Extraction of Device Structural Parameters Through DC/AC Performance Using an MLP Neural Network Algorithm

Hyundong Jang1, (Member, IEEE), Hyeok Yun1, (Member, IEEE), Chanyang Park1, (Member, IEEE), Kyeongrae Cho1, (Member, IEEE), Kihoon Nam1, (Member, IEEE), Jun-Sik Yoon1, (Member, IEEE), Hyun-Chul Choi2, (Member, IEEE), and Rock-Hyun Baek1, (Member, IEEE)

1Department of Electrical Engineering, Pohang University of Science and Technology (POSTECH), Pohang, 37673 Republic of Korea
2Department of Electronic Engineering, Yeungnam University, Gyeongsan, 38541 Republic of Korea

ABSTRACT We proposed a neural network (NN) approach that uses two multi-layer perceptron (MLP) NNs an encoder and a decoder to estimate the structural parameter (S\textsubscript{para}) of a 14-nm node fully depleted silicon on insulator (FDSOI) field-effect transistor (FET). When outputs defined by the same input exist, the proposed NN algorithm achieves loss function convergence during NN training. The decoder takes inputs of on/off current ratio, delay, and power to represent DC/AC performance for high performance (HP), low operating power (LOP), and low standby power (LSTP) applications. With the pre-trained encoder learned with R coefficients of the regression plot over 0.99 and an average percent error of approximately 1%, the decoder was modeled to estimate the S\textsubscript{para}. Our decoder successfully estimated all S\textsubscript{para} within the range that satisfies the technology node. The tendency of S\textsubscript{para} satisfying the desired figure-of-merits (FOMs) in device design can be confirmed by comparing the estimated S\textsubscript{para} of the upper 5% and 10% cases. Furthermore, it can provide device design guidance from various perspectives by presenting numerous alternatives of distinct S\textsubscript{para} sets, even when the FOM value is the same (duplicate input values). If undesirable FOMs are extracted, it is possible to determine the causal S\textsubscript{para} and provide immediate process feedback on the related unit process using the S\textsubscript{para} estimated from the lower 5% of FOMs. We performed a detailed physical analysis as an example of a delay in LOP application. NN estimation results were analyzed using gate length (\textit{L}_\text{g}) SOI thickness (\textit{T}_{\text{SOI}}), and drain-side spacer length (\textit{L}_{\text{osp}}), which mainly affect gate capacitance (\textit{C}_g) and effective current (\textit{I}_{\text{eff}}). In addition, source-side spacer length (\textit{L}_{\text{osp}}) and source/drain junction gradient (\textit{L}_{\text{adj}}) showed behaviors different from those generally selected by human experts and cases where maximal values were not estimated within the set range. The estimation of S\textsubscript{para} using the NN was effective and powerful, reducing process cost and feedback time.

INDEX TERMS Device design, device structural parameters, FDSOI FET, figure-of-merits, machine learning, manufacturing process, MLP, 14-nm node.

I. INTRODUCTION
Numerous design and production processes are required to manufacture semiconductor chips suited for various applications; these processes are large-scale, expensive, and time consuming more than a month. Various test inspections are carried out throughout the process to improve product quality. Before packing, a wafer test is typically performed, followed by a chip test. Electrical die sorting (EDS) is a test performed on wafers before packaging. It detects whether each die meets the necessary quality level by measuring the electrical parameters and determining whether the device operates appropriately. In other words, by selecting the defective die, the rate of defect that arises during the early stages of the product can be effectively limited. Thus, the product will not pass packaging unless it meets the performance requirements of the target application. In addition, identifying and resolving defects through various test inspections try to increase yield, reduce costs, and
manufacture high-quality semiconductors. Despite numerous test methods, issues to be resolved to improve yield always exist. First, when the EDS test is executed for a manufactured wafer, it is impossible to run a full test on all transistors due to time, space, and cost constraints; thus, only some transistors for the test pattern are determined. Second, while numerous figure-of-merits (FOMs) can be used to evaluate device performance, it primarily measures and assesses FOMs that can be used to reflect the qualities of an application that are appropriate for a certain purpose. Third, it is not easy to extract the structural parameter (S\textsubscript{para}) of the fabricated device from the electrical characteristics in terms of the device, which is the lowest level in semiconductors. S\textsubscript{para} can be investigated by a transmission electron microscope (TEM) or a scanning electron microscope (SEM). However, it is expensive and has the drawback of destructive testing, which necessitates wafer cutting. In addition, the existing method is difficult to apply to a large number of wafers or chips due to time and cost limitations. S\textsubscript{para} is primarily concerned with the design and affects device performance. It is also directly involved in the S\textsubscript{para}-related unit process. Thus, it is difficult to rapidly determine which S\textsubscript{para} is problematic and which unit process is related to it when an incorrect measurement result is obtained. Because different components in various production processes operate together due to the sequential process, understanding S\textsubscript{para} in the semiconductor manufacturing process is critical for determining the origin of the defect and resolving the issue on the device side.

