Delft University of Technology
Master of Science Thesis in Embedded Systems

Unsupervised Wafer Map Failure Pattern Recognition with Contrastive Learning

Zhaoxi Liu

TU Delft Delft University of Technology

Unsupervised Wafer Map Failure Pattern Recognition with Contrastive Learning

Master of Science Thesis in Embedded Systems

Embedded Systems Group
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Van Mourik Broekmanweg 6, 2628 XE Delft, The Netherlands

Zhaoxi Liu
z.liu-50@student.tudelft.nl
liuzhaoxi1205@gmail.com

24/10/2023

Author
Zhaoxi Liu (z.liu-50@student.tudelft.nl)
(liuzhaoxi1205@gmail.com)
Title
Unsupervised Wafer Map Failure Pattern Recognition with Contrastive Learning
MSc Presentation Date
30/10/2023

Graduation Committee

Leonard Lensink	ITEC
Qing Wang	Delft University of Technology
Xucong Zhang	Delft University of Technology

Abstract

This master's thesis explores the application of Self-Supervised Contrastive Learning (SSCL), specifically the SimCLR algorithm, to enhance feature representation learning from Wafer Bin Maps (WBM) in the semiconductor manufacturing process. The motivation stems from the industry's growing need for automated defect detection and root-cause analysis as electronic devices become more complex. Traditional manual inspection methods fall short in meeting these demands due to cost and time constraints. The study successfully leverages SSCL to extract meaningful feature representations, optimizing label efficiency and improving defect pattern recognition. Furthermore, a comprehensive pipeline for analysis on Nexperia's data is established, including data acquisition, preprocessing, training, testing, and interactive visualization of feature spaces. The research contributes to the automation of wafer map inspection, resulting in potential cost savings and enhanced process control in semiconductor manufacturing.

Preface

On March 2nd, Leonard, Wiljan, and I held a meeting with process engineers from Nexperia's wafer fab in Manchester. The aim was to gain a clearer understanding of our project's objectives. The engineers were troubled by the recurrent appearance of a defect in the top-right corner of wafers. Because they anticipated this issue but couldn't do anything for now, the problem is that it caused a significantly higher failure rate compared to historical data. Consequently, it triggered an alarm for manual inspections on every wafer, forcing the engineers to relax the inspection limit at the cost of potentially overlooking other defects. This situation prompted us to develop an algorithm dedicated to automating the manual inspection process, relieving the burden on the process engineers. We also confirmed in this meeting that we had no labeled data available. Consequently, we resolved to tackle this challenge in a label-efficient manner.

Throughout this journey, I developed a keen interest in deep learning and gained valuable experience in the realm of "the dark matter of intelligence" — self-supervised learning. I created practical tools and algorithms, which were rigorously validated using public datasets. Importantly, these tools and algorithms proved to be directly applicable to Nexperia's proprietary dataset. I am confident that the discoveries made during this project will facilitate the advancement of a more intelligent Industry-4.0-level wafer inspection tool, which could enhance ITEC and Nexperia's value and success.

I want to express my gratitude to my supervisors at ITEC, Dr. Leonard Lensink and Dr. Jan Driessen. Your guidance made my time at ITEC a great joy as you always gave me valuable suggestions when I encountered any trouble. I am also deeply thankful to Dr. Qing Wang for your generousity to be my supervisor and support my decisions. Our interactions have always been smooth and relaxing, and your support has meant a lot to me. Besides, I would like to extend my appreciation to Dr. Xucong Zhang for joining my thesis committee. I would like to acknowledge my friend Evander van Wolfswinkel for his help, especially during my early days in deep learning. I am also thankful to my girlfriend, Shiru as she would always cheer me up whenever I felt low. Finally, I want to thank my colleagues and family who have always supported me.

Zhaoxi Liu

Delft, The Netherlands
25th October 2023

Contents

Chapter 1

Introduction

The electronics industry is one of the most rapidly evolving, innovative, and intensely competitive sectors. Historically, semiconductor companies primarily focused on designing superior products to ensure profitability. Nonetheless, in recent decades, the mounting competition requires these companies not only to design but also to efficiently and cost-effectively manufacture their products. [26] Furthermore, as Integrated Circuits (IC) continuously evolve and scale with state-of-the-art designs and increasing complexity, there is a corresponding rise in defect complexity and frequency. The escalation of this issue highlights the urgent requirement for accurate, real-time quality monitoring and control tools. These tools are essential for achieving high yield, improving cost-efficiency, and ensuring optimal performance [21].

Inspecting Wafer Maps (WM) plays a pivotal role in enhancing production yield and assessing the manufacturing process. After the fabrication of ICs on wafers, the semiconductor devices proceed to the Wafer Probe stage. Here, they undergo a series of electrical tests to determine whether they function properly, the test results are visualized by the Wafer Map. Devices that fail these tests are prevented from advancing to subsequent stages to avoid additional costs. Defective dice on a wafer tend to form distinct spatial patterns, which indicates the underlying issues in the manufacturing process and provides valuable insights on improving yield in the foundry. Common wafer map defect patterns include: *None, Edge-Ring, Edge-Local, Center, Local, Scratch, Random, Donut* and *Near-Full*, as illustrated in the public WM-811K [38] real-world dataset. Figure 1.1 showcases representative examples of these patterns.

Traditionally, Wafer Maps with high failure rates are flagged and sent to specialized engineers for manual inspection and Root-Cause Analysis (RCA), as defect patterns can indicate potential causes of process variation. The manual inspection process is both costly and time-intensive, therefore, it has been falling short of meeting quality control demands as semiconductor production grows in complexity and scale. Consequently, the demand for automated defect detection has been growing.

In recent years, deep learning-based methods have achieved groundbreaking progress in wafer map defect detection. However, most of these approaches rely on supervised learning, particularly Convolutional Neural Networks (CNNs). The effectiveness of these methods heavily relies on the creation of a large and high-quality labeled dataset that is specifically tailored to each company's

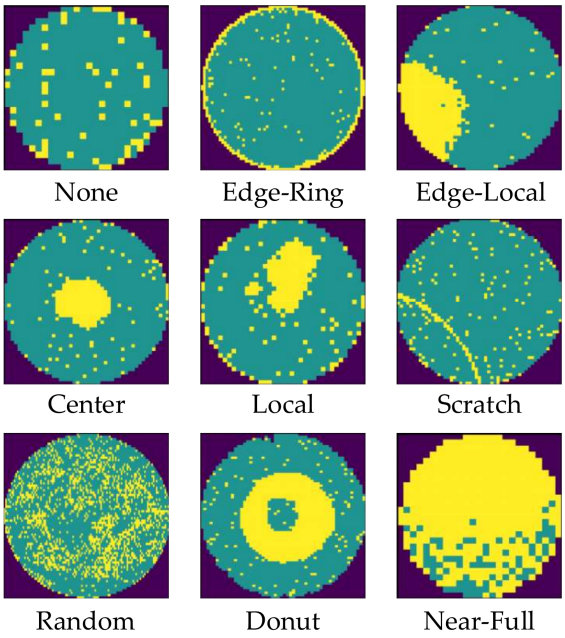


Figure 1.1: **Different Wafer Map Failure Patterns (yellow square indicate defective dice). [39]**

unique products. This step is quite costly and poses a significant challenge for practical implementation.

On the other hand, in the manufacturing process, a substantial amount of unlabeled test data is generated daily. This has led to a shift in research focus towards harnessing the potential of this abundant unlabeled data to improve wafer analysis in the past three years. One promising approach that has gained traction is contrastive learning. Contrastive learning enables the model to learn powerful representations by contrasting positive pairs (similar instances) and negative pairs (dissimilar instances), thus enabling the discrimination of subtle patterns in unlabeled data. By applying contrastive learning to wafer map analysis in a self-supervised manner, I aim to extract meaningful features of defect patterns from unlabeled data and apply the features to improve defect detection and classification.

1.1 Research Objective

In this thesis, I employ Self-Supervised Contrastive Learning (SSCL) to maximize the utility of the vast amount of unlabeled wafer test data for various downstream tasks. From the perspective of academic research, I summarize the following research objectives:

1. Apply SSCL to solve the lack of labeled data problem in the Wafer Probing stage.

2. Improve the quality of representation learning for Wafer Map Failure Pat-

tern Recognition(WMFPR) using a proper configuration for the SSCL model.

1.2 Business Objective

In collaboration with domain experts, I established the following objectives in terms of the business value of this project:

1. Study how to use the proposed method to improve the production quality and overall yield.

2. Study how to apply the extracted features to help engineers identify similar wafer maps of a given query image.

1.3 Research Challenges

- Lack of Labeled Data: A major obstacle that prevents the practical application of traditional machine learning techniques is the need to construct a significant amount of task-specific labeled data. This challenge also applies to our project. Due to the considerable cost associated with creating a large and high-quality dataset, I need to explore label-efficient solutions by harnessing the abundance of unlabeled wafer data. Our approach's effectiveness will be validated on a publicly available dataset.

