for modeling transistors [9]. The semiconductor device modeling based on ANN can easily capture the nonlinear electrical characteristic of the device by adding a nonlinear activation function [10], and predict the key device characteristics, such as current-voltage and capacitance-voltage relationships [11], [12], [13], [14], work functions [15], threshold voltage and subthreshold slope, and on-state current [16], [17], [18], [19]. Furthermore, ANN models can be used for compact modeling [20], [21], [22], [23], parameter extraction [24] and design space exploration optimization [25]. These works demonstrate the effectiveness of ANN in device modeling. Hence, further exploration of device modeling methods based on ANN is undoubtedly worthwhile.

When using ANN for device modeling, it is crucial to perform appropriate preprocessing of the data. Especially for current-voltage curve data, normalization is often done using a logarithmic transformation to eliminate the impact of data at different scales on model training. On this basis, combined with cost-sensitive learning methods [26], the ability of the model to handle complex and critical tasks can be further enhanced. By assigning different cost weights to different samples [27], [28], the model can more accurately capture key features in the curves, thereby improving the accuracy of predictions.

In this work, inspired by the densely connected neural networks [29], [30], an improvement to the traditional deep neural networks architecture is proposed, known as the DenseDNN model, which establishes dense connections between layers. By leveraging this dense direct connectivity, the performance and training effectiveness of the network are improved. We trained the DenseDNN using data generated by TCAD and further optimized the model with cost-sensitive learning methods, successfully achieving high-precision predictions of key device characteristics in a relatively short time. The remaining sections of this paper are organized as follows. Section II introduces the TCAD simulation method and data generation. In Section III, a detailed description of the proposed cost-sensitive DenseDNN is presented. Section IV discusses the performance of the cost-sensitive DenseDNN, and presents the comparison results between cost-sensitive DenseDNN and DNNs with different widths and depths. The conclusions are drawn in Section V.

II. TCAD SIMULATION AND DATA GENERATION

In this work, we focus on GAAs with nanowire in circular and triangular cross-sectional shape, which is shown in Fig. 1. We employed our in-house TCAD tool to generate the electrical characteristics of such GAA devices. Poisson equation, Schrödinger equation and drift-diffusion transport equation are self-consistent solved in TCAD within the framework of the Mode Space method [31], [32], [33], which can capture the quantum confinement phenomenon in this kind of quasi one-dimensional device. By adjusting the gate work function, oxide thickness, doping concentration and saturation velocity of carriers, the TCAD simulation can match the experimental results of a GAA nanowire device with an effective diameter of 12.8nm and a channel length of 22nm [34], as shown in Fig. 2.

To train the neural networks, the NWFETs were divided into training set, validation set, and test set based on their different cross-sectional shapes. The training set consisted of 242 NWFETs with circular cross-sections, resulting in a total of 54450 (=242 × 225) simulated data samples. The validation set comprises 27 NWFETs with circular cross-sections, generating a total of 6,075 (=27 × 225) simulated data samples. The test set included 116 NWFETs with equilateral triangular cross-sections, generating 26,100

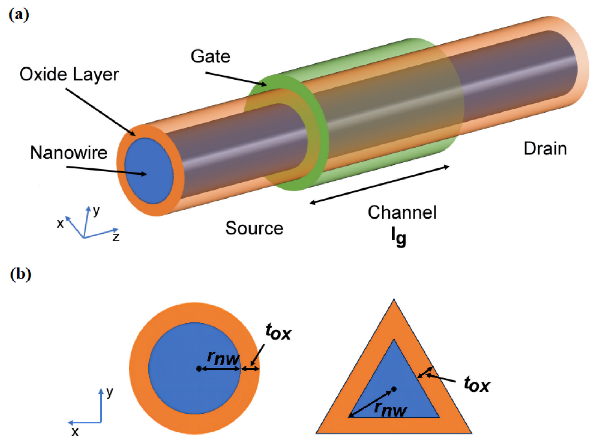

FIGURE 1. (a) The schematic structure of GAA nanowire transistor. (b) The cross-sectional view of two types of nanowires including the circle and the triangle.

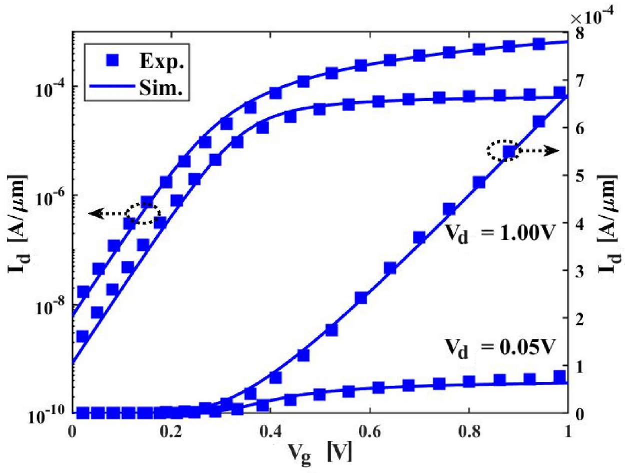

FIGURE 2. The calibration of the transfer characteristics of the GAA nanowire transistor with the experimental data [34].