Owing to the development of higher computing capacity, such as GPU parallel computing and the creation of distributed processing environments, machine learning (ML) technology [1] has recently been employed as a novel approach in different domains. ML can predict future occurrences by learning complicated correlations between inputs and outputs. It has the advantage of making accurate predictions in a short time. In the field of semiconductor devices, for example, the correlation between S\textsubscript{para} and electrical properties can be forward-estimated and backward-optimized using ML approaches and quickly and reliably applied to design and analysis. Furthermore, ML can be applied to various new devices, such as vertical nanowire FETs, to aid electrical characterization and provide insights regarding device and process design [2-4]. This study intends to provide insights for problem solving at the device level for incorrect results throughout the semiconductor test process by estimating S\textsubscript{para} using the ML technique. In addition, a guideline for device design that can satisfy the FOM of the desired application is offered through the estimated S\textsubscript{para}. We used a 14-nm node fully depleted silicon on insulator (FDSOI) field-effect transistor (FET) to achieve these goals. Its excellent performance and ultra-low leakage qualities make it popular in the network, consumer devices, a microcontroller unit (MCU), and Internet of things (IoT) goods. Particularly, the introduction of buried oxide (box) facilitates body bias control, enabling wider threshold voltage modulation and lower static power consumption, which can be applied to various applications [5-8]. We are interested in using semiconductor devices that are classified as high performance (HP), low operating power (LOP), and low standby power (LSTP) applications. Therefore, the on/off current ratio, delay, and power indicating DC/AC performance for each application are used as input. Then, we propose a neural network (NN) algorithm to quickly find the S\textsubscript{para} corresponding to the application.

II. RELATED WORK

For a 32-nm node high-k metal gate transistor, Choi et al. [2] established a new framework for semiconductor device design and analysis using the ML approach. They used NN to achieve precise electrical modeling between S\textsubscript{para} and FOMs. Using the gradient descent (GD) method and modeled NN, device optimization was performed to automatically find the optimal S\textsubscript{para} set that satisfies the specified FOM. The results of NN optimization were similar to those obtained by human experts. However, it has been demonstrated that a significant amount of time is saved. They can also analyze the tendency of changes in S\textsubscript{para} without performing numerous simulations by the sensitivity of each S\textsubscript{para} on the FOM with the modeled NN.

Yun et al. [3] used NN to estimate the relationship between the S\textsubscript{para} of 14-nm node FDSOI FETs and the on/off current ratio for three semiconductor applications. The FOMs were then improved through device optimization, which determined the best device structure for each of the three applications. Furthermore, the analysis of sensitivity of FOM to significant S\textsubscript{para} performed with NN was shown to be quite comparable to that performed using actual device physics. Choi et al. [2] assumed S\textsubscript{para} was a completely independent input feature within a specified range at the time. In contrast, Yun et al. [3] considered a design guideline that demands a fixed range or correlation of some S\textsubscript{para}s from a given technology node in actual device design. Thus, S\textsubscript{para} partially depends on existing input features. By altering the range of these S\textsubscript{para} in real-time throughout the optimization, they could find the best option for the technology node. Choi et al. and Yun et al. performed device design, optimization, and analysis for semiconductor devices using multi-layer perceptron (MLP) NN [9-10]. Furthermore, NNs comprise architectures in which the input dimensions are larger than the output dimensions, and the input features have a nearly or completely independent relationship.

We use the ML technique for failure analysis in the semiconductor process, not for semiconductor device optimization and analysis. The proposed technique can be directly used in the semiconductor test process by considering the FOMs retrieved in the actual wafer test process as input and calculating the output S\textsubscript{para}. The input dimensions of our NN are smaller than the output dimensions, and there is a positive or negative correlation between input
features. That is because only the applied voltage varies depending on the application, and each FOM uses the same formula to calculate it. Furthermore, the pair of input and output data does not have a one-to-one correlation because data have the same FOM value even when the \( S_{\text{para}} \) sets are different. This is because structures of different devices have the same FOM value. Thus, the present MLP NN fails to learn as the training loss does not converge. These challenges are solved using two MLP NNs. Finally, instead of device development and optimization, our goal is to provide a mechanism to immediately identify device-side concerns during the fabrication.

III. DATA CONFIGURATION AND NEURAL NETWORK METHODOLOGY

A. Data Configuration of the FDSOI FET Device

We used the Sentaurus TCAD simulator [11], a semiconductor device simulation tool, to simulate 40,000 massive data for NN training. Using 17 parameters related to geometry and doping in the device design and manufacturing process, we collected data through random variation. The minimum and maximum ranges were assigned to each \( S_{\text{para}} \) during parameter randomization, reflecting the technology node of the device. We set the range based on the design rule of the node because we used a 14-nm device [6, 7]. First, we obtained the current–voltage (I–V) and capacitance–voltage (C–V) curves from a TCAD simulation, tested at various gate voltages, \( V_g \), depending on the application. Then, for three applications—HP, LOP, and LSTP—FOMs of on/off current ratio (I_{ratio}), delay, and power were calculated using the current, voltage, and capacitance (Eq. 1-3). Typically, while designing a good-performance semiconductor device, an operation point (Q-point), which is a target for the optimum DC performance, is initially established. Subsequently, the small-signal (AC) characteristic is optimized. Although this technique is sequential, we estimate \( S_{\text{para}} \) using the NN to assess both performances simultaneously.

\[
I_{\text{ratio}} = \frac{I_{ON}}{I_{OFF}} \quad (1)
\]

\[
\text{Delay} = \frac{C_g}{2(V_H - V_L)} \ln(I_H - I_L) \quad (2)
\]

\[
\text{Power} = V_{dd} \frac{I_H - I_L}{\ln(I_H - I_L)} + V_{dd}I_{OFF} \quad (3)
\]

The currents, \( I_{ON} \) and \( I_{OFF} \), flow when the transistor is turned on and off, respectively. The operating voltage is \( V_{dd} \), and the gate capacitance is \( C_g \). When \( V_H \) and \( V_L \) are \( V_{dd} \) and 0.5\( V_{dd} \), respectively, \( I_H \) is the extracted current. When \( V_Q \) and \( V_g \) are 0.5\( V_{dd} \) and \( V_{dd} \), respectively, \( I_L \) is the extracted current. We

![FIGURE 1. Correlation matrix computed using Pearson coefficient. (a) Figure-of-merits (FOM), as input, with positive or negative correlations to each other. (b) Structural parameters, as output, are almost independent of each other.](image-url)
used the density-gradient model and the mobility model describing doping-dependent and high-field saturation as the device physics model in TCAD simulation. Furthermore, the doping-dependent Shockley-Read-Hall model and Auger generation-recombination model were combined with band-to-band tunneling of the Hurkx model for the recombination model. In addition, the implant technique used a gaussian-function doping profile, and the gate-electrode/dielectric interface material was HfO$_2$/SiO$_2$, which has a fixed charge concentration of $10^{12}$ cm$^{-3}$.

