Feature Extraction from Analog Wafermaps: A Comparison of Classical Image Processing and a Deep Generative Model

Tiago Santos*, Stefan Schrunner*, *Student Member, IEEE,* Bernhard C. Geiger, *Senior Member, IEEE,* Olivia Pfeiler, Anja Zernig, Andre Kaestner and Roman Kern

***Abstract*—Semiconductor manufacturing is a highly innovative branch of industry, where a high degree of automation has already been achieved. For example, devices tested to be outside of their specifications in electrical wafer test are automatically scrapped. In this work, we go one step further and analyse test data of devices still within the limits of the specification, by exploiting the information contained in the analog wafermaps. To that end, we propose two feature extraction approaches with the aim to detect patterns in the wafer test dataset. Such patterns might indicate the onset of critical deviations in the production process. The studied approaches are: (A) classical image processing and restoration techniques in combination with sophisticated feature engineering and (B) a data-driven deep generative model. The two approaches are evaluated on both a synthetic and a real-world dataset. The synthetic dataset has been modelled based on real-world patterns and characteristics. We found both approaches to provide similar overall evaluation metrics. Our in-depth analysis helps to choose one approach over the other depending on data availability as a major aspect, as well as on available computing power and required interpretability of the results.**

***Index Terms*—Automation, Data Processing, Unsupervised Learning, Feature Extraction.**

I. Introduction

PATTERN recognition is a promising, as well as challenging area in Computer Science, which has recently attracted interest in various branches of industry, including semiconductor manufacturing. In a production environment, tasks such as identification of critical conditions are supported by data analysis tools, requiring a lot of experience and human effort to take correct decisions. As a response to this, initiatives have been started to increase the degree of automation. This goal is challenging as it requires an understanding of the specifics and peculiarities of semiconductor manufacturing, in addition to in-depth know-how in data science methods.

* Both authors contributed equally to the publication.

T. Santos is with the Institute of Interactive Systems and Data Science, Graz University of Technology, Graz, Austria (e-mail: teixeiradossantos@tugraz.at)

S. Schrunner, O. Pfeiler and A. Zernig are with KAI - Kompetenzzentrum Automobil- und Industrieelektronik GmbH, Villach, Austria (e-mail: stefan.schrunner@k-ai.at; olivia.pfeiler@k-ai.at; anja.zernig@k-ai.at)

A. Kaestner is with Infineon Technologies Austria AG, Villach, Austria (e-mail: andre.kaestner@infineon.com)

B. C. Geiger is with KNOW-Center, Graz, Austria (e-mail: bgeiger@know-center.at)

R. Kern is with KNOW-Center, Graz, Austria and the Institute of Interactive Systems and Data Science, Graz University of Technology, Graz, Austria (e-mail: rkern@know-center.at)

Manuscript received Aug 08, 2018; revised Jan 17, 2019, accepted Apr 10, 2019.

At the end of semiconductor frontend production, the wafer test dataset is generated by conducting a sequence of electrical tests. Each device on the wafer is measured in order to detect quality deviations, capturing a number of parameters for each device. Depending on whether or not the measurements are within the predefined specification limits, devices are classified as pass or fail.

In addition, wafer test data might also reveal production issues visible as spatial regularities, i.e. patterns on the so-called wafermap. For many of the different products, engineers are aware of specific patterns that indicate critical process deviations and need to be traced back to their root-cause. However, manual screening of wafermaps requires a high effort and might be linked to subjective decisions. Although automated state-of-the-art procedures have been proposed in literature, these are mainly based on pass/fail or categorical data (binning), which impedes detecting process deviations at an early stage. In contrast, analog wafer test data permit to recognize patterns before they lead to violations of the specification limits. Hence, the use of analog data reduces yield loss and time until a deviation is detected. While there is a substantial body of research on binary wafermap data (pass/fail), to the best of our knowledge, there is little research available about how to make use of this potential advantage due to analog wafermaps.

In this work we fill this gap by presenting a first step towards an automatic assessment of analog wafermaps. Specifically, we propose two approaches to extract features from analog wafermaps: (A) a classical image processing approach based on restoration and specifically engineered features, and (B) a deep learning approach based on convolutional variational auto-encoders. To verify that the extracted features can distinguish different wafermap patterns, we conduct experiments on two datasets: i) a synthetic evaluation dataset based on prototype patterns, and ii) a real-world dataset of analog wafermaps. For evaluation, the features extracted via both approaches are used to cluster wafermaps, and the obtained clusters are validated internally (Average Silhouette Coefficient) and externally (Normalized Mutual Information). Our in-depth analysis of cluster assignments indicates that both approaches perform similarly, both quantitatively and qualitatively. Moreover, we show that the features extracted of both approaches can be used to distinguish different patterns and may thus be used in (semi-)supervised classification settings. We conclude our work by comparing the computational requirements of each

approach and by providing guidelines on choosing among the proposed approaches based on, e.g., the availability of data and computational resources.

II. Related Work

Several semiconductor manufacturers and research institutes have tackled the problem of automated wafer test data or wafermap analysis. While most research projects dealing with wafer test data focus on the detection of single-chip anomalies (i.e. outliers) to address product quality issues, e.g. [1], [2], a minor part utilizes automated methods for production process monitoring and failure detection.

An example for automated approaches to detect failure patterns in wafermaps is provided by Wu et al. [3]. They propose a feature extraction method to perform pattern recognition on large databases in a reasonable computation time, improving state-of-the-art approaches. In contrast to most other available research papers dealing with this topic, they consider rotation-invariant patterns. However, the presented approaches are tailored to describe patterns in discrete instead of analog wafer test data. Another closely related idea was recently introduced by Taha et al. [4], who apply an unsupervised learning (clustering) method to wafermaps. They investigate defect patterns using Voroni regions, i.e., a segmentation approach of a wafermap to detect defect areas. The developed "DDPfinder" identifies spatial clusters by their centroid points and correlates them to other wafermaps. Chen and Liu [5] present an approach for pattern recognition in wafer bin maps, using the so-called ART1 neural network pipeline. This idea is extended and refined in later works by Liu et al., e.g., in an improved network setting [6] and by introducing a wavelet transform [7]. More recently, an approach is presented by Alawieh et al. [8], who suggest wafermap clustering after removing random failures using singular value decomposition.

