2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA) | 978-1-6654-6283-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICMLA55696.2022.00025

DeepWafer: A Generative Wafermap Model with Deep Adversarial Networks

Hamidreza Mahyar*
McMaster University
Hamilton, Canada
mahyarh@mcmaster.ca

Peter Tulala
TU Wien
Vienna, Austria
ptulala@cpa.tuwien.ac.at

Elahe Ghalebi*
Vector Institute
Toronto, Canada
elahe.ghalebi@vectorinstitute.ai

Radu Grosu
TU Wien
Vienna, Austria
radu.grosu@tuwien.ac.at

***Abstract*—A certain amount of process deviations characterizes semiconductor manufacturing processes. Automated detection of these production issues followed by an automated root cause analysis has the potential to increase the effectiveness of semiconductor production. Manufacturing defects exhibit typical patterns in measured wafer test data, e.g., rings, spots, repetitive textures, or scratches. Recognizing these patterns is an essential step for finding the root cause of production issues. This paper demonstrates that combining Information Maximizing Generative Adversarial Network (InfoGAN) and Wasserstein GAN (WGAN) with a new loss function is suitable for extracting the most characteristic features from extensive real-world sensory wafer test data, which in various aspects outperforms traditional unsupervised techniques. These features are then used in subsequent clustering tasks to group wafers into clusters according to their exhibit patterns. The primary outcome of this work is a statistical generative model for recognizing spatial wafermaps patterns using deep adversarial neural networks. We experimentally evaluate the performance of the proposed approach over a real dataset.**

***Index Terms*—Generative Models, Deep Adversarial Networks, Semiconductor Wafermaps, Wafer Defect Patterns**

I. Introduction

Sustainable competitiveness in the semiconductor industry requires the rapid development of increasingly complex semiconductor products, drastically decreasing the time available for diagnosing production defects [1]. Semiconductor manufacturing is prone to production issues of two types – random defects or systematic defects. Random defects are usually attributed to the dust particles in the production environment and tend to be related to the overall cleanliness of the production environment. On the other hand, a malfunction of process equipment or human errors will cause systematic defects [2]. Due to hundreds of processing steps involved in semiconductor manufacturing, diagnosing wafers after each of these steps is not practical. Instead, equipment sensor values and electrical test data are collected only after most of the processing steps. Figure 1 depicted this procedure. It is common to treat every wafer test data (measurement) as a bitmap (called *wafermap*). Systematic defects are known to take on a typical shape in the measured wafer test data, (*e.g.* rings, spots, repetitive patterns, or scratches). Recognizing these patterns and grouping them are essential steps towards automated root cause analysis and decision-making for backtracking to which processing step caused the defects. Based on the importance of the extracted clusters, process engineers can prioritize their investigation of chip defects and pay more attention to the ongoing manufacturing processes that caused such defects. Therefore, proposing an efficient method for extracting complex structure (pattern) of a given wafer sensory data is a valuable contribution and inevitable task which intending to detect systematic defects in semiconductor manufacturing.

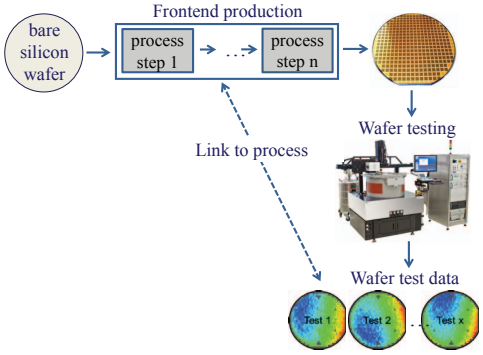


Fig. 1. Silicon wafer manufacturing process and the visualized wafermaps

II. Related work

To address the aforementioned problem, several traditional image processing approaches have been proposed [3], [4]. There exist many robust methods utilizing machine learning techniques to recognize more complex patterns in wafer test data. Most of them are based on supervised training of mixture models [5], [6], singular value decomposition [7], support vector machines [8], neural networks [9], [10] and generative adversarial network [11]. Semi-supervised learning [12] is also used by effective utilization of the unlabeled data. Although these approaches are powerful, their supervised nature still needs an expert to label the data manually.

Several unsupervised approaches [13], [14] have been proposed to eliminate subjective factors from pattern recognition task, which reduces costs and the number of clustering errors. In the industry, it is required to automatically detect the hidden dependencies between different types of wafer defects without the intervention of human expert which enables the detection of complex patterns that were unknown or overlooked before.