We adopted the $S_{\text{para}}$ that can estimate the actual length at the corresponding technology node and can be manipulated in TCAD simulation. Thus, we adopted $S_{\text{para}}$ estimated from TEM image of 14-nm node FDSOI FET hardware [5]. The definition of $S_{\text{para}}$ and the range of $S_{\text{para}}$ values for data creation are shown in Table 1. The $S_{\text{para}}$ range was set in consideration of the design rule within the range that does not deviate from the technology node. In Fig. 1, a correlation matrix depicts the Pearson correlation coefficient-calculated correlation for each input feature and output. There are positive or negative correlations between input features (Fig. 1a). First, there is a positive correlation in different applications of the same FOM (solid box). Because each FOM is calculated using the same formula, and only the applied voltage varies depending on the application. Second, there is a negative correlation between $I_{\text{ratio}}$ and power (dashed box). The original correlation between two FOMs is positively correlated because they depend on the $I_{\text{on}}$. However, we take the reciprocal of $I_{\text{ratio}}$ so that it has a small value to facilitate NN training. Therefore, it appears that $I_{\text{ratio}}$ and power have a negative correlation. The output, in contrast, exhibits essentially no correlation (Fig. 1b). However, we can confirm a weak negative correlation for some $S_{\text{para}}$s because the epitaxial length ($L_{\text{sd}}$) of each area determines the maximum source- and drain-side bottom contact ($L_{\text{comb(s/d)}}$). The overall gate pitch $L_{\text{sd}}$ is set to 70 nm, and $L_{\text{sd}}$ is the same as that in Eq. 4. In other words, $L_{\text{comb(s/d)}}$ is somewhat dependent on $L_{\text{sd}}$, $R_{\text{d}}$, and $L_{\text{sp(s/d)}}$. In addition, the maximum of $L_{\text{comb(s/d)}}$ is ($L_{\text{sd}}/R_{\text{d}}$) (Eq. 5). Thus, a weak negative correlation emerges, which is unavoidable due to the 14-nm node design rule of the device.

\[
L_{\text{s/d}} = 0.5(L_{\text{tot}} - L_{\text{g}}) - R_{\text{s/d}} - L_{\text{sp(s/d)}} \tag{4}
\]

\[
\text{Max}(L_{\text{comb(s/d)}}) = L_{\text{s/d}} - R_{\text{s/d}} \tag{5}
\]

NN, particularly MLP NN, learns the relationship between input and output from the given data by treating the input as an independent variable and output as a dependent variable. However, we make up inputs as the correlated dependent variables and the outputs as the independent ones. Because the FOMs measured in the semiconductor test process depend on $S_{\text{para}}$, we use them as inputs. Furthermore, the input and output data pairs have data pairs with identical input values, implying that the data to be trained as a one-to-many relationship can produce multiple solutions for a single input. Therefore, we enable NN training using the proposed NN algorithm and correctly estimate $S_{\text{para}}$ for these circumstances, as illustrated in the next section.

**B. Decoder Neural Network Description with Pre-trained Encoder Neural Network**
We propose an approach that allows NN training when inputs and outputs are in one-to-many correspondence using the MLP (Fig. 2a). Vanilla MLP can successfully handle the problem of non-linear relationships by adding a hidden layer with weight and bias terms between the input and output layers. It learns the correlation between input and output, assuming an independent relationship between input features and a unique solution of output to input. Thus, it can overcome one of the major weaknesses of single-layer perceptron (SLP) [12], which only works with linear relationships. Furthermore, by altering the number of perceptrons for appropriate input and output, vanilla MLP can quickly solve various issues and be flexibly applied to new fields [13-15]. However, our proposed NN algorithm deals with the case where correlations exist between input features, and the solution of the input to the output is not the only solution. At this time, if we train like vanilla MLP, it fails to train due to the characteristics of our data. Therefore, we propose an algorithm consisting of two vanilla MLPs, a pre-trained encoder with a larger input dimension (Fig. 2b) and a decoder with a larger output dimension (Fig. 2c). The pre-trained encoder supports a train for the decoder, and the decoder is a core NN that estimates $S_{\text{para}}$ by receiving FOMs from each application. In general, during NN training, the NN calculates the loss value through the set loss function, and the NN is updated in the direction to minimize this value. Also, in supervised learning, where the output ground truth ($gt$) exists, the main factors of the loss function are the output value extracted by the NN and the $gt$ of the output. Therefore, the loss is calculated according to the definition of the loss function set through these two factors. However, the loss function of our decoder is newly defined through the pre-trained encoder. In other words, the pre-trained encoder is an MLP that has learned the correlation between $S_{\text{para}}$ (input) and FOMs (output) in advance, and it contributes to updating the loss function of decoder (Input-FOMs, output-$S_{\text{para}}$).