There is a growing body of literature on deep neural networks dedicated to clustering (e.g. Guo et al. [9] and references therein) and to feature extraction with deep generative models, specifically with the variational auto-encoder [10], [11]. The work by Kyeong and Kim [12] and Nakazawa and Kulkarni [13] exemplifies the applicability of deep convolutional neural networks to classify wafer bin map defect patterns of mixed-types and to classify and retrieve images of density-based wafermap defect patterns.

All of those works use pass/fail, bin wafermaps or defect densities instead of the original measurement values. The resulting patterns show violations of the specification limits, but fail to indicate whether pass-devices are close to the limits or not. As a result, these methods cannot be applied to prevent upcoming production process issues, which do not yet violate the specification limits. Further, a distinction between different root-causes of errors is not easily possible in a setting where only pass/fail information is used.

In contrast, the solution proposed in this work focuses on the analysis of spatial patterns in the original (analog) wafer test data, i.e., a multivariate set of measurements is available for each device. Variations of single patterns are considered, such as rotations or translations, which are not covered by other approaches. Only little research has been performed along this direction. For example, Rostami et al. [14] present a whole machine learning pipeline, consisting of binary classifiers, projection and clustering methods. Their aim, however, is to detect and classify faults in wafer production data, while our focus is more generally on extracting features that can be used in unsupervised settings, too. Another interesting work on semiconductor industry is presented by Bao, Wang and Jin [15], who apply Gaussian Markov Random Fields to create a spatial model of wafer thickness. However, the work focuses on modelling material parameters instead of recognizing electrical failure patterns.

None of the cited approaches is directly comparable to our method, mainly because the target differs: state-of-the-art algorithms analyze pass/fail (or bin) data over wafermaps to detect patterns produced by devices violating the specifications. Hence, the failure mode already induces a yield loss. In this paper, a novel approach of analyzing the analog measurement data is deployed, which permits the user to detect evolving patterns before they cause a violation of the specification limits. This focus on analog wafermaps also leads to data properties fundamentally different from those of pass/fail or bin wafermaps: It is unusual in practice to observe analog wafermaps without a pattern (i.e., pure noise), as most measurements depict certain spatial dependencies. Therefore, the occurrence of patterns is the normal situation for analog wafermaps, whereas their absence is normal for pass/fail or bin wafermaps. In case that mixtures of multiple patterns on one wafermap occur, demixing such combinations can be done using independent component analysis, as demonstrated for analog wafer test data by Zernig et al. [16], or using nonnegative tensor factorization, demonstrated by Siegert et al. [17].

III. Feature Extraction Methods

For both proposed methods, the classical and the deep learning approach, we assume that there are M wafers with n devices each. The i-th device of a given wafer is represented by a real-valued measurement d_i; we collect all measurements of a given wafer in a vector $\mathbf{d} = (d_1, \ldots, d_n) \in \mathbb{R}^n$. If for a subset of devices the measurement is missing, we replace these missing values by the median measurement value of the other devices in the spatial neighborhood (approach (A)) or on the whole wafer (approach (B)), respectively.

A. Approach (A): Classical Image Processing

Our first approach is based on a classical image processing procedure. It consists of a denoising step based on a Markov Random Field model and a feature extraction step that combines Local Binary Patterns and Rotated Local Binary Patterns.

Markov Random Fields (MRFs). An MRF is a probabilistic model defined by a graph G of random variables (RVs). In our case, the set of the graph nodes corresponds to the set of devices on a wafer, $S = \{1, \ldots, n\}$. The edge set of G is specified by the spatial neighborhoods between the devices (adjacency). We assume a regular 8-neighborhood structure, i.e., the node of device i is connected to the nodes of its

horizontal, vertical, and diagonal neighbors; let $\mathcal{N}_i \subset S$ denote the index set of nodes neighboring node i. It is assumed that each (noiseless) device value x_i is a realization of an RV X_i associated with node i in the graph G. The definition of an MRF implies that in the graph G, each RV X_i, $i \in S$ is conditionally independent from all other RVs given the RVs associated with its neighbors $j \in \mathcal{N}_i$.

Denoising using MRFs. The MRF model for wafermap denoising is explained in detail in [18] which is based on the book of Li [19]. The corresponding image processing model, which was originally introduced by Geman and Geman [20], belongs to the class of Spatial Domain Filters, i.e., filtering local groups of adjacent image pixels directly from the original image to reduce noise. Specifically, the basic model describing the denoising task is given as

$$\mathbf{d} = \mathbf{x} + \boldsymbol{\epsilon}, \tag{1}$$

where $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ is the vector of noiseless device measurements and $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n) \in \mathbb{R}^n$ denotes the vector of noise values and unwanted influences. The goal is to provide an appropriate estimation of $\mathbf{x}$, assuming that $\mathbf{x}$ is such that measurements of adjacent devices on the same wafer are not too different and that the entries of $\boldsymbol{\epsilon}$ are all stochastically independent and normally distributed with zero mean and variance σ^2.

An explicit maximum a-posteriori MRF exploiting the assumption of normally distributed errors for the likelihood and a smoothing act as a prior, is given by

$$\hat{\mathbf{x}}_{\text{MAP}} = \underset{\mathbf{x}\in\mathbb{R}^n}{\arg\min} \sum_{i\in S} \left[\frac{(x_i - d_i)^2}{2\sigma^2} + \sum_{j\in\mathcal{N}_i} (x_i - x_j)^2 \right], \tag{2}$$

which is solved to infer the noiseless device values. For details, see [21].

(Rotated) Local Binary Patterns ((R)LBPs). LBPs are a texture-based image feature description method, proposed by Ojala et al. [22] based on an idea by He and Wang [23], which can be quickly calculated because of their simplicity. In the terminology of this paper, the basic principle of LBPs is to compare each denoised device measurement x_i at site i to its neighbors x_j, $j \in \mathcal{N}_i$. For each $j \in \mathcal{N}_i$, define y_j as follows:

$$y_j = \begin{cases} 1, & x_j \geq x_i \\ 0, & \text{else.} \end{cases} \tag{3}$$

For each position i, the LBP value is defined as

$$LBP_i = \sum_{j=0}^{7} \tilde{y}_j 2^j = \sum_{j\in\mathcal{N}_i} y_j 2^{\pi_i(j)}, \tag{4}$$

where $\pi_i : \mathcal{N}_i \to \{0, \ldots, 7\}$ indicates a clockwise numbering of the neighborhood set, starting with the right neighbor. Therefore, LBP_i is a decimal representation of the binary number $(\tilde{y}_0, \ldots, \tilde{y}_7)_2$, which represents the structure of the pixels neighboring the i-th device. While binary numbers containing more than two 0-1- or 1-0-transitions are often neglected as they might represent noisy image pixels (so-called non-uniform patterns), Mehta and Egiazarian [24] showed that also non-uniform patterns may contain essential information. Considering also these non-uniform patterns, we obtain 256 possible values for LBP_i.

