DMB

DATABASE MANAGEMENT
AND
BIOMETRICS

.3485

RECONSTRUCTION-BASED ANOMALY DETECTION WITH MACHINE LEARNING FOR HIGH THROUGHPUT SCANNING ELECTRON MICROSCOPE DEFECT INSPECTION

Xinyu Tian

MASTER'S THESIS ASSIGNMENT

Committee:
Dr.ir. L.J. Spreeuwers (UT)
T. Arican, MSc (UT)
Dr.ing. E.A.M. Klumperink (UT)
V. Kuiper (ASML Delft)

July, 2022

2022DMB0005
Data Management and Biometrics
EEMathCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

UNIVERSITY OF TWENTE. | DIGITAL SOCIETY INSTITUTE

List of Acronyms

SEM Scanning Electron Microscope

FOV Field of View

D2D Die-to-Die

D2DB Die-to-Database

ML Machine Learning

DL Deep Learning

SNR Signal-to-Noise Ratio

TEM Transmission Electron Microscopy

PCA Principal Component Analysis

CNN Convolutional Neural Network

AE Autoencoder

CAE Convolutional Autoencoder

VAE Variational Autoencoder

GAN Generative Adversarial Network

MSE Mean Square Error

SSIM Structural Similarity Index Measure

MMSE Modified Mean Square Error

CR Capture Rate

NR Nuisance Rate

ROC Receiver Operating Characteristic

CC Capture Count

NC Nuisance Count

PCs Principal Components

GF Gaussian Filter

Reconstruction-based Anomaly Detection with Machine Learning for High Throughput Scanning Electron Microscope Defect Inspection

Xinyu Tian

Abstract

With rapid advancement of high throughput wafer inspection systems, there are growing demands on more efficient and accurate defect inspection algorithms. This work focuses on investigating the feasibility of applying learning based algorithms to the image processing pipeline of the next generation wafer inspection tools. An end-to-end reconstruction-based anomaly detection methodology is proposed with two types of reconstruction models based on Principal Component Analysis and Convolutional Autoencoder. Both methods have demonstrated competitive performance compared to the current defect detection algorithm on multiple datasets with different patterns and various defect types.

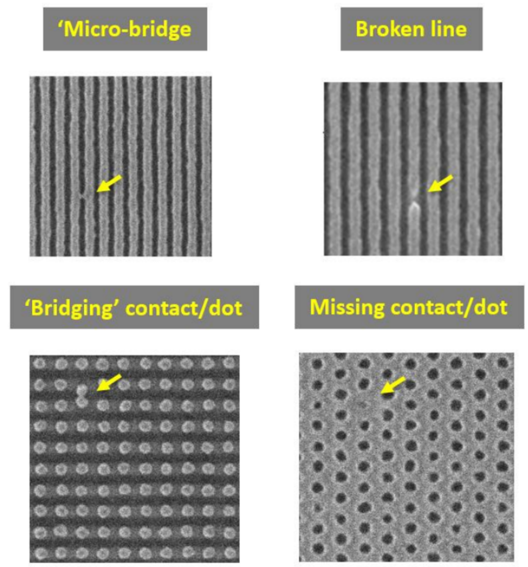


Figure 1. Examples of defects that could be found in semiconductor manufacturing [1].

1. Introduction

Chips are the core components of any electronic products that are essential in our daily life. In chip manufacturing process, defect inspection is a crucial step to guarantee a certain yield on qualified chips. With the continuous pursuit of the reduction in the size of transistors on chips, there is a growing demand on developing efficient high-resolution high-throughput defect inspection systems for the wafer fabrication process.

Wafer defect inspection systems aim to detect stochastic defects on the wafers, which could be caused by unexpected particles on the wafer, faulty exposure procedures and flaws on the mask. Some common defect examples are shown in Figure 1 [1]. The Scanning Electron Microscope (SEM) is one of the most powerful wafer inspection tools used in today's semiconductor manufacturing [2]. A SEM scans the wafer by irradiating electron beams onto the surface of the wafer and forms an image by detecting the emitted secondary and backscattered electrons. One of the major advantages of the SEM inspection tools, compared to the optical microscopes, is that SEM is capable of imaging with one nanometer resolution and finding nanoscale defects on the wafer. However, the Field of View (FOV) of one single electron beam is small, thus resulting in slow measurement and low throughput for the defect inspection. To address the issue, ASML has been developing new generation of the wafer inspection tools with multiple beams to increase the throughput.

Multi-beam wafer inspection systems gain throughput by increasing the number of electron beams and summing up the FOV of all the beams. However, as the number of electron-beams increases, the computational cost of the current algorithms for processing the produced SEM images increases correspondingly. Without adapted data processing solutions, the computation ability of the SEM image processing pipeline (referred as "datapath") will determine the throughput of the tool instead of the area of scanning in a unit time. Besides that, the computers for data analysis, that are embedded in the tool, are one of the most expensive parts of the whole system. Therefore, there is need to improve current employed image processing

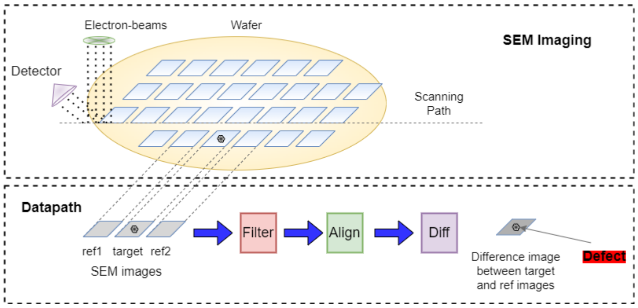


Figure 2. The principle of D2D wafer defect inspection. D2D inspection compares the SEM images of three adjacent dies and determine if there is a defect, based on the signal strength in the difference images.

pipeline to achieve reliable defect detection at a high speed with affordable computing power units.

The current detection method deployed in the tools is based on the principle of Die-to-Die (D2D) defect inspection and several denoising techniques, as demonstrated in Figure 2. We refer this method as the "baseline" method. As Machine Learning (ML) and Deep Learning (DL) bring great achievements in the field of image analysis and computer vision [3, 4], it is appealing to look into learning-based alternative algorithms for the datapath of wafer defect inspection systems. The novel methods have potentials to achieve automation, better accuracy and higher throughput.

Defect detection can be related to "anomaly detection" in ML studies, which aims for the discovery of the divergence between normal and anomalous data. Anomaly detection algorithms have been widely studied in many fields, such as cybersecurity [5], fraud detection [6], industrial fault detection [7] and medical diagnosis [8] etc. In terms of applications in semiconductor industry, deep learning methods have been investigated extensively on metrology, defect detection and SEM image analysis related topics, including defect classification [9], SEM image quality enhancement and denoising [2, 10, 11], and SEM image simulation [12]. However, few work focuses on ameliorating anomaly detection algorithms for D2D inspection with ML methods.

One SEM image from a single beam usually consists of more than a million of pixels and the defects are represented by only a few pixels. The major difficulties to apply learning-based methods for anomaly detection tasks on the SEM images are the high similarity and extreme unbalanced ratio between the normal and abnormal data. It is hypothesized that reconstruction based methods are promising to be applied in the wafer inspection applications. Therefore, the main goal of this work is to investigate the feasibility of using end-to-end reconstruction-based methods to scale the datapath of the SEM applications for the next generation high-throughput wafer inspection tools. This goal is then divided into several research questions:

1. Which reconstruction-based ML methods exist for anomaly detection for image analysis?
2. Which methods in the state of the art are best suited candidates for anomaly detection tasks on high-throughput SEM images?
3. How can reconstruction-based methods be applied to wafer defect inspection?
4. How does the choice of loss function influence the performance of the methods?
5. How well do the algorithms work on the real SEM image datasets with different pixel sizes, defect types and patterns based on current evaluation methodology?

The rest of the work is structured as follows. Section 2 analyses the related work and answers research question 1 and 2. Section 3 describes the proposed reconstruction-based methodology and answers research question 3. Section 4 outlines the available datasets and details out the experimental setup. Section 5 presents the results of the experiments and answers research question 4 and 5. Section 6 analyses the experimental results and proposes some recommendations for the further work, and Section 7 finishes with the conclusions.

2. Related work

2.1. Reconstruction based anomaly detection

Reconstruction-based methods have shown remarkable potentials on the image anomaly detection tasks [7, 13–16].

Cho et al. [14] propose a Principal Component Analysis (PCA) and Convolutional Neural Network (CNN) combined method to analyze Transmission Electron Microscopy (TEM) images of crystalline materials and predict the location of a point defect. They train a PCA reconstruction with only defect-free images to generate a residual image, and then use the same training set to create a labelled dataset with artificial circular defects on those images to train a self-supervised CNN classifier to detect the defect in the residual image. Their model is advantageous than others in the application of noisy images. The performance of a simple PCA model degrades when noise level of an image increases. Hence the CNN classifier is added to help improve the overall accuracy. However, their experiments are only proven effective on point defects and based on simulated TEM images data with regular patterns, programmed defects and noise. It can be problematic to apply the model to detect various types of defects or analyse the images with random patterns.

For high-dimensional complex data, encoder-decoder based neural networks are widely explored for reconstruction based anomaly detection. Nakazawa and Kulkarni [17] introduce a convolutional encoder-decoder architecture for anomaly detection and segmentation on binary wafer map images. The model is trained with 9 types of synthetic wafer map defects and able to successfully detect unseen defective clusters from the real wafer maps. Their experiments suggest that high frequency components and noise are less likely reconstructed by the autoencoder architecture. Liu et al. [7] and Tsai and Jen [16] present different ways to apply modified Autoencoder (AE) in practical for industrial surface inspection through improved loss functions and adding extra encoder structure. Through these modifications, the variance of the latent feature spaces is minimized so that the distance between defective and defect-free samples becomes more obvious. Then the anomaly can be found based on the distance between defective and normal samples in latent space or image space.

Generative models, such as Variational Autoencoder (VAE) [18] and Generative Adversarial Network (GAN) [15, 19] can be also considered as reconstruction based methods. Generative models is trained to learn the probability distribution of the normal samples in low-dimension latent space, and then reconstruct/generate input-alike data from the sampling of the distribution. Although GANs are capable of reconstructing high resolution realistic images, they are notorious for training instability [20]. As we are at the beginning of the research project, it is more logic to start with simpler networks.

2.2. Learning-based methods for SEM defect inspection

Recently, many works have been conducted to investigate the potential of applying ML and DL models in the field of semiconductor industry to improve lithography, especially in the Extreme Ultraviolet Lithography era [21, 22].