- Imbalanced and Unknown Data Distribution: Real-world data often exhibits a long-tail distribution, particularly in the case of wafer test data, where the majority of data samples are normal. Additionally, the occurrence of defect patterns varies depending on the specific product and the conditions in the wafer fab. With the absence of labels, I had no idea about the specific distribution of our private dataset.

- Variations in Data: Defect patterns exhibit substantial variations in terms of their area, shape, and location; Depending on the specific product, a wafer can contain anywhere from a few hundred to over ten thousand dice; Random failures may occur due to environmental factors, such as particles in the clean room, which are commonly present in all wafer maps; In some cases, wafers may include alignment markers designed for precise wafer positioning. These markers can result in square or rectangular-shaped untested regions on the wafer maps. All these diverse variations pose a challenge to the model's generalization ability.

- Design of Specific Augmentation and Hyperparameter Tuning: While contrastive learning frameworks like SimCLR (A Simple framework for Contrastive Learning of visual Representations) [5] have been made available online by their developers, there have been limited studies on the impact of specific experiment settings, such as augmentation combinations. Currently, The effectiveness of these implementations relies solely on empirical analysis.

1.4 Contributions

The main contributions of this thesis can be summarized as follows:

1. Application of the SimCLR Self-Supervised Contrastive Learning model to leverage the abundance of unlabeled Wafer Bin Map (WBM) data for the extraction of general feature representations suitable for downstream tasks.

2. Investigation of a range of domain-specific data transformations and experiment settings aimed at optimizing the quality of representation learning.

3. Establishment of a comprehensive pipeline for analyzing Nexperia's Wafer Test data, including data acquisition through Web Scraping, data conversion to wafer bin map format, and the processes of training, testing, and interactive 2D/3D visualization of the feature space.

4. Evaluation of the quality of the extracted representations on a small balanced dataset, enabling satisfactory classification performance even with very limited labeled data. Additionally, testing the model's ability to generalize to Nexperia's data and adapt to unknown/mixed-type patterns.

1.5 Thesis Content

The remainder of the thesis is structured as follows:

- Chapter 2 Background Infomation: This chapter introduces relevant background knowledge of the semiconductor manufacturing process, key concepts applied, methods, and related works on WMFPR.

- Chapter 3 Methodology: This chapter provides implementation details of the SimCLR framework, including data preprocessing steps, unsupervised pre-training phase, supervised fine-tuning phase and evaluation metrics applied.

- Chapter 4 Evaluation: This chapter covers the properties of the dataset and presents qualitative and quantitative evaluation results.

- Chapter 5 Discussion: This chapter concludes the thesis by addressing the research and business objectives, discussing limitations, and suggesting future work.

Chapter 2

Background Information

This chapter delves into the foundational concepts related to the research topic:

- Section 2.1 outlines the broader landscape of WMFPR, highlighting both the prevailing quality control methodologies in use and the rising imperative for automated detection.
- Section 2.2 focuses on the pivotal component of the project: deriving robust feature representations from raw wafer map data. This section elaborates on the significance of features and the methodologies employed for feature extraction.
- Section 2.3 shifts the spotlight to contrastive learning, in which we give an overview of its foundational theory, historical evolution, pros and cons, and explore some popular frameworks.

2.1 Wafer Map Failure Pattern Recognition

2.1.1 Semiconductor Wafer Fabrication and Inspection

The semiconductor manufacturing process can be broken down into four main stages: Wafer Fabrication, Wafer Probing, Assembly, and Final Test, as depicted in Figure 2.1 [26]. Post the fabrication stage, electrical testing is conducted to validate if the device conforms to product specifications, ensuring that only functional units progress to the subsequent manufacturing phase [9]. A Wafer Map (WM) is produced to illustrate specific failure patterns, providing essential insights to assist engineers in pinpointing root causes and enhancing production yield as well as process reliability [38]. These wafer maps, when representing dice in binary form—with defective dice marked as logic '1' and the non-defective ones as logic '0'—are also termed as Wafer Bin Map (WBM) [9].

2.1.2 Wafer Bin Map(WBM) Failure Pattern analysis

The failure patterns of WBM can be classified into three major categories [13]:

(1) Random defect: No spatial clustering or pattern exists, and the defective chips are randomly distributed in the two-dimensional map. Random defects

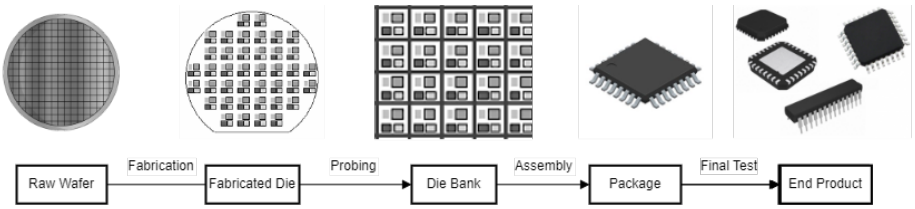


Figure 2.1: **Four main stages of IC manufacturing: fabrication, probing, assembly and final test. [2]**

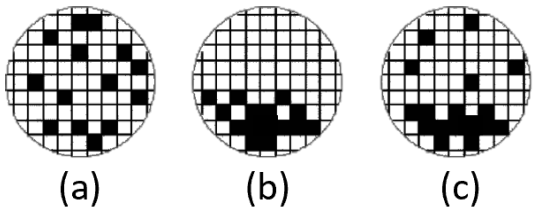


Figure 2.2: **Defect types: (a) Random, (b) Systematic, and (c) Mix of both. [2]**

are usually caused by manufacturing environmental factors. Even in a near-sterile environment, particles cannot be removed completely. Such defects are long-term and expensive to correct [9].

(2) Systematic defect: The positions of defective chips in the wafer show the spatial correlation (die-to-die correlation), for example, Center, Edge-ring, and Scratch. Systematic defects are caused by the manufacturing processes and can be prevented by controlling the causing process. For example, spatial clusters can result from uneven temperatures, chemical aging, crystalline nonuniformity, photo-mask misalignment, or particles caused by mechanical vibration. Stepper and/or probe malfunctioning and sawing imperfections are major causes of repetitive patterns. Material shipping and handling also can leave a scratch on the wafer map [15]. Thus, the identification and analysis of systematic defects facilitate process engineering and improve product quality and die yield by minimizing the rate of defective dice.

(3) Mixed defect: Consisting of a random defect and a systematic defect in one map. Most wafer maps are of this type, as shown in Figure 2.2. Engineer needs to separate random and systematic defects in the WBM, since the systematic defect's signature can reveal potential process issues.

2.1.3 Wafer-Level Quality Control

One of the major quality control metrics for wafer testing stage is Maverick Lot Handling (MLH). This statistical methodology identifies lots(a batch of wafers) exhibiting deviations from the norm. The term "Maverick" denotes outliers—either wafers or dice—that do not conform to expected behaviors or trends, suggesting potential issues in manufacturing or testing processes.

In a large-scale manufacturing environment, it's expected that a majority of the products (in this case, wafers or dice) will behave in a predictable, consistent

manner. Using statistical models and analysis, we can define a "normal" range of behaviors or results. Mavericks are those that fall outside of this normal range. For example, if a particular lot has significantly higher test failures than the average, it may be labeled a maverick.

Addressing the root causes of maverick lots can lead to increased manufacturing yields. However, while MLH proves indispensable in semiconductor manufacturing, it does come with inherent limitations:

1. Overemphasis on Outliers: While focusing on maverick lots, there's a risk of ignoring broader systemic issues that might be subtly affecting all lots, not just the outliers. MLH might also overlook more granular issues that affect only specific patterns on individual wafers.

2. Limitations in Predictive Models: Predictive models used in MLH are based on historical data. If there are unprecedented events or new defects introduced, the models might not catch them immediately.

3. Scalability Issues: MLH often works based on predefined parameters to identify outliers. As manufacturing processes evolve and become more complex, the MLH system might need continuous fine-tuning to accommodate emerging data trends and production scenarios.

In summary, while MLH provides a valuable perspective on manufacturing consistency and quality, its effective deployment requires careful consideration and continuous oversight. Moreover, even when a wafer map is flagged, a manual review is essential to identify the root causes. Given the cost of manually inspecting a vast amount of data, only a tiny fraction of test data is typically designated as maverick. Consequently, MLH could only provide a somewhat constrained and inflexible approach, potentially hindering comprehensive wafer-level quality control.

2.2 Feature

Features encapsulate the essential information of input data. Depending on the model and the chosen learning strategy, features can be obtained either through Manual Feature Generation or Representation Learning.

Consider Figure 2.4 as an illustrative example of the importance of good representations for training shallow machine learning models. If an attempt is made to use a linear model, like Logistic Regression, to delineate a boundary separating the blue and green samples, it would be evident that a linear model falls short in this scenario as the two classes can't be linearly separated in the given data representation.