TABLE 1. Device Parameters for NWFETs.

Parameter	Range (min,max)	Step
r_{nw}[nm]	(2,5)	0.5
l_g[nm]	(10,20)	1
t_{ox}[nm]	(0.5,1.5)	0.25
V_g[V]	(0,0.7)	0.05
V_d[V]	(0,0.7)	0.05

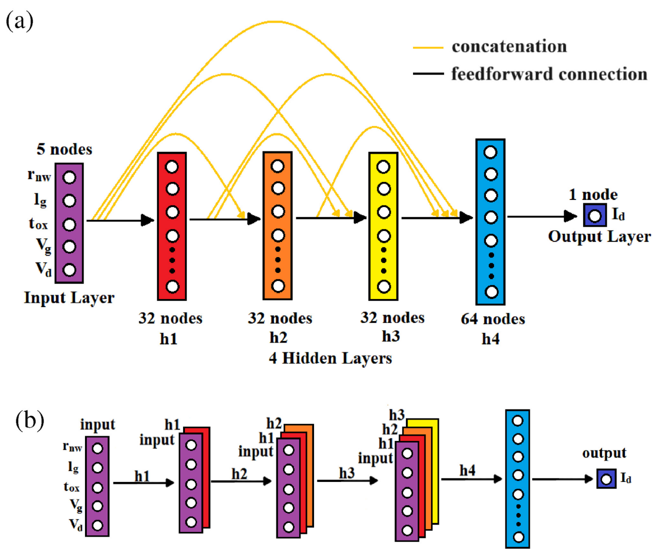

FIGURE 3. (a) A schematic view showing the structure of DenseDNN model and (b) forward process of DenseDNN.

(=116 × 225) simulated samples. It should be noted that for the equilateral triangular cross-section, the parameter 'r_{nw}' refers to the distance from the centroid to the vertex.

The TCAD tool was utilized to simulate 385 NWFETs with varying parameters in order to obtain the drain current (I_d) of each device under 225 different bias voltages. The parameter settings used are specified in Table 1, which include the nanowire radius (r_{nw}), channel length (l_g), gate oxide thickness (t_{ox}), and bias voltages (V_g, V_d).

In order to enhance the training of the neural network model, data preprocessing is performed. The input data (r_{nw}, l_g, t_{ox}, V_g, V_d) is standardized before being fed into the DenseDNN network, while the output data (I_d), due to its wide distribution span, undergoes logarithmic transformation. The model learns from the preprocessed data and finally transforms the output data back to the original range through the inverse logarithm function.

III. METHODS

A. DENSEDNN ARCHITECTURE

Fig. 3 (a) shows the structure of the DenseDNN model. The DenseDNN has four hidden layers (h1, h2, h3, h4). In h1, h2, and h3, each layer contains 32 neurons (the selection of the number of neurons is described in detail in Section IV) and each hidden layer is composed of dense connections and tanh activation functions. h4 is a fully connected layer with 64 neurons with sigmoid activation function. The model is implemented using Pytorch. In this study, the performance of training the neural networks is evaluated using the root mean square error (RMSE) as the loss function, defined as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2} \tag{1}$$

where $\hat{y}_i$ is the predicted drain current by the DenseDNN, and y_i is the actual drain current. The loss function is optimized using the Adam optimizer with an initial learning rate of 1.0×10^{-3}.

DenseDNN adopts the dense connection to establish a connection relationship between all the previous hidden layers and subsequent hidden layers. Specifically, each layer concatenates all the neurons from the previous layers and uses them as input for the subsequent layer. To better understand the dense connectivity of DenseDNN, the forward process is illustrated in Fig. 3 (b). In DenseDNN, the input of each hidden layer comes from the outputs of all previous layers. For example, in Fig. 3 (b), the input to h3 consists of the output from h2 as well as the outputs from the input layer and h1. Subsequently, the input layer, h1, h2, and h3 are concatenated as the input for h4. Similarly, direct connections are established from h4 to all previous layers. Finally, h4 and the output layer resemble fully connected layers in conventional DNN.

DenseDNN allows the features from the previous layer to be directly transmitted to all the subsequent layers, leading to a highly interactive and densely connected network structure. This is in contrast to the traditional DNN architecture, where each layer is only connected to the adjacent layer. This dense connectivity design of DenseDNN enables each layer to directly receive the information from all the previous layers, enhancing the information flow and gradient propagation, mitigating the issue of gradient vanishing to some extent. It also enhances the backward propagation of gradients, making the network easier to train, thus improving the accuracy and stability of the network.

In general, DenseDNN improves the performance and training effectiveness by introducing dense connectivity, which provides stronger feature extraction and information propagation. This makes DenseDNN an effective choice for handling complex tasks and highly nonlinear data relationships.