Therefore, the loss function process of the decoder is done in the following order (Eq.6): 1) The output ($S_{\text{para}}'$) estimated by receiving the FOM from the decoder is input to the pre-trained encoder. 2) The pre-trained encoder that receives $S_{\text{para}}$ estimates the FOM ($FOM_{\text{encoded}}$). 3) The loss function of the decoder is calculated using the FOM estimated from the pre-trained encoder ($FOM_{\text{encoded}}$) and the FOM of the $gt$ ($FOM_{\text{gt}}$) as the input of the decoder, then the NN is updated. Thus, the proposed NN algorithm can find solutions even for duplicate solutions and it can be applied to modeling any arbitrary nonlinear function. In addition, when the decoder estimates $S_{\text{para}}$, it uses a limiter, $g$, which ensures that each $S_{\text{para}}$ does not deviate from a pre-determined range. The limiter maps the existing range of each $S_{\text{para}}$ between -1 and 1 and corresponds to the hyperbolic tangent ($\tanh$) transfer function of the output layer. The $g^1$ function, which performs inverse operation of the limiter, is used to restore the original range of each $S_{\text{para}}$.

$$L_D = \frac{1}{N} \sum_{i=1}^{N} \left\| FOM_{gt} - FOM_{\text{encoded}} \right\|^2$$ (6)

A flow chart for training the decoder including the pre-trained encoder is shown in Fig. 3. We found the optimal dataset size needed for training empirically, which is obtained by acquiring additional data if the encoder is not properly trained. Then, re-training is performed by tuning the network hyper-parameters of both the decoder and the pre-trained encoder if training fails during the modeling phase. Finally, the modeled NN is evaluated through the $R$ coefficient of the regression plot and the percent error calculated from the estimated value. We used the following network hyper-parameters. First, common to both the decoder and the pre-trained encoder, the dataset is partitioned into training, validation, and test sets at ratios of 0.80, 0.10, and 0.10, respectively. The transfer function of the hidden layer is implemented using the $\tanh$ function. The mean squared error (MSE) between the output and target values is used as a function to reduce the training loss. In addition, the log-scale was applied to $S_{\text{para}}$ related to doping and FOMs to prevent NN training failure due to large-scale differences between input and output values. Second, the pre-trained encoder comprises 9-MLP NNs for each FOM, and the identical network hyper-parameters are applied across them. For training, we adopted the Levenberg-Marquardt (LM) optimizer [16-18] to solve the non-linear least-squares problem. There were fifty hidden layers, and the transfer function of the output layer was linear. Third, the decoder used a resilient back-propagation ($R_{\text{prop}}$) train optimizer [19], one of the fastest weight update mechanisms available. In most cases, the hidden layer uses a sigmoid or $\tanh$ transfer function, whose slope approaches zero when the input value becomes very large or small. While utilizing the steepest descent method to train a multi-layer NN, the magnitude of the gradient may be quite small when updating the gradient. Thus, the change amount is minimal when the weight and bias are far from optimal. Thus, during network training, the problem of updating the gradient in an undesirable direction may arise.
The trained encoder for successful decoder modeling. The training guarantee encoder influences the performance of the modeled decoder. When the decoder is trained, the pre-trained encoder and the decoder, as well as the \( S_{\text{pars}} \) found using the modeled decoder.

### IV. RESULTS AND DISCUSSION

#### A. Neural Network Model Performance Evaluation

When the decoder is trained, the pre-trained encoder is modeled to update the network performance by minimizing the loss function. Because the performance of the pre-trained encoder influences the performance of the decoder, it must guarantee network performance and reliability of the pre-trained encoder for successful decoder modeling. The training results of the pre-trained encoder are shown in Tables 2 and 3.

Table 2 shows the regression coefficient, \( R \), of the regression line for the training, validation, and test datasets and the MSE training loss for each FOM through the pre-trained encoder.

#### TABLE 2. \( R \) coefficients for training, validation, and test datasets and MSE training loss for each FOM through the pre-trained encoder.

| FOMs       | Training | Validation | Test  |
|------------|----------|------------|-------|
| Off/on current ratio | 0.99627  | 0.99622    | 0.99588  | 7.39E-3 |
|             | 0.99854  | 0.99845    | 0.99843  | 2.90E-3 |
|             | 0.99883  | 0.99874    | 0.99871  | 2.33E-3 |
| Delay      | 0.99126  | 0.99100    | 0.99017  | 1.73E-2 |
|             | 0.99777  | 0.99758    | 0.99733  | 4.46E-3 |
|             | 0.99902  | 0.99890    | 0.99889  | 1.94E-3 |
| Power      | 0.99918  | 0.99908    | 0.99905  | 1.64E-3 |
|             | 0.99910  | 0.99901    | 0.99897  | 1.80E-3 |
|             | 0.99934  | 0.99922    | 0.99923  | 1.31E-3 |

Table 3 shows the average and minimum percent error values of each FOM estimated through the pre-trained encoder for the training and test datasets.

#### TABLE 3. Average, maximum, and minimum values of the percent error of each FOM estimated through the pre-trained encoder for the training and test datasets.

| Percent error [%] | Training | Test |
|-------------------|----------|------|
| Off/on current ratio | Avg. | Min. | Avg. | Min. |
| HP                | 0.70     | 1.7E-5 | 0.71 | 1.4E-3 |
| LOP               | 0.43     | 1.5E-5 | 0.44 | 1.2E-4 |
| LSTP              | 0.54     | 1.6E-5 | 0.53 | 3.0E-4 |
| Delay             | Avg.     | Min. | Avg. | Min. |
| HP                | 1.01     | 5.5E-5 | 0.99 | 2.1E-4 |
| LOP               | 0.64     | 3.6E-5 | 0.64 | 2.2E-4 |
| LSTP              | 0.76     | 5.3E-5 | 0.77 | 7.4E-4 |
| Power             | Avg.     | Min. | Avg. | Min. |
| HP                | 0.29     | 1.1E-6 | 0.29 | 1.3E-5 |
| LOP               | 0.32     | 1.1E-6 | 0.32 | 3.4E-4 |
| LSTP              | 0.52     | 2.3E-5 | 0.53 | 4.0E-4 |