* Work done while at TU Wien.

978-1-6654-6283-9/22/$31.00 ©2022 IEEE
DOI 10.1109/ICMLA55696.2022.00025

To this end, some methods have been proposed, such as self-organizing neural networks [15], self-organizing maps [16], and techniques based on dimensionality reduction like diffusion maps, discriminant analysis, and variational auto-encoders [13]. This paper proposes an unsupervised generative wafermap model for recognizing spatial wafermap patterns based on deep adversarial neural networks.

III. Proposed Approach

Our available real dataset, provided by Infineon Technologies at SemI40 project (http://semi40.eu), consists of 6 wafer lots, each has 50 wafers containing 17509 chips. Each chip is measured with 20 different tests (as features) and its position within a wafer is stored as a tuple. We consider each test measurement of a wafer as a bitmap. Overall, we have 6000 wafermaps, where each one is represented as a bitmap of size 193x115 pixels. To address the problem of recognizing and classifying a given wafer sensory data structure, we develop a generative adversarial network (GAN) based method, which is explained below.

IV. Data Pre-processing

Data preprocessing is a fundamental step for applied machine learning in semiconductor manufacturing due to several data quality issues (*e.g.* measurement anomalies and missing values). In order to identify defect patterns, we must ensure that the wafer bitmaps are comparable. To this end, our data preprocessing involves: finding and removing outliers, imputing missing tests, denoising, and data normalization (for more details, see [13]).

A. Outlier Removal

We utilized a median absolute deviation (MAD)-based outlier detection method by modifying the common Z-score approach for outlier detection [17]. Given values of a wafer test measurements $X = x_1, \ldots, x_n$, the modified Z-score is defined as follow:

$$|z_i| = \frac{\phi^{-1}(3/4) \cdot |x_i - \text{med}(X)|}{\text{med}(\{x_j - \text{med}(X) \mid x_j \in X\})} \quad (1)$$

where ϕ^{-1} is the inverse of the cumulative distribution function of the normal distribution (also called probit function). The data points with $|z_i| > \lambda$ are considered as outliers, where threshold λ typically set to a constant cutoff value $\lambda = 3.5$. To consider skewness of the data distribution, MAD is calculated independently for data points greater (resp. less) than or equal to the median.

B. Clipping mask and biharmonic inpaiting

Wafemaps have irregular shapes containing some missing values (holes) within the wafer area. We first create a clipping mask of the wafer and then paint the masked area's missing values. To this end, we binarize a wafer by replacing the present values with one and the missing values with zero. After that, the mathematical morphology mechanism is used to close tiny holes in the wafer area and find contours of the wafer. Finally, missing values in the wafer area are inpainted with values reconstructed from neighborhood information around each missing region, using the Chui-Mhaskar inpainting algorithm via solving the biharmonic equations.

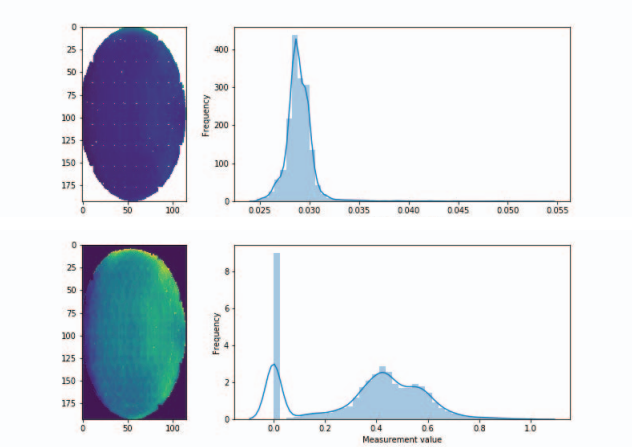


Fig. 2. A raw wafermap (top) and its cleansed one with clearly visible pattern (bottom). The resizing step was skipped for illustrative reason.

C. Median filtering

In this step, after feature normalization, wafers are smoothened in order to reduce stochastic noise. We used a simple median filtering procedure, in which each pixel of the wafer is iteratively replaced by the median value of its neighborhood in a sliding window.

