TCAD-enabled Machine Learning – An Efficient Framework to Build Highly Accurate and Reliable Models for Semiconductor Technology Development and Fabrication

Paul Jungmann*†, Jeffrey B. Johnson* *Senior Member, IEEE,*, Eduardo C. Silva*, William Taylor*, Abdul Hanan Khan*, Akash Kumar† *Senior Member, IEEE,*
*GlobalFoundries Inc.
†Chair For Processor Design, Center for Advancing Electronics Dresden (CfAED)

***Abstract*—The requirements on data-driven Machine Learning models for industrial applications are often stricter, compared to those used for academic purposes, as model reliability is critical in industrial environments. Herein is introduced a framework which enables automated data generation with the goal of efficiently providing a data set sufficient to build a reliable and actionable model. Essential to this framework is the placement of the model training/testing data points, which need to be well distributed across the defined input parameter space. The framework is applied to semiconductor fabrication, wherein TCAD, a set of simulation tools that reproduce the physical processing and the final electrical performance of semiconductor devices, is a well-established capability. Transistor-level processing data is reproduced with TCAD simulations, from which the Machine Learning model is built. The framework described here assures that the resulting Machine Learning model fulfills the accuracy requirements across the parameter space. As an example application, the final Machine Learning model is then used to modify the process for a transistor, to obtain both better electrical performance and reduced variability.**

***Index Terms*—TCAD, Technology Computer-Aided Design, TCAD-eML, TCAD-enabled Machine Learning, Reliability, Fabrication Capabilities**

I. Introduction

Technology computer-aided design (TCAD) is finding increasing use as an enabling technology for applying Machine Learning (ML) in the semiconductor industry [1]–[3]. To date, the application of ML has focused on the research and development side of the industry, where semiconductor technology, in particular core devices such as transistors, are engineered through a limited series of experiments to achieve the desired performance metrics [3]–[8]. The role of TCAD-enabled ML has yet to be articulated and demonstrated for the fabrication side of the microelectronics industry [4], [9], [10]. The work herein is intended to fill this gap, the scope is to generate sufficient data to build reliable models.

Compared to technology development, which focuses on device centering, semiconductor fabrication is focused on yield [11]–[13]. Since any reasonably complex integrated circuit will contain millions, if not billions, of semiconductor devices [14], the possible range of device variability can be expected to be completely exercised on every chip. Hence, to be applicable in real-world use cases a prerequisite for Machine Learning models is to be highly accurate and reliable over an entire input space. Unfortunately, a large part of the related publications do not focus on even one of these points. In [15] the requirement of a quick turn-around-time is added. To fulfill this prerequisite, validation samples must be essentially 'everywhere' in the input space. If an outlier is outside of the validation space, the model will not yield correct results, which leads to unreliable conclusions in later analysis or misadjustments of the processing parameters.

In this work, a framework is developed which can provide the model training data which fulfill the requirements for the application to modern semiconductor fabrication. In parallel it is shown how the framework can be modified to focus on more important sub-regions of the input space first. This allows an efficient use when the simulation resources are limited. Importance of a sub-region is not only connected to how often corresponding samples occur in the real-world use case. Often especially the outliers are of special interest, what highlights the emphasis of this work: reliability across the defined input space.

Although this publication focuses on TCAD-enabled Machine Learning the presented methodology generalizes to other design space exploration problems.

Manuscript received August 22, 2022; revised Month Day, Year.
This work has been supported by funding from the Free State of Saxony and the European Fonds for Regional Development (EFRE) under Project ARAMID, No 100375240.

Corresponding author: Paul Jungmann
Paul Jungmann is with GlobalFoundries, 01109 Dresden, Germany, the Chair For Processor Design and the Center for Advancing Electronics Dresden (CfAED), Technische Universität Dresden, 01062 Dresden, Germany (e-mail: paul.jungmann@tu-dresden.de)
Jeffrey B. Johnson, Eduardo C. Silva, William Taylor and Abdul Hanan Khan are with GlobalFoundries, Inc. Malta, NY, USA (e-mail: jeffrey.johnson@gf.com; eduardo.silva@gf.com; william.taylor@gf.com, abdulhanan.khan@gf.com)
Akash Kumar is with the Chair For Processor Design and Germany and the Center for Advancing Electronics Dresden (CfAED), Technische Universität Dresden, 01062 Dresden (e-mail: akash.kumar@tu-dresden.com)

A. Technology CAD

Technology computer-aided design (TCAD) has a history almost as long as that of solid-state microelectronics. It consists essentially of two parts [16]: 1) Process simulation, which simulates the many process steps involved in creating modern integrated circuits and 2) device simulation, which subsequently simulates the electrical characteristics of the devices. With increasing computational power has come increasingly sophisticated numerical approaches applied to both process and device simulation [17]. However at the industrial level, in both technology development and fabrication, a continuum approach dominates since the requisite level of physical accuracy can be achieved with an acceptable computational burden. Hence, the work discussed herein will be based on industry-standard continuum process and device simulation tools.

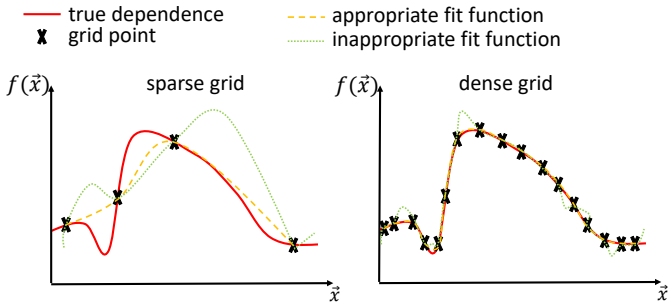

Fig. 1. Simplified visualization of fit functions $f(.)$ (ML-Models) on the training data space $\{\vec{x}\}$. The plots show the influence of the grid density on the fit function quality. In the case of a sparse grid, not only can an appropriate fit function fail to capture the true dependence but also the occurrence of inappropriate fit functions is more probable. Especially in the case of continuous fit functions an approximation based on a dense enough grid will hardly show large deviations from the truth (inappropriate fit). A very dense grid may even challenge the need for a validation and test set.

B. TCAD-enabled Machine Learning

TCAD-enabled Machine Learning carries the idea of Physics Informed Artificial Intelligence [1] into semiconductor fabrication [2], [3], [18]. The idea remains unchanged: A Regression Machine Learning model is trained to reproduce a physical dependence. TCAD Simulations are used to produce correctly labelled data, where geometry-, material-, and process-parameters are features and electrical parameters are labels.

One of the issues when working with regression problems is overfitting, in which the noise in data becomes part of the model due to an overzealous fitting [19]. Training a model on TCAD data is efficient because there is essentially no noise (very small aleatoric uncertainty) on the features and labels [20], so overfitting becomes less problematic (but not unimportant) and comparably smaller sets of training data can yield good model performance with small epistemic uncertainty. More insight about overfitting in TCAD-enabled Machine Learning may be found in [21].

TCAD-enabled Machine Learning has advantages beyond very high quality training data. The fact that the training data is based on well-understood semiconductor process and device physics allows device experts to refine models and training data using experience from semiconductor technology development.