Yet, the scenario shifts dramatically when we tweak our representation strategy. Instead of utilizing the raw data, if we were to input the square magnitude of the values, the entire landscape of the data undergoes a transformation. This new feature space clearly delineates the two classes, making a linear separation readily achievable. This underscores a salient point: representations matter.

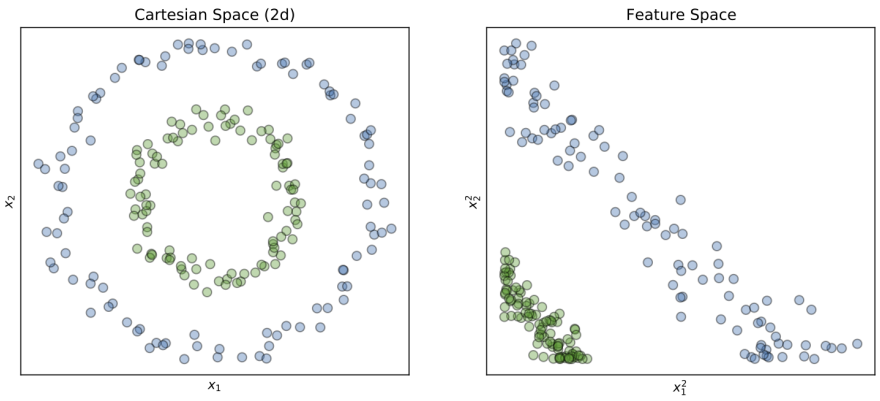


Figure 2.3: **Representations matter for shallow machine learning models such as Logistic Regression. A simple transformation, e.g. squaring the values of the raw features, may be enough to solve the problem. [30]**

2.2.1 Manual Feature Generation

Past works predominantly depended on manual feature generation for defect clusters and adopted a two-stage strategy of feature generation and classifier learning. These features include: (a) *geometrical features*, (b) *Radon projection features*, (c) *density features*, (d) *texture features*, and (e) *gray features* [21]. These handcrafted feature generation processes transform wafer maps into vector form, effectively reducing dimensionality. Then these vectors are fed into classification models for prediction. Several established machine learning algorithms can be employed in this step [16].

Wu et al. [38] utilized Support Vector Machines (SVMs) as classifiers due to their efficiency in large-scale data set applications. They developed a two-stage framework for WMFPR using rotation- and scale-invariant features. The first stage involves determining whether a wafer map exhibits a failure pattern, and the second stage involves identifying the pattern type. When tested on a dataset comprising 118,595 wafer maps, the method yielded a commendable accuracy of 94.63%. The proposed features, combined with the classifier and similarity ranking mechanism, were proved to be effective tools for large-scale data analysis, and are adopted by TSMC as an in-house tool. However, when focusing specifically on data that exhibited a defect pattern, the classification accuracy dipped to 83%. This discrepancy is largely attributed to the prevalence of the *None* pattern in the test data.

Piao et al. [28] proposed a decision tree ensemble learning scheme that achieved similar results using only radon transform based features compared to recent studice that used multi-types of features. The authors suggested that integrating multi-types of features could further enhance the discriminating capability of their method. Notably, while the approach effectively distinguished Center, Donut, Random, and None patterns in the WM-811K dataset, it faced challenges in accurately recognizing other patterns.

Saqlain et al. [29] introduced an ensemble technique that combines predictions

from four base classifiers: logistic regression, random forest, gradient enhancement, and artificial neural network. The methodology employs three distinct sets of features derived from wafer data, encompassing density-based, geometry-based, and radon-based features. Notably, their approach outperforms prior studice, primarily due to the integration of multi-type features and the comprehensive extraction from wafer data. The model attained a high level of accuracy, despite the presence of imbalanced and noisy data. Additionally, the time required for feature extraction is markedly reduced compared to previous techniques.

Clustering methods based on defect characteristics are also widely used, including density-based clustering [17] and K-means clustering [7].

While feature generation can reduce storage and computation demands, the classifier's performance is heavily dependent on the selection of features, which might not be sufficient to represent or distinguish patterns. The method also struggles to identify rare or novel defects due to their unfamiliar attributes. Moreover, selecting an appropriate classifier and fine-tuning its parameters becomes intricate, and the ensemble learning approach substantially elevates the model's complexity. Recent research has shifted from manual feature generation to feature representation learning as leveraging feature learning algorithms has proven to generate more meaningful and effective features for downstream tasks [21].

2.2.2 Representation Learning

Representation Learning is the automated feature extraction from raw data. The learned features, or representations, are often dense, compact, and can be generalized to similar data modalities. In other words, these representations can capture the essential patterns or attributes of the data, which could be easily transferred to other tasks and have been the principal method to solve problems in which data annotations are hard or even impossible to get [30].

Deep neural networks, such as the Convolutional Neural Network (CNN), are extensively utilized in various computer vision tasks to extract rich, high-level features. In a typical supervised CNN, as the raw input data passes through each layer, the features from preceding layers are refined. Initially, neurons in the early layers often detect basic features like edges and contours by analyzing high-frequency inputs. As the data advances through the network's hierarchy, subsequent layers integrate these basic detections to recognize more intricate object parts. A simple linear classifier can then be applied to map the inputs to a set of defined classes.

Nakazawa et al. [27] is the first to apply CNNs for wafer map defect pattern classification and image retrieval. They employed synthetic wafer maps with 22 defect classes, as the foundation for CNN training, validation, and testing. Their approach achieved an overall classification accuracy of 98.2% on a test dataset comprising 6600 synthetic wafer maps. To assess the generalizability of the trained model, they further evaluated its performance on 1191 real wafer maps. Their result shows that using purely synthetic data for network training can still result in high classification accuracy on real wafer maps. However, the model encountered challenges when classes had closely resembling defect patterns, leading to higher misclassification rates. Moreover, for multi-class classification scenarios, the most salient defect pattern is typically predicted,

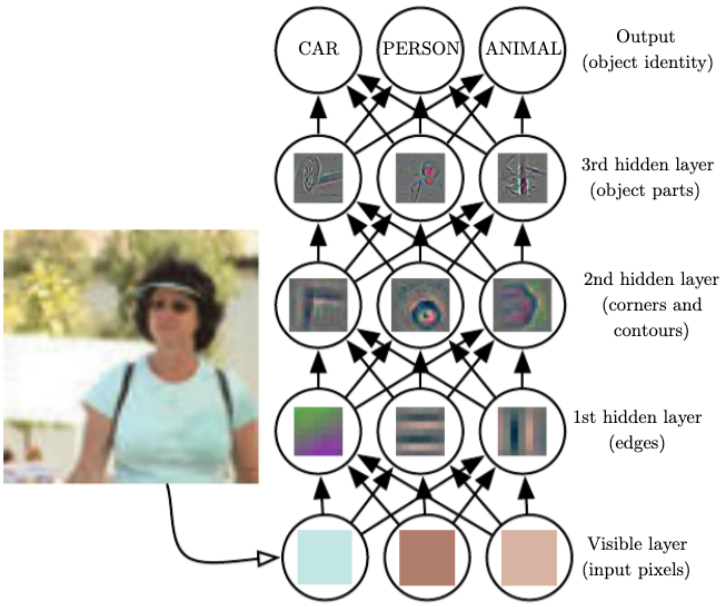


Figure 2.4: **Deep neural networks use a layer-by-layer iterative approach to create complex structures refining information from the preceding layer. [10]**

disregarding the other present defects.

Yu et al. [41] presented a two-fold approach using CNNs for wafer defect pattern analysis. First, they developed an 8-layer CNN model tailored to identify the presence of defects, and subsequently, a more intricate 13-layer model was introduced for specific defect pattern classification. This latter model yielded an average classification accuracy of 93.25%. They also studied the feature expression ability of different layers and different dimensions for retrieval and realized the root causes analysis of wafer defect pattern based on the middle layer features of the classification model and similarity ranking.

Wang et al. [35] presented a CNN-based method that uses polar mapping to transform circular wafer maps into matrices, effectively recognizing defect patterns and reducing variations in wafer sizes. Zheng et al.[44] proposed a deep CNN model for multi-class image classification of wafer map failures, outperforming other machine learning and deep learning models with fewer or much deeper layers. However, the 'Local' failure pattern type was recognized less accurately than other failure pattern types.

Representation learning can also be executed in an unsupervised manner. This approach involves automatically discerning and capturing representations or patterns in the input data without using labeled examples. Generative model such as AutoEncoders (AE) is a popular example. As illustrated in Figure 2.5 an AE consists of two main components: an encoder and a decoder. The encoder function compresses the input data into a more compact representation, while the decoder function reconstructs the original data from this compact representation.