Evanschitzky et al. [23] demonstrate an approach for photomask defect detection, based on a hybrid deep learning network. Simply put, the photomasks are the template of the integrated circuit design layout, used for guiding the patterns that need to be printed on the wafer. The defect inspection of photomasks happens prior to wafer inspection, but similar inspection tools are utilized and the method is of important values to be looked into. The authors design an model that combines CNN, a modified U-Net network and Fourier energy filters, for the tasks of pattern classification, defect detection and defect shape determination respectively. In their experiments, artificial training images with simplified characteristics are used to train the model. The model is subsequently tested with 67 real SEM images of line space pattern and 64 of contact hole patterns with respectively 453 and 179 different types of programmed defects. Their result shows their methods can reach 100% true defect detection rate with 0.44% and 4.28% false defect detection rate for the repetitive line space and contact hole patterns with a ratio of approximately 1 to 6,000 and 1 to 15,000 between defect and non-defect regions. This ratio is much larger than our case. Besides, the pixel size of the SEM images in the paper is 1.5nm and the scale of the defect size is roughly between 20×20 and 150×150 pixels. Compared to our dataset (e.g. 10nm MH dataset, see Section 4.1) that has pixel size of 10nm and defect size of 2×2 pixels, their defective targets are much larger, thereby easier to be detected.

Another research in applying deep learning model for defect detection on low Signal-to-Noise Ratio (SNR) SEM images is from Ouchi et al. [24]. They develop a Die-to-Database (D2DB) defect detection method with an encoder-decoder architecture that learns the relation between SEM images and design layouts on pixel level and predicts the luminosity distributions of an SEM image from a design layout. The defect is being investigated through the direct pixel-wise comparison between the learned distribution and target images. The model is trained on the defect free images from a small dataset that consists of 13 images (10 are defect-free) per pixel size of 1-4nm/pixel. The model only learns acceptable deformation on non-defect images, thus, prior information about defect types is not essential to have. This is an attractive attribute, as in reality, prior knowledge about all the potential defects is hard to acquire. However, one experiment with satisfying results on a really small set of data with repetitive line and space patterns can hardly be generalized to a solid conclusion that the method will work on other use cases. Follow up research are needed to prove the validity of their methods.

A study has been carried out previously under the same project, which uses a CNN-based model to classify the defect and non-defect SEM images. It is an end-to-end classification based method, and has achieved promising results of over 90% true defect and lower than 5% false defects captured on real SEM images with pixel size of 8-12nm/pixel. However, the training and evaluation datasets for the method are artificially adjusted to a ratio of 50% and 50% between defect and non-defect images. The results completely degrade if the model is applied to the representative data with a ratio of 1 to more than 10^6 between defects and defect-free samples. In the real wafer inspection process, defects are really rare comparing to defect-free pixels, leading to an extreme class imbalance problem for the binary classification models. Class imbalance means that the number of a class of positive samples is far less than the number of negative data. It is a problem because in this case learning algorithm based on the general evaluation metric fails to learn the few instances of the positive class, which is usually the class of our interest. Several works try to address class imbalance problem [25, 26]. Unsupervised learning methods for anomaly detection are suggested to solve the issue, where the objective of learning is to well model positive instances instead of discriminating between classes. Therefore, reconstruction based anomaly detection methods trained in an unsupervised manner are logical choices to be investigated next.

3. Methodology

In this section, we will introduce the reconstruction based anomaly detection methods that have been investigated in this work. The motivation behind the methodology is that we assume if we train the reconstruction models only with defect-free samples, they will not be able to reconstruct defective images correctly, then ideally we can distinguish the anomaly by outstanding reconstruction errors. Mean Square Error (MSE) and Structural Similarity Index Measure (SSIM) are commonly used to evaluate the reconstruction error between the generated image and the original image. However, the reconstruction error cannot effectively separate defective and non-defective images in our application due to the high similarity between normal and anomalous instances. Hence, we determine and localize the defect based on the prominent pixel values in the residual images between the input and reconstructed images. An overview of the proposed reconstruction based methodology for defect inspection is presented in Figure 3.

Two standard reconstruction based models, PCA and AE, are implemented on real SEM image data in this work to investigate their potentials to improve the throughput for wafer defect inspection. Both models are widely investigated and have demonstrated successful results on anomaly detection tasks [16, 17, 27, 28].

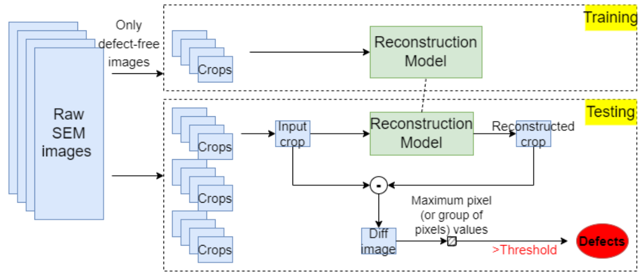


Figure 3. Overview of the reconstruction based methodology for defect detection.

3.1. PCA based reconstruction

PCA is an unsupervised dimensionality reduction technique. It is used to find an uncorrelated low-dimensional space that carries the most variance of the original high-dimensional data. Specifically, PCA is a linear transformation to transfer a data matrix X with dimension of $m \times n$ to a lower dimension representation X_k with dimension of $m \times k$, where $k << n$. The data is projected to this new feature space and the transformed features, namely Principal Components (PCs), are supposed to contain enough information to describe the original data. If we define the transformation matrix as W_k, PCA can be describe as

$$X_k = XW_k \tag{1}$$

where

$$W_k = argmax_{W_k} \|X_k\|^2 \; s.t. \, W_k^T W_k = I \tag{2}$$

To adapt PCA for unsupervised anomaly detection, PCA solution then can be rewritten as

$$W_k = argmin_{W_k} \|X - X_k W_k^T\|^2 \; s.t. \, W_k^T W_k = I \tag{3}$$

Note that $X_k W_k^T$ can be interpreted as the reconstruction of X from the low-dimentional representation X_k. Thus from the reconstruction perspective, PCA is seeking an orthogonal vector space to transform original n-dimensional data to k-dimensional data such that the reconstruction mean square error is minimized.

As reconstructed images from inverse PCA transformation usually contain much less noise than the input images, the difference images reserve the noise from the input images. If we only take the maximum pixel value as the defect candidate from one image crop, strong noise signal can be recognized as the defective signal and thus introduce nuisances. To address this problem, two modifications are made to the pipeline of using PCA as a reconstruction model. One method is to add the filtering step to enforce the strength of useful signals and reduce noise before the image reconstruction. Another idea is based on the observation that high pixel value due to noise is usually only a single pixel while the defects cause multiple high pixel values next to each other. Therefore, we can look for the maximum sum value of the defect size of $d \times d$ pixels instead of a single pixel.

3.2. The CAE model

The AE model consists of two components, the encoder and decoder, where the encoder learns to effectively encode the data to lower dimensional representation (often referred as "bottleneck"), while the decoder reconstructs the data from the compressed representation back to the original space. The output of the encoder is expected to be able to well describe the raw data so that the reconstructed data can be as close to the original data as possible. Therefore, AEs are also widely used as feature extractor for various applications [29]. As depicted in Figure 4, the architecture used in this work is based on a Convolutional Autoencoder (CAE) structure.

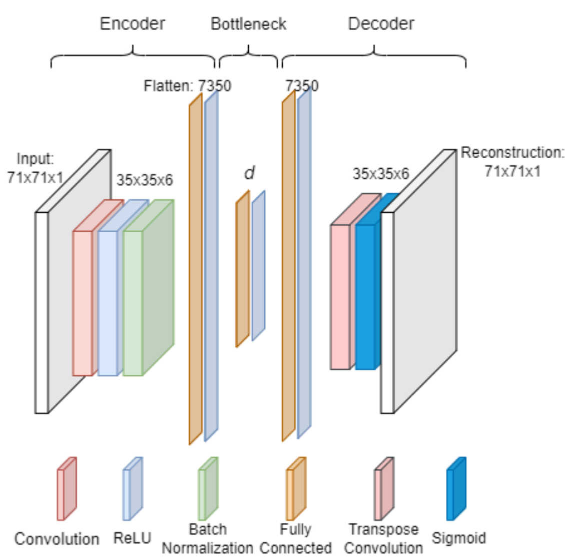


Figure 4. The architecture of the CAE network.

If we denote the input image data with dimension of $n \times n$ as $\mathbf{x}$ and the latent space representation with dimension of $k \times k$ as $\mathbf{z}$, where $k < n$, the encoder can be described mathematically as

$$\mathbf{z} = f_e(W_e\mathbf{x} + b_e) \tag{4}$$

Reconstructed image data will have the same dimensionality as the input data, denoted as $\hat{\mathbf{x}}$. The decoder can be described as

$$\hat{\mathbf{x}} = f_d(W_d\mathbf{z} + b_d) \tag{5}$$

where W_e, W_d and b_e, b_d are the weights and biases of the encoder and decoder models.

Conventionally, CAEs are trained by minimizing reconstruction error through back-propagation algorithm. The most commonly used loss function for training a reconstruction model is the mean squared per-pixel ℓ_2-distance, also known as MSE loss, i.e.

$$\mathcal{L}_{mse} = \frac{1}{N}\sum_{i=0}^{N}\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 \tag{6}$$

The choice of loss function for training a CAE model has significant influence on the performance of the reconstruction. MSE loss has weaknesses such as its sensitivity to the localization inaccuracies of edges and ignorance on meaningful features reconstruction [30]. To alleviate these problems, we experiment with two other loss functions.

One modified loss function is constructed based on MSE loss, denoted as "Modified Mean Square Error (MMSE)". The idea is to add prior information from the input images to reinforce the reconstruction of the important features. For instance, the 10nm MH dataset (see Section 4.1) has distinctive characteristics, where roughly 95% of the area of the images is background and 5% is the contact holes. With conventional reconstruction error MSE for training, the model may overlook the contact holes and focus on reconstructing background. Therefore, We introduce a continuous step function implemented by a customized inverse sigmoid function $f(x)$, which can be written as

$$f(x) = \frac{1}{1 + e^{100(x - p_{thres})}} \tag{7}$$

By applying the step function to the difference image, we put more weights (w) on reconstruction of the lower pixel values and the threshold for low pixel value (p_{thres}) is decided by 5% lowest pixels from the histogram of the input image. The modified loss function is thus

$$\mathcal{L}_{mmse} = (1 + wf(\mathbf{x}_i))\frac{1}{N}\sum_{i=0}^{N}\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 \tag{8}$$

Another experiment is to adopt SSIM as the loss function to train the CAE model as SSIM is less sensitive to edge alignment [30]. SSIM is a image quality assessment metric proposed by Wang et al. [31] to measure the difference between two images by integrating luminance, constrast and structural information instead of simple pixel values. The calculation of SSIM is defined as

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (9)$$

where μ_x, μ_y, σ_x^2, σ_y^2 are the mean and variance of x, y respectively, σ_{xy} is the covariance of x and y. c_1 and c_2 are two constants to stabilize the division. SSIM produces the similarity score between two images in a scale of -1 to 1, where 1 means they are identical and -1 means they are very different.