Exploiting domain knowledge in data science problems creates opportunities for model training and evaluation because the meaning of single features and their interactions is well understood [22], [23]. In comparison to classification problems, say in image detection, the mathematical and physical meaning of the feature space or points in that space are rather clear. With a dense enough multi-dimensional grid in that space as training data, and adequate training time, it is unlikely that a model will generate inaccurate predictions. See [24] chapter 6.5 and compare Figure 1.

This is especially true for regression models which are represented by a continuous function (say a Neural Network), due to their high interpolation capability. With that in mind, creating a model with low epistemic uncertainty comes down to creating a sufficiently dense multi-dimensional grid.

With an increasing number of features, the number of simulations needed to create a full-factorial style grid becomes computationally prohibitive. The solution is to approximate the grid with Design of Experiment (DoE) techniques. Works like [25] and [4] use commercial structure generators [26] to distribute feature vectors in the feature space. This is a solid basis for a TCAD-enabled ML model, but care has to be taken, since the positioning of the feature vectors is independent of the model performance and use case. First approaches to include domain expertise are also shown in [25]. While there have been previous frameworks for an automated data generation [15], [27], which are connected to the ML model, they target different kinds of uses. In [27] a framework is presented that focuses on creating an input which yields a device which is as different as possible to the existing devices. This leads to a coverage of the design space adapted to the variability of the device. It is therefore suited for smaller data sets or experiments with many parameters. This method does not fit the vision of this publication, as the goal is to have an accurate model which is reliable on the whole defined input space. [15] presents a framework that estimates uncertainties in addition to the predictions and generates additional TCAD data for a model update, if the uncertainty is insufficient. With that it adapts/improves based on the given design tasks. This framework is very efficient for the data generation for specific tasks, but it may not fulfill the ab initio reliability requirement in a given feature/design space.

In the following sections a methodology is presented which approximates the grid very efficiently with respect to the model performance criteria. Setting up a data set and TCAD enabled ML model to reliably full-filling these performance criteria everywhere in the feature space is the major contribution of this work. A minor, but still valuable, contribution is the inclusion of domain expertise to improve the data generation. However, one should make clear, that it is not critical how the model is built or which type of model is used. The developed methodology sets up these models using as few as possible, but well chosen, training data.

The outcome of the methodology developed here provides

the capability of engineering not only the target value, or centering, of a semiconductor device but also its variability. By efficiently and accurately explaining device variability over the complete design space, this implementation of TCAD-enabled Machine Learning models provides capabilities to improve semiconductor development and fabrication in many tangible ways:

1) Device centering.
2) Six-Sigma process variability optimization.
3) Generation and optimization of model corners [28].
4) Engineering fabrication process variability by evaluating alternative centering points and variability limits.
5) Providing fundamental explanations of outliers.

This paper is structured as follows. Section II introduces the concept of the model generation process and Section III explains the requirements on the sampling method and which method is used for that. In Section IV detailed information is given regarding the partitioned model evaluation process. Sections II - IV represent the emphasis of this work. Finally, Section V presents the application of the framework on a real TCAD device, discusses major influences throughout the data generation and model build, and shows an example use case related to semiconductor fabrication.

II. Concept

The basic idea to train a low epistemic uncertainty Machine Learning model is an iterative data generation approach [29]–[32]. Figure 2 visualizes the concept implemented and used in this work. The starting point is a TCAD simulation setup (TCAD Deck) and a (small) data set which gives information about the minimal and maximal values each feature can take (Feature Ranges). In each loop iteration the available data are evaluated. This evaluation contains a density analysis in the feature space as well as building a model on the available data and evaluating its performance. The data and the model are evaluated in disjoint sub-regions of the feature space. With the extracted information it is decided which additional data points (Feature-Label pairs) are most valuable. In the last iteration step these very points are simulated and the loop starts over. In order to better refine the generation of new training data, feature importance values are taken into account. When the model is evaluated as sufficient (i.e. it meets stopping criteria), the model is ready to use.

III. Data Generation

Generating new inputs for the simulation is a key point for the presented concept. In Section IV it is explained in detail how to find the regions in the parameter space in which the model is least reliable. Here that means both, model accuracy in the region and availability of training/test data in the region. The next step is the actual generation of new inputs. This corresponds to a traditional DoE problem: How are new data points to be distributed to gain maximal information from the experiments?

To extract most information from new samples, they should be distributed uniformly. In addition to that, it is important to have randomness in the data generation. That avoids unnecessary simulations when a specific region is identified again as the worst region, because when new samples lie on top of old ones no new information are gained.

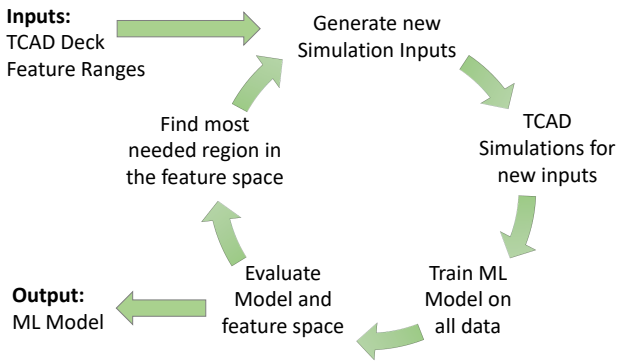

Fig. 2. Loop for efficiently training TCAD-enabled Machine Learning models. With a calibrated TCAD simulation set up (Deck) a new data point can be generated with exactly this set of parameters as needed. By analyzing the available data and the model performance based on that very training data, new simulation inputs are suggested.

A well known and established method to generate data under these circumstances is *Latin Hyper-Cube Sampling* [33], [34] and was used in this work. Especially useful is the generation of equally distributed samples even if their amount is low. Further details about this sampling method can be found in the literature.

IV. Partitioned Model Evaluation

The model evaluation for the proposed methodology needs to provide one outcome: These regions, in the feature space, have to be identified which yield poor(est) model performance.

Before developing the partitioned model evaluation, a definition of "poor model performance in regions of the feature space" has to be given. In this work "poor" stands for low reliability. This not only includes the accuracy of the model, but also how well it knows the feature space. To estimate both, for each of the following statements a measure should be calculated:

1) Accuracy of the model itself is to be found.
2) The number of samples in the region must be taken into account. Too sparsely populated regions may yield good accuracy, when the model is used close to the single points, but not when it is applied in empty areas.
3) Only counting how many samples are in the region may yield wrong conclusions when the samples are not distributed well in the region. The uniformity of the feature distribution is a key point[1]. It is not sufficient to only check the distribution of single features, but multivariate feature distributions have to be taken into account. See Figure 3.

[1]The sample distribution in the feature space does not have to be uniform in general, but samples must be available in all regions; there should not be voids. With a minimum number of samples this condition is fulfilled by a uniform distribution.

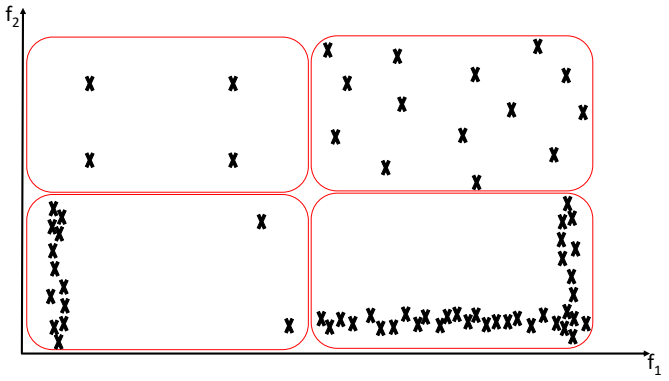

Fig. 3. Two dimensional simplification of a feature space. The four marked regions (red) show different point distributions and why it is important to also check for multivariate uniformity. The two top regions show equally distributed sample points. Here the total number of points plays an important role. One should make clear that an equidistant grid (top left) can reduce the amount of extractable information as compared to a more random grid (top right). In the bottom regions one can see that even with many available samples the the amount of extractable information can be low when the points are not distributed well.

