Artificial Neural Network-Based Compact Modeling Methodology for Advanced Transistors

Jing Wang, *Senior Member, IEEE*, Yo-Han Kim, Jisu Ryu, Changwook Jeong, Woosung Choi, and Daesin Kim

***Abstract*— The artificial neural network (ANN)-based compact modeling methodology is evaluated in the context of advanced field-effect transistor (FET) modeling for Design-Technology-Cooptimization (DTCO) and pathfinding activities. An ANN model architecture for FETs is introduced, and the results clearly show that by carefully choosing the conversion functions (i.e., from ANN outputs to device terminal currents or charges) and the loss functions for ANN training, ANN models can reproduce the current–voltage and charge–voltage characteristics of advanced FETs with excellent accuracy. A few key techniques are introduced in this work to enhance the capabilities of ANN models (e.g., model retargeting, variability modeling) and to improve ANN training efficiency and SPICE simulation turn-around-time (TAT). A systematical study on the impact of the ANN size on ANN model accuracy and SPICE simulation TAT is conducted, and an automated flow for generating optimum ANN models is proposed. The findings in this work suggest that the ANN-based methodology can be a promising compact modeling solution for advanced DTCO and pathfinding activities.**

***Index Terms*— Artificial neural network (ANN), circuit simulation, compact modeling, design-technology-cooptimization (DTCO), emerging devices, field-effect transistor (FET), machine learning, pathfinding, SPICE, statistical modeling.**

I. Introduction

TRANSISTOR compact models are indispensable for circuit simulation, which is essential for efficient analysis and design of integrated circuits (ICs). Standard compact models of field-effect transistors (FETs) (e.g., BSIM [1], PSP [2]) are composed of physics-based equations and have been widely adopted in the process design kits (PDKs) for IC product design. As CMOS technology is approaching its scaling limit, various emerging device options need to be assessed during the Design-Technology-Cooptimization (DTCO) activities with a fast turn-around-time (TAT). In this scenario, the use of standard FET compact models may face two challenges: 1) emerging devices may display electrical characteristics that are not well captured by the standard FET models, and developing the physics-based model equations for the new physical phenomena requires high expertise and a long TAT and 2) for equation-based models, it is still challenging to fully automate the model parameter extraction process while achieving a very high fitting accuracy.

The lookup table (LUT)-based method [3], [4] has been proposed as an alternative to equation-based compact models. However, it suffers a large SPICE simulation TAT and convergence issues for large-scale circuits. In addition, an LUT-based model lacks model *knobs* that can be used to manipulate the output characteristics of the model (e.g., for model retargeting and variability modeling), which can limit the application of LUT-based models for advanced technology evaluation.

Artificial neural networks (ANNs) have a history of serving as compact models for semiconductor devices, with a particular success in radio frequency (RF) device applications [5]–[15]. With the surge of machine learning applications in recent years, high-performance GPU servers and efficient software platforms for ANN training (e.g., PyTorch [16], TensorFlow [17]) have become widely available to the compact modeling community. For this reason, it is certainly worthwhile to further explore the potentials of the ANN-based methodology for advanced FET modeling.

In this work, we conduct a comprehensive evaluation of the ANN-based compact modeling methodology in the context of advanced FET modeling, with a focus on the following aspects: model fitting capability (accuracy), model generation (e.g., ANN training) TAT, SPICE simulation TAT, model retargeting, and feasibility of variability modeling. By introducing a number of key elements to the ANN-based compact modeling methodology, we have successfully achieved high model accuracy, fast ANN training TAT, and efficient SPICE simulation TAT. Our encouraging results have shown that the ANN-based compact modeling methodology may find its effective application in advanced technology DTCO and pathfinding activities.

The rest of this article is divided into the following sections. Section II introduces the ANN model architecture used in this work and presents the key results for the FET current–voltage (I–V) and charge–voltage (Q–V) fitting as well as ring

Manuscript received November 27, 2020; revised December 29, 2020; accepted December 30, 2020. Date of publication January 15, 2021; date of current version February 24, 2021. The review of this article was arranged by Editor Y. Chauhan. *(Corresponding author: Jing Wang.)*

Jing Wang and Woosung Choi are with Device Lab, DSA R&D, Samsung Semiconductor Inc., San Jose, CA 95134 USA (e-mail: jing.wang1@samsung.com).

Yo-Han Kim, Jisu Ryu, Changwook Jeong, and Daesin Kim are with the Data and Information Technology Center, Samsung Electronics, Suwon 16677, South Korea.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TED.2020.3048918.

Digital Object Identifier 10.1109/TED.2020.3048918

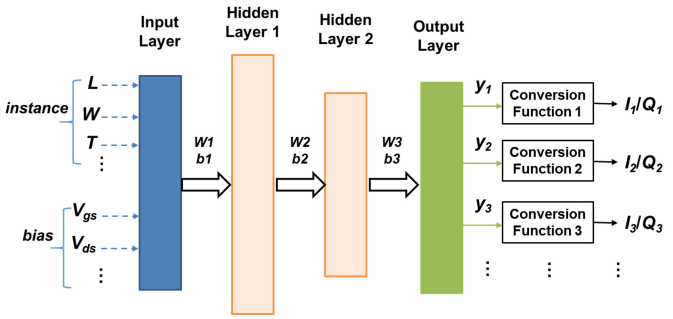


Fig. 1. ANN as the compact model for an FET. Two hidden layers are shown in this diagram, while in practice, the number of hidden layers is adjustable. The numbers of neurons in the first and second hidden layers are denoted by $N1$ and $N2$, respectively. W_i and b_i ($i = 1, 2, 3$) denote the ANN weights (in the double-precision, floating-point format), and the output variables of the ANN (e.g., y_1, y_2, y_3, . . .) are converted into a terminal current (I) or a terminal charge (Q) by using a conversion function.

