Efficient and Robust Resistive Open Defect Detection Based on Unsupervised Deep Learning

Liao, Yiwen; Najafi-Haghi, Zahra Paria; Wunderlich, Hans-Joachim; Yang, Bin

Proceedings of the IEEE International Test Conference (ITC'22); Anaheim, CA, USA; September 2022

doi: https://doi.org/10.1109/ITC50671.2022.00026

Abstract: Both process variations and defects in cells can lead to additional small delays within specifications, while the latter must be identified because they may degrade soon into critical faults for circuits and result in threat to reliability. Therefore, discriminating small delays due to defects from those due to variations has drawn increasingly attention in the test community over the recent years. One promising research direction is to formulate the task into binary classification by using delays under a few supply voltages as the only variables for datadriven algorithms. However, many approaches often assume the availability of delay information from both defective and nondefective cells or combinational circuits. This assumption implies a large time consumption for simulation, and considerable costs for manufactured defective devices. To address the issues above, this paper proposes to use unsupervised deep learning technique to train an recognizer on non-defective data only but still can identify defects during inference. Specifically, we have proposed weighted autoencoder with a novel data augmentation technique to solve this problem. Experiments show that our approach has comparable detection capability as supervised learning schemes, while our method does not require any defective data. Moreover, our approach is more robust to unbalanced datasets and to nontarget defects than other methods.

Preprint

General Copyright Notice

This article may be used for research, teaching and private study purposes. Any substantial or systematic reproduction, re-distribution, re-selling, loan or sub-licensing, systematic supply or distribution in any form to anyone is expressly forbidden.

This is the author's "personal copy" of the final, accepted version of the paper published by IEEE.[1]

[1] **IEEE COPYRIGHT NOTICE**
©2022 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Efficient and Robust Resistive Open Defect Detection Based on Unsupervised Deep Learning

Yiwen Liao*, Zahra Paria Najafi-Haghi†, Hans-Joachim Wunderlich†, Bin Yang*
*Institute of Signal Processing and System Theory, University of Stuttgart, Germany
Email: {yiwen.liao,bin.yang}@iss.uni-stuttgart.de
†Institute of Computer Architecture and Computer Engineering, University of Stuttgart, Germany
Email: {zahra.najafi-haghi,wu}@informatik.uni-stuttgart.de

***Abstract*—Both process variations and defects in cells can lead to additional small delays within specifications, while the latter must be identified because they may degrade soon into critical faults for circuits and result in threat to reliability. Therefore, discriminating small delays due to defects from those due to variations has drawn increasingly attention in the test community over the recent years. One promising research direction is to formulate the task into binary classification by using delays under a few supply voltages as the only variables for data-driven algorithms. However, many approaches often assume the availability of delay information from both defective and non-defective cells or combinational circuits. This assumption implies a large time consumption for simulation, and considerable costs for manufactured defective devices. To address the issues above, this paper proposes to use unsupervised deep learning techniques to train an recognizer on non-defective data only but still can identify defects during inference. Specifically, we have proposed to use a weighted autoencoder with a novel data augmentation technique to solve this problem. Experiments show that our approach has comparable detection capability as supervised learning schemes, while our method does not require any defective data. Moreover, in practice, our approach is more robust to unbalanced datasets and to non-target defects than other methods.**

I. Introduction

It is well known that resistive open defects can degrade during the early stage of the lifetime for circuits and the degradation can subsequently lead to catastrophic and irrecoverable faults for the devices or even the entire systems. Prior works such as [1]–[5] have already shown that the timing behavior of the circuits under varying supply voltages can be used as an indicator for identifying potential resistive open defects, because the cells with resistive open defects often result in delays of output signals of a combinational circuit.

Despite the clearly observable delays under different supply voltages for the circuits that contain resistive open defects, reliable and efficient identification remains a challenging problem. One of the major reasons is that similar timing behavior is also visible for the non-defective circuits due to process variations, including random local variations for transistors (e.g. length, width and gate oxide thickness) as stated in [6], [7]. That is to say, even a non-defective circuit can pose additional delays. Nevertheless, the delays within the given specifications for production are still acceptable in order to maintain as high yield as possible. As a result, in practice, it is effortful to discriminate the non-defective circuits with small extra delays due to process variations from the really defective circuits due to resistive opens. Specifically, Fig. 1 presents a few simulated delays respectively for non-defective (*solid orange line*) and defective (*dashed blue line*) combinational circuits. As can be clearly seen from Fig. 1, the resulting curves (delays) of both cases overlap each other for different voltages. Thereby, using a single supply voltage is infeasible to effectively distinguish delays of defective circuits from those due to process variations. Nonetheless, we can observe that the timing behavior of defective circuits still slightly differs from that of the non-defective circuits with respect to the slope and curvature for the whole curve under a few supply voltages, which is caused by the different mechanisms how process variations and resistive open defects affect the delays [4].

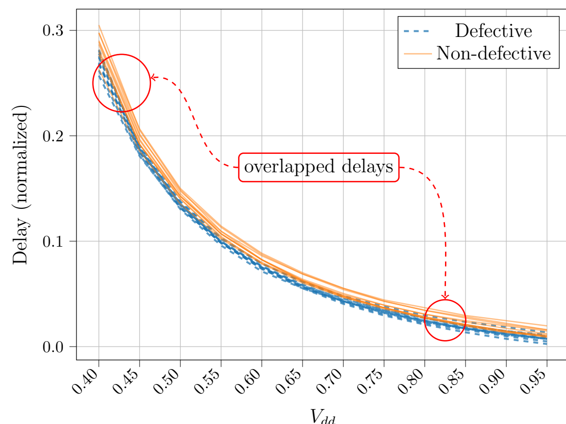

Fig. 1: Exemplary timing behavior of combinational circuits. The delays under varying voltages have overlapping parts for both defective and non-defective circuits, which is the main challenge for effectively identifying defects from defect-free circuits. Note that the delays are normalized for better visualization.