B. COST-SENSITIVE LEARNING

During the processing of training data, we performed a logarithmic transformation on the drain current data. This transformation could reduce the accuracy of predictions for the drain current in the mid-V_g and high-V_g regions. To counteract this effect and enhance the model's ability to learn key device characteristics, we employed a cost-sensitive

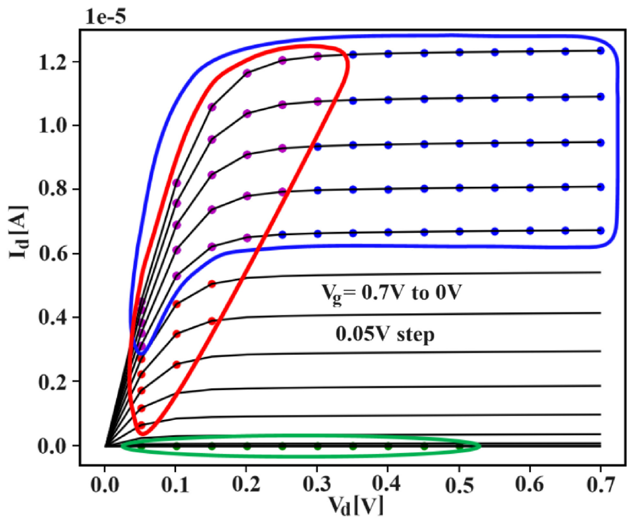

FIGURE 4. Training set samples using the sensitive learning method. The part circled in blue represents samples from the mid-V_g and high-V_g regions ($V_g \geq 0.5$V), the part circled in red represents samples from the linear region ($V_g > V_{th}$ and $V_d < V_g - V_{th}$), the green part represents samples related to the off region (V_g=0V, $V_d \leq 0.5$V), and the purple circles indicate samples from the overlapping areas.

learning approach. This method, by assigning different weights to each sample, ensures that the model pays more attention to those samples considered more important or costly during the training process, thereby optimizing the learning process.

As shown in Fig. 4, we selected all samples from the mid-V_g and high-V_g regions of the training set, thereby enhancing the prediction ability of the model for the drain current in that region. Furthermore, to accurately capture the critical device characteristics of threshold voltage (V_{th}) and off current (I_{off}), we selected relevant samples from the linear region and the off region. For the selected samples, non-overlapping regions are assigned the same weight, while weights are accumulated for overlapping regions. For samples not selected, the weight is defaulted to 1.

Next, based on the allocation of sample weights, we modified the loss function used during the training process:

$$RMSE\_sampleweight = \sqrt{\frac{1}{N}\sum_{i=1}^{N} w_i \times \left(y_i - \hat{y}_i\right)^2} \qquad (2)$$

where w_i represents the weight of each sample. This adjustment ensures that samples with higher weights have a greater impact on the total loss calculation, thereby encouraging the model to focus more on these important samples.

IV. RESULT AND DISCUSSIONS

A. EXPERIMENTAL SETTING

Regarding the depth setting of the network, we referred to the cited literature [30].

Then, to determine the optimal architecture of the proposed DenseDNN model, different hidden layer widths (number of neurons) were tested as shown in Table 2. The test results are evaluated by RMSE on the test set. The smaller the RMSE value, the smaller the gap between the predicted results of the model and the real data, and the better performance the model gets. It is observed that the RMSE is minimized when each hidden layer contains 32 neurons. Therefore, 32 was selected as the number of neurons for each hidden layer.

TABLE 2. Comparison of different widths.

Width	16	32	64	128	256	512
RMSE	0.0027	0.0021	0.0041	0.0056	0.0100	0.0101

TABLE 3. Comparison of different learning rate.

learning rate	0.0001	0.001	0.01
RMSE	0.0038	0.0030	0.0043

TABLE 4. Average, 3-sigma and Max Of prediction error.

Parameter	μ	Error 3σ	Max
I_d [μA]	- 0.0113	0.1954	1.2520

Additionally, selecting an appropriate initial learning rate is a crucial step as it directly affects the efficiency of model training and its final performance. Therefore, different initial learning rates were tested, as shown in Table 3. The test results indicate that the model's error is minimized when the initial learning rate is set to 1×10^{-3}. Consequently, the initial learning rate of the Adam optimizer was set to 1×10^{-3}.

B. EFFECTIVENESS OF DENSEDNN

Based on the mentioned structure, we utilized the trained DenseDNN model to predict the drain current on the test set in order to validate the effectiveness of the model. Comparison between the predicted drain current and the actual drain current is depicted in Fig. 5 (a), where only limited fitting results are shown. The results indicate that the trained DenseDNN can fit the key characteristic curves with high precision. Fig. 5 (b) illustrates the distribution of prediction errors in the DenseDNN model. The majority of errors are concentrated around zero and decrease rapidly, with few occurrences of larger errors. In addition, the mean error, 3σ error, and maximum error are presented in Table 4. The mean error of the prediction is close to zero, with 3σ error of 0.1954μA and maximum error of 1.252μA. These small error values demonstrate the high accuracy of the DenseDNN model in predicting drain current, validating its effectiveness in capturing the current-voltage relationship.