Shon et al. [24] proposed an unsupervised pre-training method for the identi-

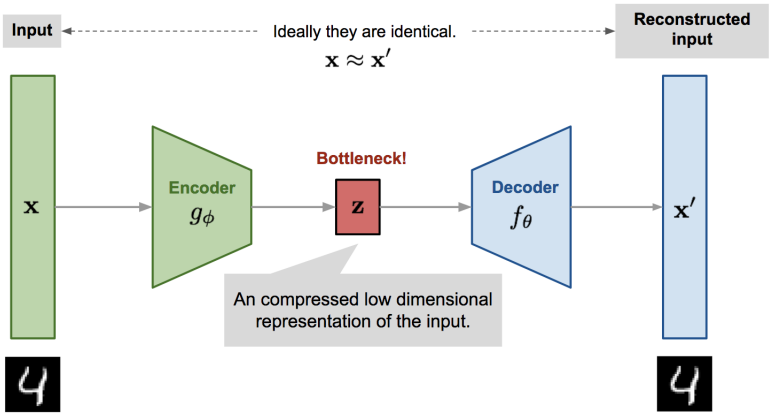


Figure 2.5: **Encoder-decoder architecture.**

fication of wafer map defect patterns. They used a stacked convolutional sparse denoising auto-encoder to learn discriminative features from wafer maps. This approach demonstrates the potential of unsupervised learning in feature extraction for wafer map defect pattern recognition.

2.2.3 Self-supervised Representation Learning

Supervised learning performs exceptionally well when given enough labeled data. However, obtaining a substantial amount of these labeled data can be both costly and challenging. Leveraging the abundance of unlabeled data, as opposed to the limited labeled datasets that are manually curated by humans, appears to be a more efficient approach. Despite the potential of unsupervised learning to leverage this unlabeled data, it often falls short in efficiency and effectiveness compared to its supervised counterpart. Therefore, Kong and Ni [22] proposed a framework combining a ladder network with a semi-supervised variational autoencoder for classifying wafer bin maps (WBMs). By combining the strengths of both supervised and unsupervised methods, their approach outperforms traditional CNN models, marking an improvement of over 5%. Additionally, the incorporation of active learning and pseudo-labeling techniques further accelerates the learning process.

In recent times, Self-Supervised Learning (SSL) has emerged as a driving force in unsupervised representation learning, particularly in domains like computer vision and NLP. The essence of SSL is harnessing the potential of unlabeled data by effectively generating labels from the data itself. A common workflow is to train a model on one or multiple pretext tasks with unlabelled images and then use one intermediate feature layer of this model to feed a multinomial logistic regression classifier on labeled data. The final classification accuracy quantifies how good the learned representation is. The design of pretext tasks stands paramount in SSL, ensuring the model learns representations with good semantic or structural meanings for downstream tasks.

Generative models can also be considered self-supervised models but with different objectives. For example in GANs, they are used to generate realistic images for the discriminator whereas the aim of self-supervised training is to

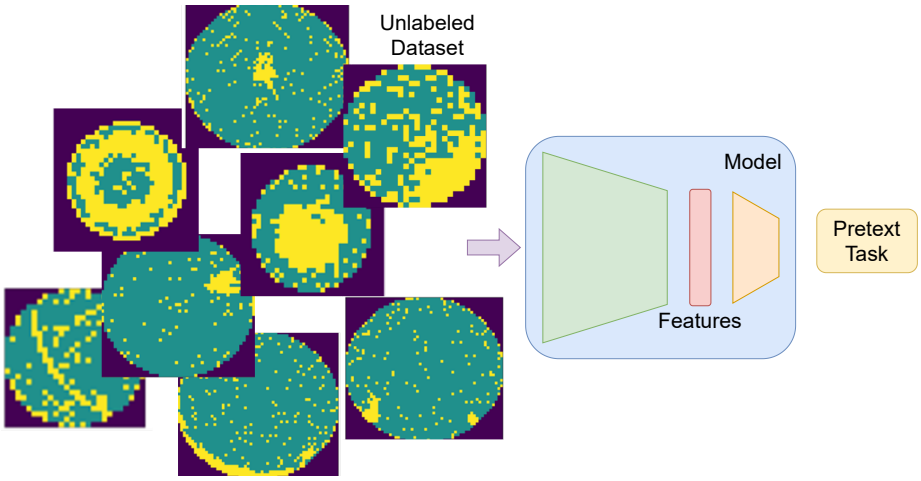


Figure 2.6: **Pretext Learning**

identify good features that can be used for a variety of tasks and not just to fool the discriminator.

Pre-training: Pretext task

The task we use for pre-training is known as the pretext task. The aim of the pretext task (also known as a supervised task) is to guide the model to learn intermediate representations of data [8]. However, we usually don't care about the final performance of this invented task. Rather we are interested in whether the learned intermediate representation is useful in understanding the underlying structural meaning that is beneficial for the downstream tasks.

For example in Figure 2.7, the image is rotated at a random degree and a model is trained to predict how each input image is rotated. The rotation prediction task is made up, so the actual accuracy is unimportant. But we expect the model to learn high-quality latent variables for real-world tasks, such as constructing an object recognition classifier with very few labeled samples [37].

Fine-tuning/Transfer Learning

Though the model acquires meaningful data representations during the pre-training phase, it often requires adjustments for specific downstream tasks. Such adaptations are typically achieved using fine-tuning or transfer learning approach.

Fine-tuning alters the entire pre-trained model using labeled data from the desired task. In other words, all layers of the pre-trained model are "unfrozen," permitting their alteration during the adaptation phase.

Conversely, transfer learning treats the pre-trained model as a static feature extractor. Here, features are extracted from the target task's labeled data using the pre-trained model. These features are then input into a new classifier, which is trained from scratch. Throughout this procedure, the pre-trained model

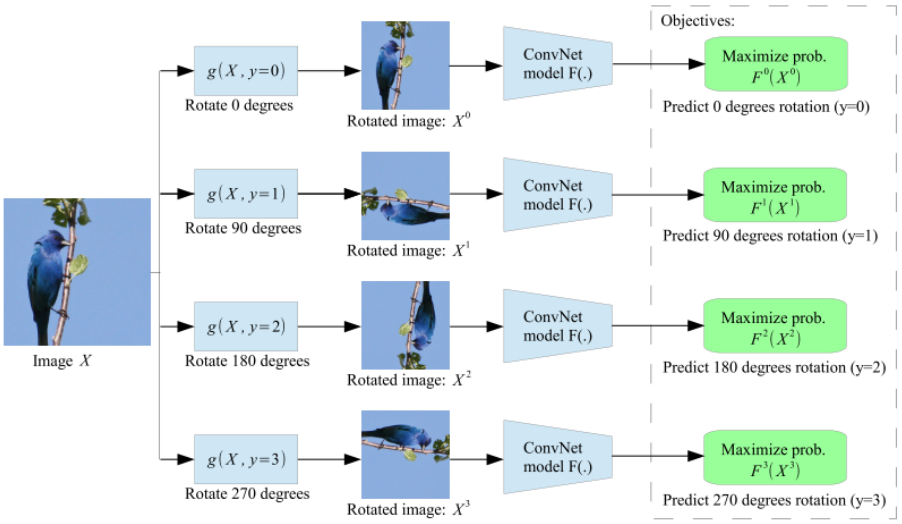


Figure 2.7: **Rotation prediction is one example of a prediction task that explores the characteristics of the data to devise supervised signals.**

remains unchanged or "frozen", with only the new model being updated. This methodology aligns with the linear evaluation process employed in this thesis.

2.3 Contrastive Learning

Recently, a surge of methods has been built on the concept of contrastive representation learning to compensate for the drawbacks induced by heuristically designed pretext tasks. The core idea behind these approaches is that different versions of an image obtainable through various data augmentations should map to a similar embedding, which distinguishes them from other contextually different images. By mapping representations of positive instances—derived from diverse augmentations of the same image—closer together and distancing them from negative instances or unrelated images, contrastive learning carves out an optimized embedding space.

Contrastive learning can be applied to both supervised and unsupervised settings. In the unsupervised domain, it stands as a formidable strategy in Self-Supervised Learning, rivaling the performance of state-of-the-art supervised deep learning frameworks. In the subsequent sections, we will delve into key components of contrastive learning, including training objectives, image augmentation, and popular architectures.