Based on this observation, some recent studies such as [4], [8] have for the first time introduced statistical approaches and machine learning to detect resistive open defects in combinational circuits with variation in presence. In particular, they recorded delays under a few different supply voltages for both the defective and non-defective circuits. Next, the detection problem was reformulated into a binary classification

task. In other words, the delays of defective and non-defective circuits were considered as two different classes. Subsequently, in a supervised manner, a classifier was trained on the delays from both classes. Finally, the trained classifier was expected to be capable of discriminating the defective and non-defective circuits during inference. Overall, these studies have shown the effectiveness of machine learning in identifying resistive open defect.

Indeed, the introduction of machine learning to resistive open defect detection is an promising idea, especially for big data and high demand on semiconductor supply. However, the approaches based on supervised learning still face several challenges. Firstly, the training requires the data from both classes, i.e. the delays of both defective and non-defective circuits. This corresponds to high time consumption in simulation and large costs on manufacturing due to the defective devices in real-world production. Secondly, it is well known that machine learning algorithms is prone to data distribution shift [9], [10]. This means that it is typically required to retrain the algorithm on the newly acquired data from scratch, if users encounter additional data or non-target defects. This is, however, common in practice, because we on one hand cannot collect all possible defect types; on the other hand, defects in the real world cannot be easily and accurately modeled in simulators. Finally, supervised classification algorithms often require balanced datasets; i.e. the numbers of data affiliated with different classes should be as similar as possible, otherwise the training can be significantly biased by the majority class. This requirement is sometimes contradictory to the real world, where defective devices are typically of small volumes.

Based on the concerns above, it is natural to consider using unsupervised methods to detect defects or anomalies in testing. There are indeed some previous studies such as [11]–[18] that used unsupervised anomaly detection algorithms in the test field. For example, early methods [11], [12] used One-Class Support Vector Machine (OCSVM) [19] to identify abnormal wafer maps. [15] used a Recurrent Neural Network (RNN) [20] to detect anomalies in in-field data by measuring the residual between original signals and their predictions. Some other studies leverage Autoencoder (AE) to identify defects for register transfer level debugging [14], screening customer returns [13] and system level tests [18].

Nevertheless, to the best extent of our knowledge, no prior works have evaluated unsupervised deep learning to detect resistive open defects. In particular, this paper proposes an unsupervised framework to detect resistive open defects in an efficient and effective way, which is based on a weighted autoencoder with a novel data augmentation technique. One of the most attractive benefits of our work is that no defective data are necessary for training, while the trained algorithm is still capable of identifying resistive open defects from the observed timing behaviors. In a nutshell, the main contributions of this paper are listed below:

- We have proposed an unsupervised framework to detect resistive open defects in combinational circuits based on deep learning with a novel weighted autoencoder;
- We control one parameter (supply voltages) and have to observe one parameter F_{max} only;
- We have designed a special data augmentation technique to enhance the overall detection performance;
- Training requires no defective data and our method thus spares time and costs for simulation and production;
- Our method has reliable and consistent detection performance facing non-target defects in contrast to supervised learning algorithms.

II. Background

In this section, we first briefly review the most typical and popular deep unsupervised learning algorithms for anomaly detection. Secondly, we present the generation process of the simulation data used in this research.

A. Unsupervised Anomaly Detection based on Deep Learning

Unsupervised anomaly techniques have become increasingly popular in the deep learning community over the last years [21]–[27]. This is due to the high demand of reliability and robustness for modern systems, including medical imaging [28], autonomous systems [25]–[27], and visual scenarios [23], [24]. Among these studies, we can observe that many state-of-the-art approaches use autoencoders as a backbone detection algorithm. The reason is intuitive: If an AE is trained on the normal data only, it can capture the most critical latent representations for the normal data. Note that the terms *normal* and *abnormal* are more common in the context of anomaly detection and deep learning. In this paper, accordingly, defective data are considered as abnormal, whereas non-defective data are considered as normal.

Generally speaking, an autoencoder is usually composed of one encoder and one decoder. Both of them can be understood as unknown nonlinear mappings to be approximated and learned from the data. In the scenario of anomaly detection, an autoencoder is trained on the normal data only so that the intrinsic structure of the normal data is captured. That is to say, the encoder learns to map normal data into some lower-dimensional latent representations, while the decoder maps them back to the original data space as reconstructions. Correspondingly, the learning objective is to maximize the similarity between the original inputs and their reconstructions. Thereby, after training, the autoencoder is expected to be capable of well reconstructing normal data only, while abnormal data are expected to have significantly larger reconstruction errors. Consequently, normal and abnormal data can be discriminated from each other by comparing the reconstruction errors to a predefined threshold.

Fig. 2 shows an intuitive example of detecting anomalies using an AE. In this example, an AE is trained with images of digit 2 only; i.e. the class of digit 2 is considered to be normal. After training, the AE has learned the most representative latent structure of digit 2 only and can therefore reconstruct digit 2 well, meaning that the reconstruction errors for the images of digit 2 are small. In contrast, the autoencoder has

no information about anomalies and is thus incapable of reconstructing abnormal samples such as digit 6, meaning that the images of digit 6 can lead to large reconstruction errors. Based on this intuition, our work also leverages autoencoder as the backbone detection algorithm and two special modifications are introduced in Section III.

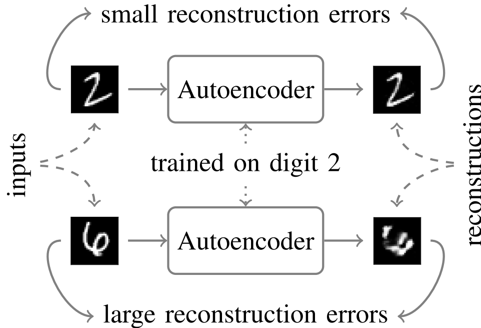

Fig. 2: An example to show how an autoencoder detects anomalies. The autoencoder is trained on the images of digit 2 only. After training, digit 2 as normal class can be well reconstructed (*top*), while digit 6 as abnormal class was poorly reconstructed (*bottom*). Intuitively, reconstruction errors can indicate the affiliation of new data.