To apply SSIM as a loss function, we define a window with size of K × K sliding across the image $\mathbf{x}$ and reconstructed image to calculate the SSIM of each pixel. Then we define the loss as 1-SSIM so that we can train the network by minimizing this loss function.

$$\mathcal{L}_{ssim} = 1 - \frac{1}{N}\sum_{i=0}^{N} SSIM(\mathbf{x}_i, \hat{\mathbf{x}}_i) \quad (10)$$

3.3. Evaluation for comparing algorithms

Under the project, there is an existing evaluation method to assess the performance and the cost of the inspection algorithms. It consists of two major key performance indicators: capture and nuisance rate, theoretical numerical complexity and average running time.

A. Capture and nuisance rate

Capture rate and nuisance rate indicate how well an algorithm can capture defects, which is the primary goal of a defect detection method. Among all of the "defects" that one detection algorithm finds, the correctly identified defects are called "capture", while the incorrect defects are called "nuisance". Capture Rate (CR) is defined as the number of capture divided by the total number of real defects, while Nuisance Rate (NR) is defined as the number of nuisance divided by the total number of found defects, including both capture and nuisance. The anomaly detection is essentially a binary classification problem, so that we can relate "CR" and "NR" to "recall" and "1-precision" in the scope of ML evaluation metrics. To write it out:

$$CR : Recall = \frac{tp}{tp + fn} \quad (11)$$

$$NR : 1 - precision = \frac{fp}{fp + tp} \quad (12)$$

We can achieve a certain capture rate at a certain nuisance rate by tuning the threshold of the determinant function. A satisfactory industry standard is a fine threshold that can achieve above 90% CR at 10% NR.

However, one pair of CR and NR based on one decision boundary is not always suffice to compare different algorithms. Thus we also plot a Receiver Operating Characteristic (ROC)-alike curve as a holistic evaluation. Saito and Rehmsmeier [32] have pointed out that using ROC to evaluate the performance of a classifier on an imbalanced dataset can be really unreliable due to the misinterpretation of specificity. Therefore, instead of ROC, we plot pairs of the absolute number of Capture Count (CC) vs Nuisance Count (NC) by moving the decision threshold. Similar to ROC curve, if the CCNC curve locates more to the top-left side, the model has achieved better performance.

B. Computational cost

Numerical complexity and average runtime are the indicators of the computational cost of a method. Numerical complexity is calculated as the number of operations needed for implementing an algorithm based on the pseudo code. For a learning-based method, we only consider the complexity of the testing process providing that training the model can be a separate procedure conducted outside of the inspection system. On the other hand, the average runtime of an algorithm can be recorded to estimate the throughput of the datapath. However, as it is still in the early stage of the research and the runtime is influenced by the detailed implementation and hardware, we will only analyse the numerical complexity of the proposed methods in this study.

4. Experimental Setup

4.1. Datasets

To evaluate the performance and generalization of the proposed reconstruction based defect detection methods, 4 real SEM image datasets that contain different types of repetitive patterns and various defects are experimented in this work.

Missing Contact Hole Datasets (MH) The MH datasets consist of 4 sets of images of repetitive contact hole pattern with the size of 4096 × 7754 pixels, each encoded with 8 bits, at 8, 10, 12 and 14-nm pixel size. They are obtained from scanning the same area on the same wafers. For larger pixel size, one pixel can cover larger area, thus the set contains less images. They contain 1044, 864, 720, 612 SEM images respectively with 252 programmed defects, where the middle contact hole out of three is missing. The size of the defect is 20nm × 20nm in this dataset. For different pixel sizes, the defect size varies in terms of the number of pixels that contain a defect. For example, for 10nm pixel size, the defect size is 2 × 2 pixels, while for 14nm

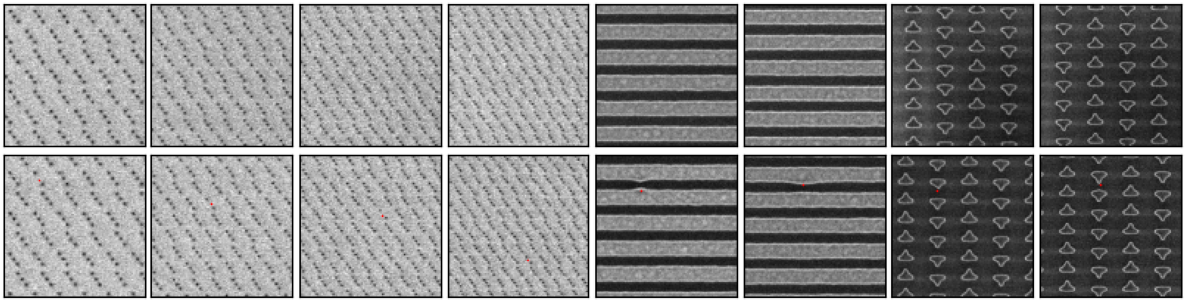


Figure 5. Examples of image crops generated from the real SEM images from the datasets. Top rows: defect-free images. Bottom rows: defective images, and the defect is marked with a red dot. The images are from MH 8nm, 10nm, 12nm, 14nm, LS, and SRAM. The types of the defects are missing, missing, missing, missing, line bridge, line wide, SRAM void and SRAM bridge.

pixel size, the defect size is 1.4×1.4 pixels. In terms of images with 10nm pixel size, there are 1008 defect pixels out of 27,440,971,776 total number of pixels, which means the defects are extremely rare.

Contact Holes with Multi-type Defects Dataset (CH-MTD) The CH-MTD dataset contains 792 real SEM images of contact hole patterns with size of 1024 x 30773 pixels at 4nm pixel size with 4 types of defects, i.e. bridge, missing, enlarge and shrinkage contact holes. The defect size varies for different defect types, where a shrinkage defect has a defect size of 3×3 pixels while a bridge defect can have a defect size up to 30×30 pixels. The dataset contains a total number of 179 defects, including 68 bridge, 80 missing, 22 enlarge and 9 shrinkage defects. Similar to the MH datasets, the number of the defective pixels with respect to normal ones is extremely low.

Line Space Dataset (LS) The LS dataset is comprised of 824 real SEM images of regular line space patterns with size of 1024×1024 pixels at 5nm pixel size, containing 825 programmed defects at the known position on the dies. In general, almost each image contains one defect and this defect is roughly in the center of the image. The dataset contains different types of defects and each type has a right and left variant. The defect types include line bridge and line wide, with defect size of 15nm and 25nm respectively.

SRAM Dataset (SRAM) The SRAM dataset consists of 1449 real SEM images of SRAM patterns with size of 512×512 pixels at 10nm pixel size with 1447 programmed defects. Similar to LS dataset, each image contains one or at most two defects and the defect is roughly in the center of the image. The dataset also contains different types of defects, including void, bridge and critical distance error, and each type has right and left variant. The defect size varies from 15nm to 30nm, based on the type of the defect.

4.2. Data preparation

For data preparation, we firstly split the raw SEM images into training set, validation set and test set. We keep 80% of images in the test set, while the rest is used for training and validation. Secondly, to avoid building a model with too many parameters, we generate image crops with a reasonable size to build the model instead of the whole image. Two cropping mechanisms are implemented for the purpose of training and testing. For training, we randomly cut a number of crops from one raw image at the locations where they do not contain defects. For testing, the entire area of the raw images in the test set should be evaluated. Therefore, we crop the image by striding a window with the same size as the training crop across the whole raw image. Cropping size and the number of crops for training are important parameters that influence the performance of the models. The initial choices for these parameters are empirical estimations, but later they are tuned based on the performance of the model.

4.3. Implementation details

To apply the PCA based reconstruction model to our pipeline of defect detection, we create a data matrix from the flattened non-defective image crops to fit a PCA model, which is implemented with scikit-learn package. The image crops for fitting a PCA model should contain all possible variants of the normal samples that may appear in the test set. Note that PCA is sensitive to translations of the image patterns, the same pattern with different shifts should be considered as different variants. Besides, image crop size, the number of PCs for reconstruction, sigma of the Gaussian Filter (GF) and defect size are tuned based on each dataset.

The CAE model is implemented with PyTorch by optimizing the networks using the Adam optimizer with a learning rate 10^{-4}, batch size 64, and weight decay 10^{-5}. We train and verify the network with two sets of 64,000 image crops randomly cropped from 40 raw images. The network is trained respectively by minimizing MSE loss, Modified MSE loss and SSIM loss functions. Meanwhile, image crop size, latent space dimension, defect size and the parameters for loss functions are tuned based on each dataset.

4.4. Post processing on edge nuisances

It is observed from the experimental results on all the datasets that many nuisances found by our methodologies are located at the edge of the SEM images. (See Appendix A for the detailed explanation and results on the edge nuisances.) These edge nuisances are not stochastic nuisances but induced by the imaging artifacts from the beam movement between lines during the scanning process. On the other hand, the convolutional operations from the algorithms may also be responsible for the edge nuisances as well. There are possible solutions to avoid these artificial edge nuisances. Firstly, we can augment more edge data for training to make the model more robust to the edge variations. Additionally, the data preparation step can be modified by removing the edge of raw images beforehand to avoid the influence of the edge artifacts. As for addressing the edge effects caused by the algorithm, we can crop the image with overlap and remove edges of the crops to avoid the nuisance introduced by the convolution operations. With time limitation, we simply remove the edge nuisances from our results in the post processing to estimate the best achievable performance.

5. Experimental Results

In this section, the performances of the PCA and CAE reconstruction based anomaly detection methods are evaluated on the datasets mentioned in Section 4.1 and assessed based on the evaluation criteria mentioned in Section 3.3.

5.1. PCA results on MH dataset

For MH datasets, the 10nm dataset is used to conduct the hyperparameter search considering 10nm pixel size is most likely to be used in practice. Based on a manual hyperparameter search, the best PCA based anomaly detection model is trained by 10,000 defect-free samples randomly cropped from 50 filtered raw SEM images, where the filter is a GF with the sigma of 0.7. The size of the crops is 256 × 256 and the number of PCs used for reconstruction is 200. Meanwhile, we identify one candidate per crop, which has the maximum value of the summed pixels for a square of the defect size 3. We apply the same model to other subsets from the MH datasets.