C. COMPARISON OF PERFORMANCE WITH DNN

To further evaluate the performance of the DenseDNN model, the DNN model was used as a reference to assess the predictive capabilities of the two trained models for key

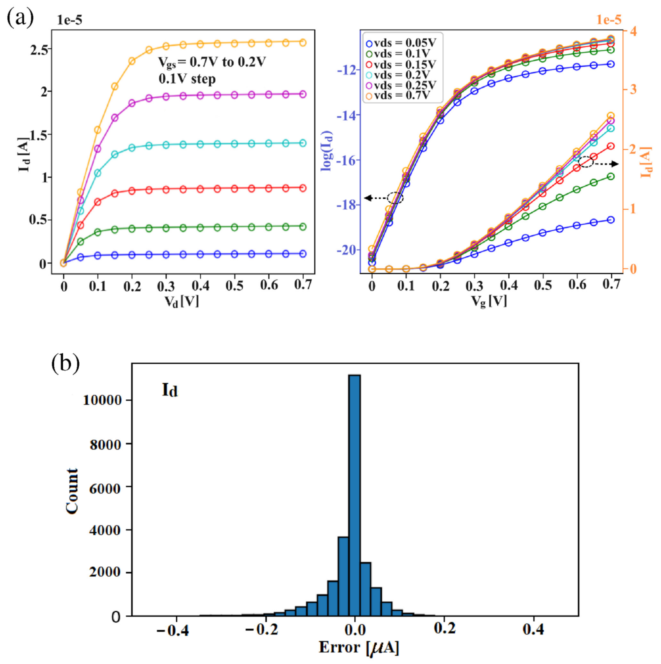

FIGURE 5. **(a) Predicted (lines) and actual (dots) drain current versus gate voltage with r_{nw}=3nm, l_g=12nm, t_{ox}=0.5nm on the test set. (b) Prediction error distribution in I_d.**

TABLE 5. **Structural parameters of all models.**

Parameter	DNN	DNNw	DNNd	DenseDNN
width	32	55	32	32
depth	4	4	9	4
total number	4481	10139	9761	10241

device characteristics. In the experiment, we configured three DNN models with varying hidden layer widths and depths. DNN has same width and depth with DenseDNN without cross-layer connections. DNNw and DNNd are designed to ensure that the number of parameters is kept as consistent as possible with DenseDNN by changing width or depth. The width, depth and the total number of parameters are presented in Table 5. The width of the last hidden layer is fixed to 64 for all models. Fig. 6 shows the loss curves of each model on the validation set during the training process. From the figure, we find that simply increasing parameters by adding width or depth does not necessarily result in better fitting and can lead to various issues like vanishing gradients or over fitting.

As observed from the figure, the loss of the DenseDNN model rapidly decreases within the first 100 epochs, then gradually stabilizes, reaching its lowest point at around 500 epochs. The losses of DNN, DNNw and DNNd reach their minima at approximately 650, 1250 and 1200 epochs, with no significant improvement afterward. Therefore, setting the training epochs of each model to 1200 is more appropriate. In the following experiment, we trained various models for

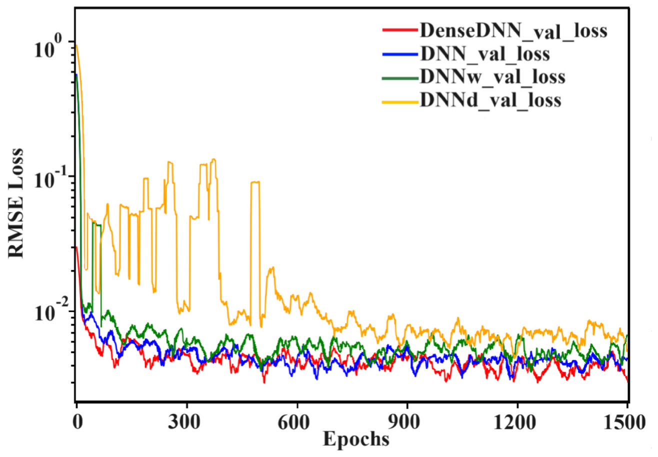

FIGURE 6. **RMSE on the validation set in the DenseDNN and DNN training.**

1200 epochs in a CPU environment and recorded the training time. Specifically, the training times of DNN, DNNw, DNNd and DenseDNN were 53.04min, 56.48min, 84.42min, and 61.81min, respectively.