B. Data Generation for Combinational Circuits

In this work, the modeling of defective and non-defective circuits as well as data generation followed a recent prior research [4]. Specifically, we connected an NAND cell to a chain of λ inverters, where the delay of each individual transistor was independent and followed a Gaussian distribution as $\tau \sim \mathcal{N}(\mu_\tau, \sigma_\tau^2)$. To generate different training samples, two process variations, the channel length L and gate width W were sampled from a Gaussian distribution by a Monte Carlo simulation on SPICE [29]. These simulated data were considered as non-defective data for training and evaluating our algorithms. It should be noted that in order to confirm the detection capability of our method, we still need defective data in this work to justify if the trained algorithm can successfully identify them. Thereby, resistive opens of different sizes were injected to the embedded NAND cells. Subsequently, similar Monte Carlo simulation was performed to obtain different samples for evaluation. The modeling of circuits and data generation are not the main focus of this paper and we encourage interested readers to refer to the previous papers [4] for more details.

III. Methodology

This section introduces our methodology to perform unsupervised open defect identification based on deep learning. Firstly, we present the overall workflow of the resistive open defect identification as an overview. Next, in order to meet the detection requirements, we propose a weighted autoencoder and a special data augmentation technique for this task. Finally, detailed implementations for our method are given for reproducibility.

A. Workflow

The entire workflow of the unsupervised open defect identification is shown in Fig. 3. As in the previous work [4], we feed the process parameters to the simulator (e.g. SPICE) to obtain the simulation results, i.e. the delays under varying voltages for circuits with random process variations. It should be emphasized that we only simulate the non-defective circuits and do not require any defective data in our case. This can save considerable time for simulation and decrease costs in practice, which is one of our major contributions. Subsequently, the simulated results are used to train our method in order to obtain a trained AE with weighted latent space (wAE). During deployment or inference, given some new measurements (delays) for a circuit, we can directly feed them to the trained AE and obtain the corresponding reconstruction errors. Based on a predefined threshold, defects can be identified. Although this paper deals with the simulation data, the overall workflow is also effective for real-world use cases by training our method using measurements from manufactured devices.

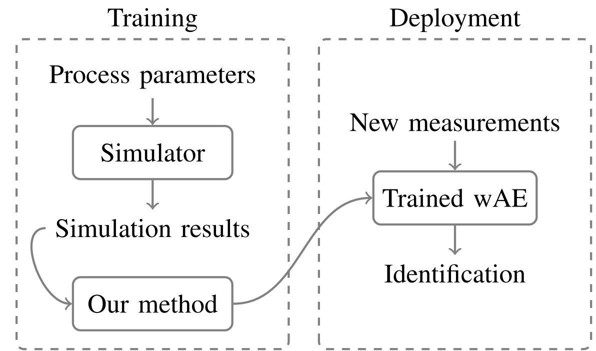

Fig. 3: Workflow for the unsupervised resistive open circuit defect identification. Simulation results for non-defective circuits only are necessary for training. Once deployed, we can feed new measurements to the wAE for identification.

B. Autoencoder with Weighted Latent Space

As mentioned before, autoencoders have been widely and successfully used for unsupervised and semi-supervised anomaly detection in many applications. Naturally, in this work, we also leverage autoencoder as the backbone algorithm. Different from a vanilla autoencoder, a learnable weighting vector $\boldsymbol{w}$ is applied to the latent space of the autoencoder in order to automatically focus on the most critical and important latent features for anomaly detection, as shown in Fig. 4, and we call this model a weighted Autoencoder (wAE).

Formally, let the delays under varying voltages of all simulated instances be denoted as a matrix $X = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]^T \in \mathbb{R}^{N \times D}$, where N is the total number of simulated instances and D is the number of varying voltages. The encoder is typically a nonlinear mapping $f_{\text{enc}}(\cdot; \boldsymbol{\Theta}_{\text{enc}})$ parameterized by $\boldsymbol{\Theta}_{\text{enc}}$ and the latent representation is

$$\boldsymbol{z} = f_{\text{enc}}(\boldsymbol{x}; \boldsymbol{\Theta}_{\text{enc}}), \tag{1}$$

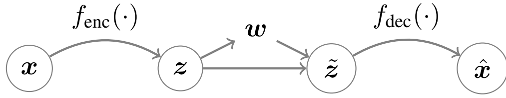

Fig. 4: The structure of an autoencoder with weighted latent space. $\boldsymbol{x}$ (delays) is mapped to a lower-dimensional representation $\boldsymbol{z}$ under the nonlinear mapping $f_{\text{enc}}(\cdot)$. Next, a weighting vector $\boldsymbol{w}$ is generated and is element-wisely multiplied to $\boldsymbol{z}$. Finally, the decoder $f_{\text{dec}}(\cdot)$ maps the weighted latent representation $\tilde{\boldsymbol{z}}$ back to the original data space as the reconstruction $\hat{\boldsymbol{x}}$.

where $\boldsymbol{z} \in \mathbb{R}^L$ is a latent representation for $\boldsymbol{x}$.

To automatically focus more on the most critical latent dimensions for better detection capability, we apply an FM-module [24] on $\boldsymbol{z}$, where the FM-module provides a batch-wise attention on the inputs. A similar design has shown its effectiveness in a two-stage semi-supervised anomaly detection algorithm proposed in [30]. However, we directly integrate the FM-module to the latent space of the autoencoder in order to obtain an end-to-end solution, which is more efficient and easier to use in practice. The effectiveness of this design has been demonstrated on image data [31] and we have for the first time introduce it to unsupervised circuit defect identification. In particular, the FM-module can generate a batch-wise weighting vector $\boldsymbol{w}$ as

$$\boldsymbol{w} = softmax\Big(\frac{1}{B}\sum_{i=1}^{B}\big(W_2 \tanh(W_1\boldsymbol{z}_i + \boldsymbol{b}_1) + \boldsymbol{b}_2\big)\Big), \quad (2)$$

where $W_1 \in \mathbb{R}^{L' \times L}$, $\boldsymbol{b}_1 \in \mathbb{R}^{L'}$, $W_2 \in \mathbb{R}^{L \times L'}$ and $\boldsymbol{b}_2 \in \mathbb{R}^L$ are trainable parameters. Based on this batch-wise weighting vector, the weighted latent representation is defined as

$$\tilde{\boldsymbol{z}} = \boldsymbol{w} \odot \boldsymbol{z}, \quad (3)$$

which can be interpreted that a batch-wise attention is applied to the latent representations during training.