Unfortunately, LBPs are not invariant w.r.t. rotations. This issue was resolved by Mehta and Egiazarian [24] who introduced a variant called Rotated LBPs (RLBPs). For RLBPs, the neighborhood is oriented along the "dominant direction": the positions of the digits in the binary number $LBP_i = (\tilde{y}_0, \ldots, \tilde{y}_7)_2$ are shifted such that the largest absolute difference, i.e. $\max_{j\in\mathcal{N}_i} |x_j - x_i|$, corresponds to the first position. Therefore we shift by D_i positions, where

$$D_i = \pi_i \left(\underset{j\in\mathcal{N}_i}{\arg\max} |x_j - x_i| \right). \tag{5}$$

In mathematical terms, this means that (4) can be reformulated for RLBPs as

$$RLBP_i = \sum_{j\in\mathcal{N}_i} y_j 2^{(\pi_i(j) - D_i)_{\text{mod}8}}, \tag{6}$$

where $(\cdot)_{\text{mod}.}$ denotes the modulus operator.

Feature Extraction using (R)LBPs and PCA. For feature extraction, we select relevant wafer regions by thresholding, extract histograms based on LBPs and RLBPs, and then reduce the resulting features via principal component analysis (PCA).

The wafer regions from which LBP- and RLBP-based histograms are computed are determined by Otsu's thresholding method [25], which is an automated criterion to perform image thresholding. This means that the smoothed image is segmented into regions, where histograms are computed only on regions $R \subseteq S$ exceeding the Otsu threshold.

We then compute LBP_i, defined in (4), for each device $i \in R$ (in the computation, we make sure not to include pixels from outside the region of interest). The LBP-based features of the wafer are finally obtained by calculating histograms of LBP_i for all $i \in R$. Since each LBP_i can take up to 256 different values, we obtain a 256-dimensional LBP-based feature for each wafer. Similarly as for the LPBs, also histograms for the RLPBs are computed from the RLPB values of the devices in R. Combined with the LPB-based histogram, this leads to a 512-dimensional feature vector for each wafer. We reduced the feature space via PCA by retaining only the first three principal components. Thus, each wafer is represented by a three-dimensional feature vector z.

B. Approach (B): Deep Learning

Our second approach is based on convolutional neural networks trained in an auto-encoding setup. As a result, every wafer is represented as a point in a low-dimensional space.

Auto-Encoder Theory. Auto-encoders are neural networks that compress their input to a lower-dimensional representation via an encoder and decompress it back to the original input space via a decoder. The encoder (decoder) is represented as neural network layers of progressively less (more) neurons, which, as with other deep neural networks, are trained via backpropagation. The loss function of auto-encoders reflects the distance between original input and the auto-encoder's decoded output (e.g., binary cross-entropy). Therefore, auto-encoders aim to faithfully reproduce given input data as far

as the constraint of encoding input data to lower-dimensional representations allows. Auto-encoders see application in dimensionality reduction and data-denoising problems.

Variational Auto-Encoder Theory. The variational auto-encoder is a Bayesian deep learning technique, which learns latent data representations given the presence of large amounts of data and intractable posterior distributions.

In variational inference, the goal is to find an approximation to an intractable probability distribution in a class of tractable probability distributions. It is assumed that data d_i, $i = 1, \ldots, n$ is generated by an unobserved continuous latent RV z via the likelihood $p_\theta(\mathbf{d}|z)$ and the prior $p_\theta(z)$. These are assumed to be parametrized by θ and to be unknown Gaussian distributions. The posterior distribution $p_\theta(z|\mathbf{d})$ and the marginal likelihood $p_\theta(\mathbf{d})$ are assumed to be intractable, so they need to be approximated with parametric families of Gaussian probability distributions $q_\phi(z|\mathbf{d})$. In variational inference, one solves this problem by maximizing the *evidence lower bound* for $p_\theta(\mathbf{d})$, which is given by:

$$\log(p_\theta(\mathbf{d})) \geq -D_{KL}(q_\phi(z|\mathbf{d})\|p_\theta(z)) + \mathrm{E}_{q_\phi}[\log(p_\theta(\mathbf{d}|z))], \tag{7}$$

where $D_{KL}(\cdot\|\cdot)$ is the Kullback-Leibler divergence. The auto-encoder encodes input $\mathbf{d}$ to a lower-dimensional representation z and then decodes it back to $\mathbf{d}$, with the goal to maximize the right-hand side of (7). First, maximizing $-D_{KL}(q_\phi(z|\mathbf{d})\|p_\theta(z))$ yields a compact representation of input data $\mathbf{d}$ by the latent variable z. Second, in a conflicting optimization objective, maximizing $\mathrm{E}_{q_\phi}[\log(p_\theta(\mathbf{d}|z))]$ leads to an accurate reconstruction $\hat{\mathbf{d}} = (\hat{d}_1, \ldots, \hat{d}_n)$ of input $\mathbf{d}$ given z via the decoder. Since (7) represents a lower bound on the log-likelihood, maximizing this objective function during training the auto-encoder ensures that the parameter estimation is getting more accurate. In particular, the so-called reparametrization trick allows for end-to-end optimization of the auto-encoder via stochastic gradient descent methods, despite the probabilistic setting. For more details, see Kingma and Welling [10] and Rezende et al. [11].

Convolutional Auto-Encoding Neural Networks. Neural networks with *convolutional layers* form the basis of many image classification neural networks. One reason for that is that they escape the curse of dimensionality present in large images. Crucially, convolutional layers leverage the spatial structure of images to constrain the neural network's architecture and thereby reduce the amount of parameters required to extract higher-level feature representations of input images. Since convolutional layers perform differentiable operations, these so-called convolutional neural networks can be trained with the same methods as used for regular multi-layer neural networks.