The \( R_{\text{prop}} \) optimizer eliminates the magnitude of the negative influence of the partial derivative. Therefore, the magnitude of the derivative does not affect the process of updating the weight. The direction of updating the weight to minimize the loss function is determined solely by the derivative sign. Thus, the weight change is halved when the sign changes and the gradient progresses from one iteration to the next. When the sign does not change, however, it increases by 1.2 times. If the slope is zero, the same updated value is maintained. Furthermore, the weight change diminishes with each vibration of the weight. Weight changes increase as the weight shifts in the same direction for multiple iterations. The decoder has 100 hidden layers, and the transfer function of the output layer, unlike the pre-trained encoder, uses the \( \tanh \) function to reflect the result of the limiter. We also set the minimum performance gradient to \( 10^{-5} \) to avoid over-fitting in training. The number of validation checks which were repeated continuously with degradation, was set to five. Thus, during training, we specified these two stop conditions to pursue network generalization. In the next section, we show the training results of the pre-trained encoder and the decoder, as well as the \( S_{\text{pars}} \) found using the modeled decoder.

#### FIGURE 4. (a) Loss reduction with increasing epochs on training, validation, and test datasets. (b) Reduction of network performance gradient and (c) the number of validation checks, which are conditions for early stopping in the training process.
reduced until training was stopped. As the epoch approached 3000, the loss began to converge. The loss on the test dataset (red line) was similar to that on the training and validation datasets. Thus, no over- or under-fitting occurred because the loss for validation does not increase or decrease in comparison to training. In addition, as the epoch rises, the network performance gradient declines progressively, and the gradient becomes saturated when the loss reaches saturation in Fig. 4(b). The best validation performance at this time was at the 3904th epoch. The validation checks surpassed the training stop condition. Hence the final epoch was halted at the 3909th epoch (Fig. 4c).

B. Analysis of Structural Parameters Estimated Through the Decoder Neural Network

Table. 4 compares the minimum and maximum of the NN estimation result with the ground truth. We can validate that all \( S_{para} \) values are within the range of the minimum and maximum values of the ground truth. Even though our goal is not device optimization, it is distinct from the related \( S_{para} \) estimation study. The \( S_{para} \) was calculated within the range of the real dataset by the limiter, unlike the study of Choi et al. [2], in which the best solution was found along the full hypersurface of the trained NN without the constraint of the range of permitted solutions. We do not intend to find a case outside the technology node of the target device. Thus, within a given technology node, we can successfully estimate \( S_{para} \) that satisfies the FOMs of the required application. Fig. 5 shows the percent error of FOMs extracted using the estimated \( S_{para} \) to ensure that the decoder results are reliable. In all FOMs, the average error is 0.1 %, and the highest error is not more than 1 %.

The trained NN deals with 17 \( S_{para} \)S for each of the 9 cases (3 FOMs for each of the 3 applications). Therefore, it is not easy to show \( S_{para} \)’s analysis for all cases due to space issues, so we selected a specific FOM, delay, for a specific application, the LOP application as an example. The selected \( S_{para} \)S are parameters that directly affect the delay, and device analysis was performed through them. Note that our NN results are not optimizations of semiconductor devices. Instead, our goal is to find the device \( S_{para} \) of a larger dimension when input FOM of a small dimension for each application. Thus, the NN result does not estimate one \( S_{para} \) set for one FOM value but can quickly find several \( S_{para} \) sets that satisfy the desired FOM. For the LOP application, the boxplots of the selected \( S_{para} \)

| Structural parameter [nm] | Minimum | Maximum |
|---------------------------|---------|---------|
| \( L_g \) | 20.2 | 20.8 |
| \( L_{res} \) | 8.6 | 8.8 |
| \( L_{app} \) | 4.9 | 4.9 |
| \( L_{spd} \) | 4.6 | 4.9 |
| \( T_{soi} \) | 3.6 | 3.5 |
| \( T_{res} \) | 13.8 | 10 |
| \( T_{id} \) | 10 | 10 |
| \( T_{cd} \) | 6.1 | 9.9 |
| \( N_{ch} \) | \( 10^{10} \) | \( 10^{10} \) |
| \( N_{gd} \) | \( 10^{-11} \) | \( 10^{-11} \) |
| \( N_{gd} \) | \( 3.61 \times 10^{-10} \) | \( 1.99 \times 10^{-10} \) |
| \( H_{gb} \) | 5.1 | 5 |
| \( L_{adj} \) | 3.1 | 11.3 |
| \( L_{soi} \) | 3 | 12 |
| \( R_{cd} \) | 1 | 3 |
| \( L_{cnd} \) | 7.1 | 7 |
| \( L_{comb} \) | 7 | 18 |

FIGURE 5. The percent error extracted when the \( S_{para} \) estimated by the decoder is the input to the pre-trained encoder.

FIGURE 6. A boxplot expressing values belonging to the FOM of upper 5 %, upper 10 %, and lower 5 % for some \( S_{para} \)S.
upper 5 % and upper 10 %, each $S_{\text{para}}$ implies the target criteria for the design it should have, considering the correlation with other $S_{\text{para}}$. Tendencies of the individual $S_{\text{para}}$ values can be observed as the boundary becomes larger. Through parameter splitting, this work can reduce the load of analyzing the influence of the related $S_{\text{para}}$. Furthermore, if the vertical range for each $S_{\text{para}}$ is narrow, it must be carefully controlled during device design. This implies that the design margin is rather large in the opposite case. Second, when comparing the upper 5 % and lower 5 %, we can confirm that $S_{\text{para}}$ has opposite tendencies or overlaps with the upper and lower cases. Therefore, when actual measurements result in unwanted FOM values such as lower 5 %, these FOM values can be entered into the trained decoder. Then, the NN can determine which $S_{\text{para}}$ is abnormal by estimating the $S_{\text{para}}$ set corresponding to the FOM value. This allows for immediate feedback on the unit process associated with the abnormal $S_{\text{para}}$. It can also help with device design considerations that should be avoided.

