Comparison of Manifold Learning Algorithms for Rapid Circuit Defect Extraction in SPICE-Augmented Machine Learning

Vasu Eranki
Electrical Engineering
San Jose State University
San Jose, USA
vasu.eranki@sjsu.edu

Thomas Lu
Stanford Online High School
Redwood City, USA
thomaslu@ohs.stanford.edu

Hiu Yung Wong *
Electrical Engineering
San Jose State University
San Jose, USA
hiuyung.wong@sjsu.edu

***Abstract*— Identifying the source of integrated circuit (IC) degradation and being able to track its degradation via its electrical characteristics (e.g. the Voltage Transfer Characteristics, VTC, of an inverter) is very useful in failure analysis. This is because the electrical measurement is non-destructive, low-cost, and rapid. However, the extraction of defects from electrical characteristics requires significant domain expertise. To reduce or even obviate the need for domain expertise so that the process can be automatic for various circuits, one may use manifold learning. As a type of machine learning (ML), manifold learning also requires a large amount of accurate training data. To obtain enough defect training data, which is almost impossible from experiments, one may use SPICE simulation. Based on our previous work of using AutoEncoder (AE) to perform SPICE-augmented ML to extract the pMOS and nMOS source contact resistances from the inverter VTC, in this paper, we compare the efficacy of using another 6 types of manifold learning. They are used to predict the experimental result and it is found that most of them have reasonable performance although the AE is still the best (R^2=0.9). However, when including also the variation of PMOS width (as a weak perturbation to the data), algorithms such as Locally Linear Embedding (LLE) are found to perform better than AE (R^2=0.72) with LLE (R^2=0.83) being the best. Therefore, multiple manifold learnings are suggested to be used in parallel in real production to enhance accuracy.**

Keywords— Contact Resistant, Defects, Machine Learning, Manifold Learning, Reverse Engineering, SPICE Simulation

I. Introduction

Semi-conductor fabrication is becoming more complicated with new 3D structures such as stacked nanosheet, FinFET, and complimentary FET [1]-[3] and the subsequent increase in process complexity has made Failure Analysis (FA) more important. Rapid FA has become an essential part of defect identification and reverse engineering. Another consequence of the increased complexity in the fabrication process is the importance of parasitic components in highly scaled devices which further complicates their analysis. Contact resistance is one such parasitic resistance that requires more attention [4][5].

Classical FA techniques such as SEM and TEM are destructive, time-consuming, and expensive. To meet the current demand in FA, a low-cost and accurate technique to identify the presence of a defect, its type, and its numerical value is highly desirable. One method is to identify defects via the corresponding electrical characteristics such as Current-Voltage, I-V, and Capacitance-Voltage, C-V, for devices and VTC and butterfly-curves for inverters and SRAM, respectively. The problem with using electrical characteristics for FA is that there is no simple correlation between the defect and the change in the electrical characteristics. As a result, significant domain expertise is required. However, this is not desirable due to the many different types of circuits and different ways of electrical characterization.

To obviate the need of developing defect-electrical characteristic correlation for every circuit and measurement, one may use ML. However, most ML algorithm requires a lot of data and it is impossible to generate well-controlled defect data experimentally. One solution is to use the well-calibrated simulation as the low-cost data [6]. However, since simulation data does not have the noise and hidden variables as in the experimental data, overfitting can be an issue and special care needs to be taken (e.g. adding noise to training data [7] or using Principal Component Analysis [8]) to apply to experiments. To avoid this, one may use manifold learning. Manifold learning is a type of machine learning for projecting high-dimensional data onto a lower dimension while preserving the information present in the data. While there are various manifold learning algorithms, each algorithm attempts to preserve certain aspects of the data such as its local shape, global shape, or distance between points [9].

Autoencoder (AE) is one of the commonly used manifold learning algorithms [10]-[13]. We have used (AE) in various TCAD-augmented (data generated by TCAD) machine learning for inverse design [10], process variation identification [11], and electrical characteristics reconstruction [12]. In [13], an AE machine trained by SPICE simulation data is used to identify the values of the pMOSFET and nMOSFET drain contact resistances of an inverter in the experiment.

However, it is unclear if AE is the most suitable manifold learning algorithm for semiconductor applications and if they will still perform well when extra variations are added. In this paper, 6 other manifold learning tools are introduced and applied to the experimental data. Moreover, the pMOSFET width is intentionally varied to test the prediction accuracy of

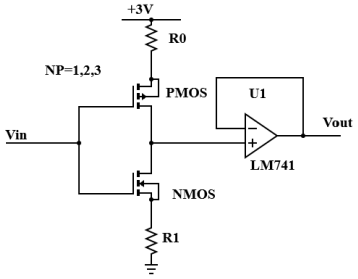

Fig. 1. Schematic of the circuit being studied. Different numbers of pMOSFETs are put in parallel (NP = 1, 2, 3) for different widths.