The decoder is a another nonlinear mapping $f_{\text{dec}}(\cdot; \boldsymbol{\Theta}_{\text{dec}})$ parameterized by $\boldsymbol{\Theta}_{\text{dec}}$ and the reconstruction is

$$\hat{\boldsymbol{x}} = f_{\text{dec}}(\tilde{\boldsymbol{z}}; \boldsymbol{\Theta}_{\text{dec}}). \quad (4)$$

As in other autoencoder-based anomaly detection algorithms, the learning objective is to maximize the similarity between the input data and their reconstructions. The differentiable similarity measure can be specifically defined by users or practitioners. In this paper, to demonstrate the effectiveness of our framework, we minimize the mean squared error (MSE) loss as our learning objective:

$$\arg\min_{\boldsymbol{\Theta}} \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i||_2^2, \quad (5)$$

where $\boldsymbol{\Theta} = \{\boldsymbol{\Theta}_{\text{enc}}, \boldsymbol{\Theta}_{\text{dec}}, W_1, W_2, \boldsymbol{b}_1, \boldsymbol{b}_2\}$ are trainable parameters of the entire model.

C. Defect Detection

The detection for defects is intuitive: Defective timing behavior is expected to have larger reconstruction errors than those of a non-defective timing behavior, since the wAE is trained on fully non-defective data. Specifically, after the deployment, given a new sample $\boldsymbol{x}_t$, the reconstruction error is calculated as $\varepsilon_t = ||\boldsymbol{x}_t - f_{\text{dec}}(\boldsymbol{w} \odot f_{\text{enc}}(\boldsymbol{x}_t))||_2^2$. Accordingly, the detection rule is defined as

$$\boldsymbol{x}_t \in \begin{cases} \text{non-defective}, & \text{if } \varepsilon_t < \varepsilon \\ \text{defective}, & \text{otherwise} \end{cases} \quad (6)$$

where ε is a predefined threshold by users or practitioners. This threshold is typically defined based on a maximally acceptable false positive rate[1] in order to maintain a high yield.

D. Data Augmentation

As introduced in Section II, the simulation of circuits is time consuming, which can take hours to days for circuits with different sizes. Therefore, we often have a limited number of non-defective instances in practice to spare simulation time, although we can theoretically generate as many instances as we need on simulators. From a practical view, manufactured prototype devices are limited as well, and measurements can be costly. This also leads to limited available data, which becomes a major challenge for DL-based approaches because DL often requires large amounts of data.

In order to tackle this problem, we propose a data augmentation method specifically designed for the timing behavior data. More precisely, given a training sample $\boldsymbol{x} = [x_1, x_2, \dots, x_D]^T \in \mathbb{R}^D$, where each element denotes the small measured or simulated delay for a given voltage as introduced before, we randomly generate a mask vector $\boldsymbol{m} \in \{0,1\}^D$ with the constraint $|\boldsymbol{m}| = K$; i.e. $\boldsymbol{m}$ is a D-dimensional vector consisting of K ones and $D-K$ zeros. Then, an augmented sample $\boldsymbol{x}_a$ with respective to the original sample $\boldsymbol{x}$ is defined as

$$\boldsymbol{x}_a = \boldsymbol{x} \odot \boldsymbol{m}. \quad (7)$$

Exemplary augmented data with $K = 2, 3, 4$ are illustrated in Fig. 5. In this sense, we can generate maximally $\binom{D}{K}$ samples for a given K from one single sample. Literally, the augmented data can be interpreted as the samples in which $D-K$ simulated delays or measurements are missing, since these $D-K$ entries are zeros. Thereby, the augmented data forces the wAE to focus on the non-zero delays and reconstruct these delays from the same latent space. In parallel, the wAE tries to recover the missing delays from the neighboring delays as well. This augmentation makes sense because we cannot guarantee each measurement is valid in a real-world test scenario. In should be additionally emphasized that the augmented data $\boldsymbol{x}_a$ must be reconstructed as $\boldsymbol{x}$ as similar as possible, instead of being reconstructed to $\boldsymbol{x}_a$. This is to avoid the case that wAE implicitly learns to reconstruct the

[1]In this case, non-defective data are denoted as negative, while defective data are denoted as positive.

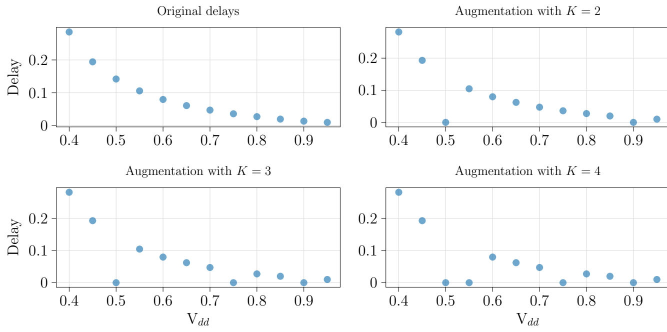

Fig. 5: Exemplary augmented data. The augmentation is performed by randomly setting K delays to zeros as missing measurements. During training, our algorithm is forced to recover these missing delays as well. Note that the delays are normalized for better visualization.

missing delays in $\boldsymbol{x}_a$, otherwise wAE would be too powerful so that both defective and non-defective data can be well reconstructed and lose the detection capability.