The convolutional layer's weights are typically termed filters. The filters function as small sliding windows which are convolved with an input matrix (or image), i.e., sequentially multiplied over subsets of its columns (width) and rows (height). The sliding window's jump size is called stride. In this paper, convolutional layers are used in the encoder and deconvolutions [26] in the decoder. Deconvolutions, also known as transposed convolutions, also perform convolutions like the encoder's convolutional layers, but they do so in the reverse direction. In particular, in our auto-encoder setup, the encoder's (decoder's) convolved output becomes lower-(higher-)dimensional with each layer.

Feature Extraction Using Convolutional Variational Auto-Encoders (CVAEs). This work employs a deep auto-encoder architecture for wafermaps as described by Santos and Kern [27]. Each device on the wafer and its measurement can be interpreted as pixel and color value, respectively, connecting the problem to image processing. For each wafer, all measurements are scaled to fall within the range $[0, 1]$, which allows interpreting them as greyscale values. Next, all wafermaps are converted to images with a resolution of 112×112 pixels. This allows to reuse the same CVAE architecture for different semiconductor products, but potentially loses the one-to-one correspondence between devices on the wafer and pixels in the image. Our experiments show that this correspondence is of little importance for the task of detecting measurement patterns.

Our CVAE has the following architecture: The encoder consists of four convolutional layers followed by two fully-connected layers of 128 neurons and two neurons, respectively. The output of the two neurons represent the latent variable z; the number of dimensions was set to two to allow visualizing the latent variable. The decoder then consists of two fully-connected layers that embed the latent variable in a higher-dimensional space. Four deconvolutional layers follow, which map the activation volumes back to the original input image size. Our architecture does not contain pooling layers: In generative models such as ours, repeated convolutions are preferred over pooling [28], a claim which we empirically confirmed in preliminary experiments with the dataset described in Section IV-B. As far as the convolutional layers are concerned, we employ 128 filters of size 3×3 with a stride of two. All layers use the rectified linear unit as activation function.

We selected an existing implementation[1] of a variational auto-encoder and adapted it to our needs. Note that this (and many other) implementations replace the second term on the right-hand side of (7) by the empirical cross-entropy, i.e., by

$$-\sum_{i=1}^{n}\Big(d_i \log(\hat{d}_i) + (1-d_i)\log(1-\hat{d}_i)\Big) \tag{8}$$

where $\hat{d}_i$ is the reconstruction of the measurement of the i-th device by the auto-encoder. This loss function was minimized using the RMSprop optimizer with a batch size of 10.

IV. Experiments & Results

To compare the classical image processing-based approach (A) from Section III-A to a representative instantiation of the deep learning approach (B) from Section III-B, we present experiments using synthetic and real-world wafer test datasets.

A. Evaluation criteria

We evaluate both approaches (A) and (B) by means of clustering the extracted feature vectors. Specifically, we perform hierarchical agglomerative clustering with average linkage

[1] github.com/keras-team/keras/blob/master/examples/variational_autoencoder.py

computed with the Euclidean distance. A predefined number of clusters is chosen for each of the datasets. The clusters are then evaluated by inspection as well as with internal and (if a ground truth is available) external validation measures: the Normalized Mutual Information (NMI) [29] and the Average Silhouette Coefficient (ASC) [30], respectively.

NMI. Let P_i denote the set of wafermaps in the test set with pattern i, $i = 1, \ldots, p$ and let C_j denote the set of wafermaps in the test set mapped to cluster j, $j = 1, \ldots, k$, where p and k are the number of patterns and clusters, respectively. Given that M_{test} is the number of elements in the test dataset, the entropy of the clustering is

$$H(\mathcal{C}) = -\sum_{j=1}^{k} \frac{|C_j|}{M_{\text{test}}} \log \frac{|C_j|}{M_{\text{test}}}$$

(and similarly, the entropy $H(\mathcal{P})$ of the pattern assignment), and the mutual information between the clustering and the pattern assignment is

$$I(\mathcal{C};\mathcal{P}) = \sum_{j=1}^{k}\sum_{i=1}^{p} \frac{|C_j \cap P_i|}{M_{\text{test}}} \log \frac{M_{\text{test}}|C_j \cap P_i|}{|C_j||P_i|}.$$

The NMI is defined as

$$NMI(\mathcal{C};\mathcal{P}) = \frac{2I(\mathcal{C};\mathcal{P})}{H(\mathcal{C}) + H(\mathcal{P})}$$

and coincides with the V-measure.

Therefore, the NMI quantifies the ability of the clustering method to identify the underlying patterns, similar to supervised learning evaluation measures like the F1-measure. However, the NMI considers that clustering might mix the label ordering or return a different number of clusters than specified patterns (ground truth). The NMI takes values in $[0, 1]$, where 1 means that the clustering perfectly matches the real labels.

ASC. In contrast to the NMI, the ASC is an internal evaluation measure, i.e., it assesses the quality of the clustering without considering a ground truth. Wafers are represented by feature vectors z in some feature space F ($\mathbb{R}^3$ for approach (A) and $\mathbb{R}^2$ for approach (B)). Let d be a distance metric in this feature space – in our case, the Euclidean metric $d(v, z) = \|v - z\|_2$ is used, where $v, z \in F$. Further, let

$$d(z, A) = \frac{1}{|A|} \sum_{v \in A} d(z, v)$$

denote the distance between wafer feature $z \in F$ and a finite set $A \subset F$. We define the silhouette s: $F \to \mathbb{R}$ of a wafer feature $z \in F$ as

$$s(z) = \frac{d(z, C(\not z)) - d(z, C(z))}{\max\{d(z, C(z)), d(z, C(\not z))\}},$$

where $C(z)$ is the cluster to which z is assigned and $C(\not z)$ is the "nearest" cluster to z excluding $C(z)$. The ASC values of the clustering are defined as

$$ASC = \frac{1}{M_{\text{test}}} \sum_{z} s(z),$$

where the sum runs over the feature vectors of all M_{test} wafers. The ASC takes values in $[-1, 1]$, with values above 0.5 and 0.75 indicating a medium and strong cluster structure.