2.3.1 Contrastive Training Objectives

NT-Xent Loss

The Normalized Temperature-scaled Cross Entropy loss (NT-Xent loss), a.k.a. the "multi-class N-pair loss", is a type of loss function, used for metric learning and Self-Supervised Learning. Kihyuk Sohn first introduced it in his paper

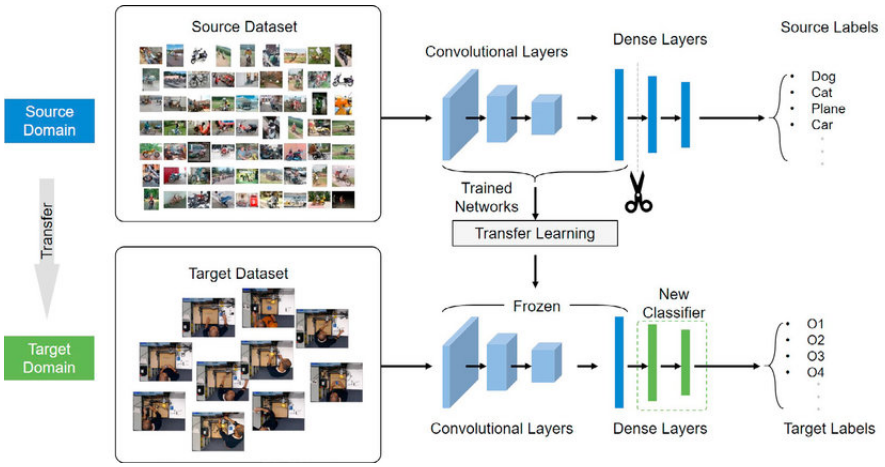


Figure 2.8: **Example architecture of a transfer learning model.**

"Improved Deep Metric Learning with Multi-class N-pair Loss Objective". It was later popularized by its appearance in the "SimCLR" paper by the more commonly used term "NT-Xent":

$$\mathcal{L}(i,j) = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}(z_i, z_k)/\tau)} \tag{2.1}$$

Cosine Similarity

$\text{sim}(z_i, z_j)$ gives us the (cosine) similarity between the vectors z_i and z_j. These vectors are usually the output of some neural network. To put it simply, the smaller the element-wise difference between the two vectors, the higher the resulting value.The expression for cosine similarity can be expressed by the equation below:

$$\text{sim}(\mathbf{A}, \mathbf{B}) = \frac{\mathbf{A} \cdot \mathbf{B}}{||\mathbf{A}||_2 \times ||\mathbf{B}||_2} \tag{2.2}$$

Another thing to note in the above equation is that, because we divide by the magnitudes of the two vectors, the cosine similarity can be considered as an L2 Normalization. Empirically, this, along with the temperature (τ), has been shown by the SimCLR paper to lead to significant improvements in contrastive accuracy.

Temperature

The similarity scores are divided and normalized by a constant parameter denoted as τ, commonly referred to as the temperature. This parameter regulates the scale of the similarity scores between feature projections. Specifically, τ impacts the distribution's sharpness of these scores. It modulates the relative weighting between positive and negative pairs during optimization. When τ is significantly less than 1, the term $\exp(\text{sim}(z_i, z_j)/\tau)$ shows a more pronounced difference between similar and dissimilar pairs, causing the distribution to be sharper. For values larger than 1, the distribution is more generalized.

2.3.2 Image Augmentation

The role of data augmentations in Contrastive Learning is crucial, as they directly influence the quality of the learned representations. Figure 2.9 shows some of the prevalent augmentation techniques leveraged in the domain including:

- *Random Cropping and Resizing*: This technique involves taking random sections of the input image and resizing them. This can help the model become invariant to the positioning and scale of the features in the image.
- *Flipping*: Images can be horizontally or vertically flipped, teaching the model not to over-rely on the orientation of objects within them.
- *Color Jittering*: Altering the brightness, contrast, saturation, and hue of images can make the model more resilient to variations in lighting and color distributions.
- *Rotation*: Rotating the image by various angles ensures the model recognizes objects regardless of their orientation.
- *Cutout*: Randomly removing parts of the image can force the model to learn from incomplete data, enhancing its ability to infer from partial information.
- *Gaussian Noise*: Random noise can be added to images, training the model to focus on essential features and disregard the insignificant perturbations.
- *Blurring*: Applying blur filters, like Gaussian blur, helps the model to generalize better by reducing high-frequency noise in the data.
- *Sobel Filtering*: Employing convolution with Sobel kernels to compute the gradient magnitude of the image at each pixel. The result emphasizes edges and transitions, which helps the model concentrate on structural features and contours, thereby enhancing its capability to recognize shapes and boundaries.
- *Geometric Transformations*: Techniques such as affine transformations can distort the geometry of the image, ensuring the model remains invariant to geometric alterations.

Chen et al. [5] observed that the choice of augmentations significantly affects the performance of SimCLR. In particular, they found that using a combination of augmentations, such as cropping with random resizing, flipping, color jittering, and blurring, was more effective than using a single augmentation. This is because a diverse set of augmentations allows the model to learn more robust and generalizable features that are useful for a wide range of downstream tasks.

Tian et al. [31] have investigated the effect of augmentations on contrastive learning performance and argued that a good set of augmentations should reduce mutual information between views while keeping task-relevant information intact. In other words, the augmentations should introduce sufficient variation between the views of the same image to challenge the model, but not to the extent that the underlying semantic content is altered or lost. This balance

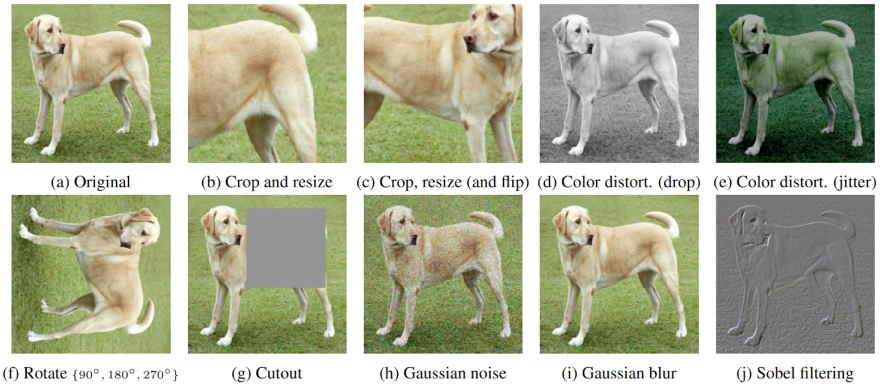


Figure 2.9: **Different data augmentations applied to an image of a dog [5].**

enables the model to focus on learning features that are relevant to the task at hand and less sensitive to irrelevant variations introduced by the augmentations.

Jing et al. [18] have found that augmentations that are too strong will result in a dimensional collapse of the feature representations. This means that, when the augmentations are overly aggressive, the learned feature representations may become overly simplified, losing their discriminative power and collapsing into a low-dimensional subspace. This limits the model's ability to distinguish between different images and may lead to poor performance on downstream tasks.

Therefore, it is important to carefully select and tune a group of semantic-preserving augmentations to achieve the optimal trade-off between encouraging the model to learn robust features and preserving the discriminative power of the learned representations.

Properly designing data transformations demands domain-specific knowledge and empirical analysis. In fact, the requirements for augmentations tailored for wafer maps significantly diverge from those designed for everyday objects, such as images from the CIFAR-10 dataset. In the CIFAR-10 dataset, images typically feature a single object centrally located and occupying a substantial portion of the image. Consequently, augmentations like cropping and resizing can isolate a relatively small part of the image while retaining the semantic feature. However, this is not the case for wafer map data. Given the defect region in wafer maps, such augmentations risk eliminating the defect region during cropping, thereby altering the semantic feature significantly from the original image, and rendering it impossible for the model to learn effectively.

As suggested by [14], traits that make a good augmentation to be considered for the representation learning task are as follows:

1. Suitability for the task: The transformations under consideration should be suitable for the data under analysis. If the transformation cannot be applied to the data sample space X, it should not be considered.

2. Capability to maintain similarity among samples: This is the most critical characteristic for a good transformation. The transformation should maintain the major sample property unchanged after transformation. In our application, wafer map patterns should be kept the same, while the

detailed good/bad die distribution in a wafer map can be varied by the transformations.

3. Flexibility to be randomized: Finally, the transformations should be easy to be randomized, adding more flexibility when transforming the details of samples, so that the similarity among samples can be fully captured by the transformations.

2.3.3 Popular Architecture

In recent years, several contrastive learning frameworks have gained prominence in deep learning due to their effectiveness in learning powerful representations. Generally, a positive pair comprises two different views of an instance, whereas a negative pair comprises the views of two different instances [5, 25]. Chen et al. [5] proposed SimCLR to train a CNN for visual-representation learning. Chen et al. [6] improved SimCLR using a larger CNN architecture and knowledge distillation. Grill et al. [11] presented Bootstrap your own latent (BYOL), which trains two CNNs that interact and learn from each other without using negative pairs. Misra and Maaten [25] proposed pre-text invariant representation learning (PIRL) that uses a memory bank of negative instances to define negative pairs. He et al. [12] proposed momentum contrast (MoCo), which uses a dictionary of negative instances that are dynamically updated with momentum values to define negative pairs. Hu et al. [14] and Kahng and Kim [19] adapted the data augmentation strategy to pre-train a CNN for wafer map defect pattern classification.

