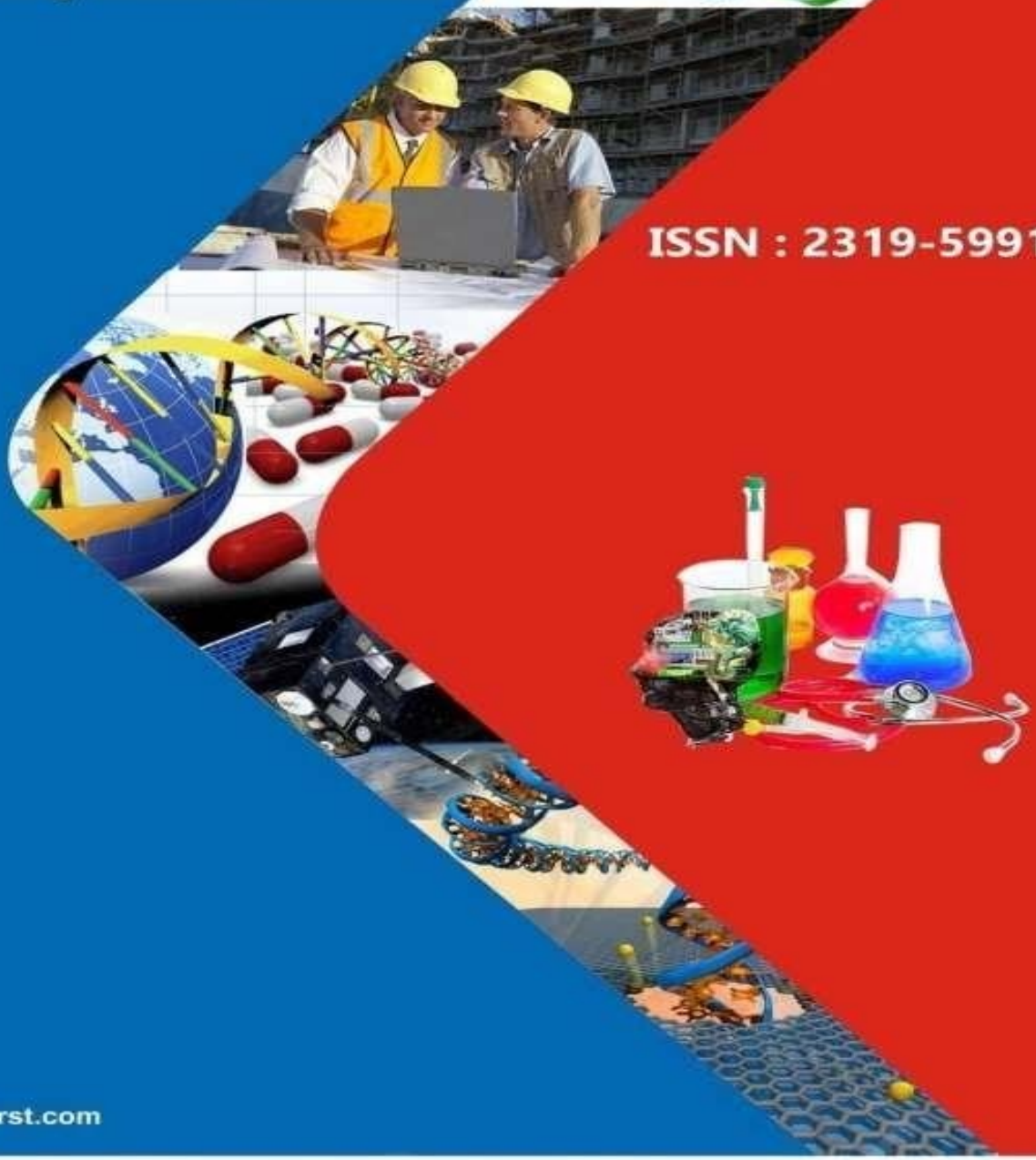
International Journal of Engineering Research and Science & Technology

IJERST

ISSN : 2319-5991



www.ijerst.com

Email: editor@ijerst.com or editor.ijerst@gmail.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

Foreseeing the Invisible Data Mistake Using Machine Learning for Devices That Are Prone

[1] Zayd Nawaz, [2] Shaik Abdul Muneeb Jeelani, [3] Mohammed Muzaffar Khan, [4] Mohammad Sufiyan Ali Nawaz, [5] Mr.Mohammed Mazheruddin,
[1]Assistant Professor,Department of CSE-AIML, Lords Institute of Engineering & Technology.
[2345] Student Department of CSE-AIML, Lords Institute of Engineering & Technology.