D. Cleansed wafer test data

The data preprocessing procedure is a cleansed set of wafermaps used for further feature extraction and clustering tasks. In order to simplify the convolutional filters during the model training phase, all wafers are resized to 128x128 pixels. Figure 2 shows a raw wafermap and its cleansed wafermap with a clear moon pattern. The cleansed wafermaps can then be used for further tasks.

V. Unsupervised Representation Learning of Wafers

The obtained dataset from preprocessing steps can be seen as N individual datasets consisting of independent identically distributed (*iid*) samples $x_1, \ldots, x_N$ from a real data distribution p_{data}. Now, our goal is to extract low-dimensional latent representation of cleansed wafers to overcome the *curse of dimensionality* [18]. Our approach is based on deep generative adversarial neural networks.

A. Generative Adversarial Networks

A Generative Adversarial Network (GAN) [19], [20] consists of two components:

- The discriminator $D(.)$ estimates the probability of a given data sample x drawn from the real dataset with distribution p_{data}.

- The generator $G(.)$ takes a code z sampled from a noise distribution p_{noise} and generates synthetic sample $G(z)$ as realistic as possible in order to fool the discriminator. The generator learns the distribution p_G that is an approximation of the real data distribution p_{data}.

These two components are simultaneously trained to compete against each other. Samples from the actual dataset and the output of the generator are randomly passed to the discriminator. The objective of GAN can be then modeled as a two-player non-cooperative minimax game where each player attempts to optimize its payoff with value function $V(D,G)$, as:

$$\begin{aligned}\min_G \max_D \; V(D,G) = \mathbb{E}_{x\sim p_{data}}[\log D(x)] \\ + \mathbb{E}_{z\sim p_{noise}}[\log(1-D(G(z)))]\end{aligned} \tag{2}$$

This minimax game is useful for theoretical analysis of the problem. However, it does not perform well in practice. When the learning process begins, discriminator D can reject all generated samples with very high confidence and hence not provide sufficient gradient to improve the performance of generator G. [19] proposed to have two different loss functions, as:

$$\begin{aligned}\mathcal{L}_D^{\text{GAN}} &= \mathbb{E}_{x\sim p_{data}}[\log D(x)] + \mathbb{E}_{z\sim p_{noise}}[\log(1-D(G(z)))] \\ \mathcal{L}_G^{\text{GAN}} &= \mathbb{E}_{z\sim p_{noise}}[\log D(G(z))]\end{aligned} \tag{3}$$

B. Information Maximizing GAN

An information-theoretic extension to GAN, known as InfoGAN [21], was proposed in order to learn disentangled representations in an unsupervised manner. InfoGAN extends the inputs of the generator by an additional *structured* code $c \sim p_{code}$ (it can be extended to multiple structured codes). The minimax game for InfoGAN is formulated by adding a regularization term, as:

$$\min_G \max_D V'(D,G,Q) = V(D,G) - \lambda I(c; G(z,c)) \tag{4}$$

where $G(z,c)$ is a generator extended with a structured code c, $\lambda \in \mathbb{R}$ is a regularization coefficient and $I(\cdot)$ is mutual information between the structured code c and a sample drawn from $G(z,c)$. However, direct calculation of $I(c;G(z,c))$ requires an access to the intractable posterior $p(c|x)$. Instead, as shown in [21], we can obtain a lower bound of the mutual information by introducing an auxiliary distribution $q(c|x)$ to approximate $p(c|x)$, as follows:

$$\begin{aligned}I(c;G(z,c)) &= \overbrace{H(c)}^{entropy} - \overbrace{H(c|G(z,c))}^{conditional\ entropy} \\ &= \mathbb{E}_{x\sim G(z,c)}[H(c|x)] + H(c) \\ &\geq \mathbb{E}_{x\sim G(z,c)}[\mathbb{E}_{c'\sim p(c|x)}[\log q(c'|x)]] + H(c)\end{aligned} \tag{5}$$

Since c is sampled from a fixed code distribution, $H(c)$ can be constant. In practice, the auxiliary distribution $q(c|x)$ is parametrized by a neural network Q that shares all convolutional layers with D extended by one additional layer. Hence it adds only a negligible computational cost.