Chapter 3

Methodology

This chapter details the training and fine-tuning process for Self-Supervised Contrastive Learning (SSCL) within the domain of WMFPR. The chapter is structured as follows:

- Section 3.1 offers an overview of the key components of the training algorithm and evaluation metrics applied.
- Section 3.2 delves into the intricacies of the training process, presenting a detailed review of the steps taken and the configurations utilized in our experimental setup.

3.1 Overview

3.1.1 Training Algorithm

As illustrated in Figure 3.1, our Self-Supervised representation learning framework for the classification of defect patterns in wafer map data is based on SimCLR [5], which consists of four key components:

1. *Random Data Augmentation Pipeline*(T): This pipeline transforms any given data example randomly, generating two correlated views of the same sample denoted as $\widetilde{x}_i$ and $\widetilde{x}_j$, considered as a positive pair. In this study, we employed six augmentations: resized crop, cut-out, die noise, rotation, horizontal flipping, and rotation.
2. *Base Encoder* $f(\cdot)$: This component extracts feature representations from the augmented data samples. Given the limited semantic information in wafer map data, we employed both ResNet-18 and a Residual-block-based CNN implemented in the CAE model to obtain $h_i = f(\widetilde{x}_i)$, where h_i represents a 512-dimensional vector derived from the ResNet output.
3. *Projection Head* $g(\cdot)$: Responsible for mapping features into a lower-dimensional representation space and computing contrastive loss based on these representations. We used two linear layers with one batch normalization layer followed by one ReLU activation layer in between. The mapping from h_i to z_i is defined as $z_i = g(h_i)$.

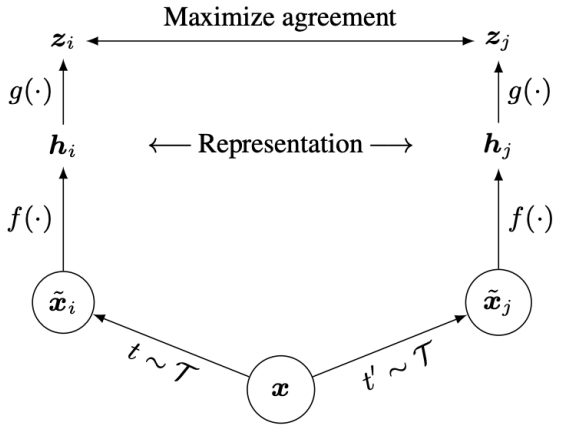


Figure 3.1: **SimCLR architecture designed by Chen et al. [5]. The input image x is augmented two separate ways by transformations randomly drawn from T, resulting in an augmented image pair x_i, x_j. A base encoder network $f(\cdot)$ and a projection head $g(\cdot)$ are trained to maximize agreement between feature projections z_i, z_j. After training, $g(\cdot)$ is discarded, and only $f(\cdot)$ is used to compute representations h for downstream tasks.**

4. *Contrastive Loss function*: We adopted the NT-Xent loss, as introduced in the original SimCLR paper, to compute the contrastive loss, as described in Section 2.3.1

3.1.2 Evaluation Metrics

After the above steps, we conduct an evaluation of the pre-trained base encoder using a linear evaluation protocol, as shown in Phase 2 of Figure 3.2. This evaluation aimed to assess the quality of the feature representations acquired by the pre-trained encoders. We achieve this by training a linear classifier attached to the base encoders to perform a supervised classification task on a labeled public dataset. Additionally, we perform image retrieval tasks based on cosine similarity to gauge the model's capability in grouping similar wafer maps with the public dataset and Nexperia's private dataset. Lastly, we visualize the feature space by applying t-SNE to map the feature dimensions to a 2-dimensional feature space, providing an overview of the distribution of all patterns.

To evaluate the classification performance, we calculate $Micro\text{-}F_1$ and $Marco\text{-}F_1$ scores on the test set. F1 score is the harmonic mean of precision and recall. The terms *Micro-* and *Macro-* indicate how metrics are averaged across multiple categories:

$$Micro\text{-}P = \frac{\sum_j TP_j}{\sum_j TP_j + \sum_j FP_j} \tag{3.1}$$

$$Micro\text{-}R = \frac{\sum_j TP_j}{\sum_j TP_j + \sum_j FN_j} \tag{3.2}$$

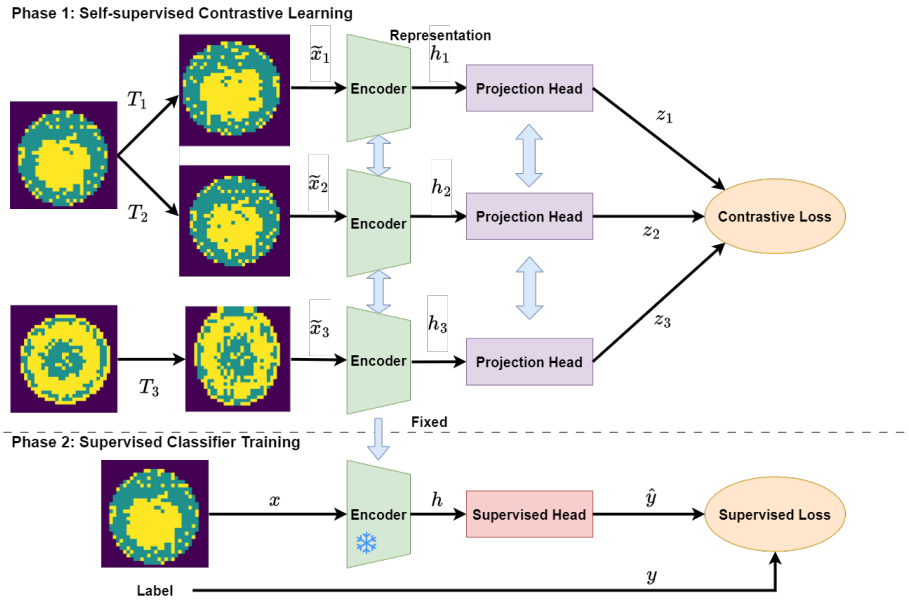


Figure 3.2: **Overview of Self-Supervised Contrastive Learning architecture.**

$$Micro\text{-}F_1 = \frac{2 \cdot Micro\text{-}P \cdot Micro\text{-}R}{Micro\text{-}P + Micro\text{-}R} \tag{3.3}$$

where TP_j, FP_j, FN_j, P_j, and R_j denote the true positive, false positive, false negative, precision, and recall, respectively, for the jth category.

$Marco\text{-}F_1$ is calculated by averaging the category-wise performances as below:

$$P_j = \frac{TP_j}{TP_j + FP_j} \tag{3.4}$$

$$R_j = \frac{TP_j}{TP_j + FN_j} \tag{3.5}$$

$$Macro\text{-}F_1 = \frac{1}{C} \sum_j \frac{2 \cdot P_j \cdot R_j}{P_j + R_j} \tag{3.6}$$

$Micro\text{-}F_1$ is heavily influenced by the majority categories, whereas $Marco\text{-}F_1$ allows each category to contribute equally to the overall score.

3.2 Implementation Details

3.2.1 Data Preprocessing

We employed two methods of image preprocessing: one-hot encoding and resizing. Additionally, we conducted a comparison of the impact of popular filters, which are commonly utilized in prior research:

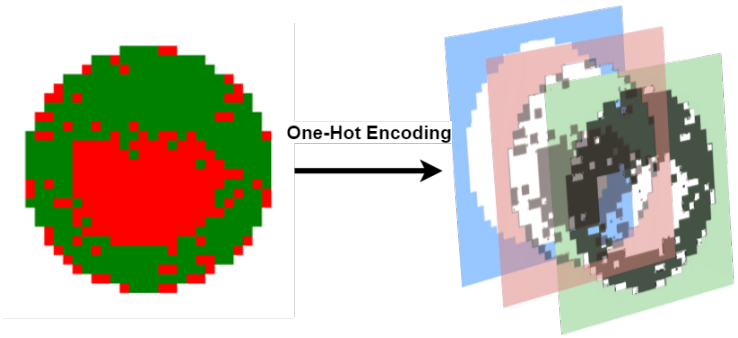


Figure 3.3: **One-Hot Encoding transformation of the original single-channel wafer map into three channels. On the left is the original wafer map, while on the right is the one-hot encoded version. The blue channel corresponds to the wafer's background, labeled as '1', with all other positions labeled as '0'. The red channel highlights defective dice with '1' labels, whereas all other positions are '0'. Conversely, the green channel labels the non-defective dice as '1' and defective dice as '0'.**

One-Hot Encoding

The input single-channel wafer map is transformed using one-hot encoding to produce an RGB-like, three-channel representation. In the original wafer map, values of '0' indicate void spaces, '1' represents normal dice, and '2' signifies defective dice. In the one-hot-encoded format, each channel consists exclusively of 0s and 1s, where each channel distinctly represents one of the three aforementioned test results, as depicted in Figure 3.3. This encoding separates the three types of data (void, normal die, defective die) into distinct channels, which can make it easier for the network to learn features that distinguish these classes. Increasing the channel count also facilitates the extraction of morphological features from the one-hot encoded image by employing a 3D filter in the convolutional layer.