Table 5 shows the median values of some $S_{\text{para}}$s for semiconductor analysis on the delay of LOP application. Delay is affected by gate capacitance ($C_g$) and effective current ($I_{\text{eff}}$), and a small value is required to enable high-speed operation (Eq. 2). Therefore, five $S_{\text{para}}$s to be analyzed in terms of semiconductor technology were selected as examples. First, we selected the $L_g$, $T_{\text{soi}}$, and $L_{\text{adj}}$, mainly affecting $C_g$ and $I_{\text{eff}}$. Second, with $L_{\text{adj}}$, we show the impressive results found by NN, usually not picked up by human experts. In devices small enough to show the short channel effect (SCE), an increase in gate length ($L_g$) decreases the area where the source/drain-substrate depletion region penetrates the channel. Thus, gate controllability increases, SCE minimizes, and $I_{\text{eff}}$ increases. Also, the $C_g$ increases due to the wider gate area at fixed oxide thickness. In Fig. 7, in devices where SCE occurs, an increase in $L_g$ affects the increase in $I_{\text{eff}}$ more than an increase in $C_g$. In general, it is attempted to decrease $L_g$ to reduce device size, but it shows that $L_g$ needs to be increased to decrease the delay within the range set from 20 to 26 nm.

Therefore, the NN estimates a large $L_g$ small, and the trained NN estimated large $L_g$ and small $T_{\text{soi}}$ to achieve the small delay.
thus, increasing $I_{\text{eff}}$. Also, it causes an increase in $C_g$. At this time, the change in $I_{\text{eff}}$ is more sensitive than the change in $C_g$, which is more pronounced when $L_g$ is small (Fig. 8). Therefore, a thin $T_{\text{tot}}$ is required to obtain a good delay characteristic (small value), which agrees with the NN results that estimate a low $T_{\text{tot}}$ value in the upper case and a high $T_{\text{tot}}$ value in the lower case. Fig. 9 shows the delay characteristics according to $L_g$ and $T_{\text{tot}}$. When $T_{\text{tot}}$ is the minimum value, it has a fairly small delay value regardless of $L_g$. However, when $L_g$ is relatively large, the change in delay according to $T_{\text{tot}}$ is less sensitive due to the improved SCE. Therefore, the NN estimates large $L_g$ and small $T_{\text{tot}}$ to find $S_{\text{para}}$ satisfying good delay characteristics.

Drain-side spacer length ($L_{\text{spd}}$) was estimated to have a large value for the upper 5 % case and a small value for the lower 5 % case. Fig. 10 shows the delay and $I_{\text{eff}}$ as a function of $L_{\text{spd}}$. When $L_{\text{spd}}$ increases, doping at the drain moves away from the gate edge, so the parasitic fringing capacitance decreases, and thus $C_g$ decreases. In addition, as $L_{\text{spd}}$ increases, the series resistance increases due to the extension of the gate underlap, resulting in a linear decrease in $I_{\text{eff}}$. However, as effective channel length increases, SCE such as DIBL and SS is minimized, and $I_{\text{eff}}$ decreases exponentially, resulting in an $I_{\text{eff}}$ increase. Therefore, delay reduction can be achieved due to a decrease in $C_g$ and an increase in $I_{\text{eff}}$. Thus, it can be seen that the trained NN adopts a large $L_{\text{spd}}$ to satisfy a small delay value and a small $L_{\text{spd}}$ value to satisfy a large delay value.

Note that our NN estimation result is not an optimization process to find the best device, and the semiconductor physics is not reflected in the NN training process. Therefore, $L_{\text{qps}}$ shows that the NN estimation results are different from the human expert selection, and $L_{\text{adj}}$ shows that the numerically found results by the NN do not necessarily have maximum or minimum values. For a device symmetry, $L_{\text{qps}}$ and $L_{\text{qpd}}$, spacer lengths in the source and drain regions, are designed to have the same value. In addition, $L_{\text{qps}}$ and $L_{\text{qpd}}$ have the same effect on the device. Especially, $L_{\text{qpd}}$ determines the overlap length in the source and drain regions and greatly affects SCE. Therefore, the NN will estimate large $L_{\text{g}}$ and $L_{\text{qpd}}$ for good delay characteristics and low for poor. However, since the NN finds a solution that satisfies the proposed NN without learning the physical mechanism of the semiconductor, different values of $L_{\text{qps}}$ and $L_{\text{qpd}}$ were estimated, as shown in Table. 5. In particular, the difference between $L_{\text{qps}}$ and $L_{\text{qpd}}$ was more than 2 nm in the lower case, and $L_{\text{qps}}$ was estimated to be relatively high. When $L_{\text{qps}}$ and $L_{\text{qpd}}$ have the same low value, the extracted delay is out of the range of the ground truth due to severe $I_{\text{eff}}$ degradation. Therefore, the NN cannot estimate the delay within the set range, and $L_{\text{qps}}$ is estimated to be relatively high to satisfy the lower 5 % case. Since a high bias is applied to the drain, the delay has a larger change rate in $L_{\text{spd}}$ than $L_{\text{qps}}$ when the spacer length increases (Fig. 11). Thus, NN satisfies the ground truth by slightly increasing the $L_{\text{qps}}$, having relatively little change rate. As a result, in the lower 5 % case, the difference between $L_{\text{qps}}$ and $L_{\text{qpd}}$ occurs, which $L_{\text{qps}}$ has a slightly higher value. This is a different result from the human expert designing symmetrically with the same values of $L_{\text{qps}}$ and $L_{\text{qpd}}$. As such, our NN can present a variety of design perspectives by providing options for $S_{\text{para}}$ that are not normally selected.