We used the normal distribution for the code distribution $q(c'|x)$ in our implementation, which yields:

$$\begin{aligned}I(c;G(z,c)) &\geq \ln\Big(\frac{1}{\sqrt{2\pi}\sigma}\cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}\Big) \\ &= -\frac{1}{2}\ln(2\pi\sigma^2) - \frac{(x-\mu)^2}{2\sigma^2} \\ &= \mathcal{L}^{\text{InfoGAN}}\end{aligned} \tag{6}$$

C. Wasserstein GAN

GANs are notoriously difficult to train. From a game-theoretic perspective, the generator and discriminator are trained to find a Nash equilibrium. However, convergence is not guaranteed due to the non-cooperative nature of the minimax game [20], [22]. There is an evidence that distributions p_G and p_{data} are concentrated on a low-dimensional manifold with disjoint supports [23]. Vanishing gradients of the generator is another common problem as the performance of the discriminator saturates [24]. Furthermore, the generator can trick the discriminator by learning only a tiny subset of the real dataset and producing samples with low variety. Although we used some techniques proposed in [20], we were not able to stabilize the training on our wafer dataset with loss function based on Kullback-Leibler divergence as used in the original GAN paper.

To this end, the idea proposed in Wasserstein GAN (WGAN) [25] is based on replacing the loss function used in the original GAN by a distance measure called Earth Mover's (EM) distance or Wasserstein-1 distance. The EM distance between the actual data and generator distributions is defined:

$$W(p_{data}, p_G) = \inf_{\gamma\sim\Pi(p_{data},p_G)} \mathbb{E}_{(x,y)\sim\gamma}[\|x-y\|] \tag{7}$$

where $\gamma(x,y)$ is a transport plan over all possible joint probability distributions $\Pi(p_{data}, p_G)$ between p_{data} and p_G. In other words, it indicates how much "*mass*" must be moved from x to y in order to transform the distribution p_{data} into the distribution p_G.

Calculating all possible distributions is intractable, instead Arjovsky [25] proposed to use a minimax game based on Kantorovich-Rubinstein duality, as follows:

$$\min_G \max_D \; \mathbb{E}_{x\sim p_{data}}[f(D(x))] - \mathbb{E}_{\tilde{x}\sim p_G}[f(D(\tilde{x}))] \tag{8}$$

where the function $f : \mathbb{R} \to \mathbb{R}$ is a 1-Lipschitz continuous function satisfying $|f(x_1) - f(x_2)| \leq \gamma \cdot |x_1 - x_2|$ for every $x_1, x_2 \in \mathbb{R}$ and some real constant $\gamma \geq 0$.

Instead of weight clipping, an improved way of enforcing 1-Lipschitz continuity was described in [26] by adding a gradient penalty regularization term to the original WGAN loss function, as:

$$\begin{aligned}\mathcal{L}_D^{\text{WGAN-GP}} &= \mathbb{E}_{x\sim p_{data}}[D(x)] - \mathbb{E}_{\tilde{x}\sim p_G}[D(\tilde{x})] \\ &\quad + \alpha\mathbb{E}_{\hat{x}\sim p_{\hat{x}}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2]\end{aligned} \tag{9}$$

$$\mathcal{L}_G^{\text{WGAN-GP}} = \mathbb{E}_{\hat{x}\sim p_G}[D(\hat{x})] = \mathbb{E}_{z\sim p_{noise}}[D(G(z))]$$

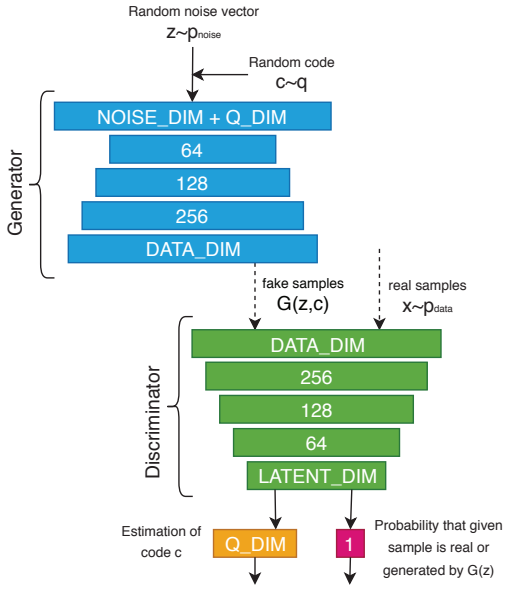


Fig. 3. The proposed generative adversarial neural network architecture for extracting low-dimensional latent representation of cleansed wafer test data.

where α is the regularization coefficient and $p_{\hat{x}}$ is a distribution laying between p_{data} and p_G, *i.e.* it can be sampled easily as $\hat{x} = \epsilon x + (1-\epsilon)\tilde{x}$ for $\epsilon \sim U[0,1]$.