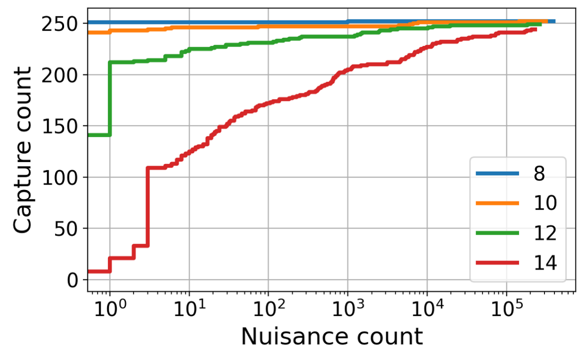


Figure 6. Performance of PCA based defect inspection method on MH datasets with 8nm, 10nm, 12nm, 14nm pixel sizes. The total number of detects is 252.

Figure 6 shows the performance of the PCA based approach on the MH datasets with different pixel sizes. The proposed PCA method can almost capture all of the missing contact hole defects without any nuisance for 8nm pixel size, whereas the performance decreases with the increasing pixel size. For 14nm pixel size dataset, the method cannot discover all of the defects anymore. This result is to be expected, as the SEM images with larger pixel size have less SNR and smaller defect size in terms of the number of pixels per defect. However, as the model was tuned based on 10nm dataset, it is possible to improve the performance for other subsets by tuning parameters based on the specific dataset.

Generally speaking, increasing the number of PCs for reconstruction can add more details at the edge of the contact holes. However, there is a trade-off between adding more components to better describe the data and retaining too many components that leads to a risk in overfitting. For instance, with image crop size of 128 × 128, the performance improves when increase the number of PCs from 50 to 200, but decreases when we further increase the number to 800. Figure 7 shows a visual comparison between the reconstruction quality of PCA model with different number of PCs. It exemplifies that with 50 PCs, the model struggles to reconstruct the edge of the contact holes, while with 800 PCs, it starts to fit the noise.

In addition, we present the trend of the performance w.r.t the numbers of PCs retained for different image crop sizes, as shown in Figure 8. The result shows that the smaller image crop size is, the less PCs needed for reconstruction to achieve similar performance in terms

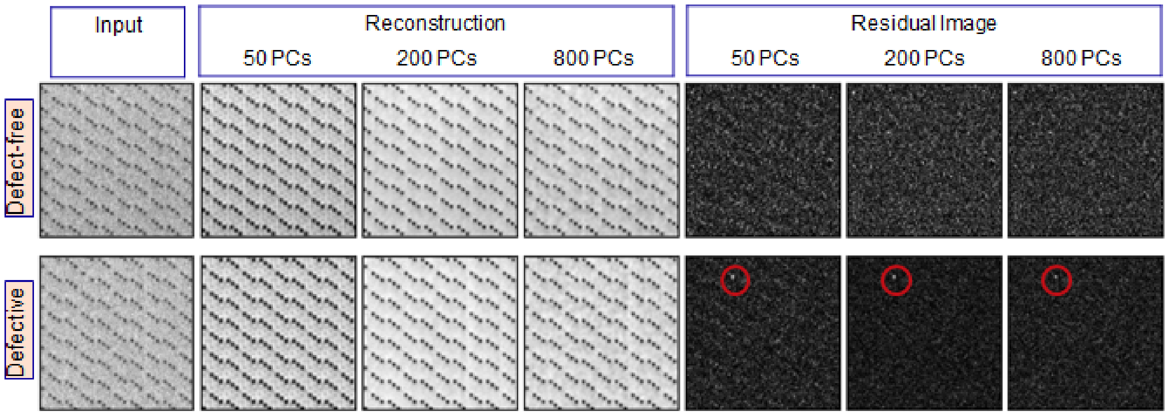


Figure 7. Examples of PCA based reconstructed images and residual images with different number of components. Note that the images are shown after a max-min normalization to [0, 255], thereby the image that is visually less noisy is due to high contrast between noise and defective signal. The higher contrast between noise and defective signal, the easier the defect can be detected from the residual image.

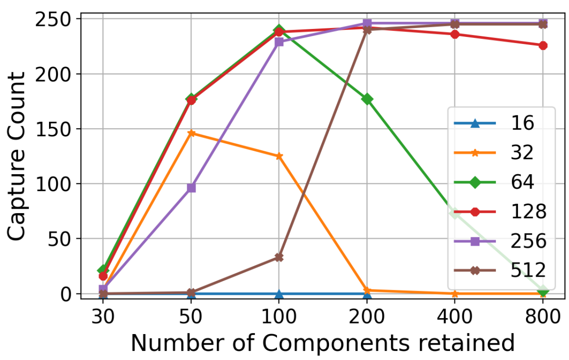


Figure 8. The number of CC at 25 NC (NR = 10%) w.r.t. the retained number of PCs over a range of {30, 50, 100, 200, 400, 800} for different image crop sizes. The image crop sizes cover a range of {16 × 16, 32 × 32, 64 × 64, 128 × 128, 256 × 256, 512 × 512}.

of capturing defects. However, if the image crop size is smaller than periodic unit of the pattern structure (e.g. 16 × 16), the PCA model starts to reconstruct defects, resulting in the poor performance of capturing defects.

5.2. *CAE results on MH dataset*

Our CAE model is implemented as mentioned in Section 4.3 and we investigate two aspects that influence the performance of the CAE model for the defect detection task, i.e. image crop size and latent vector size, and loss function.

Image crop size and Latent space dimension We plot the number of CC at 25 NC (NR = 10%) w.r.t. varying values of latent vector size for different image crop sizes, as presented in Figure 9. It appears that the performance of a CAE model mostly depends on the size of the latent vector when the image crop size is not too small. Regardless the image crop size, when the latent dimension is higher than 50, the performance of the model stays similar, where the number of CC is roughly 240 at 25 NC. We adopt the image size of 71×71 and latent vector size of 100 for the MH dataset as it needs the smallest image size and the least latent vector size, thus less operations per pixel for prediction, to achieve the best performance, where the number of CC at 25 NC is 242 in this case. Note that the CAE model with larger image crop size takes longer time to train.

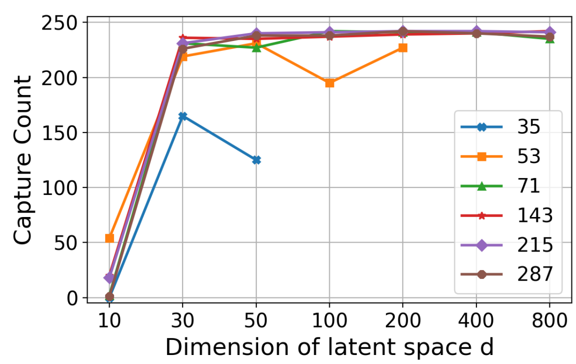


Figure 9. The number of CC at 25 NC (NR = 10%) w.r.t. the dimension of latent vector over a range of {10, 30, 50, 100, 200, 400, 800} for different image crop sizes. The image crop sizes include 35 × 35, 53 × 53, 71 × 71, 143 × 143, 215 × 215, and 287 × 287.

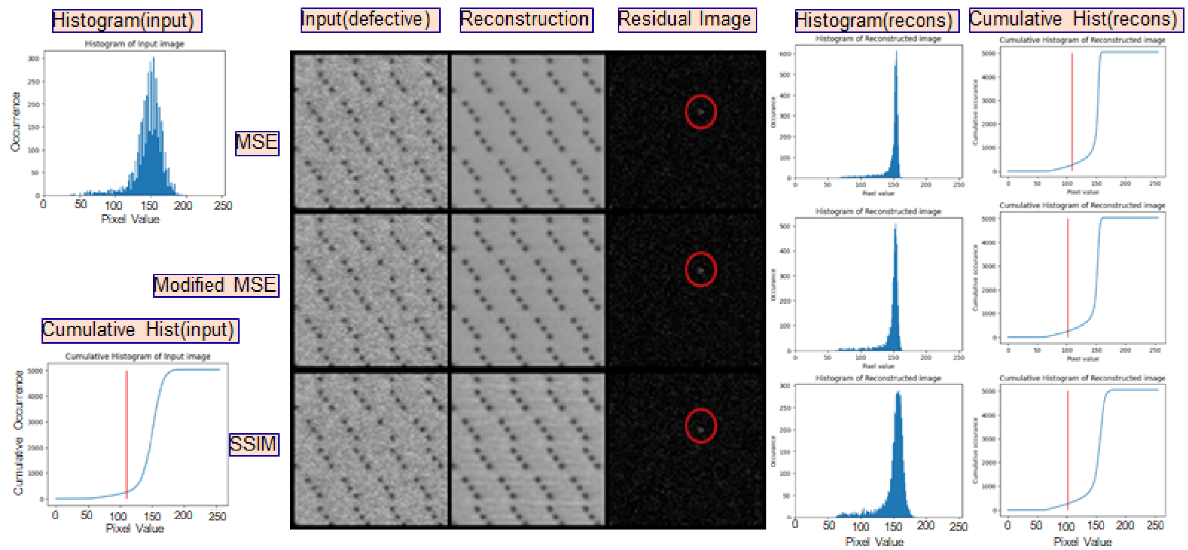


Figure 10. Examples of CAE reconstructed images and residual images with different loss function, in combination with the histogram and cumulative histogram of the input and the reconstructed images. Red vertical lines indicate the pixel value of 5% cumulative pixels, where we assume that the pixels that have values under this pixel threshold are contact holes.

Loss functions As mentioned in Section 3.2, we train the same CAE model with MSE loss (Equation (6)), Modified MSE loss (Equation (8)) and SSIM loss(Equation (10)) respectively, and compare the results to find a best suited loss function for our use case.

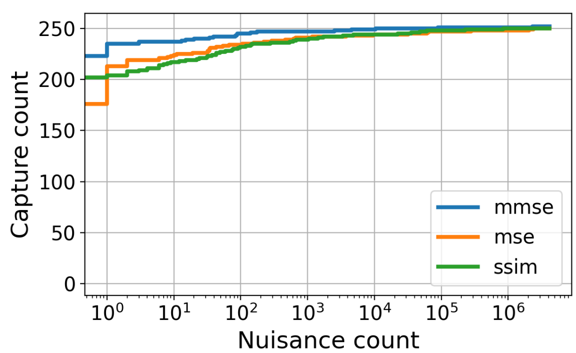


Figure 11. Performance of CAE based models trained by different loss function on 10nm MH dataset.