B. Synthetic Dataset

Dataset. The dataset consists of $M = 5000$ different wafermap images, containing one out of five distinctive patterns, see Fig. 1a-1e. The dataset is balanced, i.e., each pattern occurs on 1000 wafermaps. Each pattern is varied in size and intensity. The patterns are characterized as follows:

- Pattern 1 is a ring-pattern along the border of the wafer.
- Pattern 2 is a single circular or elliptic spot at an arbitrary position on the wafer.
- Pattern 3 is constant gradient over the whole wafermap, but changes w.r.t. the direction.
- Pattern 4 are two spots at the left and right wafer edge.
- Pattern 5 is a crescent-shaped area at the right edge of the wafer.

The selection of the five pattern types for this experiment is based on observations in real-world wafer test data. While it might not cover the whole spectrum of patterns occurring in production, these dominant characteristics shall be considered as a benchmark for method evaluation and comparison.

In addition to these specifications, Gaussian white noise and outliers are added to the simulations, as well as a linear transformation to an arbitrary data scale. The intensity of noise, as well as the number of outliers are selected in a randomized way, for each wafermap individually. This guarantees to provide realistic simulations of production data.[2]

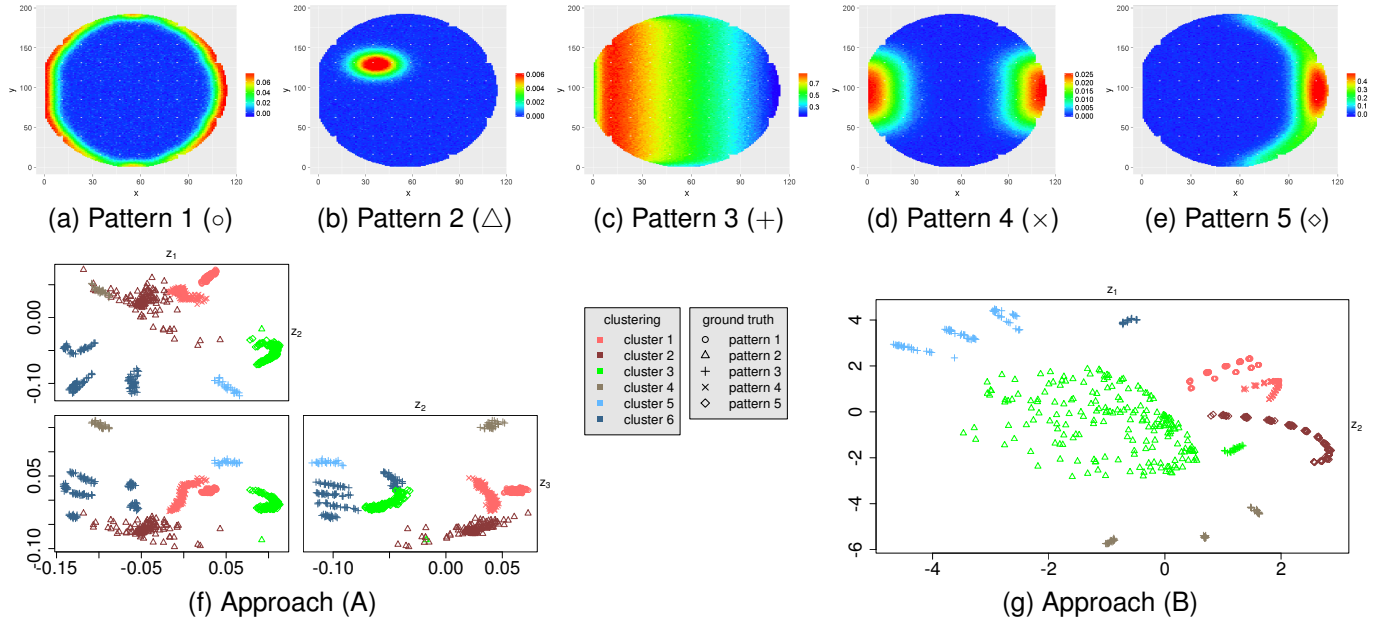
In the following experiment, approaches (A) and (B) are applied as follows: the available dataset is divided into $M_{\text{train}} = 4000$ training samples and $M_{\text{test}} = 1000$ test samples. The training set is used to estimate the PCA parameters of approach (A), as well as to train the auto-encoder for approach (B). In order to guarantee comparability, both approaches use the same split of the data samples.

Results. The results for both approaches are presented in Fig. 1 for four, five, and six clusters. Specifically, Figs. 1f and 1g show the scatterplots of the three- and two-dimensional features of approaches (A) and (B) for six clusters, with color indicating cluster assignment and marker styles indicating patterns. If, instead of six clusters, five clusters are selected, then clusters C_1 and C_2 are merged (red and dark red). If four clusters are selected, additionally, clusters C_5 and C_6 coincide (blue and dark blue).

To study the influence of individual patterns, we computed confusion matrices for both approaches for the number of clusters maximizing the NMI. The obtained confusion matrices in Figs. 1h and 1j present the number of wafermaps of each pattern assigned to each cluster. Note that – in contrast to classification problems – the order of the clusters is randomly assigned by the algorithm, i.e., any permutation of the rows is valid in the confusion matrix.

According to Fig. 1h, approach (A) distinguishes between pattern 1, 2, 3 and 5 correctly by assigning corresponding wafermaps to disjoint clusters, if the number of clusters is set to 6. However, patterns 1 and 4 are mapped to the same cluster C_1, which indicates that approach (A) is not able to

[2]For the full set of synthetic data, see https://zenodo.org/record/2542504, DOI: 10.5281/zenodo.2542504



		Pattern				
		P_1	P_2	P_3	P_4	P_5
Cluster	C_1	200	0	0	200	0
	C_2	0	199	0	0	0
	C_3	0	1	0	0	200
	C_4	0	0	27	0	0
	C_5	0	0	24	0	0
	C_6	0	0	149	0	0

(h) Approach (A)

	k	(A)	(B)
ASC	4	0.63	0.47
	5	0.70	0.45
	6	0.71	0.55
NMI	4	0.66	0.65
	5	0.70	0.64
	6	0.86	0.81

(i) Cluster Validity

		Pattern				
		P_1	P_2	P_3	P_4	P_5
Cluster	C_1	200	0	0	200	0
	C_2	0	0	0	0	200
	C_3	0	200	24	0	0
	C_4	0	0	66	0	0
	C_5	0	0	83	0	0
	C_6	0	0	27	0	0

(j) Approach (B)

Fig. 1. Clustering results of approach (A) and (B) on a synthetic dataset. (a)-(e) display prototypes of the five used patterns. (f) displays the pairwise scatterplot of the 3 principal components, resulting from approach (A), (g) displays the scatterplot of the two output features from approach (B). In (f) and (g), color indicates clusters, while marker styles indicate patterns. (h) and (j) show the confusion matrices for clustering, given an optimal number of clusters, for approach (A) and (B), respectively. The table entries in (h) and (j) indicate the number of wafermaps with a given pattern mapped to a certain cluster. Table (i) demonstrates the values of the evaluation criteria, NMI and ASC, for both approaches at a number of $k = 4$, $k = 5$ and $k = 6$ clusters.

separate them. Wafermaps showing pattern 3 are assigned to three distinct clusters C_4, C_5, and C_6.