D. Implementation

Combining the WGAN and InfoGAN objectives, we propose the following loss function for the discriminator and the generator to extract a low-dimensional latent representation of cleansed wafer test data as:

$$\begin{aligned}\mathcal{L}_D &= \mathcal{L}_D^{\text{WGAN-GP}} - \lambda\mathcal{L}^{\text{InfoGAN}} \\ &= \mathbb{E}_{x\sim p_{data}}[D(x)] - \mathbb{E}_{\tilde{x}\sim p_G}[D(\tilde{x})] \\ &\quad + \alpha\mathbb{E}_{\hat{x}\sim p_{\hat{x}}}[(\|\nabla_{\hat{x}}D(\hat{x})\|_2 - 1)^2] - \lambda\mathcal{L}^{\text{InfoGAN}} \qquad (10)\end{aligned}$$

$$\mathcal{L}_G = \mathcal{L}_G^{\text{WGAN-GP}}$$

$\mathcal{L}_D$ is a weighted sum multi-objective optimization function. As shown in [27], any convex Pareto optimal front can be obtained when α and λ are strictly positive. For simplicity, we have chosen $\alpha = 1$ and $\lambda = 1$ in our implementation.

The exact neural network architecture used in our implementation is depicted in Figure 3. The generator (depicted in blue) generates a wafer given random vectors z and c. The discriminator (depicted in green and red) then decides whether the generated wafermap is real (given the original wafermap) or fake. Classification network Q (depicted in orange) maximizes the mutual information between the output of the generator and the distribution of c that was provided to the generator. The output of Q is the statistics of distribution used to model the code c, i.e., in the case of normal distribution, its mean and standard deviation. `LeakyReLU` activation function used after each fully connected inner layer. We used hyperbolic tangent (`tanh`) activation function for the generator output (since the wafer data are normalized to interval $[-1, 1]$). The first-order gradient-based optimizer `Adam` is used for the training phase.

VI. Wafermap Clustering

The previous section described how one could assign meaningful low-dimensional codes (latent features) to high-dimensional data samples by maximizing the mutual information between the code distribution and the generator output distribution. We have also shown how to stabilize the learning on the wafer dataset by minimizing the Wasserstein distance between the two mentioned probability distributions. This section describes how to group the latent features into clusters based on a similarity measure. In theory, wafermaps with similar patterns should be contained within the same cluster, while dissimilar wafermaps should be in different groups. There are two major clustering categories – *hierarchical clustering* which groups the set of objects into a hierarchical tree (dendrogram) and *partitioning* which groups the set of objects into disjoint subsets, such that each object is precisely in one subset. In general, any clustering methods can be applied, so we used only one algorithm from each of these categories, namely *k-means* and *Hierarchical agglomerative* [28].

VII. Experimental Evaluation

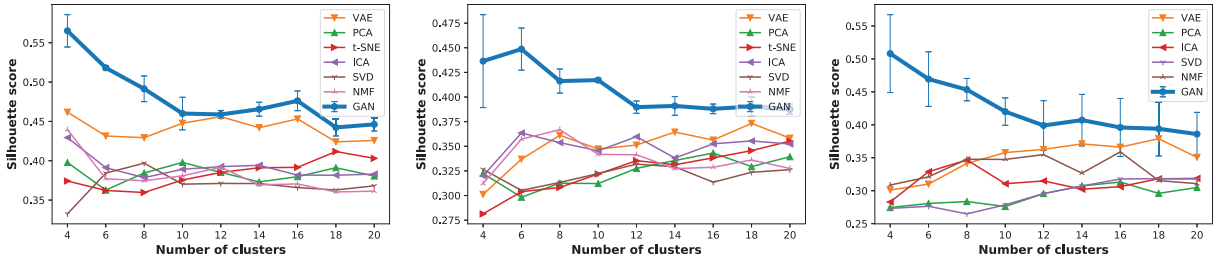
To have a quantitative analysis, we experimentally evaluated the proposed deep adversarial neural network (DeepWafer) with the best existing methods in combination with both clustering algorithms. We described the real dataset used in the experiments in Section III. We developed all codes related to this work in Python v3.5.2 and the deep learning library Keras v2.0.8 with Tensorflow v1.5.0 as backend. For comparison, we chose six well-known unsupervised methods: (1)Variational auto-encoders (VAE) [13], (2) Non-negative matrix factorization (NMF) [29], (3) Singular value decomposition (SVD) [7], (4) Principal component analysis (PCA) [30], (5) Independent component analysis (ICA) [31], (6) t-Distributed stochastic neighbor embedding (t-SNE) [32].