Saturation drain current (I_{dsat}), off-state current (I_{off}), and threshold voltage (V_{th}) are important indicators that affect device performance. Therefore, we will not only compare the fitting capabilities of all models for drain current, but also pay special attention to their ability to predict these key indicators. We evaluated the trained DenseDNN and DNN models on the same test set and extracted the values of key metrics from the predicted I-V curves using the following calculation method. V_{th} is defined as the gate voltage corresponding to the maximum transconductance (g_m) when the transistor operates in the linear region. The saturation drain current is drain current under V_d=V_{dd}=0.5V. The off-state current is drain current under V_g=0V and V_d=V_{dd}=0.5V. Table 6 presents the prediction errors of the four models for these critical device characteristics. The prediction errors are defined by the Mean Absolute Error (MAE) [see (3)] and Mean Absolute Percentage Error (MAPE) [see (4)]. To ensure the reliability of the model predictions, all models were tested 10 times, and the mean of the 10 test results was taken as the final result. It is evident from the table that, in terms of MAE, DenseDNN model performs best in the prediction error of V_{th}. For MAPE, the prediction error of the DenseDNN model does not exceed 2.05%, indicating a high prediction accuracy. Specifically, the errors for V_{th}, I_{dsat}, I_{off} and I_d are 2.00%, 0.95%, 2.05%, and 1.04%, respectively. Compared to all DNN models, the DenseDNN model exhibits lower prediction errors and better predictive capability for key device characteristics.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right| \tag{3}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{4}$$

TABLE 6. Average prediction error.

Parameter	MAE				MAPE			
	DNN	DNNw	DNNd	DenseDNN	DNN	DNNw	DNNd	DenseDNN
I_d [μA]	0.0704	0.0656	0.2089	0.0686	1.23%	1.25%	3.69%	1.04%
I_{dsat} [μA]	0.1062	0.1033	0.2905	0.1210	1.04%	0.94%	2.84%	0.95%
I_{off} [μA]	0.0012	0.0012	0.0012	0.0015	2.55%	2.76%	6.58%	2.05%
V_{th} [mV]	3.5646	5.0146	13.2100	2.5269	2.83%	3.98%	10.21%	2.00%

TABLE 7. Average prediction error on sample weight setting.

Parameter	MAE						MAPE					
	1.5	3	5	10	15	20	1.5	3	5	10	15	20
I_d [μA]	0.0491	0.0571	**0.0405**	0.0527	0.0642	0.0422	0.86%	0.91%	0.92%	0.90%	1.22%	**0.82%**
I_{dsat} [μA]	0.0624	0.0652	0.0556	**0.0520**	0.0802	0.0616	0.63%	0.63%	**0.48%**	0.57%	0.92%	0.56%
I_{off} [μA]	0.0011	0.0016	0.0011	0.0011	0.0013	**0.0008**	**1.64%**	1.96%	1.78%	2.13%	2.51%	2.33%
V_{th} [mV]	2.1566	1.9756	**1.7438**	1.8935	2.8524	2.3726	1.70%	1.61%	**1.39%**	1.54%	2.37%	1.90%

TABLE 8. Average prediction error of the optimal model.

Parameter	MAE				MAPE			
	DNN	DenseDNN	DNNcs	DenseDNNcs	DNN	DenseDNN	DNNcs	DenseDNNcs
I_d [μA]	0.0704	0.0686	0.0589	**0.0418**	1.23%	1.04%	1.07%	**0.82%**
I_{dsat} [μA]	0.1062	0.1210	0.0764	**0.0571**	1.04%	0.95%	0.73%	**0.54%**
I_{off} [μA]	0.0012	0.0015	0.0012	**0.0011**	2.55%	2.05%	2.49%	**1.62%**
V_{th} [mV]	3.5646	2.5269	2.0975	**1.8083**	2.83%	2.00%	1.69%	**1.44%**

D. COMBINED WITH COST-SENSITIVE LEARNING

We applied the cost-sensitive learning to the DenseDNN model and named it DenseDNNcs. Based on the sample selection method in Section III.B, experiments were conducted with different weight values to choose the most suitable sample weight. The performance of the model under different weight values was evaluated on the test set to measure its performance. Table 7 shows the average prediction errors of the DenseDNNcs model at various weights. Comparing the MAE error results, when the weight value is set to 5, the model has the smallest errors in predicting I_d and V_{th}. When the weight is set to 10, it is most accurate in predicting I_{dsat}. When the weight value is set to 20, the model performs best in predicting I_{off}, but with a larger error in predicting V_{th}. For MAPE, when the weight value is set to 5, the model demonstrates higher prediction accuracy, with generally smaller prediction errors than other weight settings. Based on a comprehensive evaluation of MAE and MAPE error results, 5 is determined to be the optimal weight value for the samples.

After determining the optimal weight values, the DenseDNNcs model was trained. To evaluate the performance of the DenseDNNcs model in predicting key device characteristics, Table 8 compares the prediction results of the DNN model, DenseDNN model, and the DNN model combined with cost-sensitive learning methods (DNNcs). Specifically, to ensure fair comparison, the DNNcs model utilized the same sample selection method and weight settings as the DenseDNNcs model. The results indicate that the models incorporating cost-sensitive learning methods, namely DNNcs and DenseDNNcs models, significantly improve accuracy in predicting key device characteristics. Moreover, the DenseDNNcs model exhibits the lowest prediction errors among all models. Specifically, the maximum prediction error of MAPE is reduced to below 1.62%, with MAE prediction errors for different leakage currents controlled within 0.0571μA and a significant decrease in MAE prediction error for Vth, reduced to 1.8083mV. These results demonstrate the effectiveness of cost-sensitive learning methods in enhancing model prediction capabilities.