Obviously, clustering pattern 3 is a challenge for approach (A) due to the different gradient directions, which need to be merged. For this purpose RLBP features are essential, but due to the influence of LBP, two gradient directions are separated from all others (see clusters C_4 and C_5 in Fig. 1h). The influence of gradient directions, leading to different "subclusters" of cluster C_6, is visible in the pairwise scatterplot of the PCA components used in approach (A), see Fig. 1f.

Thanks to the image segmentation step by Otsu's thresholding method, the size of the region of interest is implicitly taken into account as a minor feature for the evaluation (i.e., a larger region will result in a more accurate LBP/RLBP histogram describing the underlying distribution). This explains the fact that pattern 2 and 3 can be easily distinguished from each other and from all other pattern types.

For approach (B), the auto-encoder returns the two-dimensional feature shown in Fig. 1g. First of all, it can be seen that all patterns are well-separated even in a two-dimensional latent space. This shows that approach (B) is valid as a feature extraction method for pattern classification in a supervised or semi-supervised scenario.

Second, properties of patterns are represented in the latent space as well: The cloud of points in the center of the image corresponds to pattern 2, which is characterized by arbitrary positions on the wafer. The patterns 1, 4, and 5, which are grouped into clusters C_1 (red) and C_2 (dark red), are characterized by different sizes, which is reflected in the fact that the corresponding features z lie on approximately one-dimensional manifolds. As in approach (A), pattern 3 is split into clusters C_3 to C_6, each corresponding to a set of directions of the gradient. This suggests that the different gradient directions of pattern 3 cannot be clustered to a single cluster, but are separable and can thus easily be detected in a (semi-)supervised classification setting.

In general, approaches (A) and (B) yield similar results on the simulated dataset, which is underlined by their corresponding NMI values (see Fig. 1i). The structure of the feature space is also comparable: pattern 3 is divided into a set of subclusters, one for each gradient direction. Pattern 2, i.e., a spot at variable positions on the wafer, is represented by one widespread cluster. Pattern 1 and 4, both characterized by areas at the border of the wafer, cannot be distinguished. Regarding the quantitative comparison, approach (A) outperforms approach (B) w.r.t. the ASC evaluation. This might be caused by the different dimensions of the feature spaces (3 for approach (A), 2 for approach (B)). However, a more accurate evaluation

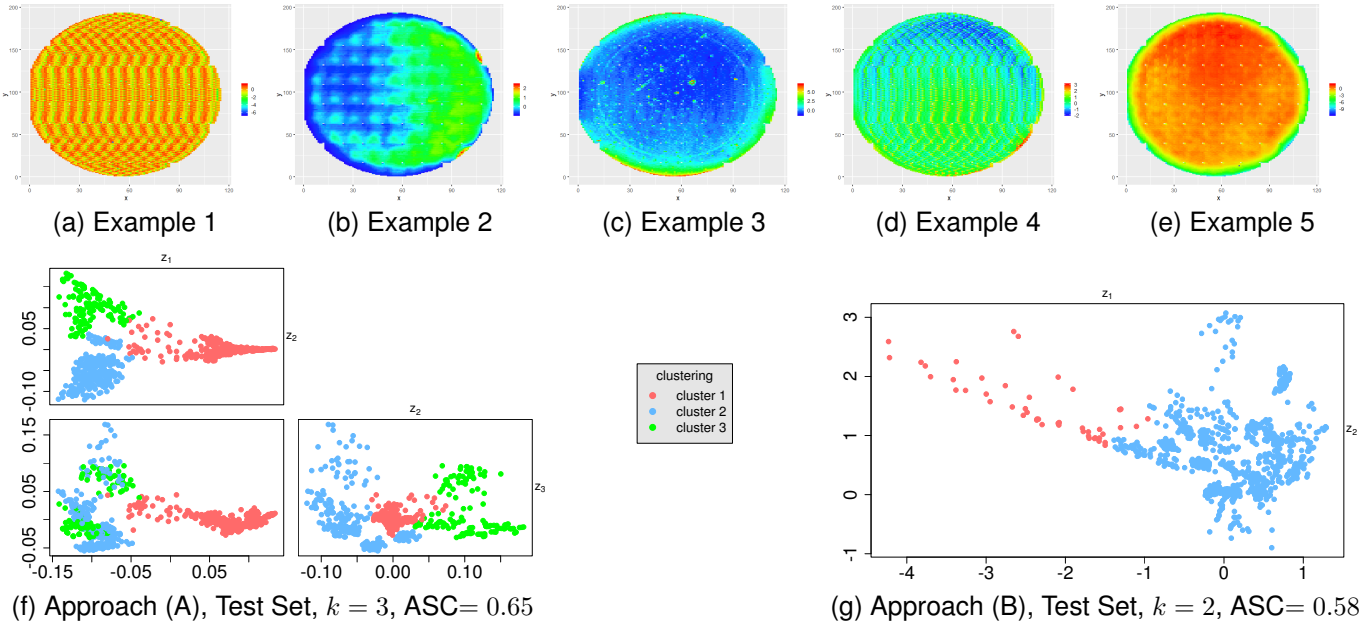


Fig. 2. Clustering results of approach (A) and (B) on a real-world dataset. (a)-(e) display five example wafermaps from the dataset. (f) displays the pairwise scatterplot of the 3 principal components, resulting from approach (A), (g) displays the scatterplot of the two output features from approach (B). In (f) and (g), color indicates clusters.

of the simulated dataset can be done by the NMI, which compares the clustering results to the ground truth. In this respect, both approaches show accurate performances, especially considering the high complexity induced by different types of variations between the patterns (e.g., position-invariance for pattern 2, rotation-invariance for pattern 3, etc.).