Quantifying 1) is rather simple - most of the classical metrics (mean absolute error (mae), root mean square error ($rmse$), ...) are suitable. An evaluation regarding 2) and 3) is more complex, since this is similar to a point density distribution analysis. Inspired by the physical description of fluids, only up to second order effects are included. Usually the inclusion of third order effects does not yield significant improvement, while being computationally expensive.

The point density distribution analysis per region is done as follows:

- 0^{th} order: count the number of points
- 1^{st} order: find for the largest "gaps" in all single feature distributions
- 2^{nd} order: measure the uniformities of all bivariate feature distributions

A more general approach – which includes an arbitrary number of dimensions – to estimate and compare point density are kernel density estimates [35]–[38]. Due to the high computational effort these techniques are not used here.

In order to find the worst performing regions, the metrics from the model evaluation and the point density distribution analysis have to be combined to a score. Then all the regions can be compared. To simplify the following evaluations, the feature space is divided into regions of equal size. In this work, the feature space is divided into hyper cubes. For better interpretability and comparison all descriptive metrics n are normalized such that $0 < n < 1$, with 0 being the worst possible performance.

A. Model Performance Metric

To define a metric for the model performance, the mae of the i-th region is normalized to the maximum mae over all regions. Subtracting the result from 1 gives the metric:

$$n_{mae,i} = 1 - \frac{mae_i}{\max\limits_{0<j\leq N_R}(mae_j)}. \tag{1}$$

Where N_R is the number of regions in the feature space.

The performance metric of the i-th region may be found using cross validation methods [38]–[40] to get a reliable estimation. Other metrics can be used as well here and as usual the metric should be chosen regarding the underlying problem. In this work the *mae* was used because it corresponds nicely to the constraints provided by the integration and device engineers (domain experts).

B. 0th Order Density - Number of points

Given N_i the number of feature-label tuples (training data points) in the i-th region and N_R the number of regions, a metric is simply:

$$n_{N_{points},i} = \frac{N_i}{\max\limits_{0<j\leq N_R}(N_j)}. \tag{2}$$

C. 1st Order Density - Gaps in Single Feature Distributions

For the first order evaluation of the point density gaps in the distribution of single feature are searched for. A gap is understood as the distance between two data points. In one dimension, e.g. for a single feature, the largest distance between two consecutive points j and $j+1$ or to the borders of the region is searched for. If the region is empty, the range $(upper\,border - lower\,border)$ of the feature is taken. The normalization is done with respect to the feature range in the region, and the feature relevance can be weighted. The maximum over all features is then used to calculate the metric:

$$n_{gap,i} = 1 - \max_{f_k\in F}\left(w_k\left(\frac{1}{f_{u,i}-f_{l,i}}\max_{0<k\leq N_i}\left(|\Delta x_{i,f}^{j,j+1}|\right)\right)\right) \tag{3}$$

With the nomenclature:

F	set of features
$\lvert\Delta x_{i,f}^{k,k+1}\rvert$	distance between two consecutive points k and $k+1$ for feature f in region i
w_k	weight of feature f_k
$f_{u,i}$ and $f_{l,i}$	upper and lower borders of feature f in region i

It is possible to use the weighted average in place of the maximum distance in Equation 3. This takes all features into account, but may hide empty spots from the final decision.

D. 2nd Order Density - Uniformities in Bivariate Feature Distributions

With the second order effects the computational effort raises, since all combinations of two features are evaluated. The bivariate uniformity measure is inspired by the chi-squared test:

Region i is projected onto the plane spanned by the features f_k and f_l. This two dimensional space is subdivided into $P\times P$ subregions, and the number of points in each subregion are counted. For each subregion the point count $E_{p,i}$ is compared to the average number of points per part $E = N_i/(P^2)$. The squared differences over all parts are summed up and weighted by the combined feature weights (this may be interpreted as a normalized variance):

$$n_{O^2,i} = 1 - \max_{(f_k,f_l)\in F\times F} \left(\mathrm{cov}(f_k, f_l) \frac{w_k + w_l}{2} \sum_{p=1}^{P^2} \frac{(E_{p,i} - E)^2}{E} \right) \tag{4}$$

The nomenclature from above is used and $\mathrm{cov}(f_k, f_l)$ is the co-variance between the two regarded features. If the co-variances are not known or neglectable the factor can be set to unity. Again the maximum can be replaced by a mean. Equation 4 can easily be expanded to higher order interactions. It should be reminded that the chi-squared test does not work well with small sample sizes. Anyhow, the created metric is a usable non-uniformity measure.

E. Dimensionality reduction and co-variance exploitation

If the co-variances between features are known, it is possible to reduce the number of points which have to be generated. For a feature with neglectable co-variances to all other features it is unnecessary to simulate the dependence of its interaction with other features. Therefore, the number of sub-regions to be populated can be reduced, and the only objective is to have uniformly distributed samples in its own 1-dimensional range. For these features the projection of the whole feature space onto the corresponding axis can be used to focus the point generation away from less important sub-regions.

The ideas from 0th and first order density evaluation (sections IV-B and IV-C) can be applied to the projection of the whole feature space. This yields these very sub-sections of the feature which should be focused on. Figure 4 visualizes this approach.

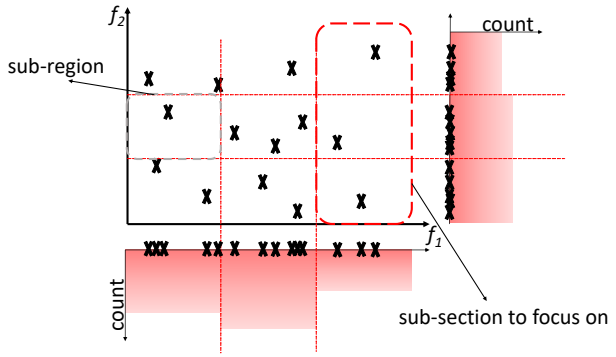

Fig. 4. Projection of a two dimensional space onto its axes. Each features is split into 3 sub-sections. The bar charts illustrate the 0th order density analysis (see section IV-B). Based on only the 0th order density analysis, the marked sub-section for f_1 would be selected.

Removing all other sub-regions from the partitioned evaluation reduces the evaluation (and simulation) effort considerably and will guarantee the creation of a uniform distribution for single features even when unimportant sub-regions are still empty. The covariance ($\mathrm{cov}(f_k, f_l)$) in Equation 4 ensures that from the (remaining) sub-regions the one is chosen where interactions between features are under represented.

In practice this allows to reduce the dimensionality of the problem. In [41] a dimensionality reduction approach is given which uses auto-encoders to find a latent representation of a device. The remaining task is the definition of *enough and well distributed points* for the feature (see Figure 1 right). That task does not only depend on the given problem, but also on the feature itself and cannot be answered in general.