$L_{\text{adj}}$ is the junction gradient, which means the distance at which S/D doping is 1/10 from the peak. As $L_{\text{adj}}$ increases, the effective doping concentration in the channel increases, and the S/D resistance decreases. Therefore, a suitably large $L_{\text{adj}}$ will achieve a small delay value. Therefore, $L_{\text{adj}}$ was estimated to be relatively higher than the lower 5 % case in the upper 5 % case. However, values above 9 nm are not estimated (Table. 5). Because, as a result of TCAD simulation according to $L_{\text{adj}}$ with other $S_{\text{para}}$s fixed as median values, the extracted delay exceeds the delay of ground truth when $L_{\text{adj}}$ is 9 nm or more. Therefore, the delay through the $L_{\text{adj}}$ in the non-estimated interval does not satisfy the ground truth, so the NN did not estimate the $L_{\text{adj}}$ for this interval. In other words, the estimation result through the numerical correlation between input and output does not always become the maximum or minimum of the range set. In addition, although NN does not learn the physical mechanism, it can confirm that the semiconductor mechanism is reflected through the ground truth that cannot
find due to the degradation caused by a specific \( S_{\text{para}} \). Actually, \( L_{\text{adj}} \) is controlled according to the annealing temperature and time in the actual process, but \( L_{\text{adj}} \) in FDSOI FET has a small value, and the value itself is not a parameter that changes as much as the set range. That is, the amount of change in \( L_{\text{adj}} \) is significantly less than that of other \( S_{\text{para}} \) changes. We gave an example of semiconductor analysis through 5 \( S_{\text{para}} \), but similarly, detailed analysis of semiconductor aspects can be applied to other \( S_{\text{para}} \) as well.

Our method can estimate the \( S_{\text{para}} \) set in different scenarios for the same FOM value (duplicate input values). Therefore, \( S_{\text{para}} \) offers numerous design alternatives to satisfy the desired conditions. The design and production of semiconductor devices is a conservative process. Thus, if unfavorable results are produced, semiconductor engineers frequently try to solve the problem by altering the cause. However, using the proposed method to rapidly and diversely provide the range of the corresponding \( S_{\text{para}} \) to satisfy the desired FOM, makes a design perspective over a wider range possible.

V. CONCLUSION

We proposed an NN algorithm using two MLP NNs to estimate the \( S_{\text{para}} \) affecting the device design and unit process of 14-nm node FDSOI FETs. The NN input is a set of FOMs with smaller dimensions than the \( S_{\text{para}} \) output. In addition, a correlation exists between the input features and duplicate input values, which overcame the problem of non-convergence and enabled NN training. For all FOMs, the pre-trained encoder used to calculate the loss function for convergence is trained with an R value close to 1. Furthermore, in both the training and test datasets, the percent errors from the actual value show average values of 1 % or less. The encoder was used to train the decoder, and the training loss of the decoder fell in line with the validation loss without over- or under-fitting. Thus, 17 \( S_{\text{para}} \) values were successfully estimated within the range specified by the 14-nm technology node. The percent errors of the decoder show averages of 0.1 % after inputting the estimated \( S_{\text{para}} \) into the pre-trained encoder. The parameter trend can be confirmed through the \( S_{\text{para}} \) that satisfies the FOM values belonging to the upper 5 % and 10 %. In addition, the \( S_{\text{para}} \) estimated from the duplicate inputs provides a different set of optional \( S_{\text{para}} \). If an abnormal FOM value is derived, as in the case of the lower 5 %, the corresponding \( S_{\text{para}} \) can be derived, and feedback on the unit process corresponding to the abnormal \( S_{\text{para}} \) is available. Therefore, the cause of failure on the device side can be immediately identified using the proposed NN algorithm in the semiconductor test process. In addition, we performed a detailed physical analysis as an example of a delay in LOP application. NN estimation results were analyzed using \( I_{\text{gs}} \), \( T_{\text{soi}} \), and \( L_{\text{adj}} \), which mainly affect \( C_{\text{g}} \) and \( I_{\text{ds}} \). \( L_{\text{adj}} \) showed behaviors different from those generally selected by human experts and cases where maximal values were not estimated within the set range. Our methodology can improve the inspection speed and yield during the test process by aggregating the estimated \( S_{\text{para}} \) values and spotting trends. This is more stable (non-destructive inspection) and economical compared to the existing methods (TEM or SEM image, destructive inspection) to extract the \( S_{\text{para}} \) of the manufactured wafer or chip. Furthermore, since our proposed method learns the relationship between input and output, the artificial neural network does not learn the physical phenomena of the device or the side effects that may occur when scaling. Therefore, this can be applied regardless of the type of device or technology node. Moreover, it is applicable to arbitrary non-linear function modeling. Finally, the proposed NN algorithm can be applied to various tasks in the semiconductor manufacturing process, including estimating and analyzing any systems with more outputs than inputs.