Intuitively, an increase of the number of PCA components in approach (A) or the dimensionality of the latent space in approach (B) should improve the clustering performance. Hence, we additionally investigate the effect of setting the feature space to 10, 15 and 20 dimensions. Our results show that for each four, five, and six clusters the NMI values decrease to approx. 0.15 – 0.64 (approach (A)) and to approx. 0.2 – 0.4 (approach (B)). In both approaches, the distinct gradient directions contained in pattern 3 are even more accentuated in the higher dimensional space, although being intra-class variations and therefore irrelevant for clustering. As a result, pattern 3 is more dispersed on several clusters, while pattern 1, 2 and 4 are successively merged to a single cluster - therefore, the negative effects observed in the lower-dimensional space are further intensified. Although three PCA components only cover 40% of the total variance in approach (A), this strong reduction of dimensionality is not only beneficial for interpretability and presentability, but rather necessary to reduce intra-class effects.

C. Real-World Dataset

Dataset. To demonstrate the performance of the presented approaches on a real-world dataset, the following use-case will be considered: wafer test data of a semiconductor product with six lots are investigated. In total, 21 electrical parameters are measured on each device, which results in a total number of approx. 6000 wafermaps. A split into a training set and a test set is performed by choosing $M_{\text{train}} = 4935$ training wafermaps (five lots) and $M_{\text{test}} = 1029$ wafermaps (one lot) for evaluation. (For approach (B), training and test datasets were reduced to be integer multiples of the batch size, i.e., to $M_{\text{train}} = 4930$ and $M_{\text{test}} = 1020$.) Visual inspection reveals different patterns at different levels of intensity on most analog wafermaps, while few show spatially independent measurement noise. Major patterns are depicted in Fig. 2. Since a correct "labeling" indicating the ground truth cannot be assumed for real data, the evaluation of the real-world dataset will be based on the ASC. The number of clusters is determined by the optimal ASC value reached on the training dataset for each method separately. Apart from this, the same experimental setup as for the synthetic dataset is applied.

Results. Using the image features from approach (A), the real-world dataset is divided into $k = 3$ clusters, which yields the best ASC value. The clustering results for the test set are depicted in Fig. 2f. The ASC value for approach (A) is 0.65, which matches the range of the ASC for the simulated data, i.e., the clustering performance is similar.

The features extraced using approach (B) for the test set are shown in Fig. 2g. On the training set, the ASC was maximized for $k = 2$ clusters to a value of 0.7. The second maximum of the ASC at 0.56 was achieved for $k = 3$. Therefore, we choose $k = 2$ for clustering the test set, achieving an ASC of 0.58 for the clustering shown in Fig. 2g.

In general, the behavior of approaches (A) and (B) on the real dataset hardly deviates from the observations on the simulated dataset. Regarding the clustering results, the ASC yields similar values, while the number of clusters is reduced for both approaches. As no ground truth is available, an external evaluation measure (e.g., NMI) is not applicable. Nevertheless, a verification via expert judgement is possible,

evaluating whether intra-cluster similarities exist between the wafermaps. Indeed, known relations from domain knowledge between the wafermaps (e.g., wafermaps originating from closely related electrical measurements on the same wafer) can be retrieved by the clustering.

In detail, the largest clusters (w.r.t. their cardinalities) in the real-world dataset are cluster C_1 in approach (A) and cluster C_2 in approach (B). These collect mainly wafermaps with unspecific, noisy or weak patterns, see Fig. 2a, depicting an example of cluster C_2 in approach (B), and Fig. 2d, depicting an example of cluster C_1 in approach (A). The other clusters, i.e., cluster C_2 and C_3 in approach (A) and cluster C_1 in approach (B) contain patterns showing a higher level of intensity: Fig. 2b demonstrates one example of cluster C_2 in approach (A), characterized by a gradual increase of the measurement values from the left to the right side of the wafer. Fig. 2c, in contrast, shows an evolving border pattern, mainly affecting the bottom side of the wafer, assigned to cluster C_3 in approach (A). Similarly, Fig. 2e, shows a ring-pattern, similar to pattern 1 in Fig. 1a from the simulated dataset, which is assigned to cluster C_1 by approach (B). Hence, both approaches are able to distinguish between noisy patterns and those depicting e.g. regions of interest, i.e., potential process patterns. In summary, the suggested methods deliver a plausible partitioning of the wafermaps.

V. Discussion & Conclusion

In a first step towards automated root-cause analysis of deviations in semiconductor frontend production, this work addressed the problem of extracting patterns from wafermaps, i.e., a spatial view of test measurements of devices on the wafer. The problem was approached with methods based on classical as well as deep learning-based image processing and evaluated in an unsupervised clustering framework. Regarding computational complexity, we note that the bulk of the computational overhead of approach (B) lies at training time and not in the evaluation. To train on the real-world dataset using commodity hardware, approach (A) requires 1.5–2 hours and approach (B) requires 3 hours, whereas both, once trained, generate outputs within few minutes. Hence, we believe both approaches are applicable for productive usage. Although the quantitative and qualitative performances are similar, the classical image processing approach shows two beneficial characteristics: First, a lower amount of training data is required (in detail, only the estimation of the PCA parameters requires training), which is advantageous if a new product is launched or a production process step is modified. Second, the features extracted from the wafermaps are more comprehensible and interpretable, which makes them more accessible for wafer production experts. However, the deep learning-based approach can benefit from an increasing amount of training data in a way impossible for the classical approach. Moreover, the applicability of the classical approach is limited by the complexity of the pattern structure. Thus, we believe that, e.g., arrangements of multiple spots, lines, etc. or combinations of those on the wafer might be hard to recognize or distinguish.

In summary, the auto-encoder approach (approach (B)) is more suitable under the following three conditions:

- large amounts of training data are available, together with high-performance hardware to process it,
- training time is rather uncritical,
- complexity of the patterns requires a more flexible model.

However, in case that one of these aspects is not fulfilled (especially if only little data is available), the classical approach should be preferred.