E. Implementations

In this work, the simulation data are not of extremely high dimensionality, so the implementation of our method was based on a few simple fully connected layers[2]. Specifically, as shown in Fig. 6, $f_{\text{enc}}(\cdot)$ consisted of three fully connected layers with 64, 32 and 16 neurons, respectively. After each dense layer, a Leaky-ReLU [32] activation with a rate of 0.02 was used. The decoder consisted of two hidden dense layers with 32 and 64 neurons, respectively, and one output layer. Same activation layers were used after each hidden layer in the decoder as well. The latent weighting block consisted of two dense layers with 8 and 16 neurons, respectively. After the first dense layer in the weighting block, $\tanh$ was used as the activation function as the original paper. Specifically, a detailed architecture can be found in TABLE I, where B denotes the minibatch size during training. In total, our implementation had 7156 trainable parameters, corresponding to about 28 Kibibyte only. It should be noted that our method is generic and thus not restricted to the aforementioned architecture, and in this work, the implementation aims to justify the effectiveness of our idea on the simulated combinational circuits.

IV. Experiments

This section evaluates the proposed framework for unsupervised resistive open circuit defect identification. The conducted experiments aim to answer the following four questions:

i) Can our unsupervised method achieve comparable performance as supervised approaches?
ii) Can our method outperform supervised methods when limited defect data only are available?
iii) Can our method provide robust identification performance for novel (unknown) defect types?

[2]Fully connected layer is also known as dense layer and we use both terms equally through out this paper.

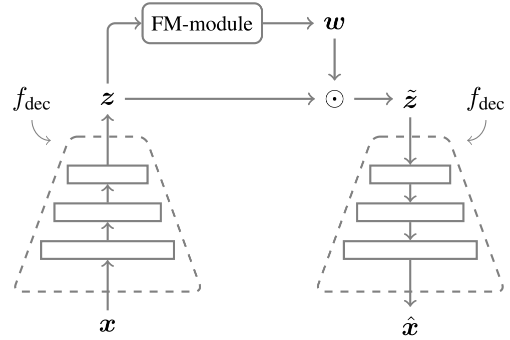

Fig. 6: The architecture of our method. Both the encoder and decoder are composed of multiple dense layers with corresponding activations (*rectangular blocks in dashed frames*). The FM-module is applied to the latent space to automatically weight different latent dimensions to further improve the capability of detecting defects.

TABLE I: Architecture of the implementation.

	Layer type	Output shape	# Parameters
Encoder	Input layer	$(B, 12)$	0
	Dense layer	$(B, 64)$	832
	Leaky-ReLU	$(B, 64)$	0
	Dense layer	$(B, 32)$	2080
	Leaky-ReLU	$(B, 32)$	0
	Dense layer	$(B, 16)$	528
Weighting	FM	$(B, 16)$	280
Decoder	Dense layer	$(B, 32)$	544
	Leaky-ReLU	$(B, 32)$	0
	Dense layer	$(B, 64)$	2112
	Leaky-ReLU	$(B, 64)$	0
	Dense layer	$(B, 12)$	780

iv) How do the novel augmentation and weighting in latent space facilitate the overall identification ability?

A. Settings

In an unsupervised setup, non-defective data only were used for training our method and no defective data were accessible during training. In total, 802 defect-free training samples were generated and we augmented the data by randomly setting 2 or 3 input features (delays) to zeros and the training dataset was thus increased to 4010. Note that 5% of the training data were used for validation. The test set consisted of 193 non-defective and 205 defective samples. The Area Under Receiver Operating Characteristic Curve (AUC) [33] was used as the metric to measure the defect identification performance. AUC is bounded in the range $[0, 1]$ and a higher AUC indicates better anomaly detection performance.

1) Supervised Reference Methods: As supervised reference methods, we considered three popular algorithms, i.e. k-Nearest Neighbor (kNN) [34], Support Vector Machine (SVM) [35] and Random Forest (RF) [36] from literature be-

cause they are popular supervised classifiers with simple training pipelines. Grid search was performed for the aforementioned reference methods. In particular, for RF, we selected the number of estimators from $\{5, 10, 25, 50, 100\}$ and depth from $\{5, 10, 20, 50, 100\}$; for SVM, we used radius basis function as the kernel and selected C from $\{0.01, 0.1, 1, 10, 20, 50\}$; for kNN, we used Euclidean distance as metric and selected the number of neighbors from $\{5, 10, 15, 20, 25, 50, 100\}$.

2) Unsupervised Reference Methods: In order to justify the effectiveness of the proposed unsupervised method, we selected three unsupervsied anomaly detection algorithms from literature for comparison, namely One-Class Support Vector Machine (OCSVM) [19], Isolation Forest [37] and Local Outlier Factor (LOF) [38]. For each reference method, we did a grid search on the key hyperparameters as follows. For OCSVM, we chose the optimal hyperparameters combinations γ and ν from $\{0.001, 0.01, 0.1, 0.5, 1, 10., 100, 500\}$ and $\{0.01, 0.1, 0.5, 0.75\}$, respectively; for Isolation Forest, the optimal number of estimators was determined from the set $\{5, 10, 15, 25, 50, 75, 100, 150, 200, 250, 500\}$; for LOF, we searched different numbers of neighbors from $\{5, 10, 25, 50, 75, 100, 150, 200, 250, 500\}$.

3) Hyperparameters: Our method was implemented using TensorFlow 2 [39] in Python. Both the supervised and unsupervised reference methods were implemented using the scikit-learn package [40] in Python. The optimizer for training the proposed wAE was Adam [41] with a learning rate of 0.0001. Batch size was 512 for all experiments. We stopped training when validation losses converged. The input data were preprocessed by standard-scaling for an efficient training.

B. Comparison with Unsupervised Methods

TABLE II: Comparison with unsupervised methods.

	OCSVM	Isolation Forest	LOF	Ours
AUC	0.895 (± 0.000)	0.880 (± 0.008)	0.889 (± 0.000)	0.942 (± 0.005)

In order to investigate the superiority of the proposed wAE, we compared wAE to three other popular unsupervised anomaly detection algorithms introduced above. TABLE II presents the detection performance in terms of AUC for all considered methods. As can be easily seen, our method significantly outperformed the other three data-driven unsupervised anomaly detection algorithms for resistive open defect identification with about $5.3\% \sim 7.0\%$ higher AUC, which confirms the effectiveness of our method for detecting defects.