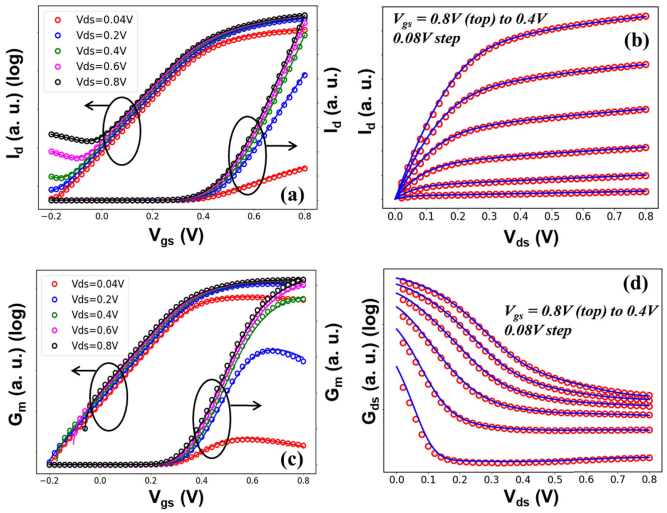


Fig. 2. ANN model results (lines) versus targets (circles) for an advanced n-type FET. (a) I_d versus V_{gs}. (b) I_d versus V_{ds}. (c) G_m versus V_{gs}. (d) G_{ds} versus V_{ds}.

oscillator (RO) simulation. In Section III, we propose several key techniques that are critical for improving ANN model training TAT and model capability. Section IV covers some important topics related to variability modeling, SPICE simulation TAT, and automated optimum ANN model generation. Our key conclusions are summarized in Section V.

II. ANN Model Architecture and Results

Fig. 1 illustrates an example of using ANNs as the compact model for an FET. The neurons of the input layer are for voltage biases applied to the FET terminals and the instance parameters (e.g., L, W, T). The number of hidden layers (i.e., fully connected) in the ANN and the number of neurons in each hidden layer are hyperparameters that can be tuned to achieve the optimum model accuracy and SPICE simulation TAT (to be discussed in detail in Section IV). Each neuron in the output layer corresponds to a terminal current (I) or a terminal charge (Q). Considering the fact that some terminal currents or charges may vary by many orders of magnitude during the FET operation, a conversion function needs to be introduced to improve the ANN model fitting accuracy. The details of the conversion functions used in this work will be discussed in the following section.

Samples for ANN training are obtained from I–V or Q–V (C–V) data generated by Technology Computer-Aided Design (TCAD) simulation or hardware measurements, and the ANN training is conducted by using backward propagation algorithms available in the standard ANN software packages (e.g., PyTorch, TensorFlow). In order to achieve high ANN model accuracy, it is important to define a proper loss function for ANN training. In this work, the loss function is defined as follows:

$$\text{loss}(M) = \frac{1}{N_S}\sum_{i=1}^{N_S}\Bigg[(1-d1)\cdot \text{err}(M^{(i)}) + (d1-d2) \cdot \text{err}\left(\frac{dM^{(i)}}{dV_{gs}}\right) + d2\cdot \text{err}\left(\frac{dM^{(i)}}{dV_{ds}}\right)\Bigg]. \quad (1)$$

Here M represents a terminal current or a terminal charge of the modeled device (e.g., I_d, Q_g, Q_d, or Q_s), N_S is the number of training samples, i denotes the ith training sample, $d1$ and $d2$ ($0 \le d2 \le d1 \le 1$) are weights used to indicate the importance of the derivatives of M with respect to V_{gs} and V_{ds} for model fitting, and err() refers to the formula for calculating the relative fitting error of the model value with respect to the target. If one ANN model is used to fit multiple terminal currents and/or charges, the total loss function used for the ANN training should be a sum of all relevant loss(M) functions.

A. ANN Model for FET I–V Characteristics

Fig. 2 illustrates the ANN I–V model fitting results for an advanced n-type FET at various bias conditions. It is clear that high model accuracy has been achieved for the drain current (I_d) as well as its derivatives with respect to V_{gs} and V_{ds} (i.e., G_m and G_{ds}). It is worth mentioning that the selection of the conversion function is critical for achieving high I–V model accuracy. In this case, the following conversion function is used for the channel (drain–source) current

$$I_{ds} = I_0 \cdot V_{ds} \cdot 10^y. \quad (2)$$

Here, I_0 is a normalization factor (e.g., 1 pA), and y is the output from the corresponding neuron in the ANN output layer (see Fig. 1). This conversion function guarantees a zero I_{ds} when $V_{ds} = 0$ V, and limits the range of y even when I_{ds} varies by many orders of magnitude during the FET operation. (For simplicity, the gate leakage currents and the substrate currents of the FET are not included in this study. Therefore, the drain current, I_d, equals to the channel current, I_{ds}.)

If the ANN FET model is to be used in RF distortion simulations [18], it is important to ensure the I–V model passes the Gummel symmetry test (GST) at $V_{ds} = 0$ V [19]. In this work, we introduce the following voltage smoothing functions for V_{ds} and V_{gs}, respectively:

$$V_{ds\_sm} = \sqrt[\epsilon]{V_{ds}^{\epsilon} + \eta^{\epsilon}} - \eta \quad (3)$$

$$V_{gs\_sm} = V_{gs} + (V_{ds\_sm} - V_{ds})/2. \quad (4)$$

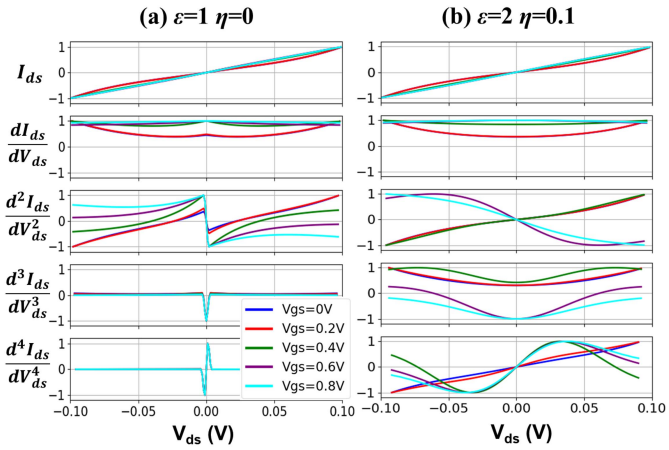


Fig. 3. GST results for the ANN I–V model (a) without the voltage smoothing functions (i.e., $V_{ds\_sm} = V_{ds}$ and $V_{gs\_sm} = V_{gs}$) and (b) with $\varepsilon = 2$ and $\eta = 0.1$ for the voltage smoothing functions. The *normalized* derivatives of I_{ds} with respect to V_{ds} are shown up to the fourth order.