To further validate the performance of DenseDNNcs, the DNNcs model was selected for comparison. Fig. 7 displays the predicted results of both models for I_d, I_{dsat}, I_{off}. The black line in figure represents the ideal target where the

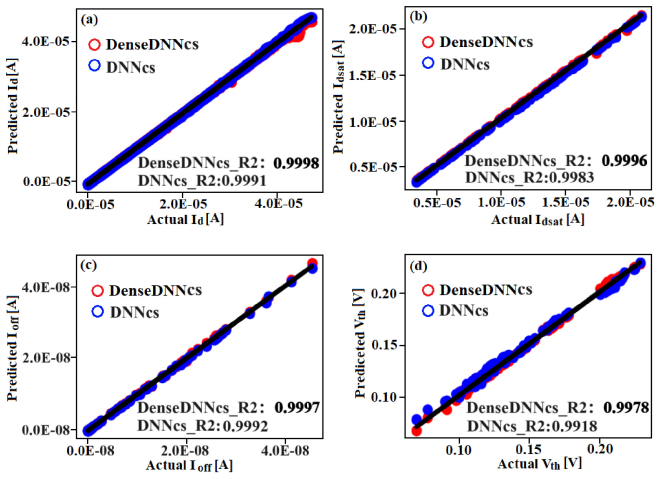

FIGURE 7. The comparison of the predicted I_d, I_{dsat}, I_{off}, V_{th} by DenseDNNcs and DNNcs on the same test set are shown in the (a)-(d). The blue circles represent the prediction result of DNNcs, the red circles represent the prediction result of DenseDNNcs, and the black line represents the ideal target where the ML predictions match the TCAD results.

TABLE 9. The comparison of model training and testing time.

Models	Training time	Testing time	Speedup	
			with train time	without train time
TCAD [h]	-	871.31	-	-
DNN [h]	0.88	1.74e-4	9.90×10^2	5.01×10^6
DenseDNN [h]	1.03	2.03e-4	8.46×10^2	4.29×10^6
DenseDNNcs [h]	1.34	4.51e-4	6.65×10^2	1.93×10^6

predicted values match the actual values. Considering that most values of I_{off} are concentrated below 5e-8, some values higher than 5e-8 were removed in Fig. 7 (c) to allow a clearer comparison of the prediction results. By comparing the distribution of prediction results around the target line for both models, the prediction results of DenseDNNcs are closer to the target line, indicating a stronger fitting capability. To evaluate the prediction accuracy of the two models, the value of R^2 was calculated, and R^2 is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y}_i)^2} \tag{5}$$

where $\bar{y}_i$ is the mean value of the actual drain current in the test set. The calculation results also presented in Fig. 7. As shown in the figure, R^2 obtained from the predictions of the DenseDNNcs model are greater than 0.99 and higher than those of the DNN model, indicating superior accuracy compared to DNN. It is worth noting that the R^2 of I_{ds}, I_{dsat} and I_{off} predicted by DenseDNNcs are almost 1, further proving that the use of cost-sensitive learning methods can predict key device characteristics more accurately.

E. COMPARISON OF SIMULATION TIME WITH TCAD

Furthermore, the DenseDNNcs model can predict key device characteristics faster with relatively small errors compared to TCAD simulations. As shown in Table IX, based on the trained DenseDNNcs model, it takes about 1.6 seconds to predict the key characteristics of 116 NWFETs, whereas TCAD simulations may require approximately 871.3 hours. While the complexity of the DenseDNNcs model leads to slightly longer prediction times compared to the DNN model, the results from the aforementioned experiments demonstrate that DenseDNNcs exhibits higher accuracy and superior fitting capabilities when predicting key device characteristics. It is important to note that the TCAD simulation times for different devices varied in this experiment. The minimum time to simulate one device is 1 hour, while the maximum time reaches 28.69 hours.

V. CONCLUSION

In This paper, we propose a cost-sensitive DenseDNN model to learn the compact current-voltage relationship of GAA NWFETs from TCAD simulation data. The proposed model leverages the information exchange among all layers through direct cross-layer connections, and focus more on deterministic samples due to cost-sensitive learning. Experimental results demonstrate that the proposed model can better fit the TCAD simulation data with the prediction errors for key device characteristics no more than 1.62%, demonstrating good predictive capability. Furthermore, compared to TCAD simulation, the proposed model achieves a significant improvement in prediction efficiency, which speeds up $10^6\times$ while ensuring accuracy.

In future research, by analyzing the impact of samples from different working regions of the device on the overall performance of the model and accordingly adjusting the sample weights, the learning process of the model regarding the relationships of device characteristics can be effectively optimized. This approach is expected to further improve upon the existing performance of the model.