C. Comparison with Supervised Methods

TABLE III: Comparison with supervised methods.

	kNN	SVM	Random Forest	Ours
AUC	0.948 (± 0.000)	0.947 (± 0.000)	0.949 (± 0.002)	0.942 (± 0.005)

Unsupervised anomaly detection is typically more challenging than anomaly detection in supervised manner due to the lack of information of missing classes. Thereby, the binary

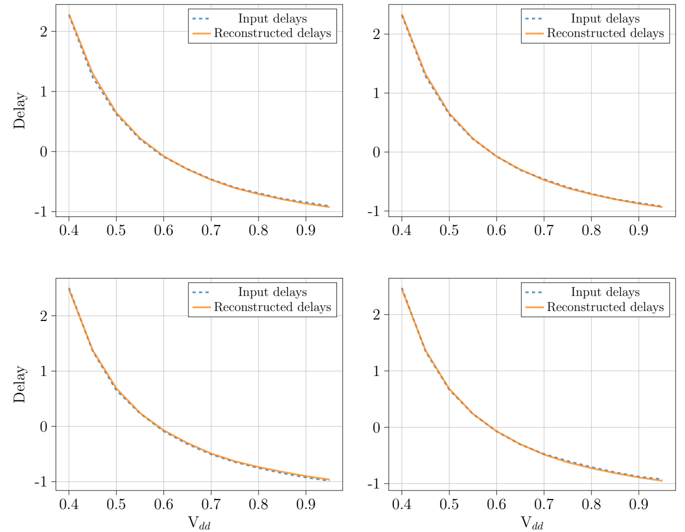

Fig. 7: The delays and reconstructions for non-defective data. The delays were well reconstructed with invisible differences.

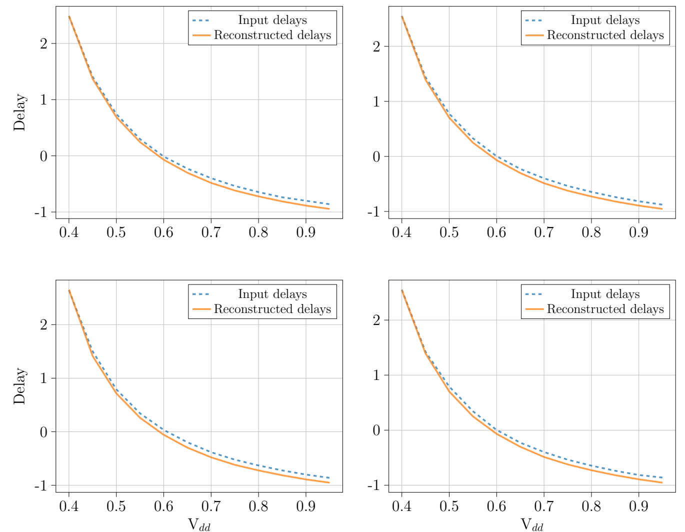

Fig. 8: The delays and reconstructions for defective data. The delays were poorly reconstructed with notable differences.

classification performance of supervised learning algorithms can be considered as an upper bound for the anomaly detection performance of unsupervised approaches. Nevertheless, it is still interesting to justify whether our method can have comparable performance as that of the supervised approaches. TABLE III shows the resulting AUCs on the test set for our method and other three supervised reference methods. All three reference methods outperformed our method without surprise. Nevertheless, we can still clearly observe that the performance difference is minor; i.e. our method achieved more than 99.3% anomaly detection ability of the reference supervised methods in terms of AUC. This indicates that our method already captured the key discriminative latent features for the non-defective data, although we did not

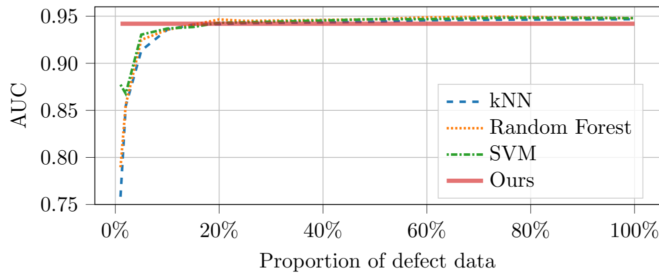

Fig. 9: The comparison of the defect identification ability under different amounts of defect data during training. Our method (*red solid line*) was independent of defect data and had consistent AUC. On the contrary, supervised methods had poor performance when limited defect data only were available.

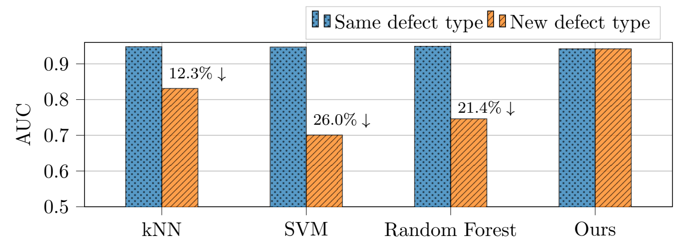

Fig. 10: The performance comparison for detecting non-target defects. Blue bars demonstrate the case where same defect type was encountered during test, while orange bars shows the performance for non-target defects during test. Notable performance drops can be easily observed by the supervised methods, while our method had consistent AUCs due to the unsupervised nature.

use any label information, meaning that using our method only requires a few non-defective simulation samples but can provide comparable defect identification performance as using both defective and non-defective simulation samples. Clearly, using our method is valuable for users and practitioners to spare much time and cost.

Furthermore, Fig. 7 and Fig. 8 demonstrates some exemplary defective and non-defective delays (*blue dashed line*) and their reconstructions (*orange solid line*). In Fig. 7, the reconstructed delays well match the original input delays, indicating small reconstruction errors. On the contrary, in Fig. 8, there is obvious discrepancy between the original delays and their reconstructions. As introduced before, no defective data were available during training, so delays of defective data cannot be well reconstructed. As a result, by comparing the reconstruction errors of a new test sample (i.e. delays under various voltages), we can justify whether the test sample is defective.