Here, ε and η are constants (i.e., $\varepsilon \geq 1$ and $\eta \geq 0$), and the modified voltage biases (i.e., V_{ds_sm} and V_{gs_sm}) are used as the inputs of the ANN model. (Following the convention in the standard FET models, when the FET is in the reverse mode (i.e., $V_{ds} < 0$ for nFET, $V_{ds} > 0$ for pFET), the source terminal and the drain terminal of the FET are swapped for the voltage bias calculation.) Fig. 3 clearly shows that by introducing the voltage smoothing functions and setting $\varepsilon = 2$ and $\eta = 0.1$ [Fig. 3(b)], the ANN model can provide smooth derivatives of I_{ds} with respect to V_{ds} to at least the fourth order, implying a significant improvement of the GST results over the case without voltage smoothing [Fig. 3(a)]. (It has been confirmed that the subthreshold current (e.g., at $V_{gs} = 0.1$ V) from the ANN model has a monotonic dependence on V_{ds} with $\varepsilon = 2$ and $\eta = 0.1$.) It should be noted that the voltage smoothing functions mentioned above should only be applied to the I–V ANN model, not the Q–V ANN model (to be covered in Section II-B). Otherwise, the C–V characteristics of the ANN model may be distorted to such an extent that makes it difficult to fit the target C–V data.

B. ANN Model for FET Q–V (C–V) Characteristics

In transient circuit simulation, a transistor compact model ought to provide the terminal charges (e.g., Q_g, Q_d, Q_s) of the device at each given voltage bias. However, the Q–V data are not directly available from TCAD simulation or hardware measurements; instead, the capacitance–voltage (C–V) data need to be used for Q–V model calibration. For the LUT-based models, some authors have proposed a method to compute the terminal charges by integrating the C–V data over the V_{ds} and V_{gs} space [3]. However, the computed integrals depend on the integration path, so it may result in errors for certain bias conditions. In this work, we will show that the calibration of ANN-based Q–V models can be well conducted by using the C–V data only, which is a significant advantage of ANN models over LUT-based models.

In Fig. 4, we compare the ANN model results (lines) versus the targets (circles) for the Q–V and C–V characteristics of an advanced n-type FET. (For comparison purposes, the targets are generated from a BSIM-CMG [1] model, so that the Q and C values are consistent.) First, we set $d1 = 0.5$ and $d2 = 0.25$ for the loss function [see (1)] calculation. By doing so, both the Q (e.g., Q_g) targets and the C (e.g., C_{gg}, C_{gd}) targets are used for training the Q–V ANN model. As shown in Fig. 4(a) and (b), the obtained ANN model can well fit both the Q_g–V_{gs} and C_{gg}–V_{gs} targets in this case. Next, by setting $d1 = 1.0$ and $d2 = 0.5$, we exclude the Q targets from the loss function calculation (i.e., the err($M^{(i)}$) terms in (1) become zero). It is clear from Fig. 3(d)–(f) that the trained ANN model with this setting can accurately capture the C–V characteristics of the device, while the Q values from this model may have a constant offset from the Q targets. From a circuit simulation perspective, what truly affects the simulation outcome is the derivatives of the device terminal charges with respect to voltage biases, not the charge values themselves. (This has been verified by our RO simulations, to be shown

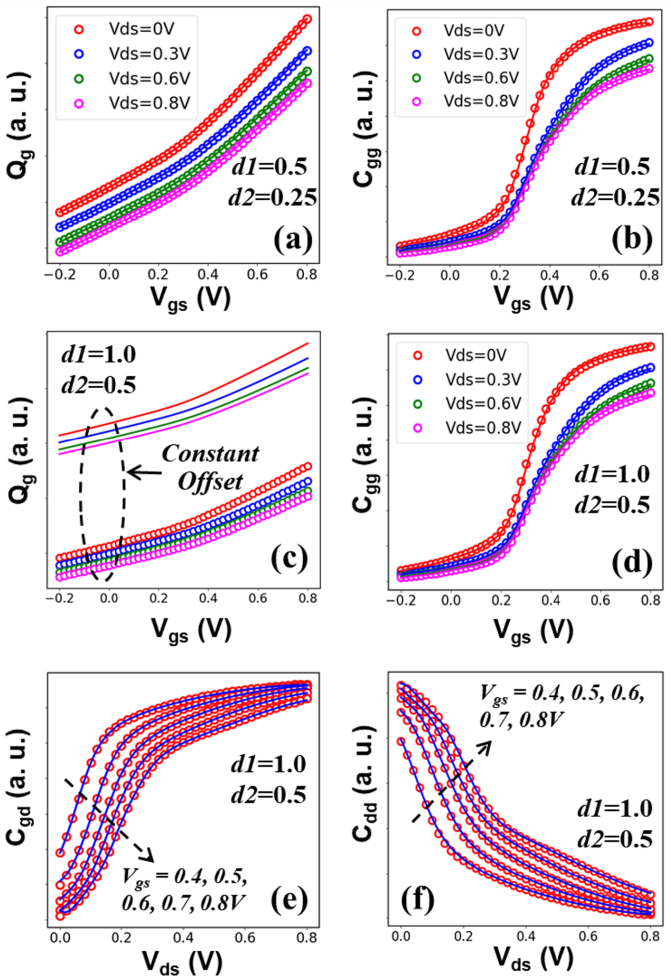
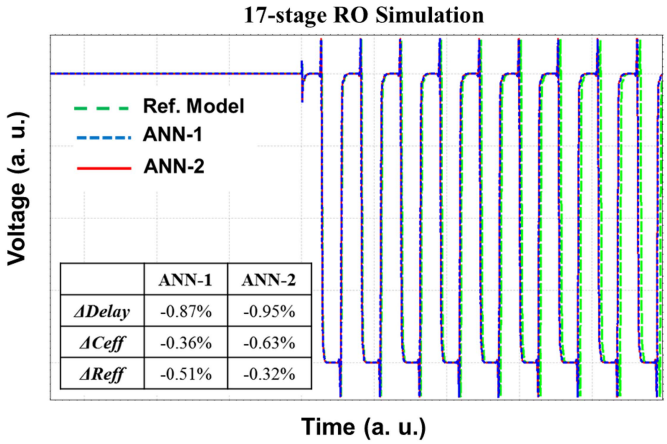