*Corresponding Author; This material is based upon work supported by the National Science Foundation under Grant No. 2046220.

the algorithms (note that in [13], the pMOSFET width is fixed to different values during the corresponding training and testing).

II. Experimental Setup and Training Data Generation

A setup using discrete components is used to generate the experimental data as this allows precise control of contact resistances. The PMOS and NMOS devices are from Advanced Linear Devices, Inc. (ALD1103). The devices were fabricated with an enhanced ACMOS silicon gate CMOS process. Extra resistors with 10% or better accuracies are added to the setup to model the defect-induced drain contact resistances of pMOSFET and nMOSFET devices. These resistances are labeled as R_0 and R_1, respectively. To prevent any loading during data acquisition, a unity gain buffer is used at the output. The experimental setup used in this experiment is described in detail in [13] and depicted in Fig. 1.

The training data is generated using SPICE simulation in Cadence [14]. Because ALD1103 does not have a SPICE model, a level 3 SPICE model is developed to capture the threshold voltage and the equivalent on-state resistance at $V_{DS} = V_{GS} = V_{DD}/2$ as R_p and R_n. The values of the drain contact resistance, R_0 and R_1, are varied from 10Ω to 1MΩ logarithmically. The effective width of the pMOSFET is varied by having 1, 2, or 3 pMOSFET in parallel in the circuit (NP = 1, 2, 3). Further details regarding how the training dataset is generated can be found in [13]. Selected experimental and simulation VTC are shown in Fig. 2(a) and Fig. 2(b), respectively.

Two different sets of data have been studied. The first set has NP = 1 and only R_0 and R_1 are varied. The second set has also the NP changed among the values of 1, 2, and 3.

III. Manifold Learning Algorithms and Results

In this paper the following algorithms are reviewed for their efficiency at dimensionality reduction in addition to the AE in [13]:

1. Principal Component Analysis
2. Isometric Mapping
3. Locally Linear Embedding
4. Multi-Dimensional Scaling
5. T-Stochastic Neighbour Embedding
6. Graph Convolutional Neural Networks

The VTC curve which had 51 discretized points generated across different input voltages is assumed to be a singular point in 51 dimensions, which are then projected onto a lower-dimensional plane using dimensionality reduction techniques listed above. After dimensionality reduction, the latent variables are mapped to R_0 and R_1 (Fig. 3) using KNN regression. During regression training, 10% of the training data was kept aside to form the validation dataset for hyperparameter tuning. The experimental dataset which contains actual VTC data was used to compare the performance of different algorithms.

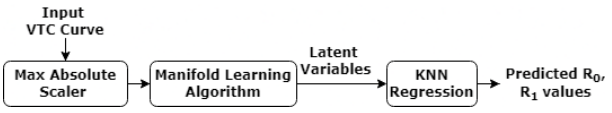

Fig. 3. The algorithmic flow for manifold learning prediction.

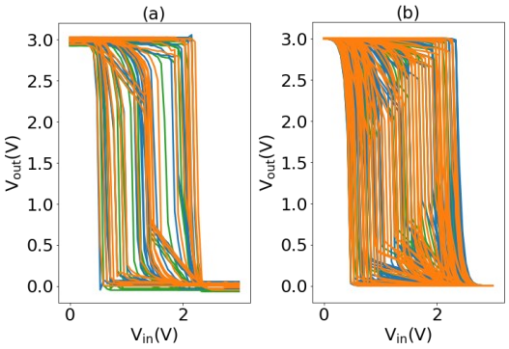

Fig. 2. Selected experimental (a) and simulated (b) VTCs for different NP's. Green :NP=1, Blue: NP=2, Orange=NP=3

A. Principal Component Analysis (PCA)

Most manifold learning algorithms are an extension of PCA [15] to deal with non-linear data and to effectively carry out dimensionality reduction for such data. For the first set of data, the first two principal components which accounted for 87.34% of the variance were retained and used for dimensionality reduction as there are only two variables (R_0 and R_1). The experimental prediction accuracy is depicted in Fig. 4. When extending the capabilities of PCA to predict the contact resistance values by introducing an additional variable of the effective pMOSFET width (NP = 1, 2, 3), it is found that four components are needed (instead of 3) for accurate prediction. These four components accounted for 91.49%. Moreover, the number of neighbors (k) was increased from 3 to 6 to overcome issues of underfitting. The