REFERENCES

[1] G. E. Moore, “Cramming more components onto integrated circuits,” *Proc. IEEE*, vol. 86, no. 1, pp. 82–85, Jan. 1998, doi: 10.1109/JPROC.1998.658762.
[2] M. Li et al., “Sub-10 nm gate-all-around CMOS nanowire transistors on bulk Si substrate,” in *Proc. Symp. VLSI Technol.*, 2009, pp. 94–95.
[3] S.-G. Hur et al., “A practical Si nanowire technology with nanowire-on-insulator structure for beyond 10nm logic technologies,” in *Proc. IEEE Int. Electron Devices Meeting*, 2013, pp. 26.5.1–26.5.4, doi: 10.1109/IEDM.2013.6724698.
[4] S. Johansson, E. Memisevic, L.-E. Wernersson, and E. Lind, “High-frequency gate-all-around vertical InAs nanowire MOSFETs on Si substrates,” *IEEE Electron Device Lett.*, vol. 35, no. 5, pp. 518–520, May 2014, doi: 10.1109/LED.2014.2310119.
[5] N. Loubet et al., “Stacked nanosheet gate-all-around transistor to enable scaling beyond FinFET,” in *Proc. Symp. VLSI Technol.*, 2017, pp. T230–T231, doi: 10.23919/VLSIT.2017.7998183.
[6] D. Yakimets et al., “Power aware FinFET and lateral nanosheet FET targeting for 3nm CMOS technology,” in *IEDM Tech. Dig.*, Dec. 2017, pp. 20.4.1–20.4.4, doi: 10.1109/IEDM.2017.8268429.