Fig. 4. ANN model results (lines) versus targets (circles) for the Q–V and C–V characteristics of an advanced n-type FET. The $d1$ and $d2$ values used in the loss function [see (1)] calculation are listed for each case. (a) Q_g versus V_{gs} ($d1 = 0.5$, $d2 = 0.25$). (b) C_{gg} ($=dQ_g/dV_{gs}$) versus V_{gs} ($d1 = 0.5$, $d2 = 0.25$). (c) Q_g versus V_{gs} ($d1 = 1.0$, $d2 = 0.5$). (d) C_{gg} ($=dQ_g/dV_{gs}$) versus V_{gs} ($d1 = 1.0$, $d2 = 0.5$). (e) C_{gd} ($=dQ_g/dV_{ds}$) versus V_{ds} ($d1 = 1.0$, $d2 = 0.5$). (f) C_{dd} ($=dQ_d/dV_{ds}$) versus V_{ds} ($d1 = 1.0$, $d2 = 0.5$).



	ANN-1	ANN-2
ΔDelay	-0.87%	-0.95%
ΔCeff	-0.36%	-0.63%
ΔReff	-0.51%	-0.32%

Fig. 5. 17-stage RO simulation results using the reference BSIM-CMG models (green dashed line), the ANN-1 models (blue dotted line), and the ANN-2 models (red solid line). Key circuit metrics such as RO *Delay*, *Ceff* (effective capacitance), and *Reff* (effective resistance) are extracted from the simulation results, and the differences between the ANN model results and the reference are summarized in the inset table.

in Section II-C.) Hence, it can be concluded that by adopting the loss function in (1) and setting $d1 = 1.0$ ($0 < d2 < d1$), the training of a Q–V ANN model can be well conducted with C–V data only, which can be directly obtained from TCAD simulation or hardware measurements.

C. RO Simulation

After the training of the ANN I–V and Q–V models, the ANN weights (e.g., $W1$, $b1$, $W2$, $b2$, $W3$, $b3$ as shown in Fig. 1) can be ported into a Verilog-A [20] model for circuit simulation. (This process is automated by using a Python script.) Fig. 5 shows the voltage waveforms of a 17-stage RO simulated using the reference BSIM-CMG models as well as the ANN models. The I–V and Q–V target data of both nFET and pFET are generated from the reference models, and the ANN I–V and Q–V models are trained with the relevant target data. The difference between the ANN-1 and ANN-2 models is that the Q–V models in ANN-1 are trained with the inclusion of the Q values in the loss function ($d1 = 0.5$, $d2 = 0.25$), while those in ANN-2 are trained with the C–V data only ($d1 = 1.0$, $d2 = 0.5$). The results clearly show that the ANN models offer very high accuracy for RO simulations (i.e., less than 1% error for Delay, Ceff, and Reff) as compared with the reference models, and ANN-2, with the Q–V models trained with C–V data only, performs equally well as ANN-1.

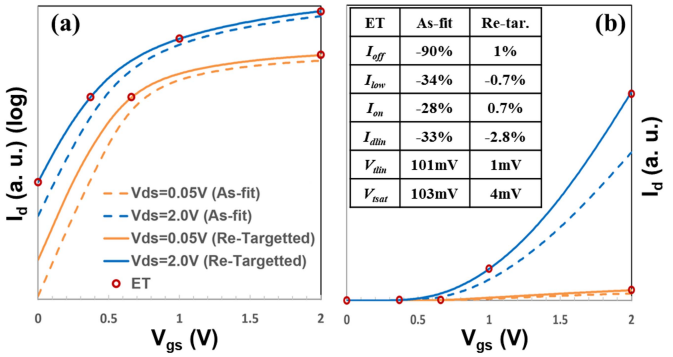
III. Advanced Features

This section covers several key elements we introduce to the ANN-based compact modeling methodology for improving the model capability and the model creation (i.e., ANN training) efficiency.

A. Model Retargeting

When developing compact model libraries for advanced technology, it is common practice that the model is first fit to I–V and C–V data measured from hardware (the so-called as-fit model). Then, a carefully selected subset of model parameters are tuned so that the output of the model can match a list of electrical targets (ETs). This process is the so-called model retargeting. For ANN-based compact models, if we retrain the ANN to fit the ETs, then overfitting is nearly inevitable due to the limited number of ETs. To address this issue, we introduce the following methodology.

1) When constructing an ANN model that is expected to go through the model retargeting process, limit the number of neurons in the final hidden layer to the number of ETs minus one.
2) After the as-fit ANN model is created based on the I–V and C–V data, only adjust the weights related to the output layer (i.e., W_{output}, b_{output}) of the ANN to match the ETs.
3) If the ET list includes multiple device instances (e.g., L), separate W_{output} and b_{output} values can be extracted for each device instance to best fit its ETs if needed. Then, an analytical or table-based model for W_{output} and b_{output} (as a function of the device instances) can be built and included as part of the ANN model.