Abstract—

One kind of test escape that corrupts data without anyone noticing is known as a Silent Data Error (SDE). At the scales used by cloud service providers, they are noticeable even at very low DPPM levels. The costly system level test (SLT) is used to screen for certain faults in high-volume production that appear as system dependent errors (SDEs). Because these faults are so subtle, semiconductor devices that are susceptible to SDEs don't show any obvious patterns or abnormalities in the test data distributions. Therefore, it is difficult to screen these defective devices using ATE utilizing statistical death limits. We suggest using a Supervised Machine Learning (ML) strategy to discover inherent patterns in a real-world test dataset in order to speed up the process of identifying such defective devices before system testing. Results from experiments reveal that when compared to both conventional supervised and unsupervised approaches, the adopted supervised learning framework using an ensemble of feature selection algorithms performs far better.
Index Terms—machine learning, silent data mistakes

INTRODUCTION

When large-scale infrastructure systems suffer data loss without warning, it might be due to the sinister and destructive Silent Data Error (SDE) [1] [2]. System Level Tests (SLT) need a specialized test suite to identify these devices due to their covert nature, which evades detecting circuitry inside CPUs [3].

In order to identify and screen devices that may have faults that appear as SDEs, SLT employs a number of SDE detecting test suites, such as the Data center Diagnostics Project (DCDIAG) [4,5]. Nevertheless, owing to the lengthy time-to-fail, such tests come with a hefty price tag [6]. However, we still need lengthy testing to identify the failures, even with architectural developments that use Lockstep [7] and INFIELD testing [8] strategies to decrease SDE susceptibility. Those SDE failure devices at SLT are all marked as known good dies and have previously been checked on the ATE. This is because there aren't enough reliable tests or kill limits to detect these devices.

As a result, we use ML modeling on ATE test data in an effort to identify devices that could be SDE failures at SLT. This forecast may be useful for preventing these gadgets from reaching SLT in the first place. Relevant but smaller-scale research on consumer returns [9] has failed [10] [11] without focusing on SDE. Experimental findings from a large industrial dataset, as well as our Machine Learning technique, data labeling, data pre-processing, and feature engineering are detailed in this article.

PROPOSED METHODOLOGY

Predicting devices that appear as SDEs in the system or platform is the goal of this early effort. Initially, we exclude devices that have been shown to be problematic by the use of conventional ATE structural testing, such as SCAN and BIST. next this, we apply the supervised learning method to the test data in order to discover its inherent patterns, as explained in the next section.

A. Proposed Method Flow

Our proposed methodology is shown in the Figure 1 & 2.

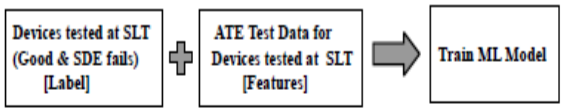


Fig. 1. Phase 1: Model Training

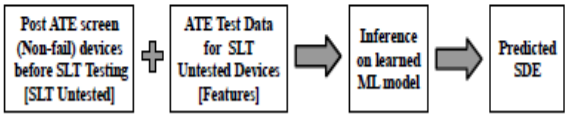


Fig. 2. Phase 2: Inference

Using the ground truth label as input, our study use multivariate test results (features) to train a prediction

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

function. Our data set's structure is shown in Figure 3. Consider a set of n SLT tests T1, T2,...., Tn in which M devices pass (G1,... GM) and W devices fail (D1,... DW) due to SDE. The results of the ATE tests, as they pertain to each device Gj and Dj, are saved as −→fj = [fj1,..., fjn] and −→kj = [kj1,..., kjn] correspondingly. We also assign a value of 1 to devices that pass and a value of 0 to devices that fail as a result of SDE. We get rid of the devices that don't have an SDE failure or a Good status. Our goal is to create a generalizable binary classifier H using the data from the dataset mentioned above, where H(→→T = T1, T2,...., Tn) → {0, 1}. A prediction label (either 0 or 1) is produced by the classifier, which takes input in the form of a test result vector →→T.