It is possible to generalize the method to decoupled sets of features. The result is the partitioned evaluation on disjoint sub sets from the training data. Afterwards the evaluation results just have to be combined to yield an optimal sub-region for the data generation step. With a known co-variance matrix, a way to identify disjoint sub sets are clustering algorithms, such as *hierarchical clustering* with distance criteria. [42]

F. Scores for the Regions

Once the previously described metrics have been calculated, they have to be combined in order to find the regions with the poorest performance. Two simple but meaningful scores are given by the arithmetic and/or geometric means with optional weighting factors ω_a, ω_b, ω_c and ω_d:

$$s_{a,i} = \frac{\omega_a n_{mae,i} + \omega_b n_{N_{points},i} + \omega_c n_{gap,i} + \omega_d n_{O^2,i}}{\omega_a + \omega_b + \omega_c + \omega_d} \tag{5}$$

$$s_{g,i} = \left(n_{mae,i}^{\omega_a} \cdot n_{N_{points},i}^{\omega_b} \cdot n_{gap,i}^{\omega_c} \cdot n_{O^2,i}^{\omega_d} \right)^{\frac{1}{\omega_a+\omega_b+\omega_c+\omega_d}} \tag{6}$$

Finally $s_i \in [0, 1]$ provides a quantitative measure for the model performance in the i-th region of the feature space and it is normalized to the overall model performance: the smaller n_i, the worse is the performance of a given region. Especially for high dimensional feature spaces it can be likely that multiple regions have the same score, e.g. when regions are empty. When the sub-regions are ordered in a deterministic way it can be beneficial to shuffle their order before the scores are calculated. This ensures a more equal distribution of new samples over many loop (Figure 2) iterations. By construction, the above metrics are relative to the best/worst performing region and so are the scores. This means it is neither possible to compare different instances of the framework, nor can the scores be used as a goodness threshold to stop the loop. To address the second point an exit criterion must be used. See section IV-H.

G. Integration of Domain Knowledge

Physics Informed Artificial Intelligence connects data science with background knowledge from domain experts. The first step for this connection is to generate a large amount ($\sim 10^6$) of artificial inputs for the trained model. These inputs must represent the whole feature space, ideally a dense multi-dimensional grid. For all the generated inputs an output is predicted and the resulting cloud of output points is analyzed to identify any un-physical regions. The feature values for which the model yields un-physical predictions are then simulated in TCAD.

For semiconductor devices two methods are presented to do so. The idea for both methods is an evaluation of predicted data points. In the first method a domain expert may identify these regions just by visually inspecting the point cloud (see

Figure 5 left). This method is simple and often effective. The second method uses real world hardware measurements for comparison, here electrical test data. This data is collected over long production times, and it includes many process variations which are assumed to represent the feature space. From the electrical test data a second point cloud is plotted and compared to the cloud from prediction points. Predictions which do not fall inside the cloud of electrical test data are then assumed to be un-physical. See Figure 5 right.

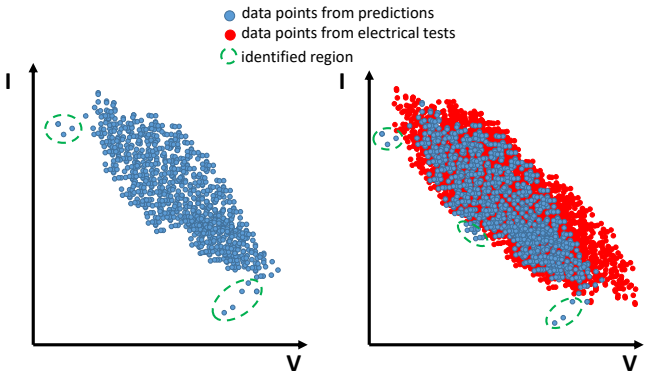

Fig. 5. Current-Voltage plot with predicted and real world data points. The identification of un-physical regions is done by domain expert experience (left) and by comparing with real world data (right).

While both methods fail to identify problems inside the data cloud, they are good to identify outliers. This approach has a nice synergy with the data driven evaluation.

H. Exit Criterion

The metric based estimation of the model performance, as in section IV-A, can be used to quantify if the generated model can be used, but without normalization. This may be done by setting a threshold value for the metric and stop the loop (Figure 2) automatically when the threshold is reached. Depending on the application, it can be interesting to demand this not only globally, but for every sub-region.

However, only estimating the model performance based on a metric can be misleading, even when a given threshold is reached. Sub-regions in the feature space that are too sparsely populated may yield a distorted view on the metric, and in empty sub-regions the model performance can not be estimated at all. It is therefore important to verify that enough training and test data points are available in all sub-regions of importance, and that they are well distributed.

The authors suggest an adapted k-fold cross validation scheme for the model as an exit criterion. In this scheme, the feature space is split into sub-regions, and for each k, a k-th part of the data is held back for model evaluation. A new model is built with the remaining data over the whole feature space, but evaluated separately in each sub-region. This yields k metric scores per sub-region which are averaged. If there are less than k samples in a sub-region all the data is used for validation. Figure 6 visualizes the adapted cross validation for $k = 2$.

Since the simulation and corresponding regression model have multiple outputs, a sufficiency threshold for every output has to be reached. An additional requirement is to have the metric evaluated in each sub-region to be sure the model is valid for each sub-region.

I. Model Building

To obtain a model of very high performance, fully automated excessive hyper parameter optimization [43], [44] is done in two major steps. First, multiple hundreds of different model configurations (sets of hyper parameters) are trained and evaluated in parallel. The best performing configurations are picked. Second, each of the best performing configurations is used as a starting point for a Bayesian search [45], also in parallel. As usual for optimization procedures, good starting points lead to fast convergence. This two step approach requires high processing power, but brings advantages beyond a time improvement as compared to a standard Bayesian optimization: When multiple Bayesian optimizations converge in the same final configuration, a high confidence is given, that a global optimum was found. When multiple local optima were found, a specific one can be chosen based on requirements (accuracy, hardware,...) or an ensemble model can be constructed. In case of TCAD-enabled ML, a custom loss/objective function can be used tailor the model even more to the use case. For this work fully connected Neural Networks are used, to apply transfer learning for the later setup of TCAD-enabled ML model for similar transistors.

V. Evaluation

The generation of training data, as described in Section III, is very efficient regarding the given task. Within a few initial loop iterations new samples are generated for empty sub-regions. After all sub-regions were selected one time for data generation, a uniform distribution was obtained and then the framework started to add more samples in underperforming regions. New inputs are generated not only for one, but multiple sub-regions and so a large amount of simulations can run in parallel, and a linear scaling regarding the data generation is achieved.

With an increasing number of input parameters dimensionality becomes problematic: the amount of simulations needed to generate a dense enough grid (see Figure 1) scales as $N_{div}^{N_{par}}$, where N_{div} is number of sub-sections a feature parameter is divided into, and N_{par} is the number of features.

For the semiconductor fabrication application presented in this work, a total of 9 parameters with 3 sub-sections per parameter were used, for a total of nearly 20,000 sub-regions. This large number underlines the importance of having an automated data generation method, and gives an idea about the amount of simulations needed to provide a trustable model. When using Latin Hyper-Cube Sampling to generate new TCAD simulation inputs, more than two new inputs per sub-region should be generated to obtain an equal distribution in the training data.