ET	As-fit	Re-tar.
I_{off}	-90%	1%
I_{low}	-34%	-0.7%
I_{on}	-28%	0.7%
I_{dlin}	-33%	-2.8%
V_{tlin}	101mV	1mV
V_{tsat}	103mV	4mV

Fig. 6. Example of ANN model retargeting. The I_d versus V_{gs} curves [(a) log-scale and (b) linear-scale] are generated from both the as-fit ANN model (dashed lines) and the retargeted ANN model (solid lines). The data points for the ETs (circles) are included for comparison. The differences between the model (as-fit and retargeted) results and the ET values are summarized in the inset table.

An example of ANN model retargeting is shown in Fig. 6. There are six ETs to be matched in this case, so the number of neurons in the second (final) hidden layer of the ANN is set to five. Although the as-fit model results (dashed lines) are significantly different from the ETs (symbols), the retargeted ANN model (solid lines) matches the ETs very well.

B. Initial Weights Setting for Training TAT Improvement

For ANN training, the initial ANN weights (i.e., W_i, b_i) are usually assigned to random numbers generation from a certain distribution, and their values are then adjusted gradually during the learning process to minimize the loss function value. In the practice of compact model development, it is not uncommon that a model fitted to a device with similar characteristics is already available before the model parameter extraction is conducted. In this case, using a prefitted model as a start model may effectively speed up the parameter extraction

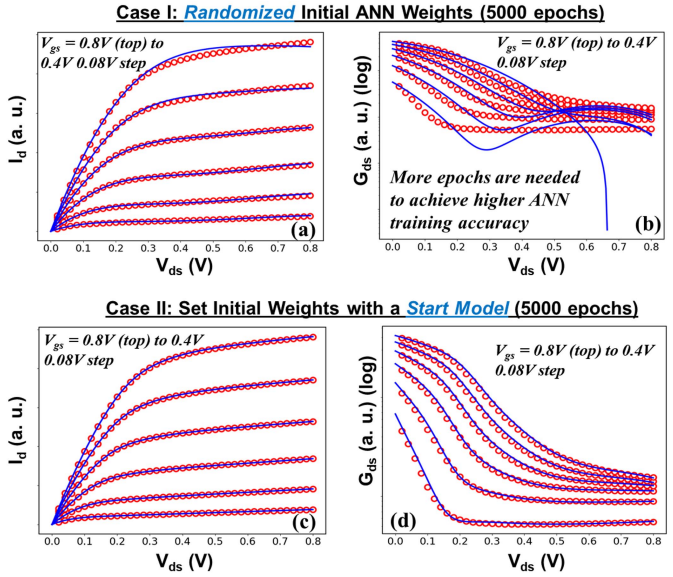


Fig. 7. ANN model results (lines) versus targets (circles) for I_d versus V_{ds} and G_{ds} versus V_{ds} characteristics of an advanced n-type FET. The ANN models are trained with two different settings: 1) with randomized initial ANN weights [(a) and (b)] and 2) initial ANN weights are set based on a start model [(c) and (d)]. The ANN training has stopped after 5000 epochs in both cases.

process. In this section, we apply this idea in the context of ANN-based compact modeling, by introducing an option to load a pretrained ANN model as the initial weights for the ANN training process.

Fig. 7 shows an example of using a start model as the initial ANN weights to expedite the ANN training. In Case I [Fig. 7(a) and (b)], the ANN model is trained with randomized initial ANN weights and the training has stopped after 5000 epochs. The results in Fig. 7(b) clearly show that the obtained ANN model displays substantial errors for G_{ds} fitting, implying that more training epochs are needed to achieve better model accuracy. In Case II [Fig. 7(c) and (d)], we set the initial ANN weights by using a start model, which has been fitted to a device with similar I–V characteristics as the device to be modeled (e.g., V_{th} and I_{ON} differences between the two devices are ~80 mV and ~20%, respectively). After running 5000 epochs, the trained ANN model in this case shows high fitting accuracy for both I_d [Fig. 7(c)] and G_{ds} [Fig. 7(d)], a significant improvement over Case I. (If randomized initial ANN weights are used, about 200 000 to 500 000 epochs are needed to achieve the same level of model accuracy as in Case II, implying that choosing a proper start ANN model may help improve the ANN training efficiency by ~40–100×.) It is worth mentioning that in DTCO activities, gradual updates on device characteristics from run to run are mostly expected. Therefore, the ANN training efficiency may be greatly improved by adopting an early version of the ANN model as the start model for initial ANN weights setting.

C. Global Fitting Versus Local Fitting

When creating compact models for multiple device instances (e.g., W, L, and T), two schemes can be used: 1) generating one single model that fits the data for all device instances (the so-called *global fitting*) and 2) creating a separate model for each device instance (the so-called *local fitting*). In this section, we compare the global fitting versus local fitting schemes in the context of ANN model generation and simulation.

TABLE I
EXAMPLE OF GLOBAL FITTING VERSUS LOCAL FITTING (36 DEVICE INSTANCES, W/L/T, TO BE FITTED)

	Global Fitting	Local Fitting
No. of ANN Models	1 (a single ANN model for all 36 device instances)	36 (one separate ANN model for each device instance)
ANN Size	*N1*=20, *N2*=15	*N1*=10, *N2*=5
TAT per Epoch (PyTorch)	~0.015 sec.	~0.004 sec.
No. of Epochs Needed	~2,000,000	~50,000
Total Training TAT	~8 hours	~200 sec. (per model), ~2 hours for all 36 models (sequentially trained), can be parallelized
SPICE Simulation TAT	~5x	1x (reference)