The clustering performance is measured with the Silhouette metric [33], which is a number on an interval $[-1, 1]$. It shows how similar a latent feature is to other features within the same cluster compared to the other clusters. The higher this value is, the better the clustering performance will be. Values around 0 indicate overlapping clusters, and negative values mean that the sample was likely assigned to a wrong cluster. Formally, this measure can be written as follows:

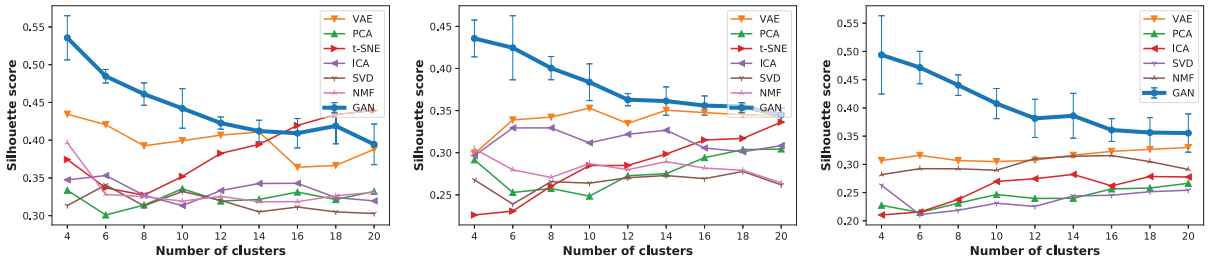
$$\text{sil}(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \qquad (11)$$

where $a(i)$ is the average distance between feature vector x_i to the other vectors in the same clusters and $b(i)$ is the average distance between x_i to the others in the nearest cluster.

The average values of $\text{sil}(i)$ measured overall latent features with different latent dimensions (*i.e.* 2, 3, and 4) for our method and the competing methods are shown in Figure 4.



(a) k-means, LATENT_DIM = 2 (b) k-means, LATENT_DIM = 3 (c) k-means, LATENT_DIM = 4



(d) Agglomerative, LATENT_DIM = 2 (e) Agglomerative, LATENT_DIM = 3 (f) Agglomerative, LATENT_DIM = 4

Fig. 4. Evaluation of different techniques in terms of Silhouette score with two clustering methods k-means and agglomerative clustering in different latent space dimensions. Our GAN-based approach yields better-separated clusters compared to the competing methods in the majority of cases.

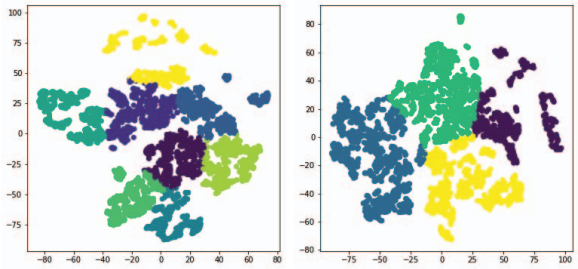


Fig. 5. Visualizations of 2-dimensional t-SNE embedding of high-dimensional latent space. Colors represent different clusters identified by k-means method.

For this experiment, we trained the model on 1000 epochs with batches of size 32. The results show that our approach outperforms the best existing methods for efficiently recognizing and clustering spatial wafermap patterns in terms of the Silhouette score, even in small clusters. One can easily see that our GAN-based method can get a higher score in both partitioning and hierarchical clusterings than the competing methods. Moreover, the results show a decreasing trend on the Silhouette score when the number of clusters increases on any latent space dimension. Our previous work on variational auto-encoders [13] has better performance in most of the cases among the competing methods.

Figure 5 shows t-SNE visualization of the output of the penultimate layer in the discriminator network (see Figure 3), projected into two-dimensional latent representation. It is observed that wafers associated with the same behavior (pattern) form a cluster while wafers belonging to different patterns are separated to some degree.