The discussion about the number of divisions per feature and of the number of training data samples in each sub-region is worth a whole discussion on its own and is just briefly done here. One should make clear again, that classical

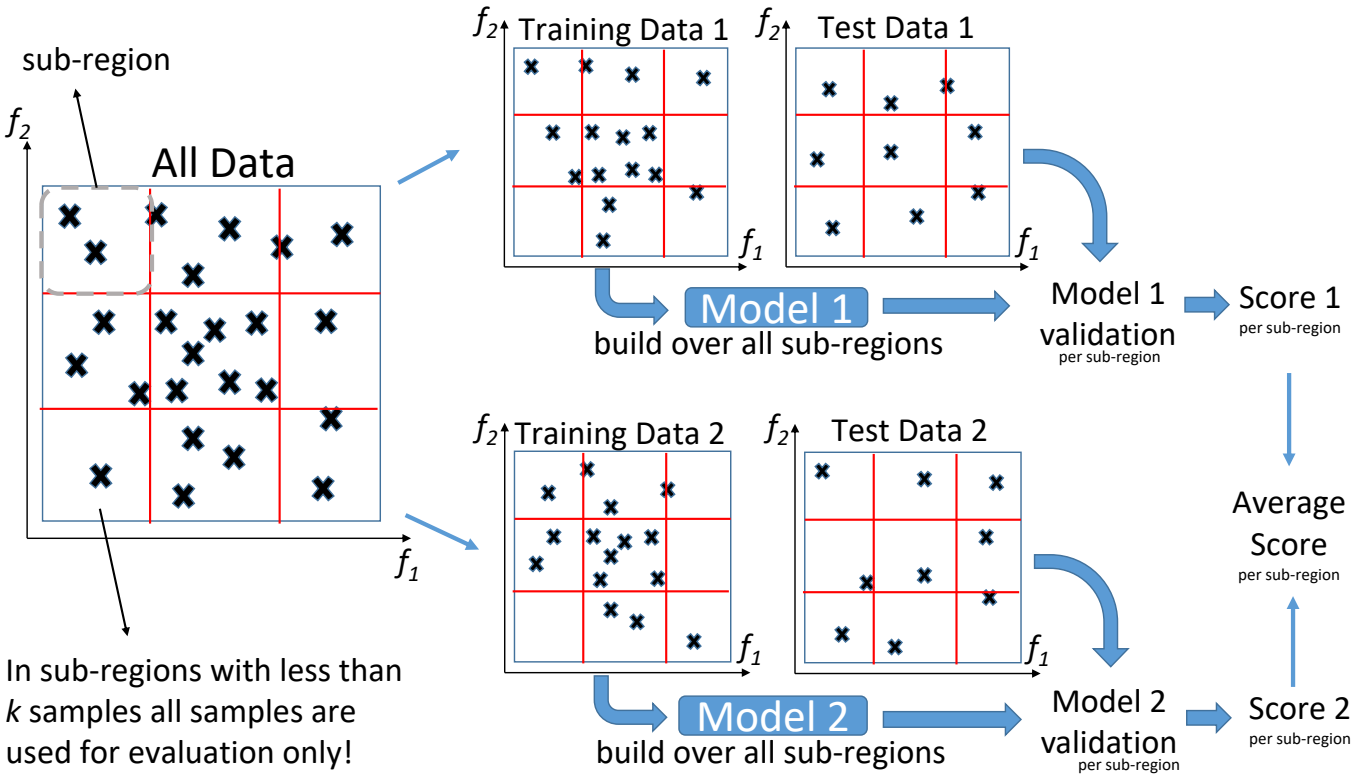

Fig. 6. The used cross validation yield an accuracy measure (score) for each sub-region. The visualization shows two dimensional feature spaces (spanned by f_1 and f_2) and a 2-fold cross validation. Two dimensions combined with three sub-sections per feature yields an accuracy score for each of the nine sub-regions.

Machine Learning approaches for model validation (e.g. cross validation), detection of underfitting and overfitting, or model generalization to independent or unseen data can be very limited by both undersampling and/or poorly distributed samples in the parameter space. A very simple case is an empty sub-region in the feature space: without any samples for training or testing it is impossible to validate the model. With just a moderate number of features (e.g. 5) the task of having an adequate population of the feature space becomes a non-trivial problem because of the computational effort required to finding new simulation inputs, running the TCAD simulations, and evaluating the model on all sub-regions.

This discussion is strongly connected to the point density in the feature space (see again Figures 1 and 3), but should also include arguments regarding the use case. Some sub-regions may just be un-physical or extremely improbable to occur, so there is no need to generate data there. Another point is the relative importance of features, which may make it unnecessary to focus on the equal distribution of unimportant features. Additionally, co-variances should be analyzed carefully to check if the dimensionality can be reduced: being able to separate an input space with 15 dimensions into 5 and 10 dimensional independent sub spaces not only reduces total number of needed samples, but also allows to generate a sample in each sub space with only one simulation. Therefore, the authors recommend to carefully evaluate each TCAD-eML related publication regarding the number and distribution of generated data points.

In summary, the authors suggest a split into 3 to 5 sub-sections per feature and a generation of at least 3 new samples per chosen sub-region. This depends on the number of dimensions of the feature space and the available computational power. To ensure a high scalability for parallel running simulations, it was very helpful to generate new inputs for multiple bad performing sub-regions instead of just one. When the total number of sub-regions is large, say $> 5,000$, it can be valuable to generate new simulation inputs already when previous simulations are still running. This has to be done with care to avoid unnecessary simulations.

Model Build

The output of the presented framework is rather a valuable data set than a Machine Learning model. However, training a regression-model with this data is rather straightforward when the data is understood. The procedure from Section IV-I was used to find optimal hyper parameters for the model. There are two major differences as compared to more classical Machine Learning applications. As presented in Section I-B the training and test data are noiseless, and the feature data are uniformly distributed with every sub-region in the feature space occupied with a sufficient number of points. The latter point needs to be regarded in more detail.

Usually, (numerical) training data follow a Gaussian distribution, possibly including co-variances. Splitting these data randomly in two sets, say training and test set, will generate two Gaussian distributions with smaller altitudes. Doing such

a split with the data generated by the presented framework is problematic because the point density throughout the feature space is rather constant but very small. Taking out single points leaves holes in the distribution.

In the classical case (Gaussian distributed data) this is less problematic, as the removed points are backed up by nearby points. For the given case, however, there are no back up points nearby, and the model will underperform on the test data when compared to a model build on a data set with a classical distribution. In the end this leads, again, to the question of how dense the grid has to be. See Figure 1. Because of this, regression techniques which hold back data during training (e.g bootstrap based training [46]) yield less good performing models.

VI. Application to Semiconductor Fabrication

The utility of a TCAD-enabled Machine Learning model will be demonstrated on device optimization within the constraints of processing capabilities – that is, parametric variability cannot be modified for improved nominal performance. Specifically, the following example shows a re-centering of a particular transistor to a higher performance level, while simultaneously not degrading, and preferably reducing, that transistor's variability across the range of critical electrical characteristics.

A. Technology Platform and Model Build

The simulation subject is the low-threshold-voltage (LVT) logic NFET of a widely-used 22nm full-depleted SOI (FDSOI) CMOS technology [47]. A model for the 22nm FDSOI LVT NFET was built based on the nine features shown in Table I.