Figure 10 exemplifies reconstructed images and residual images from the models trained by the different loss functions. To quantify the difference between them, we also plot the histograms and cumulative histograms for the input images and the reconstructed images. Regardless the histogram of the input image or the reconstructed images, there is only one obvious peak at around 150 that is the average pixel value of the background, while the peak for the pixel values of the contact holes is immersed in the noise deviation of the background. Hence, we assume all the dark pixels up to 5% fraction of the cumulative histogram are the contact holes, which the pixel value at 5% is then defined as "pixel threshold" for MMSE loss. This value is roughly 110 in the input image and reconstructed image from MSE loss, and approximately 102 in the reconstructed images from MMSE and SSIM loss. This result shows that the models trained with MMSE and SSIM loss generate more contrast between the background and the contact holes in the reconstructed images, resulting in more distinctions for missing contact defects. In addition, all the reconstructed images contain much less noise than the input image, while the result from SSIM loss retains the most deviations in the background.

MMSE and SSIM have their own hyperparameters, the weight and the pixel threshold for MMSE and window size for SSIM. As mentioned in Section 3.2, the setting of the pixel threshold is set to 110. Experimental results indicate that the model is not so sensitive to this parameter. In another words, slight changes on this value do not influence the performance of the model significantly. The weight for MMSE and the window size for SSIM are set as 10 and 5 respectively. Figure 11 illustrates the performance of the models trained by different loss functions. It can be seen that the model trained by MMSE loss outperforms the other two models. This means manually manipulating the loss function based on prior knowledge can improve the performance of the model. However, at the same time, the manipulation is highly dedicated to one dataset, which makes the model less generalizable to other use cases. In addition, based on the results from MSE loss and SSIM loss, it can be inferred that for MH dataset, which has limited variations on luminance and contrast, a pixel-based method works better than structural-based method.

5.3. Comparison between methods

To validate the effectiveness of the proposed methods, we compare the experimental results from our approaches with the baseline method on the MH datasets. A detailed description about the baseline method is given in Appendix B. Simply put, the baseline method firstly uses filters to suppress noise and integrate signals, and then compares the target inspection image with the reference images. For repetitive patterns, where the exact number of pixels per repetition is known, the baseline method can be adapted to use the shifted target SEM images as the reference images instead of taking from different dies. There are different filtering options for the baseline method, such as Gaussian filters and the averaging filter. The averaging filter cancels out the noise by averaging the images with 6 times periodic shifted version of themselves. Gaussian filters can be used in both repetitive and non-repetitive patterns, while the averaging filter can only be applied to the repetitive patterns. Since the MH datasets have repetitive patterns, we will compare our methods with the baseline method with Gaussian filters and the averaging filter. All the methods are tested on the same test set.

Table 1. NR@90%CR of different methods on MH dataset. The best result is marked in bold, and the results that not the best but meet that requirement of lower than 10%NR@90%CR is underlined.

Method	8nm	10nm	12nm	14nm
Baseline-GF	**0%**	<u>3.4%</u>	60.7%	98.2%
Baseline-averaging	**0%**	<u>2.2%</u>	**0.4%**	**69.1%**
PCA	**0%**	**0%**	<u>8.5%</u>	97.7%
CAE-mse	**0%**	12.3%	18.6%	97.3%
CAE-mmse	**0%**	**0%**	<u>2.2%</u>	91.8%
CAE-ssim	**0%**	17.8%	41.5%	99.5%

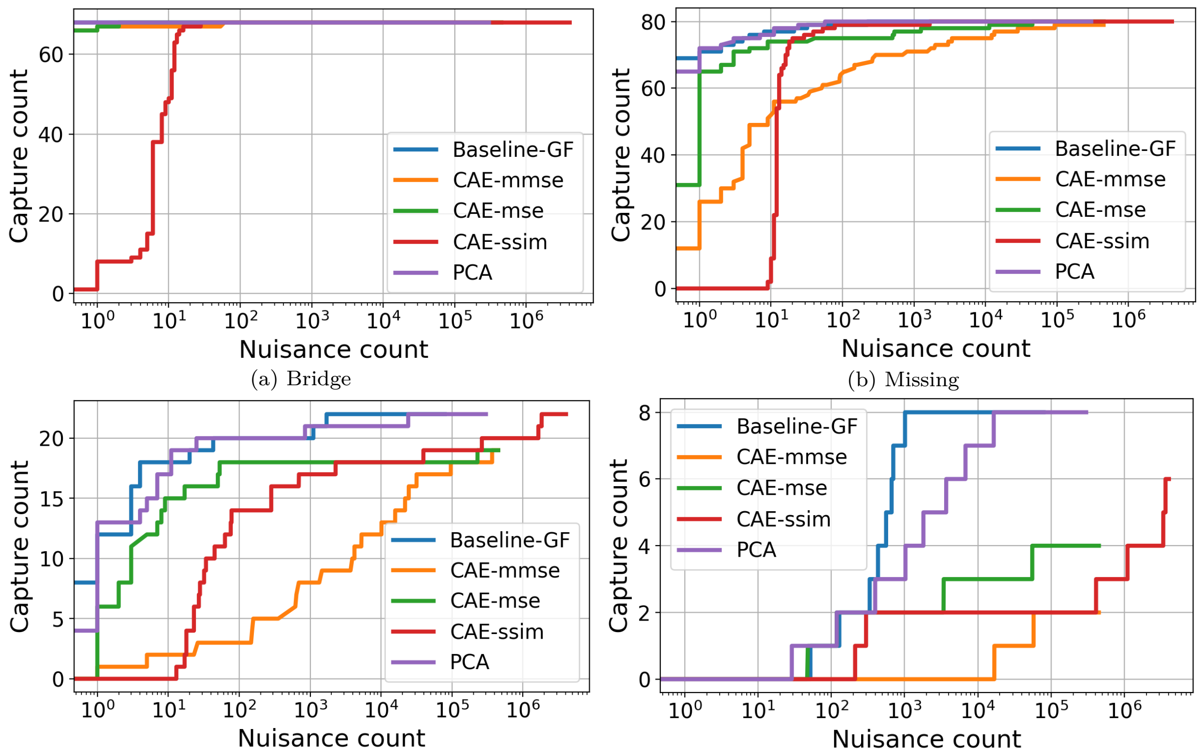
Table 1 summarizes the NR@90%CR achieved by different defect detection methods on 8nm, 10nm, 12nm and 14nm MH datasets. The results presented here are acquired from the models with the best settings from the hyperparameter search on 10nm MH dataset. For 8nm pixel size, all of the methods can locate 90% real defects without any nuisances. For 10nm pixel size, PCA and CAE with MMSE loss reconstruction based methods outperform the baseline method no matter the filter options. For 12nm pixel size, the baseline method with averaging filter achieves the best performance, yet PCA and CAE with MMSE loss methods have satisfactory performance with 8.5% and 2.2% NR@90%CR. When SNR is low and defect size is so small as in 14nm pixel size case, no methodology attains 90%CR@10%NR standard, and the baseline method with the averaging filter performs the best among all the approaches.

5.4. Performance on different defect types based on CH-MTD dataset

To evaluate how well the proposed methods perform on different defect types, we test our approaches on the CH-MTD dataset, which contains the real SEM images of the contact hole pattern and four different types of defects, i.e. bridge, missing, enlarged and shrinkage contact holes. For both PCA and CAE based models, we conduct a hyperparameter search based on CCNC curves for this new dataset.

For the PCA based method, we fit the model with 1,000 image crops from 50 raw images with the size of 128×128. The number of the PCs used for reconstruction is 30, sigma for the filter is 1.5 and the defect size is set as 5. As for the CAE based method, the experimental results show that when different random 64,000 training image crops are used to train the model, the model produces different results. Weight initialization variation and different hyperparameter settings show similar impact on the performance. Therefore, we cannot conclude which hyperparameters give better results. We use one of the best performed experimental results to gain insights into how the method works on different defect types. In addition, we adopt the same way to construct a MMSE loss function for this dataset by setting the pixel threshold at 128. However, it does not outperform MSE loss in this use case. This is possibly due to the fact that there is obvious charging effect around contact hole and the defects in this dataset, so that the found defect locations are actually where these charging effects are obvious instead of the locations of the actual flawed patterns. On the other hand, as we have four different defect types, it is not straightforward how we can better modify the MSE loss that emphasize on the important features (maybe not the contact holes) for the novelty detection task.

Figure 12 demonstrates the performance of the defect detection per defect type achieved by the proposed methods and the baseline method. It is evident that strong defects such as bridge and missing contact holes



(a) Bridge (b) Missing

(c) Enlarge (d) Shrinkage

Figure 12. Performance on the defect detection per defect type achieved by the proposed methods and baseline method with GF filters on CH-MTD dataset.

are the first defects found by all the methods, while the softly enlarge and shrinkage contact hole defects are more challenging. The results meet our expectation since the bridge and missing holes defects usually have larger defect size, hence are easier to locate, especially with the algorithm that looks for a group of high pixel values in the residual image. Overall, the baseline method achieves the best results on all types of the defects and the PCA based method has comparable performance as the baseline method on bridge, missing and enlarged holes. The CAE model with MSE loss and MMSE loss performs very well on the bridge and missing holes defects but struggle to find the defects with small defect size, whereas the CAE model trained by SSIM loss displays the most inferior performance overall in this experiment.

5.5. Performance on different patterns

To examine the generalization performance of the models on different patterns, we further test them on another two datasets, i.e. LS and SRAM datasets. Note that the parameters of our models are tuned based on the different datasets.

Table 2 summarizes the NR@90%CR performance achieved by different defect detection methods on the contact hole pattern dataset CH-MTD, the line space pattern dataset LS and the memory pattern dataset SRAM. We can see that the performance achieved by the PCA based reconstruction method fulfills the industrial standard among all the datasets. In particular, it has slightly higher NR@90%CR than the baseline method on the CH-MTD dataset, but outperforms all the approaches on the LS and SRAM datasets. The

Table 2. NR@90%CR for different methods on different datasets with different patterns. The number in the bracket is the number of defects contained in the testset for each dataset. The best result is marked in bold, and the results that not the best but meet that requirement of lower than 10%NR@90%CR is underlined.

Method	CH-MTD (179)	LS (661)	SRAM (1159)
Baseline-GF	**3.0%**	34.7%	2.5%
PCA	4.7%	**3.1%**	**1.3%**
CAE-mse	12.4%	17.7%	5.8%
CAE-ssim	46%	20.1%	4.3%

CAE based reconstruction method achieves the worst performance among all the datasets. In addition, the CAE model with MSE loss performs better than with SSIM loss on the CH-MTD and LS datasets, but underperforms on the SRAM dataset, where the features of the image data have more complex structures.