D. Case Study I: Lack of Defect Simulations

In this case study, we compared our method to the other supervised reference methods when only limited defective data were available during training. This is a typical and important scenario in practice, since it is time consuming to generate many enough defective instances or difficult to collect real devices with defects, which is a threat to yield. Thereby, the defect identification ability under different amounts of available defect data is a critical criterion for anomaly detectors. Importantly, we should emphasize that our method did not use any defect data and different amounts of the defect data were used for the supervised reference methods only. Specifically, we considered 11 cases; i.e. the ratio between defective and non-defective data was defined as ρ with $\rho \in \{1\%, 2\%, 5\%, 10\%, 15\%, 20\%, 25\%, 50\%, 60\%, 75\%, 100\%\}$. That is to say, for example, $\rho = 20\%$ indicates that the number of defective samples is only 20% of that for the non-defective data. Fig. 9 presents the overall AUCs versus different proportions of defective data. Since our method did not require any defective data, it resulted in a consistent performance (*red solid line*) of 0.942, while the performance of the reference methods was affected by the amounts of defective data available for training as expected. It can be observed that references methods required at least about 20% defective data (i.e. #defect : #defect-free = 1 : 5) to have a comparable AUC to that of our method. This implies that using our unsupervised defect identification methods can spare at least about 16.7% simulation time in practice. Moreover, for different proportion of defective data, the hyperparameters of each individual supervised learning algorithms must be separately fine-tuned. Therefore, the actual time consumption for supervised algorithms is significantly larger than the results shown above. It should be additionally noted that we only focus on the training time for simplicity. However, in practice, using our method means that much time of simulation and costs for real defective devices can be significantly spared as well.

E. Case Study II: Non-Target Defects

Machine-learning-based approaches often suffer from poor performance in presence of test data distribution shifts [9], [10]. This phenomenon is notably severe in anomaly detection because we cannot cover all possible or potential defect types during simulation and there can be non-target defects in the real world. In this case study, we designed two experiments. First, for the supervised methods, they were trained and tested on the defective and non-defective data from the same distribution (i.e. exactly the dataset introduced above); i.e they encounter same defect types during both training and test. Second, for the supervised methods, they were trained on a dataset, in which defect type was easier to detect, while tested on a harder test set; i.e. they encounter different defect types during training and test. The resulting AUCs are shown in Fig. 10. The blue bars present the first experiment (i.e. same defect type for both training and test) and this results are exactly the same results in TABLE III. The orange bars show the second case (i.e. different defect types during training and test). We clearly see that the AUCs for all three supervised methods significantly dropped by up to 26%, meaning that these methods are not robust and cannot deal with non-

target defects in practice. On the contrary, the training of our method was independent of defective data and it can thus have consistent AUCs for both cases. This robustness is valuable in practice because it means that our method does not require to collect all different defect types (which is infeasible in the real world) but can still achieve satisfied identification performance. More importantly, less modeling of the defect data also means notably less simulation time and less costs on defective devices.

F. Ablation Study

TABLE IV: Ablation study.

	without Augmentation	with Augmentation
without $\boldsymbol{w}$	0.890 (± 0.006)	0.927 (± 0.005)
with $\boldsymbol{w}$	0.913 (± 0.004)	0.942 (± 0.005)

As presented in Section III, the conspicuous design of our methodology is twofold in comparison with a vanilla autoencoder for detecting defects: *i*) We applied a batch-wise weighting block in latent space; *ii*) The simulated data were augmented by using random masks. In order to justify whether both designs contribute to the detection performance in comparison with a vanilla autoencoder, we conducted the ablation study. Note that ablation study is a common paradigm in Artificial Intelligence (AI) to identify the contribution of individual component to the entire AI system. In particular, we considered four cases: *i*) our method without data augmentation *and* without weighting, which can be understood as a vanilla autoencoder; *ii*) our method without data augmentation; *iii*) our method without weighting in latent space; *iv*) our method, i.e. wAE with data augmentation. TABLE IV shows the overall results. A vanilla autoencoder only achieved an AUC of around 0.890. Nevertheless, this anomaly detection performance is already comparable to supervised learning algorithms if only a few defective data are available. More importantly, we can observe that using either augmentation or weighting can moderately improve the identification capability in comparison with a vanilla autoencoder, respectively. For example, the novel augmentation led to about 4.2% improvement in terms of AUC for a vanilla autoencoder. In our case, the proposed novel data augmentation played a more critical role than weighting in latent space. One feasible reason might be that the original dataset was of small scale, i.e. having less than 1000 training samples, which made the training difficult. As expected, by using both the novel data augmentation and the batch-wise weighting mechanism in latent space, the overall detection capability was significantly improved.

V. Conclusion

In this paper, we present an unsupervised resistive open defect identification framework by introducing a novel data augmentation and an automatic weighting mechanism in latent space for autoencoders. The proposed method has shown comparable identification performance in comparison with supervised approaches. Our method requires no defective data and thus has consistent performance regardless of how many defective data are available in practice. Furthermore, the proposed method has shown robust performance w.r.t non-target defects, which is not possible for supervised approaches. Finally, ablation studies have shown the necessity of integrating both data augmentation and weighting to obtain a superior defect detection capability. Future work may include further evaluating the proposed method for real-world data measured from manufactured devices. In addition, the entire method is generic and can be applied to the test of different levels.

Acknowledgment

This research was supported by Advantest as part of the Graduate School "Intelligent Methods for Test and Reliability" (GS-IMTR) at the University of Stuttgart. This work is partially supported by the project grant WU 245/19-1 (FAST) funded by the German Research Foundation (DFG).