TABLE I
THE NINE FEATURES USED IN THE 22NM FDSOI LOW-VT NFET

Feature Name	Physical Meaning
Lgate	On-wafer metal gate length
SOI	Thickness of silicon layer on buried oxide
TIL	Thickness of gate oxide layer under high-k dielectric
Implant Dose	Implant to adjust V_t
Spacer	Thickness of nitride spacer between metal gate and source/drain epitaxial layer
RTA peak	Final high-temperature anneal peak temperature
RDF	Variability arising from random doping fluctuations and metal gate granularity, used as work function standard deviation
ρ_{c}	Silicide contact resistivity
$\mathrm{R}_{\mathrm{con}}$	Lumped resistance of metal layers above silicide

Each of these features was split into three sub-sections. In every loop iteration (Figure 2) the 60 lowest-scoring sub-regions are identified and 4 new inputs were generated per sub-region. The framework ran constantly on a high-performance computing cluster with up to 30 simulations in parallel.

As an exit criterion and for model validation an adapted k-fold cross validation was used. The feature space is split into 3^9 sub-regions. For each sub-region the k-th part of its data is held back for model validation as explained in IV-H and Figure 6.

For this example, approximately 150,000 distinct TCAD process/device simulations were performed. In modern integrated circuits the number of devices on a chip is so large that the entire design space for each feature will be sampled on each chip. Therefore, a rather strict exit criterion was employed for our framework in order to properly sample the feature space. The benefits of such a strict criterion are confirmed when a regression model is built/evaluated on the whole data set in a classical Machine Learning way: Train-Test-Splits [48]. The calculated R^2-values lay above 0.99 and in the majority of the tests even above 0.999, when 10% of the data is used for validation only.

B. Model Application

The model, as an output of the presented framework, brings most advantages when used for tasks for which TCAD environment is unsuitable or impractical. Among other things this can be providing TCAD-based insights to non-TCAD experts or approaching problems where large numbers of TCAD simulations are needed. A classical use case for the model is device optimization/variability analysis [3]–[8]. This work presents a use case where a very large number of simulations have to be repeatedly produced.

In the following it is shown how the model can be used to provide guidance for design and process experts in the case of a processing environment where both the transistor variability and centering are critical for circuit. The model inputs are characterized statistically as Gaussian distributions. The nominal values of the inputs reflect processing targets and the variability of inputs are based on tool capabilities. The corresponding outputs are standard transistor electrical characteristics, e.g. $V_{t,sat}$, or $I_{d,sat}$, having their own distribution. The mean of each of the resulting electrical parametrics will be referred to as its 'centering' and its standard deviation as its 'variability'. The goal of the processing operation is to keep the transistor variation clustered as tightly around the centering target as possible.

The starting point for the study is shown in Figure 7a, the standard $I_{d,sat}$ vs. $V_{t,sat}$ plot for the transistor being studied. This plot is generated from the TCAD-eML model which is derived from the fitting process described above, combined with knowledge of the processing variability of the inputs in Table I. Both nominal (or centered) $I_{d,sat}$ and $V_{t,sat}$, as well as their initial dispersion or variability compares well with data collected from manufactured devices. This dispersion is the result of 10^5 randomly generated samples of the features defined in Table I, varied according their assumed feature variabilities.

The optimization goal is twofold: It is desired to improve device current drive $I_{d,sat}$ (a performance enhancement) without reducing the device turn-on point $V_{t,sat}$ (a power consumption degradation). This is a desirable and straightforward improvement in the power-performance characteristics of the technology. In meeting this performance-enhancement goal, the variability of the electrical parametrics of the device should

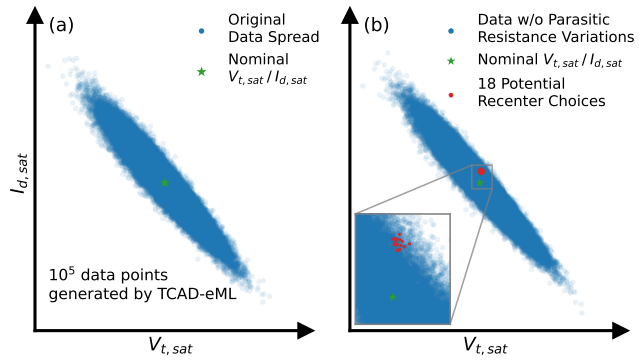

Fig. 7. Distributions of 22nm FDSOI low-V_t NFET generated by TCAD-enabled Machine Learning model. The point spreads arise from the Gaussian distributions used to fit the model variability to the equivalent distribution based on hardware test structures. (a) 10^5 sets of the nine features described in Table I are used to generate this distribution, which nicely overlay the real fabrication data point cloud (not shown). (b) Distribution generated by omitting the variability arising from the parasitic resistance features of ρ_c and R_{con}. From these 10^5 cases, there are 18 cases with higher $I_{d,sat}$ and equal or greater $V_{t,sat}$ which also meet other design restrictions. These 18 feature sets are re-centering candidates.

not increase. Decreasing variability, as characterized by their distribution standard deviation, is desirable. By exercising the model over the generated data points, a set of points with similar $V_{t,sat}$ but enhanced $I_{d,sat}$ can be identified (Figure 7b). This potential device re-centering sample forms a subset of only 0.018% of the population. Such a small opportunity for optimization is not unexpected from a successful technology that has been in fabrication for several years. This subset of devices are used as the starting point for the device re-centering exercise.

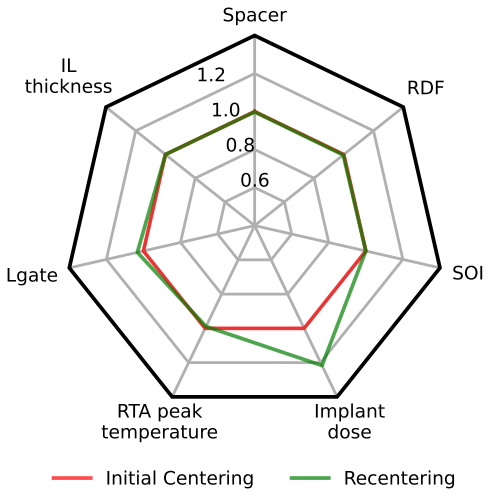

Fig. 8. Re-centered device features. The centering point was chosen from the candidate combination of features identified in Figure 7b.

Two details of the generation of the potential re-centering should be noted. Only seven out of nine features of Table I were used to generate the distribution of Figure 7b, since the features ρ_c and R_{con} have a monotonic effect on performance

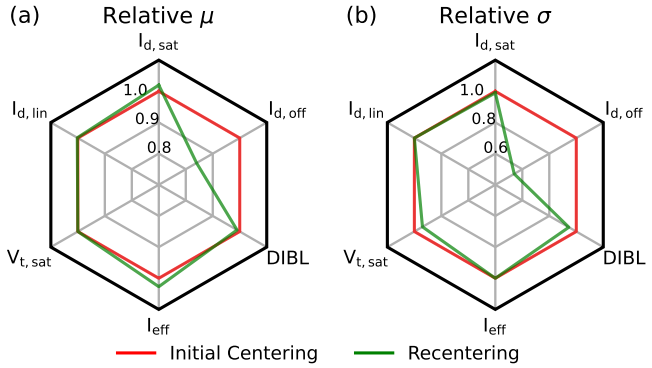

Fig. 9. Relative comparison of mean (μ) and standard deviation (σ) values of the critical electrical parametrics of the original and re-centered 22nm FDSOI NFET. (a) Re-centered mean values show a 2.0% $I_{d,sat}$, and 2.7% I_{eff} performance improvement, achieved at reduced off-current $I_{d,off}$, as well as reduced drain induced barrier lowering ($DIBL$). (b) Note that none of the electrical parameters shows worse variability as quantified by their standard deviation, and most have substantially improved.

and are assumed to be fully optimized. Additionally, the TIL parameter, the interfacial oxide thickness in the high-k metal-gate stack, is constrained to be not less than a certain value in order to meet reliability targets. While TIL has a Gaussian distribution in the data of Figure 7b, only devices with a TIL value greater than the target will be considered as potential re-centering options.