5.6. Computational cost

In this section, we present the evaluation of the computational cost of the proposed reconstruction based anomaly detection methods through conducting a numerical complexity analysis.

An overview of the required number of operations per pixel for the baseline method with a GF and the averaging filter, the PCA based method and the CAE based method is given in Table 3. The numbers of operations per pixel for the baseline method with different filtering strategies are adopted from the previous study under the same project. Appendix C presents the detailed numerical complexity analysis for the PCA and CAE based methods on 10nm MH dataset. The result shows that the computational costs of the PCA and CAE based methods in terms of operations per pixel are approximately 143 and 638 respectively, which are within one order of magnitude higher than the baseline method.

Table 3. Comparison between the baseline method with different filtering strategies and the proposed PCA and CAE models in terms of numerical complexity based on MH 10nm dataset.

Algorithm	Operations Per Pixel
Baseline-GF	27
Baseline-Averaging	56
PCA	143
CAE	638

6. Discussion and Future work

In this work, we propose two reconstruction based anomaly detection methods to locate the defects in the SEM images of the wafer. For repetitive patterns, compared to the baseline method that uses the shifted images as the reference images, the proposed methods generate a golden reference image based on the input image. In the case that the repetition of the pattern is not accurate, for instance for 10nm MH dataset, the proposed approaches demonstrate a superior performance. Meanwhile, compared to use CNNs for defect detection from the previous study, the proposed methodology has notable benefits. In particular, the models are only trained on the defect-free data, thereby they do not suffer from the class imbalance problem and the shortage of the defective samples. Besides, the model can be trained in unsupervised manner and detect unknown types of defects. However, the proposed methods are mainly experimented on the SEM images that contain periodic structures. A simple simulated dataset with random contact hole pattern and random missing hole defects is adopted to examine the feasibility of applying the proposed methods to non-repetitive patterns. The preliminary experimental results show that the proposed reconstruction based methods fail to function on non-repetitive patterns, as both PCA and CAE models are not able to learn random structures.

Both PCA and CAE reconstruction based methods have demonstrated their abilities to locate the extremely small unknown defects that consist of several pixels on the SEM images with different repetitive patterns. However, we notice that the CAE based model has inferior performance compared to the PCA based model and unstable behavior when implemented on the CH-MTD dataset. More work is needed to investigate and understand why the model is unstable and where the variations of the performance based on the same hyperparameters come from. Additionally, Reddi, Kale, and Kumar [33] state that Adam optimizer, which is used for training the CAE model in this work, has failed to converge to an optimal solution in many applications. This can be a possible reason that the model performs differently with the same hyperparameters as it converges to different local optima each time. Besides, better hyperparameter search strategy should be considered to further optimize the model.

In addition, there are still many potentials to improve the proposed CAE based model. Firstly, some regulation terms can be added to the loss function in order to get better reconstruction results [16] or include more use case specific information for training in favor of capturing defects to improve overall performance. Secondly, we can add a second step to remove excessive nuisances. In stead of a single threshold on the maximum pixel values of the residual image as the determinant for defects, we can introduce a CNN classifier to better separate the two classes. To avoid the class imbalance problem, we can use the proposed method as the first step and throw away most of the nuisances, thus generate a new dataset that has more balanced ratio between defective and defect-free samples. In this way, we expect the performance on capturing defects will be improved with some extra computational cost.

Along with the investigation on the methodology, we discover that the characteristics of the datasets have great impact on the performance of the detection algorithms. Some individual nuisance is worth looking at to better understand how the PCA model or the neural network works and where the nuisance comes from. It can be caused by noise, inaccurate ground truth, or some operations from the algorithm. For the edge nuisances, we add a post-processing step to remove them to further improve the model towards the best results that are possibly achieved by the approaches. In the future, we should investigate them further and work on strengthening the algorithm by data augmentations to overcome the edge nuisances.

7. Conclusion

With rapid advancement of high throughput wafer inspection systems, there are growing demands on more efficient and accurate defect inspection algorithms. This work focuses on investigating the feasibility of applying novel learning based algorithms to the datapath of the next generation wafer inspection tools. An end-to-end reconstruction-based anomaly detection methodology is proposed with two types of reconstruction models based on PCA and CAE. Although the methodology can only be applied to repeating patterns, yet this covers a large range of use cases in electronic design and manufacturing. We evaluate both models on multiple datasets of the real SEM images that contain different repetitive patterns and diverse defect types, as well as compare the results to the baseline method in terms of the performance and the computational cost. The proposed PCA based method achieves promising results with above 90%CR@10%NR for all the datasets containing different patterns and diverse unknown defect types experimented in this work, except from MH 14nm dataset, where the SNR of the SEM images is so low that the baseline method also struggles to effectively locate the real defects. Besides, it outperforms the baseline method on 10nm MH dataset, LS dataset and SRAM dataset. On the other hand, the overall performance of the CAE based model is not as good as the PCA based method. However, for 12nm and 14nm MH datasets where the SNR of the images is relatively low, the CAE based model trained by MMSE loss function achieves better results than PCA. To conclude, the proposed reconstruction based anomaly detection methods have demonstrated the potentials to achieve better performance on defect inspection tasks on high-throughput SEM images with reasonable computational cost.

Acknowledgements

I would like to express my gratitude to all the people that have supported me during my graduation internship and contributed to this research. Many thanks to my supervisor Dr.Ir. Luuk Spreeuwers for the academic guidance and constructive advises during every progress meeting. Many thanks to Vincent Kuiper for sharing his knowledge and providing me constant support whenever I needed it. Thanks to Marija Trajanoska, Tiago Botari and Moritz Kiehn for their generous helps and the fruitful discussions. I am grateful to the people of ASML Delft for offering me the opportunity to learn at their company. Finally, I would like to thank my parents and my partner for their consistent support, as well as my friends for their company throughout my study.

References

[1] Willem de Moor. *Extreem-ultravioletlithografi.* [Online; accessed September 03, 2019]. 2019. URL: https://tweakers.net/reviews/7282/7/extreem-ultravioletlithografie-zin-en-onzin-van-euv-chipproductie-euv-in-de-praktijk.html.

[2] Liangjiang Yu et al. "SEM image quality enhancement: an unsupervised deep learning approach". In: *Metrology, Inspection, and Process Control for Microlithography XXXIV*. Ed. by Ofer Adan and John C. Robinson. Vol. 11325. International Society for Optics and Photonics. SPIE, 2020, pp. 388–396. DOI: 10.1117/12.2552883. URL: https://doi.org/10.1117/12.2552883.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25 (2012).

[4] Joseph Redmon et al. "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[5] Jonathan Goh et al. "Anomaly detection in cyber physical systems using recurrent neural networks". In: *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*. IEEE. 2017, pp. 140–145.

[6] Mohiuddin Ahmed, Abdun Naser Mahmood, and Md Rafiqul Islam. "A survey of anomaly detection techniques in financial domain". In: *Future Generation Computer Systems* 55 (2016), pp. 278–288.

[7] Jie Liu et al. "Semi-supervised anomaly detection with dual prototypes autoencoder for industrial surface inspection". In: *Optics and Lasers in Engineering* 136 (2021), p. 106324.

[8] Mohammad Havaei et al. "Brain tumor segmentation with Deep Neural Networks". In: *Medical Image Analysis* 35 (2017), pp. 18–31. DOI: 10.1016/j.media.2016.05.004.

[9] Yuansheng Ma et al. "Machine learning based wafer defect detection". In: *Design-Process-Technology Co-optimization for Manufacturability XIII*. Vol. 10962. SPIE. 2019, pp. 38–45.

[10] Yoshihiro Midoh and Koji Nakamae. "Image quality enhancement of a CD-SEM image using conditional generative adversarial networks". In: *Metrology, Inspection, and Process Control for Microlithography XXXIII*. Ed. by Vladimir A. Ukraintsev and Ofer Adan. Vol. 10959. International Society for Optics and Photonics. SPIE, 2019, pp. 37–46. DOI: 10.1117/12.2515152. URL: https://doi.org/10.1117/12.2515152.

[11] Hairong Lei et al. "Denoising sample-limited SEM images without clean data". In: *Metrology, Inspection, and Process Control for Semiconductor Manufacturing XXXV*. Vol. 11611. International Society for Optics and Photonics. 2021, 116111A.

[12] Zhe Wang, Liangjiang Yu, and Lingling Pu. "Defect simulation in SEM images using generative adversarial networks". In: *Metrology, Inspection, and Process Control for Semiconductor Manufacturing XXXV*. Ed. by Ofer Adan and John C. Robinson. Vol. 11611. International Society for Optics and Photonics. SPIE, 2021, pp. 113–119. DOI: 10.1117/12.2581881. URL: https://doi.org/10.1117/12.2581881.

[13] David Zimmerer et al. "Unsupervised anomaly localization using variational auto-encoders". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2019, pp. 289–297.

[14] Philip Cho et al. "Defect Detection in Atomic Resolution Transmission Electron Microscopy Images Using Machine Learning". In: *Mathematics* 9.11 (2021), p. 1209.

[15] Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. "Ganomaly: Semi-supervised anomaly detection via adversarial training". In: *Asian conference on computer vision*. Springer. 2018, pp. 622–637.

[16] Du-Ming Tsai and Po-Hao Jen. "Autoencoder-based anomaly detection for surface defect inspection". In: *Advanced Engineering Informatics* 48 (2021), p. 101272.

[17] Takeshi Nakazawa and Deepak V Kulkarni. "Anomaly detection and segmentation for wafer defect patterns using deep convolutional encoder–decoder neural network architectures in semiconductor manufacturing". In: *IEEE Transactions on Semiconductor Manufacturing* 32.2 (2019), pp. 250–256.

[18] Jinwon An and Sungzoon Cho. "Variational autoencoder based anomaly detection using reconstruction probability". In: *Special Lecture on IE* 2.1 (2015), pp. 1–18.

[19] Lucas Deecke et al. "Image anomaly detection with generative adversarial networks". In: *Joint european conference on machine learning and knowledge discovery in databases*. Springer. 2018, pp. 3–17.

[20] Martin Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein generative adversarial networks". In: *International conference on machine learning*. PMLR. 2017, pp. 214–223.

[21] Tsuyoshi Kondo et al. "Massive metrology and inspection solution for EUV by area inspection SEM with machine learning technology". In: *Metrology, Inspection, and Process Control for Semiconductor Manufacturing XXXV*. Vol. 11611. International Society for Optics and Photonics. 2021, p. 1161111.