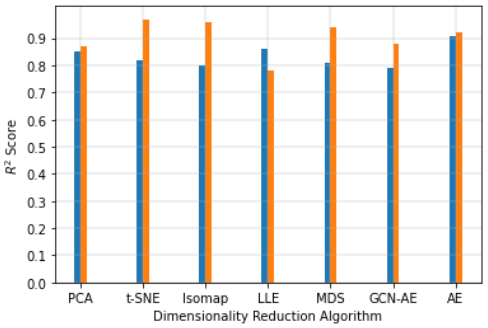

Fig. 4. Experimental prediction accuracy of R_0 and R_1 for NP=1 (first set of data with only R_0 and R_1 varied). Blue: R_0; Orange: R_1.

experimental results of the second set of data are shown in Fig. 5.

B. Isometric Mapping (Isomap)

Isomap is a non-linear dimensionality reduction technique that attempts to preserve the geodesic structure of high-dimensional data in lower dimensions [16]. The geodesic distance is calculated using the Minkowski distance with p set to 2, which then is equivalent to calculating the Euclidean distance between points. The experimental results for the first and second set of data are depicted in Fig. 4. and Fig. 5, respectively. During hyperparameter tuning, it is found that setting the number of neighbors to 128 was optimal and this could prevent the algorithm from underfitting or overfitting the data.

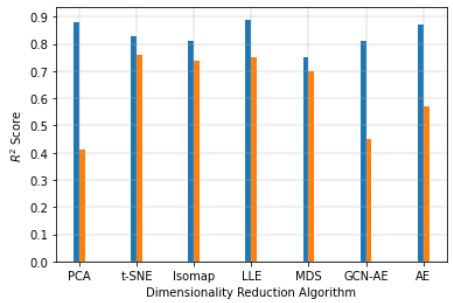

Fig. 5. Experimental prediction accuracy of R_0 and R_1 for the second set of data with R_0, R_1 and NP varied. Blue: R_0; Orange: R_1.

C. *Locally Linear Embedding (LLE)*

LLE is another type of manifold learning algorithm which attempts at preserving the local distance or 'information' within local neighborhoods [17]. In this study, the number of neighbors was set to 256 and because the number of neighbors was greater than the number of dimensions (51), the modified LLE was used instead which adds a regularization term to the calculation to get a stable non-rank-deficient solution [18]. The experimental results for the first and second set of data are depicted in Fig. 4. and Fig. 5, respectively.

D. *Multi-Dimensional Scaling (MDS)*

MDS is a non-linear dimensionality reduction manifold learning algorithm that attempts at preserving the actual distances between points from the higher dimension to its lower dimension projection [19]. In this study, the metric MDS was used wherein the dissimilarity matrix was calculated by using the Euclidean distance between two neighboring points. Since MDS is a tool to find the relative positions of the training data after finding their best lower dimensional representation, it cannot be used to predict new data. Therefore, to use it in the experimental data, one possibility is to add the experimental data (scarce) to the simulation data (abundant) to perform the projection. Once the data is reduced to a simpler representation, the two datasets are then separated and the regression model is trained only on the training data and evaluated on the experimental data. The number of neighbors was set to 128. The experimental results for the first and second set of data are depicted in Fig. 4. and Fig. 5, respectively.

E. *t-Distributed Stochastic Neighbour Embedding (t-SNE)*

t-SNE is a non-linear dimensionality reduction manifold learning technique that attempts at preserving the local distribution of the data points from the higher dimension to its lower-dimensional projection [20]. Similar to MDS, since t-SNE is a data visualization tool, the experimental data was appended to the training data during training and then ran separately. For this experiment, the perplexity of the algorithm was set to 50. The experimental results for the first and second set of data are depicted in Fig. 4. and Fig. 5, respectively.

F. *Graph Convolutional Neural Networks (GCN)*

Another algorithm that was a possible dimensionality reduction technique was highlighted in [21] wherein a graph neural network algorithm was used for doing dimensionality reduction on a graph. For this experiment, graph convolutional neural networks were used [22] and the VTC curve was modeled as a graph. The adjacency matrix was varied to accommodate the number of potential neighbors. The number of neighbors of the VTC curve was varied from 1 (i.e. the adjacency matrix resembled an identity matrix) to where every VTC sample is a neighbor to every other point (i.e. adjacency matrix becomes a unit matrix). Other numbers of neighbors such as 2, 5, 10, and 25 were all tried as well. However, the GCN did not perform well. It is found that the most efficient algorithm is the GCN-based Autoencoder whose model is shown in Fig. 6. Experimental results are shown in Fig. 4 and Fig. 5.