The feature re-centering that meets the above constraints and has the largest $I_{d,sat}$ improvement is shown in Figure 8. The electrical parametrics arising from the new re-centering are shown in Figure 9, which shows both the relative shift in the parametric means of the new device centering, as well as the changes in parametric variabilities of this device. Figure 9a shows a desirable increase in current drive, $I_{d,sat}$ and I_{eff} [49], with a concomitant, desirable, reduction in off current, $I_{d,off}$, and drain-induced-barrier-lowering, DIBL. Figure 9b indicates that in achieving this performance enhancement the variability of the electrical parametrics tracked in this study do not get worse, and for several parameters show improvement. Note that this pattern of re-centering improvement has been observed in large samples of complex circuits over multiple CMOS technology generations. [50]

The plot of Figure 7a is thus regenerated, again with a sample of 10^5 random feature samples around the mean (i.e., the new re-centering point), and the variability of the electrical parameters recalculated. In Figure 10 the result is visualized. It should be noted that this optimization was done by a device expert in less then 10 minutes. This is an extreme time reduction as compared to the standard optimization procedure, where actual material has to be run through the whole processing line.

VII. Conclusion

In this work, a data and model generation framework is developed and applied to analysis of a high-performance FDSOI CMOS technology. The focus throughout lies on model accuracy and reliability in order to be applicable for real world

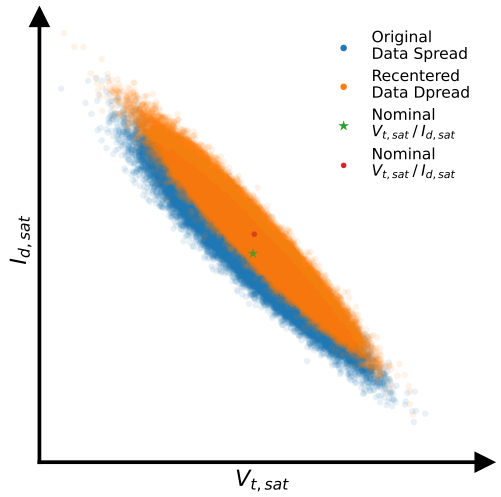

Fig. 10. Final result for re-centering and variability for a 22nm FDSOI NFET based on calibrated process and device TCAD combined with TCAD-enhanced ML. The resulting distribution, the orange points, is overlaid on the original distribution, the blue points, and their centered values noted. The re-centered device provides a noticeable increase in current drive with slightly less variability in both saturated threshold voltage and drive current (see Figure 9b).

use cases. Still the major constraint is: "As few simulations as possible but as many as needed." When the simulation capacity is limited, sub-sections IV-E and IV-G show which simulations should be prioritized. In general the framework scales very well with the available simulation resources. The defined exit criterion allows to enforce minimal model accuracy tailored to the needs of the use case.

The power of the methodology was demonstrated on a high-performance FDSOI CMOS technology. It was shown that despite a long period of fabrication optimization the methodology was able to identify a device centering solution that promises to deliver approximately 2% higher performance with reduced parametric variability. The details of that improved centering solution was found to be consistent with trends retrospectively observed in many other CMOS technology optimization efforts based on traditional hardware experimentation.

ACKNOWLEDGMENTS

This work has been supported by funding from the Free State of Saxony and the European Fonds for Regional Development (EFRE) under Project ARAMID, No 100375240.

REFERENCES

[1] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Reviews Physics*, vol. 3, no. 6, pp. 422–440, 2021.

[2] C. Jeong, S. Myung, I. Huh, B. Choi, J. Kim, H. Jang, H. Lee, D. Park, K. Lee, W. Jang *et al.*, "Bridging tcad and ai: its application to semiconductor design," *IEEE Transactions on Electron Devices*, vol. 68, no. 11, pp. 5364–5371, 2021.

[3] R. Ghoshhajra, K. Biswas, and A. Sarkar, "A review on machine learning approaches for predicting the effect of device parameters on performance of nanoscale mosfets," *2021 Devices for Integrated Circuit (DevIC)*, pp. 489–493, 2021.

[4] H. Y. Wong, M. Xiao, B. Wang, Y. K. Chiu, X. Yan, J. Ma, K. Sasaki, H. Wang, and Y. Zhang, "Tcad-machine learning framework for device variation and operating temperature analysis with experimental demonstration," *IEEE Journal of the Electron Devices Society*, vol. 8, pp. 992–1000, 2020.

[5] J. K. Lee, K. Ko, and H. Shin, "Analysis on process variation effect of 3d nand flash memory cell through machine learning model," in *2020 4th IEEE Electron Devices Technology & Manufacturing Conference (EDTM)*. IEEE, 2020, pp. 1–4.

[6] J. Yoo, Y. Jeon, D. Jung, J. Kim, J. Ryu, U. Kwon, D. S. Kim, K. Kim, Y. Kim, K. Lee *et al.*, "Machine-learning based tcad optimization method for next generation bcd process development," in *2021 33rd International Symposium on Power Semiconductor Devices and ICs (ISPSD)*. IEEE, 2021, pp. 279–282.

[7] 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. 29 071–29 077, 2021.

[8] R. Ghoshhajra, K. Biswas, and A. Sarkar, "Device performance prediction of nanoscale junctionless finfet using miso artificial neural network," *Silicon*, pp. 1–10, 2022.

[9] C.-W. Teo, K. L. Low, V. Narang, and A. V.-Y. Thean, "Tcad-enabled machine learning defect prediction to accelerate advanced semiconductor device failure analysis," in *2019 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*. IEEE, 2019, pp. 1–4.

[10] Y. Bankapalli and H. Wong, "Tcad augmented machine learning for semiconductor device failure troubleshooting and reverse engineering," in *2019 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*. IEEE, 2019, pp. 1–4.

[11] W. Kuo and T. Kim, "An overview of manufacturing yield and reliability modeling for semiconductor products," *Proceedings of the IEEE*, vol. 87, no. 8, pp. 1329–1344, 1999.

[12] N. Kumar, K. Kennedy, K. Gildersleeve, R. Abelson, C. Mastrangelo, and D. Montgomery, "A review of yield modelling techniques for semiconductor manufacturing," *International Journal of Production Research*, vol. 44, no. 23, pp. 5019–5036, 2006.

[13] J. Nduhura-Munga, G. Rodriguez-Verjan, S. Dauzere-Peres, C. Yugma, P. Vialletelle, and J. Pinaton, "A literature review on sampling techniques in semiconductor manufacturing," *IEEE Transactions on Semiconductor Manufacturing*, vol. 26, no. 2, pp. 188–195, 2013.