[22] Zhigang Wang et al. "High-accuracy, high-speed, and smart metrology in the EUV era". In: *Metrology, Inspection, and Process Control for Microlithography XXXIV*. Vol. 11325. International Society for Optics and Photonics. 2020, 113251Q.

[23] Peter Evanschitzky et al. "Mask defect detection with hybrid deep learning network". In: *Journal of Micro/Nanopatterning, Materials, and Metrology* 20.4 (2021), p. 041205.

[24] Masanori Ouchi et al. "A trainable die-to-database for fast e-Beam inspection: learning normal images to detect defects". In: *Metrology, Inspection, and Process Control for Microlithography XXXIV*. Ed. by Ofer Adan and John C.

Robinson. Vol. 11325. International Society for Optics and Photonics. SPIE, 2020, pp. 437–445. DOI: 10.1117/12.2551456. URL: 10.1117/12.2551456.

[25] Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. “A systematic study of the class imbalance problem in convolutional neural networks”. In: *Neural Networks* 106 (2018), pp. 249–259.

[26] Justin M Johnson and Taghi M Khoshgoftaar. “The effects of data sampling with deep learning and highly imbalanced big data”. In: *Information Systems Frontiers* 22.5 (2020), pp. 1113–1131.

[27] Nina Shvetsova et al. “Anomaly detection in medical imaging with deep perceptual autoencoders”. In: *IEEE Access* 9 (2021), pp. 118571–118583.

[28] Manpreet Singh Minhas and John Zelek. “Semi-supervised anomaly detection using autoencoders”. In: *arXiv preprint arXiv:2001.03674* (2020).

[29] Xinghui Dong, Chris J Taylor, and Tim F Cootes. “Small defect detection using convolutional neural network features and random forests”. In: *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*. 2018, pp. 0–0.

[30] Paul Bergmann et al. “Improving unsupervised defect segmentation by applying structural similarity to autoencoders”. In: *arXiv preprint arXiv:1807.02011* (2018).

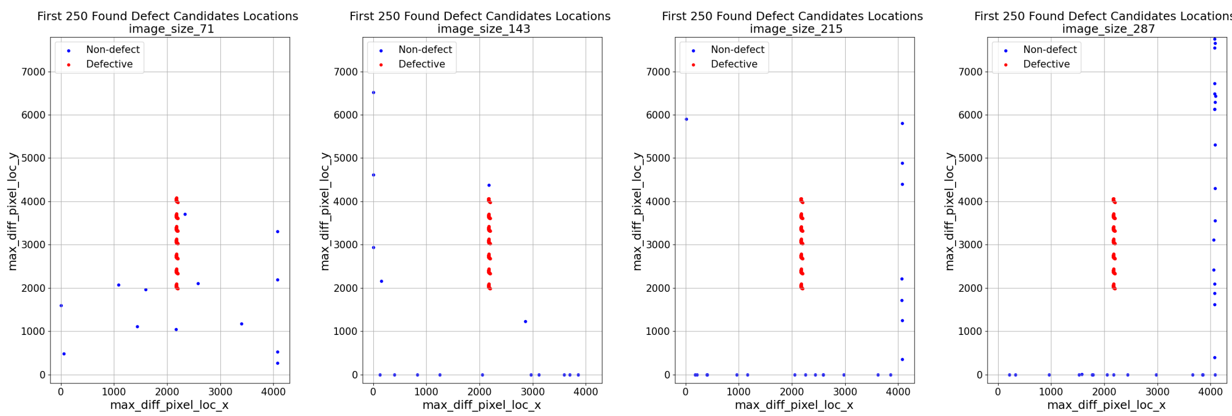
[31] Zhou Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE transactions on image processing* 13.4 (2004), pp. 600–612.

[32] Takaya Saito and Marc Rehmsmeier. “The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets”. In: *PloS one* 10.3 (2015), e0118432.

[33] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. “On the convergence of adam and beyond”. In: *arXiv preprint arXiv:1904.09237* (2019).

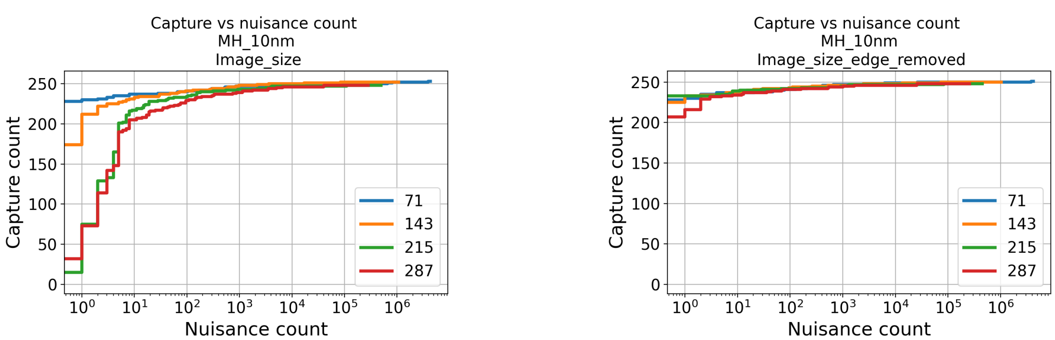
A. Edge nuisances

In this appendix, we demonstrate the detailed experimental results on edge nuisances. The phenomenon that many nuisances found at the edge of the SEM images exists for all the datasets with both PCA and CAE methods. However, the CAE based model suffers from it more than the PCA based method. Here we use the MH 10nm dataset with the CAE based model as an example to illustrate the edge nuisances. There are three findings about the edge nuisances. Firstly, it is found that the larger image crop size we choose, the more edge nuisances are found. Secondly, increasing the convolutional layers of the CAE model can introduce more nuisances at the edge of each image crop. Finally, it is noticed that using SSIM loss function to train the models arouse more edge nuisances than MSE loss function. Figures 13 to 18 presents the plot of the locations of the first 250 found defect candidates who have the strongest signals in the residual images and the influence of removing the edge nuisances on the CCNC curves for each aforementioned case. These nuisances are not actual nuisances considering that they are known to be caused by edge artifacts, distortions or deconvolutional operations. We can remove them by improving the algorithm. Therefore, we remove them at the post processing step for fair evaluation and comparison.



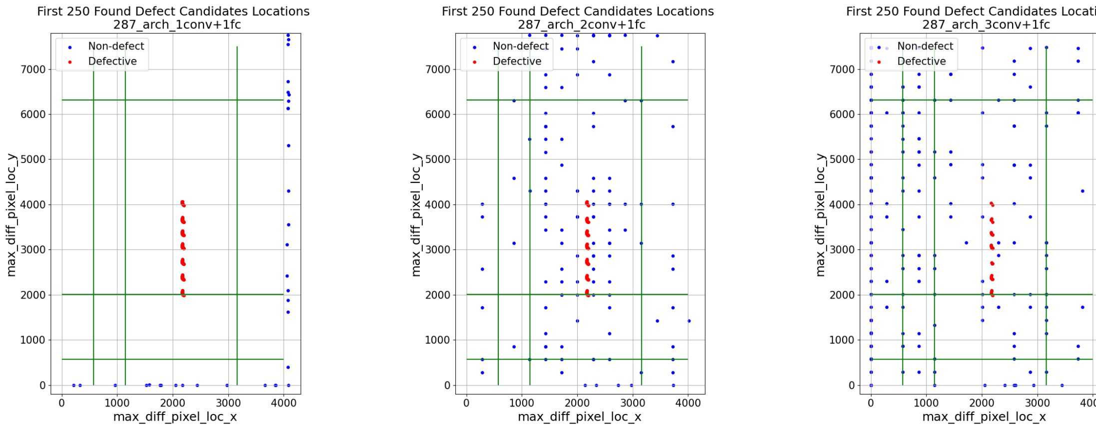
(a) 71 × 71 (b) 143 × 143 (c) 215 × 215 (d) 287 × 287

Figure 13. First 250 found defect candidates locations for different image crop sizes. The candidates include defects marked as red and nuisance marked as blue. Notice that with larger image size, there more nuisance found by the algorithm and they are located mostly at the edge of the SEM images.



(a) Before removing edge nuisance (b) After removing edge nuisance

Figure 14. The influence of removing the edge nuisances on the CCNC curves for different image crop sizes. Before removing the edge nuisances, smaller image crop size yields better results. This is due to the fact that with smaller image crop, the model can learn the edge artifacts better. After removing the edge nuisances, the models with different image crop sizes have similar performance. In another word, the choice of image corp size has little influence on the overall performance of the CAE model.