Table I summarizes the global fitting versus local fitting comparison results for a test case with 36 device instances (i.e., $W/L/T$). To achieve the same level of model accuracy, the ANN size of the global model (i.e., $N1 = 20$, $N2 = 15$) has to be larger than that of a local model (i.e., $N1 = 10$, $N2 = 5$), simply because more tunable parameters (i.e., ANN weights) are needed to fit the data of all device instances than just one. The larger ANN size and the larger number of training samples (e.g., 36×) for the global fitting lead to a significantly longer training TAT per epoch than the local fitting (i.e., 0.015 s versus 0.004 s, by using PyTorch). In addition, more epochs are needed in ANN training for the global fitting, since it takes more iterations to train a larger ANN with more training samples. For these reasons, the total training TAT for the global model (i.e., ~8 h) is substantially larger than that of the local models (i.e., ~2 h if training all 36 local models sequentially). Considering the fact that the training of different local models is independent, local model training can be further expedited by running the 36 jobs in parallel, if the computational resources permit. Finally, our results show that the SPICE simulation TAT for the global model is ~5× of that for a local model, due to the fact that the SPICE simulation TAT of an ANN model strongly depends on the ANN size, which will be covered in detail in Section IV-B. Based on this study, we can conclude that using a local fitting scheme is highly preferred for ANN-based modeling, which can offer a significantly better ANN training TAT and a faster SPICE simulation TAT than the global fitting scheme.

IV. DISCUSSION

A. Feasibility of Capturing Device Variability in ANN Models

Accurately modeling device variations is critical for circuit performance benchmark and yield analysis. With

equation-based compact models such as BSIM and PSP, a selected set of model parameters can be used to capture both the global (process) variations and the local variations (e.g., random dopant fluctuations, metal gate granularity, line-edge roughness, and so on) [21]. In contrast, LUT-based models lack tunable model parameters, making variability modeling a challenge. In [4], a generic method was proposed to capture process variations (PVs) in LUT-based FET models, while the methodology for accurately treating local variations with LUTs remains unavailable to our knowledge. Different from LUT models, ANN-based compact models acquire tunable model parameters (e.g., ANN weights) as well as a flexible model architecture (e.g., the number of neurons in each layer is adjustable), making it feasible to create accurate, ANN-based variability models.

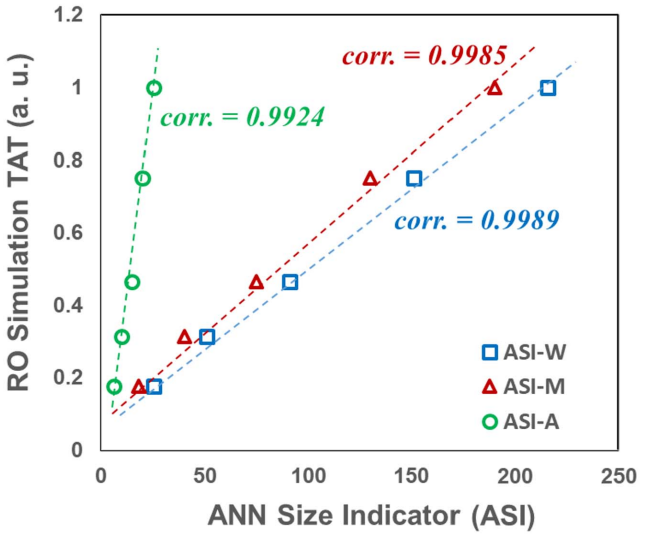
To treat global (process) variations in an ANN-based FET model library, one of the following two methods can be adopted.

1) The methodology introduced in [4] for capturing PVs in LUT-based FET models is directly applicable to ANN-based FET models. To do so, the I–V and Q–V LUTs for each process split need to be replaced by ANN models, and the rest of the I–V and Q–V calculation remains the same as in [4]. (Please refer to [4] for the details of the methodology.)
2) The method described in 1) requires generating separate ANN models for each process split. Alternatively, the PVs may also be captured by expanding the nominal (i.e., without variations) ANN models using the following approach. In addition to the neurons for the device instances and voltage biases (see Fig. 1), more neurons can be added to the input layer of the ANN for all PV sources (the so-called PV *neurons*). In the nominal case, the input for each PV neuron is zero. During the ANN training, the ANN weights associated with the PV neurons are adjusted together with other ANN weights, so that the output of the ANN model matches the FET I–V/Q–V characteristics for all process splits. (Please refer to [4] for the method for generating process splits.)

The treatment of local variations in an ANN-based FET model is similar to that in an equation-based model. As shown in Section III-A, the output of an ANN model can be effectively manipulated, by tuning the ANN weights related to the output layer (i.e., W_{output}, b_{output}), for ET retargeting. As we can expect, the same set of ANN weights can be used in the variability models (e.g., Monte Carlo, corners) that capture the local and global variations. (To be concise, the details of variability model library generation and calibration are not included here.)

B. SPICE Simulation TAT as a Function of ANN Size

Computational efficiency of a compact model will directly impact the circuit (SPICE) simulation TAT. Therefore, it is important to measure the SPICE simulation TAT of some benchmark circuits when developing new compact models. In an ANN model, the number of computations (e.g., multiplications, activation function evaluations) is determined by the ANN size, as measured by the number of hidden layers (N_{HL}) and the number of neurons in each hidden layer. In general, a larger ANN may offer a higher fitting capability due to its larger number of fitting parameters (i.e., ANN weights). On the other hand, a larger ANN size is expected to degrade the SPICE simulation TAT. In this section, we conduct a study to explore the quantitative dependence of the SPICE simulation TAT on the ANN size, and in Section IV-C, we will provide a guideline for generating an optimum ANN model with the consideration of both model accuracy and SPICE simulation TAT.



Name	*Definition*	*Formula*
ASI-W	No. of ANN weights	$N1\cdot(Ni+1)+N2\cdot(N1+1)+No\cdot(N2+1)$
ASI-M	No. of multiplications	$N1\cdot N2+N1\cdot Ni+N2\cdot No$
ASI-A	No. of activation function evaluations	$N1+N2$

Fig. 8. SPICE simulation TAT for a 17-stage RO versus ASIs. The definition and the formula for each ASI are summarized in the inset table. Here, *Ni*, *N*1, *N*2, and *No* are for the number of neurons in the input layer, the first hidden layer, the second hidden layer, and the output layer of the ANN, respectively. The correlation coefficients between the RO simulation TAT and each ASI are shown in the plot.