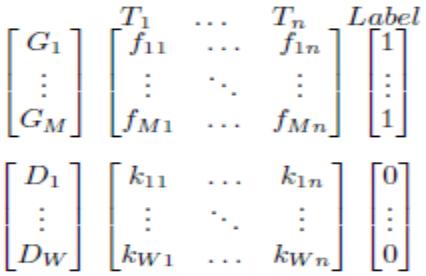


Fig. 3. Data Mapping

In our training dataset, there is a significant imbalance between the number of samples for devices that fail SDE and those that pass (W <<< M). Despite our best efforts, we were unable to get satisfactory training results with any of the oversampling strategies we tried, including K-NearestNeighbor (KNN) synthetic minority over-sampling technique (SMOTE) [12], K-Means SMOTE, and Adaptive Synthetic (ADASYN) [13]. Since the likelihood of SDE failure is greater in LOTs with devices that display it, we made the first order assumption that these LOTs were the only ones from which to choose wafers. Afterwards, we achieved device class parity by using KNN SMOTE oversampling.

D. Data preprocessing

We removed features with more than 12% missing values and imputed with the mean value to handle missing values in the test data [14]. Duplicate features have been eliminated. Additionally, we have eliminated features with low variance, which are variables that may not add anything to the model and are eliminated when the p-value is less than 0.95 according to Equation 1.

$$Var[T_1,\ldots,T_n] = \left[\frac{\sum_{v=1}^{M+W}(T_u^v - T_u^{Mean})^2}{M+W}; u \in [1,n]\right] \quad (1)$$

In order to increase relevance and avoid duplication in input features, feature engineering is a vital step. Using classifiers such as tree-based, MLP, etc., [16] and Principal Component Analysis (PCA) [15] to reduce features to 300 principal components while maintaining a variance of 95% yielded only 54% area under the receiver operating characteristic (ROC) using the CatBoost model. We have used an ensemble of feature selection approaches, including L1-norm [17] and Gini impurity [18], to enhance the model's performance. These methods are detailed below. L1-Norm: 1) We have penalized classifiers using L1-Norm to estimate the sparse weights (→→W = [w1,w2,...,wn]) of tests (→→T) in order to decrease the data's dimensionality and identify significant features. After that, we separate the weights with non-zero values and the tests that go along with them so we can focus on the important characteristics. We have expressed the L1-Norm to the cost function of the binary classifier like Equation 2 in order to locate relevant features.

$$\min_W \sum_{j=1}^{M+W} (-y_j * log(\hat{p}) - (1-y_j) * log(1-\hat{p})) + ||\vec{W}||_1 \quad (2)$$

In this context, ˇp represents the hypothesis function, yj stands for the ground truth, and ||→→W||1 is the L1-Norm.

2) Gini-Impurity: This metric allows us to calculate the relevance of features based on impurities; thereafter, we may use tree-based classifiers to remove characteristics that aren't relevant. Based on the results of Equation 3, we may calculate the split's impurity; the feature with the lowest impurity would be considered prime.

$$\overline{Gini(X)} = 1 - \sum_{j=0}^{|C|-1} p_j{}^2 \quad (3)$$

In this way, we may use the impurity score to choose the top n tests, or features. The probability of samples belonging to class j is denoted as pj, while the cardinality of classes is represented by |C|.