[14] S. Li, "From moore's law to function density law," in *MicroSystem Based on SiP Technology*. Springer, 2022, pp. 3–27.

[15] S. Myung, J. Kim, Y. Jeon, W. Jang, I. Huh, J. Kim, S. Han, K.-h. Baek, J. Ryu, Y.-S. Kim *et al.*, "Real-time tcad: a new paradigm for tcad in the artificial intelligence era," in *2020 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*. IEEE, 2020, pp. 347–350.

[16] C. K. Maiti, *Introducing Technology Computer-Aided Design (TCAD): Fundamentals, Simulations, and Applications*. Jenny Stanford Publishing, 2017.

[17] V. Moroz, S. Smidstrup, M. Choi, and A. Svizhenko, "Atomic level material and device analysis for finfet and nanowire design," 06 2018.

[18] C. Jeong, S. Myung, I. Huh, B. Choi, J. Kim, H. Jang, H. Lee, D. Park, K. Lee, W. Jang *et al.*, "Bridging tcad and ai: its application to semiconductor design," *IEEE Transactions on Electron Devices*, vol. 68, no. 11, pp. 5364–5371, 2021.

[19] X. Ying, "An overview of overfitting and its solutions," in *Journal of Physics: Conference Series*, vol. 1168, no. 2. IOP Publishing, 2019, p. 022022.

[20] M. F. Korns, "Highly accurate symbolic regression with noisy training data," in *Genetic Programming Theory and Practice XIII*. Springer, 2016, pp. 91–115.

[21] S. S. Raju, B. Wang, K. Mehta, M. Xiao, Y. Zhang, and H.-Y. Wong, "Application of noise to avoid overfitting in tcad augmented machine learning," in *2020 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, 2020, pp. 351–354.

[22] T. Yu, S. Simoff, and T. Jan, "Vqsvm: a case study for incorporating prior domain knowledge into inductive machine learning," *Neurocomputing*, vol. 73, no. 13-15, pp. 2614–2623, 2010.

[23] C. M. Childs and N. R. Washburn, "Embedding domain knowledge for machine learning of complex material systems," *MRS Communications*, vol. 9, no. 3, pp. 806–820, 2019.

[24] I. F. Sbalzarini, "Basic numerical methods," https://sbalzarini-lab.org/docs/numerics/script.pdf, pp. 71–73, 2021, accessed: 2022-01-28.

[25] H. Dhillon, K. Mehta, M. Xiao, B. Wang, Y. Zhang, and H. Y. Wong, "Tcad-augmented machine learning with and without domain expertise,"

IEEE Transactions on Electron Devices, vol. 68, no. 11, pp. 5498–5503, 2021.
[26] *Sentaurus Structure Editor User Guide*.
[27] T. Hirtz, S. Huurman, H. Tian, Y. Yang, and T.-L. Ren, "Framework for tcad augmented machine learning on multi-i–v characteristics using convolutional neural network and multiprocessing," *Journal of Semiconductors*, vol. 42, no. 12, p. 124101, 2021.
[28] S. A. Weissman and N. G. Anderson, "Design of experiments (doe) and process optimization. a review of recent publications," *Organic Process Research & Development*, vol. 19, no. 11, pp. 1605–1633, 2015.
[29] M. Zhou and T. Goh, "Iterative designed experiment analysis (idea)," *Quality and Reliability Engineering International*, vol. 32, no. 8, pp. 2977–2986, 2016.
[30] S. Goswami, S. Ghosh, and S. Chakraborty, "Reliability analysis of structures by iterative improved response surface method," *Structural Safety*, vol. 60, pp. 56–66, 2016.
[31] M. Yliruka, N. Asprion, R. Böttcher, J. Höller, P. Schwartz, J. Schwientek, and M. Bortz, "Increasing the reliability of parameter estimates by iterative model-based design of experiments using a flowsheet-simulator," in *Computer Aided Chemical Engineering*. Elsevier, 2019, vol. 46, pp. 637–642.
[32] F. vom Lehn, L. Cai, and H. Pitsch, "Iterative model-based experimental design for efficient uncertainty minimization of chemical mechanisms," *Proceedings of the Combustion Institute*, vol. 38, no. 1, pp. 1033–1042, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1540748920302790
[33] M. D. McKay, R. J. Beckman, and W. J. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 42, no. 1, pp. 55–61, 2000.
[34] M. Stein, "Large sample properties of simulations using latin hypercube sampling," *Technometrics*, vol. 29, no. 2, pp. 143–151, 1987.
[35] A. J. Izenman, "Review papers: Recent developments in nonparametric density estimation," *Journal of the american statistical association*, vol. 86, no. 413, pp. 205–224, 1991.
[36] Y.-C. Chen, "A tutorial on kernel density estimation and recent advances," *Biostatistics & Epidemiology*, vol. 1, no. 1, pp. 161–187, 2017.
[37] ——, "Modal regression using kernel density estimation: A review," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 10, no. 4, p. e1431, 2018.
[38] A. Q. del Río and J. V. Fernández, "A local cross-validation algorithm for dependent data," *Test*, vol. 1, no. 1, pp. 123–153, 1992.
[39] M. Stone, "Cross-validation: A review," *Statistics: A Journal of Theoretical and Applied Statistics*, vol. 9, no. 1, pp. 127–139, 1978.
[40] M. W. Browne, "Cross-validation methods," *Journal of mathematical psychology*, vol. 44, no. 1, pp. 108–132, 2000.
[41] K. Mehta and H.-Y. Wong, "Prediction of finfet current-voltage and capacitance-voltage curves using machine learning with autoencoder," *IEEE Electron Device Letters*, vol. 42, no. 2, pp. 136–139, 2020.
[42] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: an overview," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 86–97, 2012.
[43] T. Yu and H. Zhu, "Hyper-parameter optimization: A review of algorithms and applications," *arXiv preprint arXiv:2003.05689*, 2020.
[44] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, 2020.
[45] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2015.
[46] B. Efron and R. Tibshirani, "The bootstrap method for assessing statistical accuracy," *Behaviormetrika*, vol. 12, no. 17, pp. 1–35, 1985.
[47] E. M. Bazizi, A. Zaka, T. Herrmann, I. Cortes, L. Jiang, M. H. J. Goh, S. Deb Roy, E. Nowak, G. Kluth, P. Javorka, L. Pirro, J. Mazurier, D. Harame, T. Kammler, J. Hoentschel, J. Schaeffer, F. Benistant, and B. Rice, "Versatile technology modeling for 22fdx platform development," in *2017 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, 2017, pp. 365–368.
[48] A. Rácz, D. Bajusz, and K. Héberger, "Effect of dataset size and train/test split ratios in qsar/qspr multiclass classification," *Molecules*, vol. 26, no. 4, p. 1111, 2021.
[49] M. Na, E. Nowak, W. Haensch, and J. Cai, "The effective drive current in cmos inverters," in *Digest. International Electron Devices Meeting,*, 2002, pp. 121–124.
[50] K. Peterson, R. Logan, X. Yu, K. Dezfulian, G. Bazan, J. Winslow, N. Zamdmer, L. Dubuque, B. Walsh, A. Norfleet, F. Clougherty, B. Bayat, A. Mocuta, and K. Rim, "Device-design metrics to improve manufacturability," in *2010 IEEE/SEMI Advanced Semiconductor Manufacturing Conference (ASMC)*, 2010, pp. 179–183.