Resizing

To ensure consistent input dimensions for downstream models, the original Wafer-Based Maps (WBMs) undergo resizing using nearest-neighbor interpolation. In our experiments, we employed wafer maps with dimensions of both (32,32) and (64,64), a choice influenced by the prevalent distribution of wafer sizes. Figure 3.4 provides insight into the distribution of the number of dice in the WM-811K Dataset.

Considering that a significant portion of the test data approximates the shape of (32,32), and the majority of data falls below (64,64), these dimensions effectively preserve information across the majority of wafer maps. While there is potential for performance improvement by further increasing the dimensionality of wafer maps, as suggested by Kahng et al. [19], who propose using (96,96) for reshaping, as smaller sizes have been observed to lead to a degradation in classification performance, and larger sizes offer only marginal improvement.

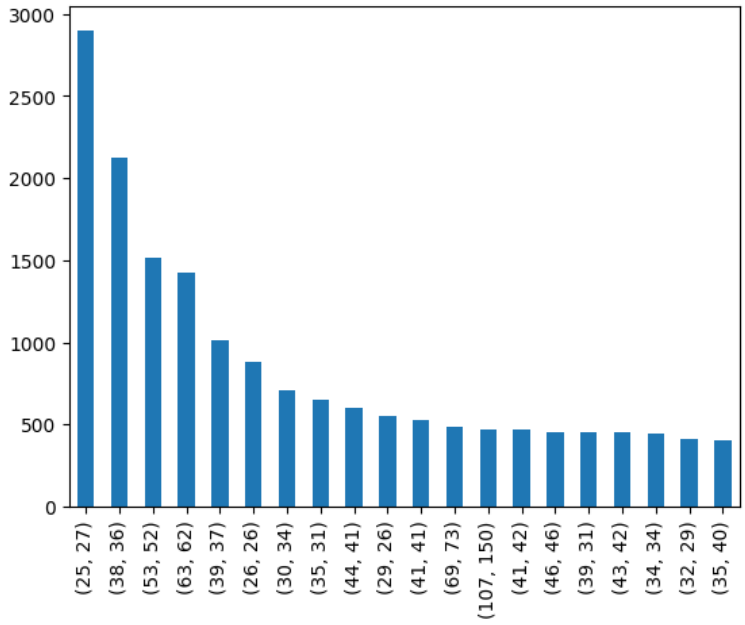


Figure 3.4: **Distribution of the top-10 most frequently seen wafer map shape in WM-811K dataset.**

However, it's important to note that increasing the size of the wafer map also escalates computational complexity. Thus, we believe that (64,64) strikes a practical and balanced compromise.

Through our experiment, we found that the shape of the wafer map has a significant impact on the model's performance. As shown in Figure 3.5, increasing the wafer map size improved the classifier's performance by 7% in macro-F1 score. This gap is narrowed with more training data to about 1% when 1600 training data samples are available.

Filtering

Another widely adopted preprocessing method in WMFPR is filtering. As elaborated in Section 2.1.2, one of our primary objectives is to direct the model's attention toward systematic defects while ignoring random die failures. Consequently, numerous studice have applied noise-removing filters to raw data before sending it for model learning. Common choices encompass the Median Filter with kernel sizes of 2×2 and 3×3, the connected-path filter introduced by Kim et al. [20], and density-based methods such as DBSCAN [4].

Notably, the simplest and most commonly used method is the Median Filter. However, its effectiveness can be influenced by the substantial variability in the size of wafer maps. As illustrated in Figure 3.6 and 3.7, the 2×2 filter struggles to completely eliminate noise, whereas the 3×3 filter demonstrates better performance while removing smaller defect patterns like scratches. An alternative approach proposed by Xu Q et al. [39] involves the use of a constrained mean filter (C-mean filtering) as a preprocessing step across the entire

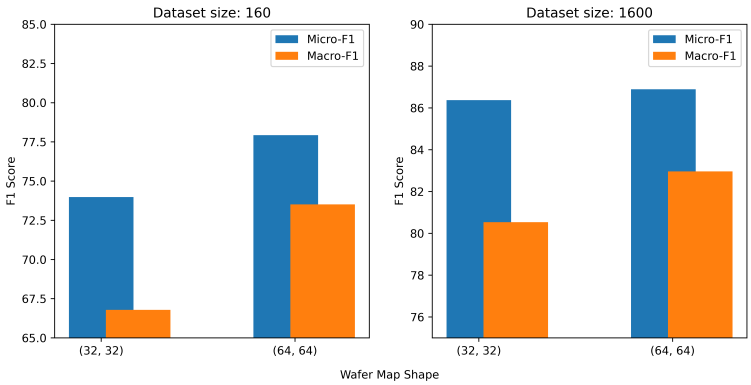


Figure 3.5: **Effect of wafer map shape on classification results with different amounts of training data.**

dataset. This preprocessing step, coupled with a vision transformer, has yielded an exceptionally high overall classification accuracy of 95.46%.

The C-mean filter was initially applied to WMFPR by Yu N et al. [40], unlike other filters, it focuses only on the neighborhood of defective dice. In this method, the mean value of pixels within the filter window is computed. If the mean value falls below a preset threshold, the target defective die is converted into a normal die; otherwise, it remains unchanged. Figure 3.8 illustrates results obtained using a 3×3 filtering window and a mean threshold of 1.25, through our experiment, we found that kernel size of 3×3 is better at preserving the shape of the defect area than 5×5 kernel.

However, our experimentation has revealed a drawback of the C-mean filter. It significantly reduces the identification of defective dice at the edge of the wafer due to the presence of background areas labeled with 0 in the neighboring region, which lowers the mean value. Since the edge region plays a crucial role in determining the wafer's defect pattern, we have enhanced the C-mean filter to preserve these patterns. This enhancement involves patching the 0-labeled locations with 1 during mean calculation, resulting in improved filtering capabilities. The outcome of this enhancement is depicted in Figure 3.9, showcasing the filter's ability to effectively eliminate irrelevant random noise while preserving the shape of the defect region. Although the C-mean filtering was not ultimately included in our model, we firmly believe it retains its value as a useful tool for Preprocessing.

In our experiments, we explored a partial augmentation setting where one of the positive samples underwent a set of augmentations, including the addition of random die noise, while the other positive sample, instead of adding random die noise, was subjected to the C-mean filter. However, our results demonstrated that this partial setting did not yield improvements in the classification outcomes. Additionally, as discussed in Section 4.3.2, our model possesses the capability to ignore random noise, making it unnecessary to apply any filtering before training, as such filtering could potentially alter the feature representation.

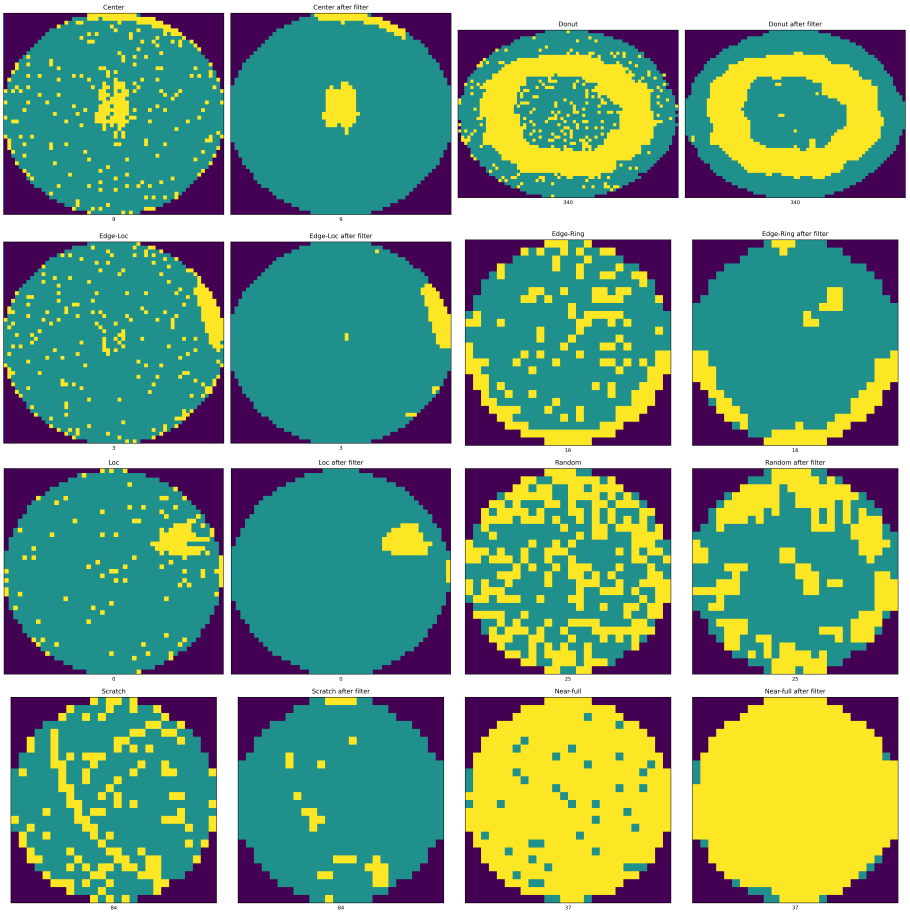


Figure 3.6: **Filtering results obtained using a 2 × 2 Median Filter. While the filtering does reduce noise to a certain extent, it also introduces alterations in the shape of the defect area. This alteration is of particular concern because it has the potential to impact the model's performance, especially when dealing with smaller wafer patterns.**