We trained our GAN-based DeepWafer model (described in Section V) on the cleansed wafermap dataset. The model learns to generate realistic wafermaps with visible patterns successfully, and some randomly selected real and generated wafermaps are shown in Figure 6. We have two significant observations from this figure: (1) the generated wafermaps by DeepWafer are clear and plausible, compared to real wafermaps; (2) we can easily see the patterns in the generated images, which can lead to having a better clustering result.

VIII. Conclusion

Systematic defects in the manufacturing industry are caused by a malfunction in process equipment or human errors. Automated detection of such production issues and automated root cause analysis will improve the efficiency of semiconductor production. Manufacturing defects often exhibit patterns in measured test data. Recognizing these patterns and their categorization are essential tasks in root cause identification of the production issues. In this paper, we proposed a deep generative adversarial network methodology to recognize the spatial wafermap patterns and cluster them. We experimentally evaluated the performance of the proposed approach over a real dataset compared to well-known unsupervised methods.

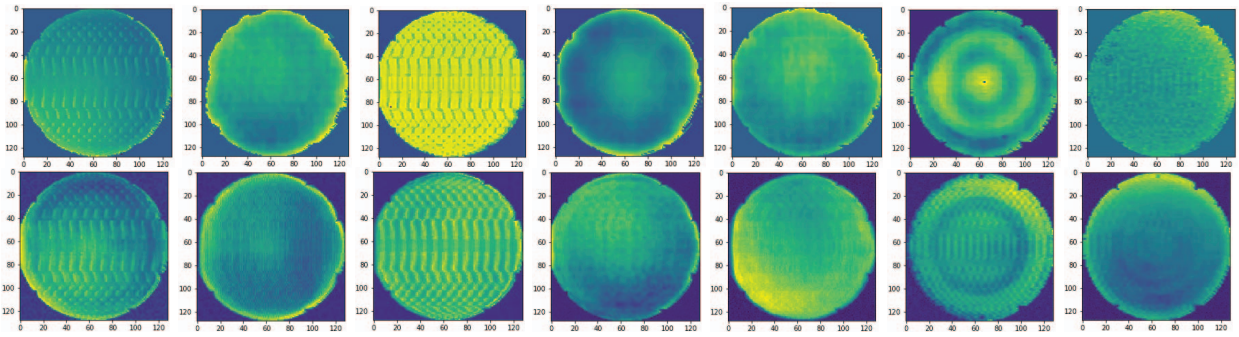


Fig. 6. Visualization of samples from the real wafermaps dataset (top row) and generated wafermap samples after our GAN-based DeepWafer neural network model (bottom row). We can observe that the quality of generated wafermaps by our model is plausible in comparison with real wafermaps.

REFERENCES

[1] C.-Y. Hsu, "Clustering ensemble for identifying defective wafer bin map in semiconductor manufacturing," *Mathematical Problems in Engineering*, 2015.

[2] C.-H. Wang, "Recognition of semiconductor defect patterns using spectral clustering," in *Industrial Engineering and Engineering Management, IEEE International Conference on*, 2007, pp. 587–591.

[3] F. Duvivier, "Automatic detection of spatial signature on wafermaps in a high volume production," in *Defect and Fault Tolerance in VLSI Systems, 1999. DFT'99. International Symposium on*. IEEE, 1999, pp. 61–66.

[4] Mickelson and A.R, "Defect recognition and image processing in semiconductors 1995," *Crystal Research & Technology*, vol. 33, no. 1, pp. 58–58, 2015.

[5] M. Saqlain, B. Jargalsaikhan, and J. Y. Lee, "A voting ensemble classifier for wafer map defect patterns identification in semiconductor manufacturing," *IEEE Transactions on Semiconductor Manufacturing*, pp. 1–1, 2019.

[6] D. Jiang, W. Lin, and N. Raghavan, "A novel framework for semiconductor manufacturing final test yield classification using machine learning techniques," *IEEE Access*, vol. 8, pp. 197 885–197 895, 2020.

[7] 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, 2018.

[8] L.-C. Chao and L.-I. Tong, "Wafer defect pattern recognition by multiclass support vector machines by using a novel defect cluster index," *Expert Systems with Applications*, vol. 36, no. 6, 2009.

[9] N. Yu, Q. Xu, and H. Wang, "Wafer defect pattern recognition and analysis based on convolutional neural network," *IEEE Transactions on Semiconductor Manufacturing*, vol. 32, no. 4, pp. 566–573, 2019.