Although in principle, N_{HL} is an adjustable hyperparameter, it is found in our work that $N_{\text{HL}} = 2$ is optimal because when $N_{\text{HL}} > 2$, the ANN model may suffer larger training and SPICE simulation TATs, while when $N_{\text{HL}} = 1$, the fitting capability of the ANN model may be degraded. For this reason, we choose to focus on ANN models with two hidden layers in this study. Five ANN models with various numbers of neurons (i.e., $N1/N2$ = 15/10, 10/10, 10/5, 5/5, 3/3) are created, and SPICE simulations of a 17-stage RO circuit are performed with each ANN model option. To represent the ANN size quantitatively, we introduce three ANN size indicators (ASIs), based on the number of ANN weights (ASI-W), as well as the number of multiplications (ASI-M) and the number of activation function evaluations (ASI-A) in one ANN model inference. Fig. 8 plots the normalized SPICE

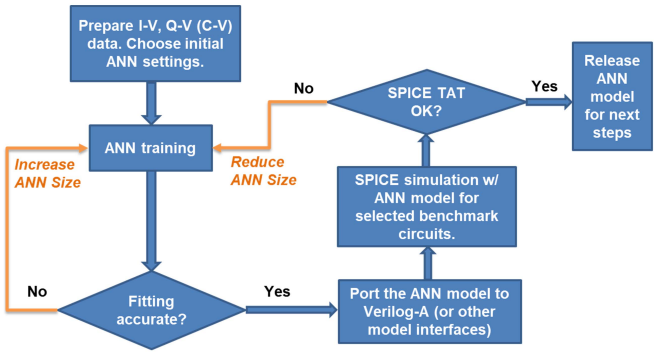


Fig. 9. Automated flow for generating an optimum ANN model with the consideration of both model accuracy and SPICE simulation TAT.

simulation TATs for the RO versus the three ASIs. It is clear that the RO simulation TAT has a nearly perfect correlation with all three indicators, implying that these ASI indicators are effective for estimating the SPICE simulation TAT as a function of ANN sizes, without the need of conducting the actual ANN model training and SPICE simulation. (In our study, the memory usage of the RO SPICE simulation is found insensitive to the ANN model size.)

To investigate the impact of the ANN size on model accuracy, we compare the simulated RO delays from the five ANN model options with that from the reference model. A less than 0.5% error is achieved for all cases except the one with the smallest ANN size ($N1 = 3$, $N2 = 3$), which leads to a 1.4% error. In practice, once a model accuracy target is determined, there should exist an optimum ANN size that can be used to deliver an *accurate enough* ANN model with the best possible SPICE simulation TAT.

C. Automated Flow for Generating an Optimum ANN Model

As shown in Section IV-B, when adjusting the ANN size, there is a tradeoff between ANN model accuracy and SPICE simulation TAT. Therefore, it is highly beneficial to develop an automated flow to decide the *optimum* ANN configuration for each modeling task. Fig. 9 illustrates such a flow, which includes initial data preparation and ANN setup, ANN training, model accuracy/quality check, model porting (e.g., to Verilog-A), SPICE simulation validation, and TAT verification. During this process, the ANN size is increased if the model accuracy is unsatisfactory, and it is reduced if the SPICE simulation TATs do not meet the model users' expectation. Using this automated flow, an optimum ANN model with the best balance of model accuracy and TAT can be obtained, which is critical for accurate and efficient SPICE simulations.

D. Comparison With Other Compact Model Types

In Table II, we compare the key properties of ANN-based FET models with those of the standard, physical equation-based FET models, as well as LUT models. Clearly, the ANN model is at an advantage over the LUT counterpart in most aspects, except for its modest overhead in model parameter extraction (i.e., ANN training). In addition, the results shown in Section II-B indicate that ANN models may offer higher Q–V model accuracy than LUTs. For these reasons, it can be concluded that ANN is a superior option to LUT for *data-based* compact modeling, given that the hardware and software infrastructure for ANN training is available. (More discussions on the advantages of ANN-based compact models over the LUT models are available in [13].)

TABLE II
COMPARISON AMONG ANN, STANDARD, AND LUT MODELS

	Standard	LUT	ANN
Model Equation Creation	long TAT, expertise needed	no need	no need
Model Parameter Extraction	long TAT, expertise needed[a]	no need	short TAT, fully automated
Data Requirement	medium	very high	high
Variability Modeling Capability	high	low	medium
SPICE simulation TAT	fast	slow	medium[b]

[a]It may be automated to some extent, but it can be challenging to obtain very high model accuracy with a fully automated flow.
[b]It may be further improved by converting the Verilog-A based ANN models into hand-coded C code [22], and the extent of the improvement needs to be characterized in future work.

As compared with the standard FET models, ANN models hold an apparent advantage in the efficiency of model equation creation and parameter extraction, while the standard models offer a faster SPICE simulation TAT, a higher capability for variability modeling, and less data requirement due to their physics-based nature. For this reason, we expect that the standard FET models will remain the best choice for the PDK model libraries, while ANN-based FET models may find their superiority in advanced DTCO and pathfinding activities, due to their excellent model generation efficiency and fitting capabilities. (Demonstration of ANN model fitting capabilities for various emerging devices is beyond the scope of this article. We would like to refer the interested readers to [11], [15] for examples of ANN-based Tunnel FET [23] modeling.)

V. CONCLUSION

We have evaluated the ANN-based compact modeling methodology for advanced FET modeling. Our results clearly show that ANN-based FET models well reproduce the I–V and C–V characteristics of the modeled devices, and the SPICE simulation results based on the ANN models match the RO delay target with a less than 1% error. We have extended the capabilities of ANN models for model retargeting and variability modeling, and several key techniques for improving ANN training TAT and SPICE simulation TAT have been introduced. We have also studied the impact of the ANN size on ANN model accuracy and SPICE simulation TAT, and an automated flow for generating optimum ANN

models has been proposed. Based on our findings in this work, we conclude that the ANN-based compact modeling methodology shows promises for advanced DTCO and pathfinding activities.