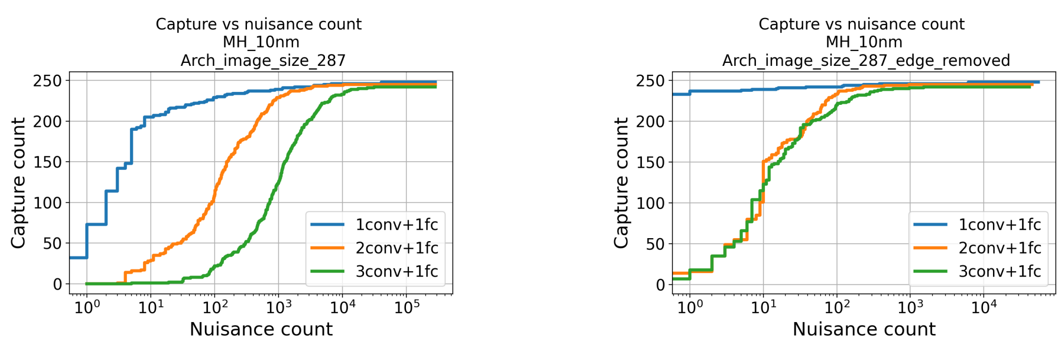


(a) 1 conv + 1 FC

(b) 2 conv + 1 FC

(c) 3 conv + 1 FC

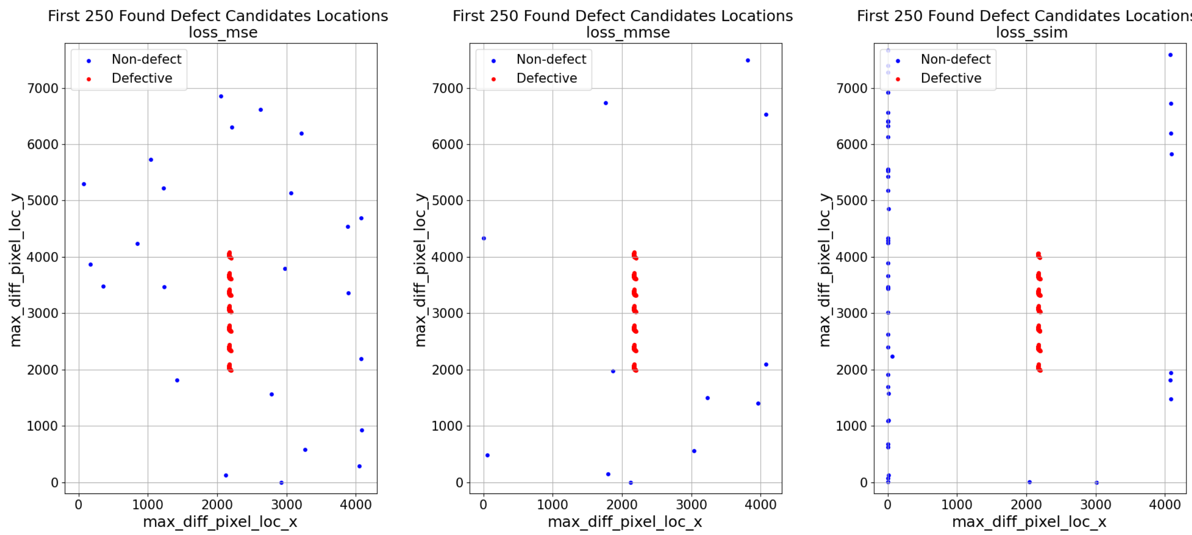
Figure 15. First 250 found defect candidates locations for image crop size of 287×287 with different numbers of convolutional/deconvolutional layers. The candidates include defects marked as red and nuisance marked as blue. Notice that with increasing convolutional layers, we can see a checkerboard pattern of the nuisances that are located at the corner edge of each image crop.



(a) Before removing edge nuisance

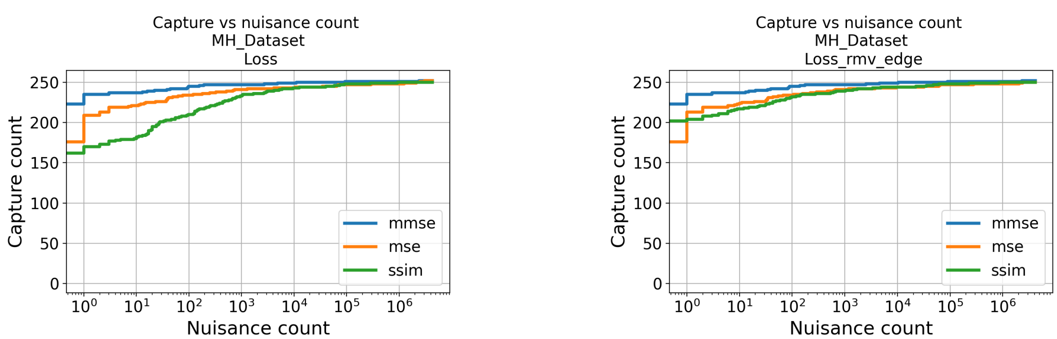
(b) After removing edge nuisance

Figure 16. The influence of removing the edge nuisances on the CCNC curves for different architectures of the CAE model with image crop size of 287×287. Based on the results, after removing the checkerboard pattern of nuisances caused by deconvolution operations, the proposed architecture still outperforms the others.



(a) MSE loss (b) MMSE loss (c) SSIM loss

Figure 17. First 250 found defect candidates locations for applying different loss functions. The candidates include real defects marked as red and nuisances marked as blue. We can see the results from the MMSE loss function have the least nuisances. The CAE model trained by SSIM loss function produces edge nuisances as well. It can be due to the fact that the structure-based loss learns more global features than the pixel-based loss MSE, thereby easier influenced by edge artifacts.



(a) Before removing edge nuisance (b) After removing edge nuisance

Figure 18. The influence of removing the edge nuisances on the CCNC curves for using different loss functions to train the CAE model. After removing the edge nuisances, the Modified MSE loss function is still the best loss function to use in the case of MH 10nm dataset.

B. The baseline method

The baseline method is based on the basic principle of wafer defect inspection, i.e comparison between a target image and two reference images. It is a D2D inspection with both target images and reference images acquired from the wafer at different dies of identical circuits. Two reference images are used to avoid the situation that the reference image itself has defects. It is believed that the chance is extremely rare that both reference images has the same defects.

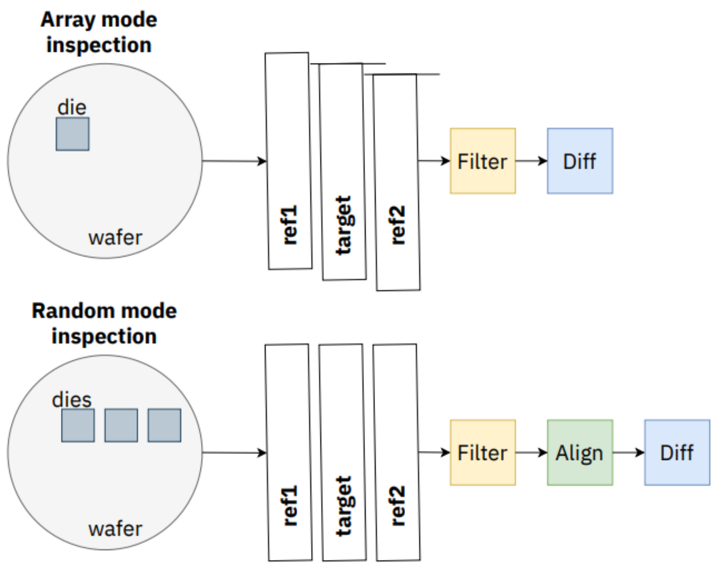


Figure 19. Inspection modes overview.

The baseline method includes two inspection modes defined on if the patterns on the wafer are repetitive or non-repetitive, which are named "array mode" and "random mode" respectively. The major difference between the two modes is that array mode utilizes the prior knowledge about the repetitiveness over the patterns etched on the wafer. A single image then can be compare to itself with a shift of one repetition period. The overview of the two inspections modes is demonstrated in Figure 19. For both modes, filtering is an essential step for a decent performance as the process noise is inevitable for the wafer inspection tools.

In the array mode setup, a target image is compared to the shifted versions of itself as the reference images. There are obvious gains to consider repetitive use cases separately. First of all, the target image and the reference images are captured by the same beam, which means the variations due to the beam characteristics, image brightness and contrast are limited, consequently resulting in less nuisance count. Secondly, as we have the privilege of knowing exactly the repeating period to the pixel scale, image alignment and subsequently interpolation step can be omitted. Even better, we could generate a golden reference image with so-called "the averaging filter", which computes an average image of multiple shifted ones. Thirdly, the resources for the storage of multiple images can be saved.

As for random mode, the target image is compared with the reference images captured from the same location/pattern at the different dies by other beams. Therefore, image alignment is needed before calculating the difference image, and an interpolation step is added to improve the accuracy of the image alignment. There are multiple available interpolation techniques, such as the nearest neighbour, bilinear and bicubic etc., but choice should be made with the awareness of the trade-off between accuracy and computational cost.

C. Numerical complexity analysis

C.1. PCA

1. Filter the whole SEM image with kernel size k, computational cost is $2k^2$ operations per pixel, then crop the whole image into test image crops;

2. Feed a test image crop $x_i(N \times N)$ flattened to $1 \times N^2$ into the PCA model trained by M image crops, which contains eigenvector and mean: $V(K \times min(N^2, M))$ and $\mu(1 \times N^2)$;

3. Calculate the weights $W(1 \times K)$ on the trained K PCs for the test image: $W = (x_i - \mu) \cdot V^T$. Computational cost is $\frac{(2 \times min(N^2,M)-1)K+N^2}{N^2}$ operations per pixel;

4. Reconstruct the image: $x_{i\_re} = \mu + W \cdot V$. The calculation has the same number of the operations as the previous step;

5. Calculate difference between the test image and reconstructed image: $x_i - x_{i\_re}$, which cost 1 operation per pixel;

6. Take n candidates to calculate the summation of the block with defect size $d \times d$: $n(d^2 - 1)$ in total, then divided by the total number of pixels for the averaged operations per pixel. Assume $n << N$ and $d < N$, $\frac{n(d^2-1)}{N^2}$ is less than 1 operation per pixel;

7. With thresholding, we can then decide if this test image contains a defect or not.

As detailed out above and N=256, K=200, M=10,000, n=5, d=3, the total computational cost for the PCA reconstruction-based defect detection method is roughly 143 operations per pixel.

C.2. CAE

For neural networks, the testing stage is a process of forward propagation. Note that for one neuron in the neural network, the output can be written as:

$$y = \sigma(\sum_{i=0}^{n} w_i x_i - w_0) \tag{13}$$

where σ is the activation function, w_i are the weights and w_0 is the bias. To compute one output from one neuron (neglect the activation function), the number of operations is roughly two times of the number of weights, including both multiply and sum operations.

For the convolutional layer with K filters that has kernel size of $k \times k$, the total number of weights is $K \times k^2$ per neuron, thus the total number of operations per neuron is then $K \times 2k^2$. Considering the input image with size of $m \times m$ is fed into this convolutional layer with the stride s, the total number of neurons is $(\frac{m-k}{s} + 1)^2$. Hence in total, a convolutional layer consist of $K \times 2k^2 \times (\frac{m-k}{s} + 1)^2$ operations. For dense layers, the number of weights equals to the number of neurons in the previous layer, denoted as n_{l-1}, where l refers to the current layer, then the operations for this dense layer in total is $n_l \times 2n_{l-1}$. Now consider our CAE model with image crop size of 71×71 and latent dimension of 100:

1. Convolutional layer with 6 filters with kernel size of 3 has $6 * 2 * 3^2 * 35^2 = 132,300$ operations;

2. Dense layer with 100 neurons has $2 * 6 * 35^2 * 100 = 1,470,000$ operations;

3. Dense layer with 7350 neurons has $2 * 6 * 35^2 * 100 = 1,470,000$ operations;

4. Transpose convolutional layer is as the same as the first convolutional layer, thereby has $6*2*3^2*35^2 = 132,300$ operations;

5. Calculating difference between the test image and reconstructed image costs 1 operation per pixel;

6. Taking n candidates to calculate the summation of the block with defect size $d \times d$ costs less than 1 operation per pixel;

7. With thresholding, we can then decide if this test image contains a defect or not.

The total number of the operations for the proposed CAE model is 3,204,600, totalling approximately 636 operations per pixel. Then the holistic CAE reconstruction-based defect detection method costs roughly 638 operations per pixel.