[10] M. Bellini, G. Pantalos, P. Kaspar, L. Knoll, and L. De-Michielis, "An active deep learning method for the detection of defects in power semiconductors," in *2021 32nd Annual SEMI Advanced Semiconductor Manufacturing Conference (ASMC)*, 2021, pp. 1–5.

[11] J. Kim, Y. Nam, M. Kang, K. Kim, J. Hong, S. Lee, and D.-N. Kim, "Adversarial defect detection in semiconductor manufacturing process," *IEEE Transactions on Semiconductor Manufacturing*, pp. 1–1, 2021.

[12] Y. Kong and D. Ni, "A semi-supervised and incremental modeling framework for wafer map classification," *IEEE Transactions on Semiconductor Manufacturing*, vol. 33, no. 1, pp. 62–71, 2020.

[13] P. Tulala, H. Mahyar, E. Ghalebi, and R. Grosu, "Unsupervised wafermap patterns clustering via variational autoencoders," in *International Joint Conference on Neural Networks (IJCNN)*, 2018.

[14] S. Wang, S. Yan, Q. Shen, C. Luo, J. Ai, L. Li, D. Wang, S. Ding, and Q. Xia, "Wafer defect map similarity search using deep learning in semiconductor manufacturing," in *2021 China Semiconductor Technology International Conference (CSTIC)*, 2021, pp. 1–4.

[15] C.-Y. Chang, C. Li, J.-W. Chang, and M. Jeng, "An unsupervised neural network approach for automatic semiconductor wafer defect inspection," *Expert Systems with Applications*, vol. 36, no. 1, pp. 950–958, 2009.

[16] F. Di Palma, G. De Nicolao, G. Miraglia, E. Pasquinetti, and F. Piccinini, "Unsupervised spatial pattern classification of electrical-wafer-sorting maps in semiconductor manufacturing," *Pattern recognition letters*, vol. 26, no. 12, pp. 1857–1865, 2005.

[17] C. Leys, C. Ley, O. Klein, P. Bernard, and L. Licata, "Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median," *Journal of Experimental Social Psychology*, vol. 49, no. 4, pp. 764–766, 2013.

[18] M. Rezanejad, M. Khodadad, H. Mahyar, H. Lombaert, M. Gruninger, D. Walther, and K. Siddiqi, "Medial spectral coordinates for 3d shape analysis," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 2686–2696.

[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[20] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems*, 2016, pp. 2234–2242.

[21] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Advances in neural information processing systems*, 2016, pp. 2172–2180.

[22] L. M. Mescheder, "On the convergence properties of gan training," *ArXiv*, vol. abs/1801.04406, 2018.

[23] H. Narayanan and S. Mitter, "Sample complexity of testing the manifold hypothesis," in *Advances in Neural Information Processing Systems*, 2010, pp. 1786–1794.

[24] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," *arXiv:1701.04862*, 2017.

[25] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *stat*, vol. 1050, p. 26, 2017.

[26] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems*, 2017, pp. 5767–5777.

[27] R. T. Marler and J. S. Arora, "The weighted sum method for multi-objective optimization: new insights," *Structural and Multidisciplinary Optimization*, vol. 41, no. 6, pp. 853–862, Jun 2010. [Online]. Available: https://doi.org/10.1007/s00158-009-0460-7

[28] G. J. Szekely and M. L. Rizzo, "Hierarchical clustering via joint between-within distances: Extending ward's minimum variance method," *Journal of classification*, vol. 22, no. 2, pp. 151–183, 2005.

[29] R. Schachtner, "Extensions of non-negative matrix factorization and their application to the analysis of wafer test data," Ph.D. dissertation, 2010.

[30] T. J. Rato, J. Blue, J. Pinaton, and M. S. Reis, "Translation-invariant multiscale energy-based pca for monitoring batch processes in semiconductor manufacturing," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 2, pp. 894–904, 2017.

[31] J.-M. Lee, S. J. Qin, and I.-B. Lee, "Fault detection and diagnosis based on modified independent component analysis," *AIChE journal*, vol. 52, no. 10, pp. 3501–3514, 2006.

[32] G. C. Linderman, M. Rachh, J. G. Hoskins, S. Steinerberger, and Y. Kluger, "Efficient algorithms for t-distributed stochastic neighborhood embedding," *arXiv preprint arXiv:1712.09005*, 2017.

[33] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.