REFERENCES

[1] J. Schmidhuber, “Deep learning in neural networks: An overview,” Neural Networks, vol. 61, pp. 85-117, 2015.
[2] H.-C. Choi, H. Yun, J.-S. Yoon and R.-H. Baek, “Neural approach for modeling and optimizing Si-MOSFET manufacturing,” IEEE Access, vol. 8, pp. 159351-159370, 2020, doi: 10.1109/ACCESS.2020.3019933.
[3] H. Yun, J. Yoon, J. Jeong, S. Lee, H. Choi and R. Baek, “Neural network based design optimization of 14-nm node fully-depleted SOI FET for SoC and 3DIC applications,” 2020 4th IEEE Electron Devices Technology & Manufacturing Conference (EDTM), pp. 1-4, 2020, doi: 10.1109/EDTM47692.2020.9117935.
[4] J.-S. Yoon, S. Lee, H. Yun and R.-H. Baek, “Digital/Analog Performance Optimization of Vertical Nanowire FETs Using Machine Learning,” IEEE Access, vol. 9, pp. 29071-29077, 2021, doi: 10.1109/ACCESS.2021.3059475.
[5] Q. Liu, M. Venet, J. Gimbert, N. Loubet, R. Wacquez, L. Grenouillet, Y. Le Tiec, A. Khakhifrozoo, T. Nagumo, K. Cheng, H. Kothari, D. Chanemougeme, F. Chafik, S. Guillaumet, J. Kuss, F. Allibert, G. Tsutoi, J. Li, P. Morin, S. Mehta, R. Johnson, L.F. Edge, S. Pono, T. Levin, S. Kanakasabapathy, B. Hanan, H. Bu, J.-L. Bataillon, O. Weber, O. Faynot, E. Josse, M. Haond, W. Kleemeier, M. Khare, T. Skotnicki, S. Luning, B. Doris, M. Celik, and R. Sampson, “High performance UTBB FDSOI devices featuring 20nm gate length for 14nm and beyond,” 2013 IEEE International Electron Devices Meeting, pp. 9.2.1-9.2.4, 2013, doi: 10.1109/IEDM.2013.6724592.
[6] O. Weber, E. Josse, F. Andreu, A. Cross, E. Richard, P. Perreau, E. Baylac, N. Degors, C. Gallon, E. Perrin, S. Chhun, E. Petitprez, D. Delacroix, J. Simond, J. Mazurier, N. G. Druais, S. Lasserre, J. Mazurier, N. Ginot, E. Bernard, R. Bianchini, L. Parmigiani, X. Gerard, C. Pribat, O. Gourhant, F. Abbate, C. Gaumer, V. Beugin, P. Gouarda, P. Maury, S. Lagrasta, D. Barge, N. Loubet, R. Beneyton, D. Benoît, S. Zoll, J.-D. Chapon, L. Babaud, M. Bidaud, M. Gregoire, C. Monget, B. Le-Gratiet, P. Brun, M. Mellier, A. Polfelski, L.R. Clement, R. Bingert, S. Puget, J.-F. Kruck, D. Hoguet, P. Scheer, T. Poiroux, J.-P. Manceau, M. Rafik, D. Rideau, M.-A. Jaud, J. Lacord, F. Monsieur, L. Grenouillet, M. Venet, Q. Liu, B. Doris, M. Celik, S.P. Fetterolf, O. Faynot, and M. Haond, “14nm FDSOI technology for high speed and energy efficient applications,” 2014 Symposium on VLSI Technology (VLSI-Technology): Digest of Technical Papers, pp. 1-2, 2014, doi: 10.1109/VLSIT.2014.6894343.
[7] O. Weber, E. Josse, J. Mazurier, N. Degors, S. Chhun, P. Maury, S. Lagrasta, D. Barge, J.-P. Manceau, and M. Haond, “14nm FDSOI upgraded device performance for ultra-low voltage operation,” 2015 Symposium on VLSI Technology (VLSI-Technology), pp. T168-T169, 2015, doi: 10.1109/VLSIT.2015.7223564.
[8] S. Monfray, and T. Skotnicki, “UTBB FDSOI: Evolution and opportunities,” 2016 Solid-State Electronics, vol. 124, pp. 63-72, 2016, doi: 10.1016/j.sse.2016.07.003.
[9] K. I. Funahashi, “On the approximate realization of continuous mappings by neural networks,” Neural Networks, vol. 2, no. 3, pp. 183-192, 1989, doi: 10.1016/0893-6080(89)90003-8.
[10] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: a new learning scheme of feedforward neural networks,” 2004 IEEE
International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541), vol.2, pp. 985-990, 2004, doi: 10.1109/IJCNN.2004.1380068.

[11] Synopsys, Inc., Mountain View, CA, Version N-2017.09.

[12] Š. Raudys, “Evolution and generalization of a single neurone: I. Single-layer perceptron as seven statistical classifiers,” Neural Networks, vol. 11, no. 2, pp. 283-296, 1998, doi: 10.1016/S0893-6080(97)00135-4.

[13] U. Orhan, M. Hekim, and M. Ozer, “EEG signals classification using the K-means clustering and a multilayer perceptron neural network model,” Expert Systems with Applications, 2011, 38.10: 13475-13481.

[14] R. Velo, P. López, and F. Maseda, “Wind speed estimation using multilayer perceptron,” Energy Conversion and Management, 2014, 81: 1-9.

[15] CO. Sakar, SO. Polat, M. Katircioglu, and Y. Kastro, “Real-time prediction of online shoppers’ purchasing intention using multilayer perceptron and LSTM recurrent neural networks,” Neural Computing and Applications, 2019, 31.10: 6893-6908.

[16] M. T. Hagan, and M. B. Menhaj, “Training feedforward networks with the Marquardt algorithm,” IEEE Transactions on Neural Networks, vol. 5, no. 6, pp. 989-993, Nov. 1994, doi: 10.1109/72.329097.

[17] A. Ranganathan, “The levenberg-marquardt algorithm,” Tutorial on LM algorithm, 2004, 11.1: 101-110.

[18] H. Yu and B. M. Wilamowski, “Levenberg–marquardt training,” In: Intelligent systems, CRC Press, 2018. p. 12-1-12-16.

[19] M. Riedmiller, and H. Braun, “A direct adaptive method for faster backpropagation learning: the RPROP algorithm,” IEEE International Conference on Neural Networks, vol. 1, pp. 586-591, 1993, doi: 10.1109/ICNN.1993.298623.