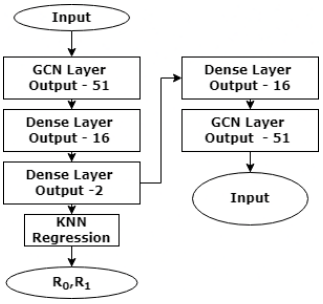

Fig. 6. GCN based Autoencoder

IV. Discussion

There were other experiments done as well but eventually led to poor algorithmic performance on the experimental data. The first such experiment was standardizing the input zero mean and unit variance. Once standardized, although all of the algorithms did perform well on the training data, the performance degraded for the experimental data. This is because the latter's distribution is different from the training dataset. The second experiment is the use of least-square regression instead of KNN regression. While the former and latter had similar performances, upon close inspection, the former was underfitting on the data.

As seen in Fig. 4, most manifold learnings have similar performance although AE is the best when there are two variables (i.e. the first set of data with only R_0 and R_1 varied). Note that for R_1 prediction, the t-SNE, MDS, and Isomap perform better than AE. This might be due to the fact that the experimental dataset was appended to the training dataset during training and the algorithm had *a priori* information about the experimental dataset via the calculated dissimilarity matrix. This shows that if some experimental data are

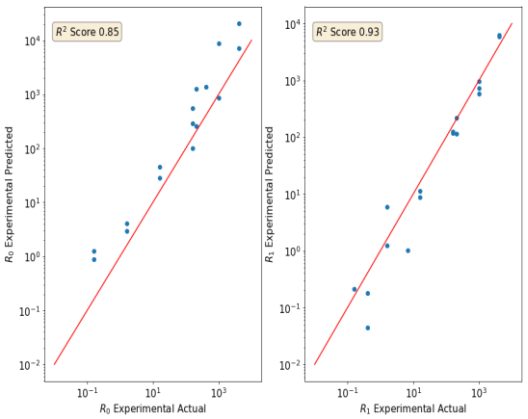

Fig. 7. Prediction of experimental R_0 and R_1 by Isomap with NP=1.

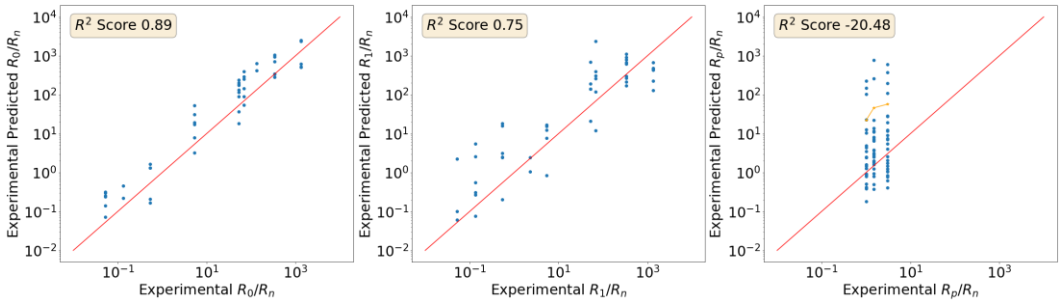

Fig. 8. Experimental prediction of R_0, R_1, and NP (which is translated to normalized R_P. The machine is trained by the second set of data.

available, appending them to the training data set generated by simulation might improve the inference accuracy.

Besides the data visualization techniques, between PCA, Isomap, LLE, and GCN-AE, the best performing algorithm overall was Isomap (which attempts at preserving the geodesic distances between points) with predicted experimental R^2 scores of 0.8 and 0.96 for R_0 and R_1, respectively (Fig. 7).

However, their performances start to differ in the second set of data (i.e. R_0, R_1, and NP are varied). It is found that AE prediction degrades substantially for R_1 prediction. LLE has the best overall performance with R^2 scores of 0.89 and 0.75 for R_0 and R_1, respectively (Fig. 8). Isomap and GCN-AE degrade the most when the variation of PMOS width is introduced.

For the second set of data, R_0, R_1, and NP (transformed to R_P) are the variables and are all predicted by the machine. Fig. 8 shows that the prediction of NP is not good probably because the variation of the corresponding R_P is not large enough. LLE can predict the trend of the average and it can be seen that the average R_p/R_n prediction increases for each NP value (golden line). This shows that even there is an extra relatively insignificant variable, the machine can still predict R_0 and R_1 well.

V. CONCLUSION

In this paper, SPICE simulation data is generated to train machines to predict the experimental contact resistances in inverters. Various algorithms capable of carrying out dimensionality reduction were experimented with. When there are only noise and regular hidden variables in the experimental data, while AE performs the best, all the algorithms have reasonable performance. When a minor variation is introduced (PMOS effective width), their performance differs a lot and LLE is found to be the best.