REFERENCES

[1] J. P. Duarte *et al.*, "BSIM-CMG: Standard FinFET compact model for advanced circuit design," in *Proc. 41st Eur. Solid-State Circuits Conf. (ESSCIRC)*, Sep. 2015, pp. 196–201, doi: 10.1109/ESSCIRC.2015.7313862.

[2] G. Gildenblat *et al.*, "PSP: An advanced surface-potential-based MOSFET model for circuit simulation," *IEEE Trans. Electron Devices*, vol. 53, no. 9, pp. 1979–1993, Sep. 2006, doi: 10.1109/TED.2005.881006.

[3] R. A. Thakker, C. Sathe, A. B. Sachid, M. S. Baghini, V. R. Rao, and M. B. Patil, "A novel table-based approach for design of FinFET circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 7, pp. 1061–1070, Jul. 2009, doi: 10.1109/TCAD.2009.2017431.

[4] J. Wang, N. Xu, W. Choi, K.-H. Lee, and Y. Park, "A generic approach for capturing process variations in lookup-table-based FET models," in *Proc. Int. Conf. Simulation Semiconductor Processes Devices (SISPAD)*, Washington, DC, USA, Sep. 2015, pp. 309–312, doi: 10.1109/SISPAD.2015.7292321.

[5] Q. J. Zhang and K. C. Gupta, *Neural Networks for RF and Microwave Design*. Norwood, MA, USA: Artech House, 2000.

[6] J. Xu, M. C. E. Yagoub, R. Ding, and Q. J. Zhang, "Exact adjoint sensitivity analysis for neural-based microwave modeling and design," *IEEE Trans. Microw. Theory Techn.*, vol. 51, no. 1, pp. 226–237, Jan. 2003, doi: 10.1109/TMTT.2002.806910.

[7] J. Xu, D. Gunyan, M. Iwamoto, J. M. Horn, A. Cognata, and D. E. Root, "Drain-source symmetric artificial neural network-based FET model with robust extrapolation beyond training data," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Honolulu, HI, USA, Jun. 2007, pp. 2011–2014, doi: 10.1109/MWSYM.2007.380244.

[8] M. Li, O. Irsoy, C. Cardie, and H. G. Xing, "Physics-inspired neural networks for efficient device compact modeling," *IEEE J. Explor. Solid-State Comput. Devices Circuits*, vol. 2, pp. 44–49, Dec. 2016, doi: 10.1109/JXCDC.2016.2636161.

[9] Y. Lei, X. Huo, and B. Yan, "Deep neural network for device modeling," in *Proc. IEEE 2nd Electron Devices Technol. Manuf. Conf. (EDTM)*, Kobe, Japan, Mar. 2018, pp. 154–156, doi: 10.1109/EDTM.2018.8421454.

[10] 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, Apr. 2017, doi: 10.1007/s10825-017-0984-9.

[11] Y. Kim, S. Myung, J. Ryu, C. Jeong, and D. S. Kim, "Physics-augmented neural compact model for emerging device technologies," in *Proc. Int. Conf. Simulation Semiconductor Processes Devices (SISPAD)*, Kobe, Japan, Sep. 2020, pp. 257–260, doi: 10.23919/SISPAD49475.2020.9241638.

[12] 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)*, Beijing, China, Oct. 2016, p. 1, doi: 10.1109/CAD-TFT.2016.7785057.

[13] D. Root, "Future device modeling trends," *IEEE Microw. Mag.*, vol. 13, no. 7, pp. 45–59, Nov. 2012, doi: 10.1109/MMM.2012.2216095.

[14] A. Huang, Z. Zhong, Y.-X. Guo, and W. Wu, "A dimension-reduced artificial neural network for the compact modeling of semiconductor devices," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Chengdu, China, May 2018, pp. 1–4, doi: 10.1109/IEEE-IWS.2018.8400840.

[15] Z. Zhang *et al.*, "New-generation design-technology co-optimization (DTCO): Machine-learning assisted modeling framework," in *Proc. Silicon Nanoelectron. Workshop (SNW)*, Kyoto, Japan, Jun. 2019, pp. 1–2, doi: 10.23919/SNW.2019.8782897.

[16] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, H. Wallach, ed. Red Hook, NY, USA: Curran Associates, 2019, pp. 8024–8035. [Online]. Available: https://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[17] M. Abadi *et al.*, "Tensorflow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Operating Syst. Design Implement. (OSDI)*, Nov. 2016, pp. 265–283.

[18] P. Bendix *et al.*, "RF distortion analysis with compact MOSFET models," in *Proc. IEEE Custom Integr. Circuits Conf.*, Orlando, FL, USA, Oct. 2004, pp. 9–12, doi: 10.1109/CICC.2004.1358719.

[19] C. C. Mcandrew, "Validation of MOSFET model source–drain symmetry," *IEEE Trans. Electron Devices*, vol. 53, no. 9, pp. 2202–2206, Sep. 2006, doi: 10.1109/TED.2006.881005.

[20] (May 2014). *Verilog-AMS Language Reference Manual*. [Online]. Available: https://www.accellera.org/downloads/standards/v-ams

[21] X. Wang *et al.*, "FinFET centric variability-aware compact model extraction and generation technology supporting DTCO," *IEEE Trans. Electron Devices*, vol. 62, no. 10, pp. 3139–3146, Oct. 2015, doi: 10.1109/TED.2015.2463073.

[22] C. C. McAndrew *et al.*, "Best practices for compact modeling in Verilog—A," *IEEE J. Electron Devices Soc.*, vol. 3, no. 5, pp. 383–396, Sep. 2015, doi: 10.1109/JEDS.2015.2455342.

[23] E. Ko, H. Lee, J.-D. Park, and C. Shin, "Vertical tunnel FET: Design optimization with triple metal-gate layers," *IEEE Trans. Electron Devices*, vol. 63, no. 12, pp. 5030–5035, Dec. 2016, doi: 10.1109/TED.2016.2619372.