[7] R. H. Dennard, F. H. Gaensslen, H.-N. Yu, V. L. Rideout, E. Bassous, and A. R. LeBlanc, "Design of ion-implanted MOSFET's with very small physical dimensions," *IEEE J. Solid-State Circuits*, vol. 9, no. 5, pp. 256–268, Oct. 1974, doi: 10.1109/JSSC.1974.1050511.
[8] Q. Huo et al., "Physics-based device-circuit cooptimization scheme for 7-nm technology node SRAM design and beyond," *IEEE Trans. Electron Devices*, vol. 67, no. 3, pp. 907–914, Mar. 2020, doi: 10.1109/TED.2020.2964610.
[9] V. B. Litovski, J. I. Radjenovic, Z. M. Mrcarica, and S. L. Milenkovic, "MOS-transistor modeling using neural network," *Electron. Lett.*, vol. 28, no. 18, pp. 1766–1768, Aug. 1992, doi: 10.1049/el:19921124.
[10] H. Huang and C. X. Wu, "Approximation capabilities of multilayer fuzzy neural networks on the set of fuzzy-valued functions," *Inf. Sci.*, vol. 179, no. 16, pp. 2762–2773, Jul. 2009, doi: 10.1016/j.ins.2009.04.004.
[11] J. Wei et al., "Advanced MOSFET model based on artificial neural network," in *Proc. China Semicond. Technol. Int. Conf. (CSTIC)*, Shanghai, China, 2020, pp. 1–3, doi: 10.1109/CSTIC49141.2020.9282457.
[12] K. Mehta and H.-Y. Wong, "Prediction of FinFET current-voltage and capacitance-voltage curves using machine learning with autoencoder," *IEEE Electron Device Lett.*, vol. 42, no. 2, pp. 136–139, Feb. 2021, doi: 10.1109/LED.2020.3045064.
[13] C. Akbar, Y. Li, and W. L. Sung, "Machine learning aided device simulation of work function fluctuation for multichannel gate-all-around silicon nanosheet MOSFETs," *IEEE Trans. Electron Devices*, vol. 68, no. 11, pp. 5490–5497, Nov. 2021, doi: 10.1109/TED.2021.3084910.
[14] M. -Y. Kao, H. Kam, and C. Hu, "Deep-learning-assisted physics-driven MOSFET current-voltage modeling," *IEEE Electron Device Lett.*, vol. 43, no. 6, pp. 974–977, Jun. 2022, doi: 10.1109/LED.2022.3168243.
[15] R. Butola, Y. Li, and S. R. Kola, "A machine learning approach to modeling intrinsic parameter fluctuation of gate-all-around Si nanosheet MOSFETs," *IEEE Access*, vol. 10, pp. 71356–71369, 2022, doi: 10.1109/ACCESS.2022.3188690.
[16] H. Carrillo-Nuñez, N. Dimitrova, A. Asenov, and V. Georgiev, "Machine learning approach for predicting the effect of statistical variability in Si junctionless nanowire transistors," *IEEE Electron Device Lett.*, vol. 40, no. 9, pp. 1366–1369, Sep. 2019, doi: 10.1109/LED.2019.2931839.
[17] K. Ko, J. K. Lee, M. Kang, J. Jeon, and H. Shin, "Prediction of process variation effect for ultrascaled GAA vertical FET devices using a machine learning approach," *IEEE Trans. Electron Devices*, vol. 66, no. 10, pp. 4474–4477, Oct. 2019, doi: 10.1109/TED.2019.2937786.
[18] M.-H. Oh, M.-W. Kwon, K. Park, and B.-G. Park, "Sensitivity analysis based on neural network for optimizing device characteristics," *IEEE Electron Device Lett.*, vol. 41, no. 10, pp. 1548–1551, Oct. 2020, doi: 10.1109/LED.2020.3016119.
[19] G. Choe et al., "Machine learning assisted statistical variation analysis of ferroelectric transistors: From experimental metrology to predictive modeling," in *Proc. IEEE Symp. VLSI Technol.*, Honolulu, HI, USA, 2022, pp. 336–337, doi: 10.1109/VLSITechnologyandCir46769.2022.9830392.
[20] Q. Chen and G. Chen, "Artificial neural network compact model for TFTs," in *Proc. 7th Int. Conf. Comput. Aided Design Thin-Film Transistor Technol. (CAD-TFT)*, 2016, p. 11, doi: 10.1109/CAD-TFT.2016.7785057.
[21] L. Zhang and M. Chan, "Artificial neural network design for compact modeling of generic transistors," *J. Comput. Electron.*, vol. 16, no. 3, pp. 825–832, Sep. 2017, doi: 10.1007/s10825-017-0984-9.
[22] J. Wang, Y.-H. Kim, J. Ryu, C. Jeong, W. Choi, and D. Kim, "Artificial neural network-based compact modeling methodology for advanced transistors," *IEEE Trans. Electron Devices*, vol. 68, no. 3, pp. 1318–1325, Mar. 2021, doi: 10.1109/TED.2020.3048918.
[23] Q. Yang et al., "Transistor compact model based on multi-gradient neural network and its application in SPICE circuit simulations for gate-all-around Si cold source FETs," *IEEE Trans. Electron Devices*, vol. 68, no. 9, pp. 4181–4188, Sep. 2021, doi: 10.1109/TED.2021.3093376.
[24] M.-Y. Kao, F. Chavez, S. Khandelwal, and C. Hu, "Deep learning-based BSIM-CMG parameter extraction for 10-nm FinFET," *IEEE Trans. Electron Devices*, vol. 69, no. 8, pp. 4765–4768, Aug. 2022, doi: 10.1109/TED.2022.3181536.
[25] S. Guglani et al., "Artificial neural network surrogate models for efficient design space exploration of 14-nm FinFETs," in *Proc. Int. Conf. Device Res. Conf. (DRC)*, Columbus, OH, USA, 2022, pp. 1–2, doi: 10.1109/DRC55272.2022.9855816.
[26] C. Elkan, "The foundations of cost-sensitive learning," in *Proc. Int. Joint Conf. Artif. Intell.*, vol. 17, no. 1, pp. 973–978, 2001.
[27] V. S. Sheng and C. X. Ling, "Thresholding for making classifiers cost-sensitive," in *Proc. 21st Nat. Conf. Artif. Intell.*, vol. 6, 2006, pp. 476–481.
[28] Z.-H. Zhou and X.-Y. Liu, "On multi-class cost-sensitive learning," *Comput. Intell.*, vol. 26, no. 3, pp. 232–257, 2010. doi: 10.1111/j.1467-8640.2010.00358.x.
[29] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 4700–4708, doi: 10.1109/CVPR.2017.243.
[30] Z. Yao, J. Li, Z. Guan, Y. Ye, and Y. Chen, "Liver disease screening based on densely connected deep neural networks," *Neural Netw.*, vol. 123, pp. 299–304, Mar. 2020, doi: 10.1016/j.neunet.2019.11.005.
[31] R. Venugopal, Z. Ren, S. Datta, M. S. Lundstrom, and D. Jovanovic, "Simulating quantum transport in nanoscale transistors: Real versus mode-space approaches," *J. Appl. Phys.*, vol. 92, no. 7, pp. 3730–3739, 2002, doi: 10.1063/1.1503165.
[32] J. Lee, Y. Kim, and S. Cho, "Design of poly-Si junctionless fin-channel FET with quantum-mechanical drift-diffusion models for sub-10-nm technology nodes," *IEEE Trans. Electron Devices*, vol. 63, no. 12, pp. 4610–4616, Dec. 2016, doi: 10.1109/TED.2016.2614990.
[33] N. Bousari, M. Anvarifard, and S. Haji-Nasiri, "Improving the electrical characteristics of nanoscale triple-gate junctionless FinFET using gate oxide engineering," *AEU-Int. J. Electron. Commun.*, vol. 108, pp. 226–234, Aug. 2019, doi: 10.1016/j.aeue.2019.06.017.
[34] S. Bangsaruntip et al., "Density scaling with gate-all-around silicon nanowire MOSFETs for the 10 nm node and beyond," in *Proc. IEEE Electron Devices Meeting (IEDM)*, Washington, DC, USA, 2013, pp. 20.2.1–20.2.4, doi: 10.1109/IEDM.2013.6724667.