In a practical environment in semiconductor industry, pattern types are not constant over time. While known, product-specific process patterns are likely to reoccur, previously unknown patterns need to be considered in addition. Therefore, future work will consider methods for online or incremental learning with a focus on small datasets.

The presented clustering results indicate that important patterns are separable in feature space for both the classical and the deep learning-based approach. This suggests that both types of features are applicable in (semi-)supervised settings for automatic pattern recognition as well.

Acknowledgment

The authors thank Martin Pleschberger (KAI) for supporting our experiment by providing the simulated dataset. The work has been performed in the project Power Semiconductor and Electronics Manufacturing 4.0 (SemI40), under grant agreement No 692466. The project is co-funded by grants from Austria (BMVIT-IKT der Zukunft, FFG project no. 853338), Germany, Italy, France, Portugal and - Electronic Component Systems for European Leadership Joint Undertaking (ECSEL JU). The Know-Center is funded within the Austrian COMET Program - Competence Centers for Excellent Technologies - under the auspices of the Austrian Federal Ministry of Transport, Innovation and Technology, the Austrian Federal Ministry of Digital and Economic Affairs, and by the State of Styria. COMET is managed by the Austrian Research Promotion Agency FFG.

References

[1] A. Zernig, "Device level maverick screening," Ph.D. dissertation, Alpen-Adria-Universität Klagenfurt, 2016.
[2] M. J. Moreno-Lizaranzu and F. Cuesta, "Improving electronic sensor reliability by robust outlier screening," *Sensors*, vol. 13, 2013.
[3] M.-J. Wu, J.-S. R. Jang, and J.-L. Chen, "Wafer map failure pattern recognition and similarity ranking for large-scale data sets," *IEEE Transactions on Semiconductor Manufacturing*, vol. 28, no. 1, pp. 1–12, Feb. 2015.
[4] K. Taha, K. Salah, and P. D. Yoo, "Clustering the dominant defective patterns in semiconductor wafer maps," *IEEE Transactions on Semiconductor Manufacturing*, vol. 31, no. 1, pp. 156–165, Feb. 2018.
[5] F. L. Chen and S. F. Liu, "A neural-network approach to recognize defect spatial pattern in semiconductor fabrication," *IEEE Transactions on Semiconductor Manufacturing*, vol. 13, no. 3, pp. 366–373, Aug. 2000.
[6] S. F. Liu, F. L. Chen, and W. B. Lu, "Wafer bin map recognition using a neural network approach," *International Journal of Production Research*, vol. 40, no. 10, pp. 2207–2223, 2002.
[7] S. F. Liu, F. L. Chen, and A. S. Chung, "Using wavelet transform and neural network approach to develop a wafer bin map pattern recognition model," in *Proc. Int. MultiConf. of Engineers and Computer Scientists (IMECS)*, Hong Kong, Mar. 2008.

[8] M. B. Alawieh, F. Wang, and X. Li, "Identifying wafer-level systematic failure patterns via unsupervised learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 4, pp. 832–844, Apr. 2018.
[9] X. Guo, X. Liu, E. Zhu, and J. Yin, "Deep clustering with convolutional autoencoders," in *Proc. Int. Conf. on Neural Information Processing (ICONIP)*, Guangzhou, Nov. 2017, pp. 373–382.
[10] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. on Learning Representations (ILCR)*, Banff, Apr. 2014.
[11] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Proc. 31st Int. Conf. on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research, E. P. Xing and T. Jebara, Eds., vol. 32, no. 2, Bejing, China, Jun. 2014, pp. 1278–1286.
[12] K. Kyeong and H. Kim, "Classification of mixed-type defect patterns in wafer bin maps using convolutional neural networks," *IEEE Transactions on Semiconductor Manufacturing*, vol. 31, no. 3, pp. 395–402, 2018.
[13] T. Nakazawa and D. V. Kulkarni, "Wafer map defect pattern classification and image retrieval using convolutional neural network," *IEEE Transactions on Semiconductor Manufacturing*, vol. 31, no. 2, pp. 309–314, 2018.
[14] H. Rostami, J. Blue, and C. Yugma, "Equipment condition diagnosis and fault fingerprint extraction in semiconductor manufacturing," in *Proc. 15th IEEE Int. Conf. on Machine Learning and Applications (ICMLA)*, Anaheim, CA, Dec. 2016, pp. 534–539.
[15] L. Bao, K. Wang, and R. Jin, "A hierarchical model for characterising spatial wafer variations," *International Journal of Production Research*, vol. 52, no. 6, pp. 1827–1842, 2014.
[16] A. Zernig, O. Bluder, J. Pilz, and A. Kaestner, "Device level maverick screening - detection of risk devices through independent component analysis," in *Proceedings of the Winter Simulation Conference 2014*, Dec 2014, pp. 2661–2670.
[17] T. Siegert, R. Schachtner, G. Pöppel, and E. W. Lang, "A nonnegative tensor factorization approach for three-dimensional binary wafer-test data," in *15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec 2016, pp. 842–845.
[18] S. Schrunner, O. Bluder, A. Zernig, A. Kaestner, and R. Kern, "Markov random fields for pattern extraction in analog wafer test data," in *Proc. 7th Int. Conf. on Image Processing Theory, Tools and Applications (IPTA)*, Montreal, Nov. 2017, pp. 1–6.
[19] S. Z. Li, *Markov Random Field Modeling in Image Analysis*. Springer, 2001.
[20] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 6, 1984.
[21] M. Pleschberger, "Runtime optimization for automated pattern analysis," Master's thesis, Alpen-Adria-Universtät Klagenfurt, 2018.
[22] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, 2002.
[23] D.-C. He and L. Wang, "Texture unit, texture spectrum, and texture analysis," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 28, 1990.
[24] R. Mehta and K. Egizarian, "Dominant rotated local binary patterns (drlbp) for texture classification," *Pattern Recognition Letters*, vol. 71, 2016.
[25] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, 1979.
[26] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, Jun. 2010, pp. 2528–2535.
[27] T. Santos and R. Kern, "Understanding wafer patterns in semiconductor production with variational auto-encoders." in *Proc. 26th European Symp. on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, Bruges, Apr. 2018, pp. 141–146.
[28] J. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," in *ICLR (workshop track)*, 2015.
[29] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005.
[30] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, 1987.