REFERENCES

[1] S. .-W. Chang *et al.*, “First Demonstration of CMOS Inverter and 6T-SRAM Based on GAA CFETs Structure for 3D-IC Applications,” 2019, pp. 11.7.1-11.7.4, doi: 10.1109/IEDM19573.2019.8993525.

[2] A. Guler and N. K. Jha, "Three-Dimensional Monolithic FinFET-Based 8T SRAM Cell Design for Enhanced Read Time and Low Leakage," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 27, no. 4, pp. 899-912, April 2019, doi: 10.1109/TVLSI.2018.2883525.

[3] A. Elwailly, J. Saltin, M. J. Gadlage, and H. Y. Wong, “Radiation Hardness Study of L G= 20 nm FinFET and Nanowire SRAM Through TCAD Simulation,” vol. 68, no. 5, pp. 2289–2294, 2021.

[4] Z. Liu *et al.*, “Dual beam laser annealing for contact resistance reduction and its impact on VLSI integrated circuit variability,” 2017, pp. T212–T213.

[5] S. Dev and S. Lodha, “Process Variation-Induced Contact Resistivity Variability in Nanoscale MS and MIS Contacts,” vol. 66, no. 10, pp. 4320–4325, 2019.

[6] Y. S. Bankapalli and H. Y. Wong, “TCAD augmented machine learning for semiconductor device failure troubleshooting and reverse engineering,” 2019, pp. 1–4.

[7] 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,” 2020, pp. 351–354.

[8] Hiu Yung Wong, et al., "TCAD-Machine Learning Framework for Device Variation and Operating Temperature Analysis With Experimental Demonstration," in *IEEE Journal of the Electron Devices Society*, vol. 8, pp. 992-1000, 2020.

[9] Y. Ma and Y. Fu, *Manifold learning theory and applications*, vol. 434. CRC press Boca Raton, 2012.

[10] K. Mehta, S. S. Raju, M. Xiao, B. Wang, Y. Zhang and H. Y. Wong, "Improvement of TCAD Augmented Machine Learning Using Autoencoder for Semiconductor Variation Identification and Inverse Design," in *IEEE Access*, vol. 8, pp. 143519-143529, 2020, doi: 10.1109/ACCESS.2020.3014470.

[11] Harsaroop Dhillon, Kashyap Mehta, Ming Xiao, Boyan Wang, Yuhao Zhang, and Hiu Yung Wong, "TCAD-Augmented Machine Learning with and without Domain Expertise," in IEEE Transactions on Electron Devices, doi: 10.1109/TED.2021.3073378.

[12] K. Mehta and H. -Y. Wong, "Prediction of FinFET Current-Voltage and Capacitance-Voltage Curves Using Machine Learning With Autoencoder," in IEEE Electron Device Letters, vol. 42, no. 2, pp. 136-139, Feb. 2021, doi: 10.1109/LED.2020.3045064.

[13] T. Lu, V. Kanchi, K. Mehta, S. Oza, T. Ho, and H. Y. Wong, “Rapid MOSFET Contact Resistance Extraction From Circuit Using SPICE-Augmented Machine Learning Without Feature Extraction,” vol. 68, no. 12, pp. 6026–6032, 2021.

[14] “Cadence Spectre Simulation Platform.” https://www.cadence.com/en_US/home/tools/custom-ic-analog-rf-design/circuit-simulation/spectre-simulation-platform.html.#

[15] I. Jolliffe, “Principal component analysis,” 2005.

[16] Xin Geng, De-Chuan Zhan and Zhi-Hua Zhou, "Supervised nonlinear dimensionality reduction for visualization and classification," in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 35, no. 6, pp. 1098-1107, Dec. 2005.

[17] S. T. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” vol. 290, no. 5500, pp. 2323–2326, 2000.

[18] B. Schölkopf, J. Platt, and T. Hofmann, *MLLE: Modified Locally Linear Embedding Using Multiple Weights*. 2007, pp. 1593–1600.

[19] M. A. Cox and T. F. Cox, *Multidimensional scaling*. Springer, 2008, pp. 315–347.

[20] L. Van der Maaten and G. Hinton, “Visualizing data using t-SNE.,” vol. 9, no. 11, 2008.

[21] S. Yan, D. Xu, B. Zhang, H. Zhang, Q. Yang, and S. Lin, “Graph Embedding and Extensions: A General Framework for Dimensionality Reduction,” vol. 29, no. 1. pp. 40–51, 2007, doi: 10.1109/TPAMI.2007.250598.

[22] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” 2016