EXPERIMENTAL RESULTS

We ran trials on the "SDE fails" and "Non-SDE fails" failure signature buckets in SLT to see how well our

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

suggested technique worked. Data mistakes that do not affect SDE are quiet and are tested for at SLT using specialized DCDIAG test suites; data errors that do affect non-SDE are discovered deterministically at SLT using signatures from self-checking internal fault detection hardware. When creating the ML models, we relied on sklearn [19] and the Keras framework [20]. In the sections that follow, we detail those tests. First Experiment (A) A total of twenty-one thousand devices examined at SLT were used in this investigation. "Good" and "Non-SDE fails" are the device labels that were assigned based on the SLT test findings. After that, we took ATE test data for each of these devices and used it to extract characteristics. These characteristics include data from process monitors, power and temperature readings, all structural tests, logic at speed testing, minimum operating voltage (Vmin) from memory BIST tests, and more. There are around 13,000 rows in the training dataset, 4,000 rows in the validation dataset, and 4,000 rows in the test dataset. First, we have trained the system to focus on the top fifty characteristics by using the feature selection methods described in Sections II–E. After that, we trained a plethora of models, including a neural network, XGBoost, RandomForest, LightGBM, CatBoost, and Variational Autoencoder, an unsupervised method based on mean construction loss. However, when all test features are used from feature selection, an XGBoost model (with 100 trees, a booster learning rate of 0.2, and a booster of gbtree) performs better than the others.

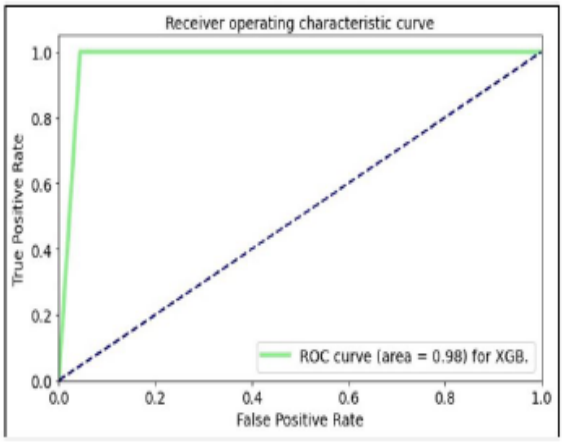


Fig. 4. ROC of XGBoost model

2) Evaluation: With an unknown test dataset consisting of 4K devices, we evaluated the aforementioned trained XGBoost Model. In order to assess how well the model performed, we have used ROC. This statistic provides a visual representation of our model's accuracy in class distinction, broken down into true positives and false positives. The model can distinguish between classes with a 98% confidence level, as shown in Figure 4 by the area under the ROC curve, which is 0.98. Gini impurity and L1-Norm features may be model agnostic on their own, but when combined with additional features and fed into XGBoost, they allowed the model to get a greater area under the receiver operating characteristic (ROC) curve. A True Negative rate of 96% and a True Positive rate of 100% for "Non-SDE fails" branded devices and "Good" devices, respectively, were achieved by our trained model. Second Experiment (B)

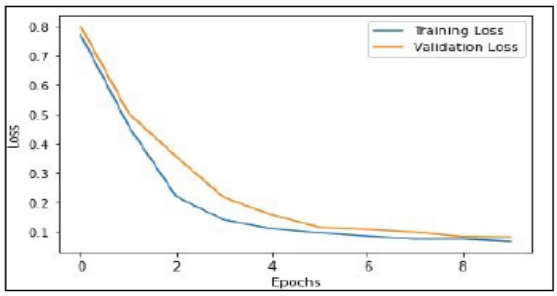


Fig. 5. Convergence of the MLP model

We have taken into account a larger amount of devices in this trial compared to the first, which included screening 1.1 million devices at SLT. We have marked the devices as "Good" or "SDE fails" according to the outcomes of the SLT tests. Further characteristics were subsequently derived from the ATE test data for each of these devices. These characteristics include data from process frequency monitors, power and temperature measurements, all structural tests, logic at speed testing, minimum operating voltage (Vmin) from memory BIST tests, and more. For the purpose of hyper-tuning the parameters, we have divided the dataset into two parts: training (80K) and validation (~20K). A test set of one million devices has been used. Selecting a large test set allows for the incorporation of data drift (process-related) [21], [22] occurring over a longer period of time. In doing so, we may evaluate the model's capacity to generalize and its performance on new data.