References

[1] U. Ingelsson and B. M. Al-Hashimi, "Investigation into voltage and process variation-aware manufacturing test," in *2011 IEEE International Test Conference*, 2011, pp. 1–10.
[2] S. Kundu, P. Engelke, I. Polian, and B. Becker, "On detection of resistive bridging defects by low-temperature and low-voltage testing," in *14th Asian Test Symposium (ATS'05)*, 2005, pp. 266–271.
[3] S. Deutsch and K. Chakrabarty, "Contactless pre-bond tsv test and diagnosis using ring oscillators and multiple voltage levels," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 5, pp. 774–785, 2014.
[4] Z. P. Najafi-Haghi and H.-J. Wunderlich, "Resistive open defect classification of embedded cells under variations," in *2021 IEEE 22nd Latin American Test Symposium (LATS)*, 2021, pp. 1–6.
[5] A. Karel, F. Azais, M. Comte, J.-M. Galliere, M. Renovell, and K. Singh, "Detection of resistive open and short defects in fdsoi under delay-based test: Optimal vdd and body biasing conditions," in *2017 22nd IEEE European Test Symposium (ETS)*, 2017, pp. 1–2.
[6] K. J. Kuhn, M. D. Giles, D. Becher, P. Kolar, A. Kornfeld, R. Kotlyar, S. T. Ma, A. Maheshwari, and S. Mudanai, "Process technology variation," *IEEE Transactions on Electron Devices*, vol. 58, no. 8, pp. 2197–2208, 2011.
[7] V. Champac and J. G. Gervacio, *Timing Performance of Nanometer Digital Circuits Under Process Variations*. Springer, 2018.
[8] Z. P. Najafi-Haghi, M. Hashemipour-Nazari, and H.-J. Wunderlich, "Variation-aware defect characterization at cell level," in *2020 IEEE European Test Symposium (ETS)*, 2020, pp. 1–6.
[9] J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset shift in machine learning*. Mit Press, 2008.
[10] M. Sugiyama and M. Kawanabe, *Machine learning in non-stationary environments: Introduction to covariate shift adaptation*. MIT press, 2012.
[11] J. Tikkanen, N. Sumikawa, L.-C. Wang, L. Winemberg, and M. S. Abadir, "Statistical outlier screening for latent defects," in *2013 IEEE International Reliability Physics Symposium (IRPS)*, 2013, pp. 2E.1.1–2E.1.4.
[12] N. Sumikawa, L.-C. Wang, and M. S. Abadir, "A pattern mining framework for inter-wafer abnormality analysis," in *2013 IEEE International Test Conference (ITC)*, 2013, pp. 1–10.
[13] H. Hu, N. Nguyen, C. He, and P. Li, "Advanced outlier detection using unsupervised learning for screening potential customer returns," in *2020 IEEE International Test Conference (ITC)*, 2020, pp. 1–10.
[14] C.-H. Shen, A. C.-W. Liang, C. C.-H. Hsu, and C. H.-P. Wen, "Fae: Autoencoder-based failure binning of rtl designs for verification and debugging," in *2019 IEEE International Test Conference (ITC)*, 2019, pp. 1–10.
[15] F. Su and P. Goteti, "Improving analog functional safety using data-driven anomaly detection," in *2018 IEEE International Test Conference (ITC)*, 2018, pp. 1–10.

[16] N. Sumikawa, M. Nero, and L.-C. Wang, "Kernel based clustering for quality improvement and excursion detection," in *2017 IEEE International Test Conference (ITC)*, 2017, pp. 1–10.

[17] Y. J. Zeng, L.-C. Wang, C. J. Shan, and N. Sumikawa, "Learning a wafer feature with one training sample," in *2020 IEEE International Test Conference (ITC)*, 2020, pp. 1–10.

[18] C. Liu and J. Ou, "Smart sampling for efficient system level test: A robust machine learning approach," in *2021 IEEE International Test Conference (ITC)*, 2021, pp. 53–62.

[19] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[20] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[21] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[22] D. Gong, L. Liu, V. Le, B. Saha, M. R. Mansour, S. Venkatesh, and A. v. d. Hengel, "Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection," in *The IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1705–1714.

[23] P. Schlachter, Y. Liao, and B. Yang, "Deep one-class classification using intra-class splitting," in *2019 IEEE Data Science Workshop (DSW)*. IEEE, 2019, pp. 100–104.

[24] Y. Liao, A. Bartler, and B. Yang, "Anomaly detection based on selection and weighting in latent space," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, 2021, pp. 409–415.

[25] Y. Liao, E. Hashemi, T. Wang, and B. Yang, "A learning-aided generic framework for fault detection and recovery of inertial sensors in automated driving systems," *IEEE Systems Journal*, vol. 15, no. 2, pp. 3001–3011, 2020.

[26] X. Cui, J. Cao, T. Wang, and X. Lai, "Robust randomized autoencoder and correntropy criterion-based one-class classification," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 4, pp. 1517–1521, 2020.

[27] A. B. Chowdhury, B. Tan, S. Garg, and R. Karri, "Robust deep learning for ic test problems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 1, pp. 183–195, 2022.

[28] T. Fernando, H. Gammulle, S. Denman, S. Sridharan, and C. Fookes, "Deep learning for medical anomaly detection–a survey," *ACM Computing Surveys (CSUR)*, vol. 54, no. 7, pp. 1–37, 2021.

[29] L. W. Nagel and D. Pederson, "Spice (simulation program with integrated circuit emphasis)," EECS Department, University of California, Berkeley, Tech. Rep. UCB/ERL M382, Apr 1973. [Online]. Available: http://www2.eecs.berkeley.edu/Pubs/TechRpts/1973/22871.html

[30] Y. Liao, R. Latty, and B. Yang, "Feature selection using batch-wise attenuation and feature mask normalization," in *2021 International Joint Conference on Neural Networks (IJCNN)*, 2021, pp. 1–9.

[31] Y. Liao and B. Yang, "To generalize or not to generalize: Towards autoencoders in one-class classification," in *2022 International Joint Conference on Neural Networks (IJCNN)*, 2022.

[32] A. L. Maas, A. Y. Hannun, A. Y. Ng *et al.*, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, no. 1. Citeseer, 2013, p. 3.

[33] N. Goix, "How to evaluate the quality of unsupervised anomaly detection algorithms?" *arXiv preprint arXiv:1607.01152*, 2016.

[34] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[35] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[36] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[37] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 413–422.

[38] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 93–104.

[39] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "{TensorFlow}: A system for {Large-Scale} machine learning," in *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, 2016, pp. 265–283.

[40] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.