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

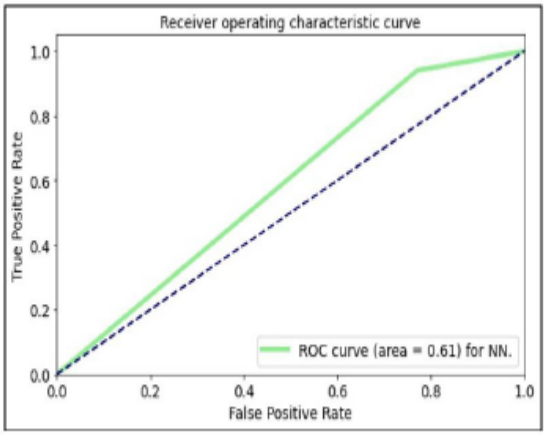


Fig. 6. ROC of MLP model

First, education: In order to speed up the training process, we used the feature selection strategy outlined in II-E to decrease 100,000 features to around 15,000 features. The dataset was then trained using a variety of models, including neural networks, XGBoost, CatBoost, LightGBM, and RandomForest. A 6-layer Multi-Layer Perceptron (MLP) model was the most effective of the bunch. In order to avoid overfitting, it incorporates a dropout layer at each hidden layer, which has a variable number of nodes and varying probabilities. Every hidden layer uses the Gaussian Error Linear Unit activation function as its output. Figure 5 displays the model's convergence with respect to epochs, and it is trained using entropy loss with a learning rate of 1e−4. At the ninth epoch, we see a near-perfect match between the training data loss and validation data loss, marking an ideal point at which to terminate training.

IV. CONCLUSION

By training and inferring on a large-scale industrial ATE test data set using ML, we aimed to anticipate SDE failure devices at SLT in this article. This is the first effort of its kind, as far as we are aware. We get a good performance score when using our proposed approach to build a model to predict between Good and Non-SDE (functional detectable fail) devices. However, when we apply the same methodology to predict SDE failing devices, we only get 26.61% of the SDE failing devices correctly with the test data that is available. We want to enhance the ML model's performance in the future by expanding the framework to cope with bigger datasets and adding elements from pre-silicon design data, architectural error detection hardware capabilities, Burn-In parameters, sophisticated structural fault models, etc. To avoid wasting test time, ML model training and inference will take place offline, meaning after the ATE test, while devices are on their way to the SLT test. With better results, we may utilize this approach to identify which devices are likely to fail SDE tests early on and subject them to rigorous testing. As a result, the likelihood of SDE DPPM is reduced in that group. Beyond that, we can reduce the amount of time it takes to apply tests on "Good" devices, which in turn reduces the cost of effective test time at SLT.

REFERENCES

[1] Peter H. Hochschild, Paul Turner, Jeffrey C. Mogul, Rama Govindaraju, Parthasarathy Ranganathan, David E. Culler, and Amin Vahdat. 2021. Cores that don't count. In Proceedings of the Workshop on Hot Topics in Operating Systems (HotOS '21). Association for Computing Machinery, New York, NY, USA, 9–16.

[2] Dixit, H. D., Pendharkar, S., Beadon, M., Mason, C., Chakravarthy, T., Muthiah, B. and Sankar, S. (2021). "Silent data corruptions at scale". arXiv preprint arXiv:2102.11245.

[3] D. Appello, H. H. Chen, M. Sauer, I. Polian, P. Bernardi and M. S. Reorda, "System-Level Test: State of the Art and Challenges," 2021 IEEE 27th International Symposium on On-Line Testing and Robust System Design (IOLTS), 2021, pp. 1-7, doi: 10.1109/IOLTS52814.2021.9486708.

[4]https://www.intel.com/content/www/us/en/support/articles/000058107/processors/ intel-xeon-processors.html

[5] https://github.com/opendcdiag/opendcdiag

[6] Lerner, David P., Benson Inkley, Shubhada H. Sahasrabudhe, Ethan Hansen, Luis D. Rojas Munoz, and Arjan van de Ven. "Optimization of Tests for Managing Silicon Defects in Data Centers." In 2022 IEEE International Test Conference (ITC), pp. 578-582. IEEE, 2022.

[7] A. Sinha, "Innovative Practices Track: Silent Data Errors," 2022 IEEE 40th VLSI Test Symposium (VTS), 2022, pp. 1-1.

[8] R. T. Veetil, R. Sharma and S. Gundeboyina, "Comprehensive In-field Memory Self-Test and ECC Self-Checker -Minimal Hardware Solution for FuSa," 2021 IEEE International Test Conference India (ITC India), 2021, pp. 1-6.

[9] H. -C. Hsu, C. -C. Lu, S. -W. Wang, K. Jones, K. -C. Wu and M. C. . -T. Chao, "Rule Generation for Classifying SLT Failed Parts," 2022 IEEE 40th VLSI Test Symposium (VTS), 2022, pp. 1-7, doi: 10.1109/VTS52500.2021.9794184.

[10] H. Hu, N. Nguyen, C. He and P. Li, "Advanced Outlier Detection Using Unsupervised Learning for

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

Screening Potential Customer Returns," 2020 IEEE International Test Conference (ITC), 2020, pp. 1-10, doi: 10.1109/ITC44778.2020.9325225.
[11] Lv, S.; Kim, H.; Zheng, B.; Jin, H. A Review of Data Mining with Big Data towards Its Applications in the Electronics Industry. Appl. Sci. 2018, 8, 582. https://doi.org/10.3390/app8040582
[12] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique.
J. Artif. Int. Res. 16, 1 (January 2002), 321–357.
[13] Haibo He, Yang Bai, E. A. Garcia and Shutao Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), 2008, pp. 1322-1328, doi: 10.1109/IJCNN.2008.4633969.
[14] Roderick J A Little and Donald B Rubin (1986). "Statistical Analysis with Missing Data". John Wiley & Sons, Inc., New York, NY, USA.
[15] Mac kiewicz, Andrzej and Ratajczak, Waldemar, Principal components analysis (PCA), Computers & Geosciences journal, volume= 19,number=3, pages=303-342, year=1993.
[16] V. Borisov, T. Leemann, K. Sesler, J. Haug, M. Pawelczyk, and G. Kasneci, "Deep neural networks and tabular data: A survey," arXiv
preprint arXiv:2110.01889, 2021
[17] Richard G. Baraniuk "Compressive Sensing", IEEE Signal Processing Magazine [120] July 2007
[18] Breiman, L., J. H. Friedman, R. A. Olshen, and C. J. Stone. Classification and Regression Trees. Boca Raton, FL: Chapman & Hall, 1984.
[19] Pedregosa, F et al., "Scikit-learn: Machine Learning in Python" in Journal of Machine Learning Research, 2011, pp. 2825-2830.
[20] Gulli, A., & Pal, S. (2017). Deep learning with Keras. Packt Publishing Ltd.
[21] Kim, Y., Lee, H., & Kim, C. O. (2021). A variational autoencoder for a semiconductor fault detection model robust to process drift due to incomplete maintenance. Journal of Intelligent Manufacturing, 1-12.
[22] Mallick, A., Hsieh, K., Arzani, B. and Joshi, G., 2022. Matchmaker: Data Drift Mitigation in Machine Learning for Large-Scale Systems. Proceedings of Machine Learning and Systems, 4, pp.77-94.
[23] F. Hapke et al., Defect-Oriented Test: Effectiveness in High Volume Manufacturing, in IEEE Transactions on Computer-Aided Design of
Integrated Circuits and Systems, vol. 40, no. 3, pp. 584-597, March 2021, doi: 10.1109/TCAD.2020.3001259.
[24] P. G. Ryan, I. Aziz, W. B. Howell, T. K. Janczak and D. J. Lu, Process defect trends and strategic test gaps, 2014 International Test Conference, 2014, pp. 1-8.