3.2.2 Unsupervised Pre-training

The SimCLR pre-training procedure is depicted in Figure 3.10. Within the same batch, wafer maps with defect patterns like Center and Donut each undergo two distinct random augmentations before being fed into the encoder, typically a CNN architecture such as ResNet18. Subsequent to this, the extracted feature representations are processed through a projection head, which is an MLP. The objective is to ensure that representations of the same pattern (e.g., two instances of the Center or Donut patterns) are brought closer together, while representations from different patterns are driven apart. It's noteworthy that during training, all modules denoted as CNN and MLP utilize shared weights. Post training, the intermediate feature representations produced by the encoder CNN can be employed as input for various networks tailored for downstream tasks.

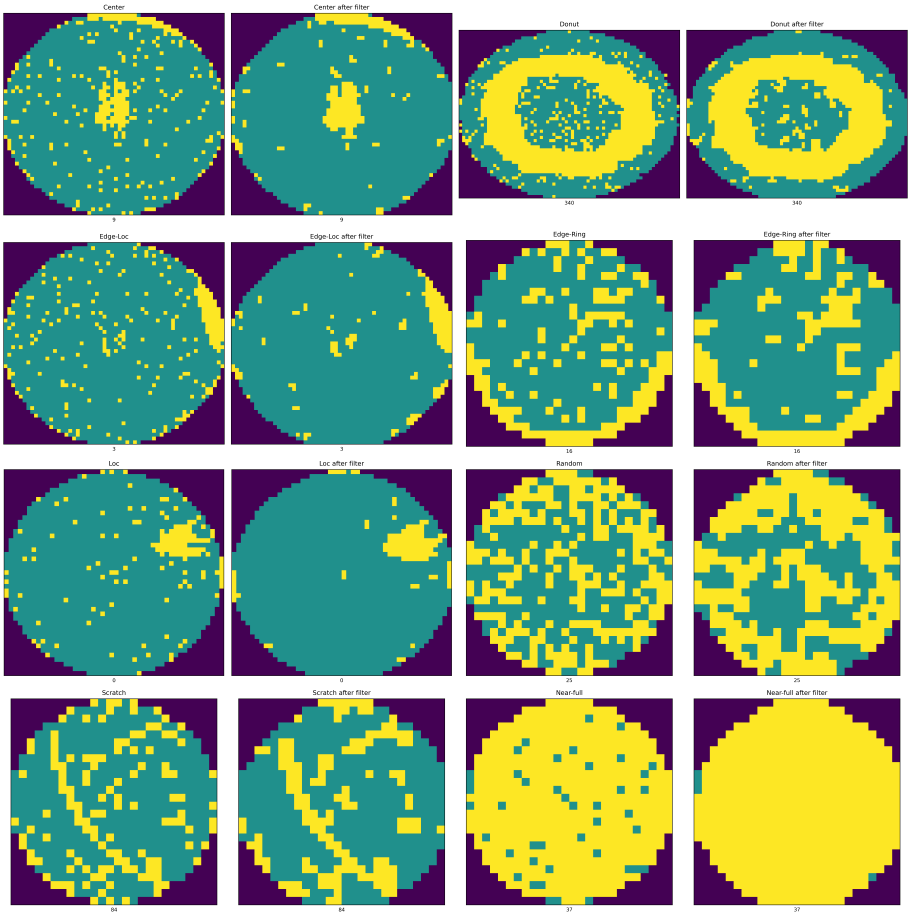


Figure 3.7: **Filtering results using of a 3 × 3 Median Filter. This filtering approach proves highly effective in noise reduction, particularly benefiting large round patterns such as Center and Donut. However, it is crucial to note that the filter exerts a more substantial impact on the shape of defect patterns characterized by thin borders. A striking example of this effect is observed in the seventh image pair, where the scratch pattern is entirely eliminated.**

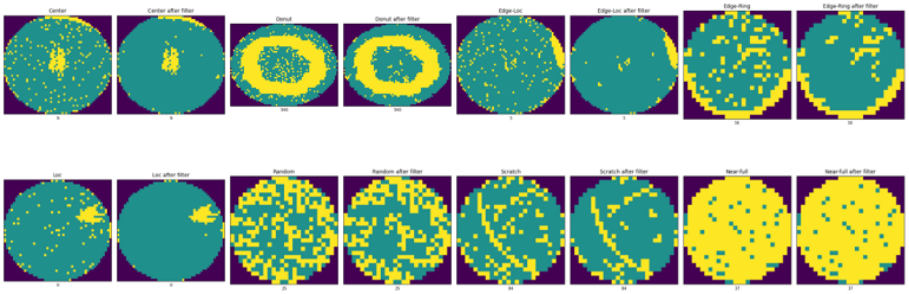
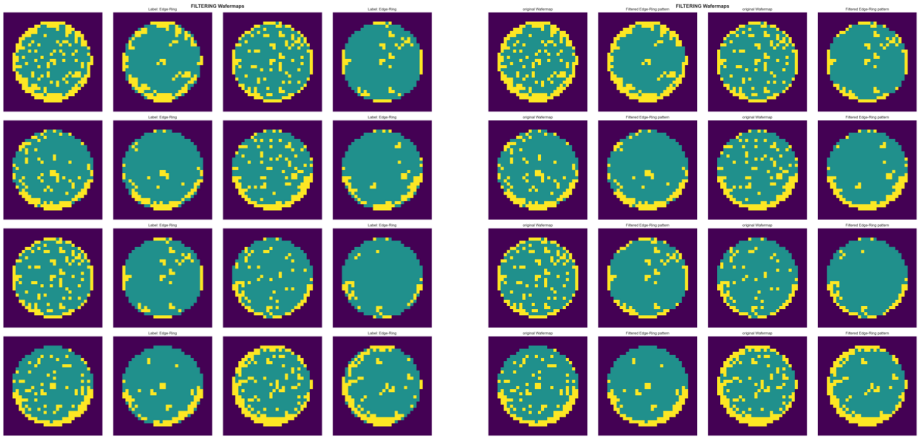


Figure 3.8: **Filtering results of the original C-mean filtering for wafer maps.**



(I) **Original C-mean filtering** (II) **Improved C-mean filtering**

Figure 3.9: **Comparison results obtained from wafer maps after applying the original and improved C-mean filtering. The samples are all Edge-Ring pattern wafer maps. It is evident that the edge pattern was well preserved with the improved C-mean filter.**

Image Augmentation Pipeline

As discussed in Section 3.1 (or appropriately formatted reference) on background augmentation, augmentation plays a pivotal role in Contrastive Learning. Notably, within the realm of research pertaining to WMFPR, there is an absence of comprehensive studice that systematically investigate and analyze various augmentation configurations. Ideally, an exhaustive examination of all potential augmentations and their combinations would be conducted. However, compared to the expansive hyperparameter space encompassing various augmentations, levels, and combinations, the limited time and computational resources forced us to conduct only a few exploratory experiments. Nevertheless, despite these limitations, the results obtained from these experiments proved immensely valuable. They provided significant insights that played a pivotal role in guiding the selection of optimal augmentation strategies for the domain of WMFPR.

In these experiments, our objective was to gain insights into the most effective augmentation combinations for WMFPR. Five augmentations were carefully selected and tested, The visualization of applying these augmentations to wafer maps is depicted in Figure 3.11.

(a) *Horizontal Flipping*: Horizontal Flipping is a widely recognized technique, recommended by influential works such as SimCLR [5]. Experiments conducted on general image recognition tasks, such as ImageNet, have demonstrated the effectiveness of this augmentation approach.

(b) *Resized Cropping*: Resized cropping is another augmentation technique recommended by SimCLR and has been identified as one of the most effective augmentations. Its effectiveness has also been confirmed in other studice focusing on WMFPR tasks [14, 19, 23]. However, it's worth noting that there is no universally recognized ratio for the cropped region, and

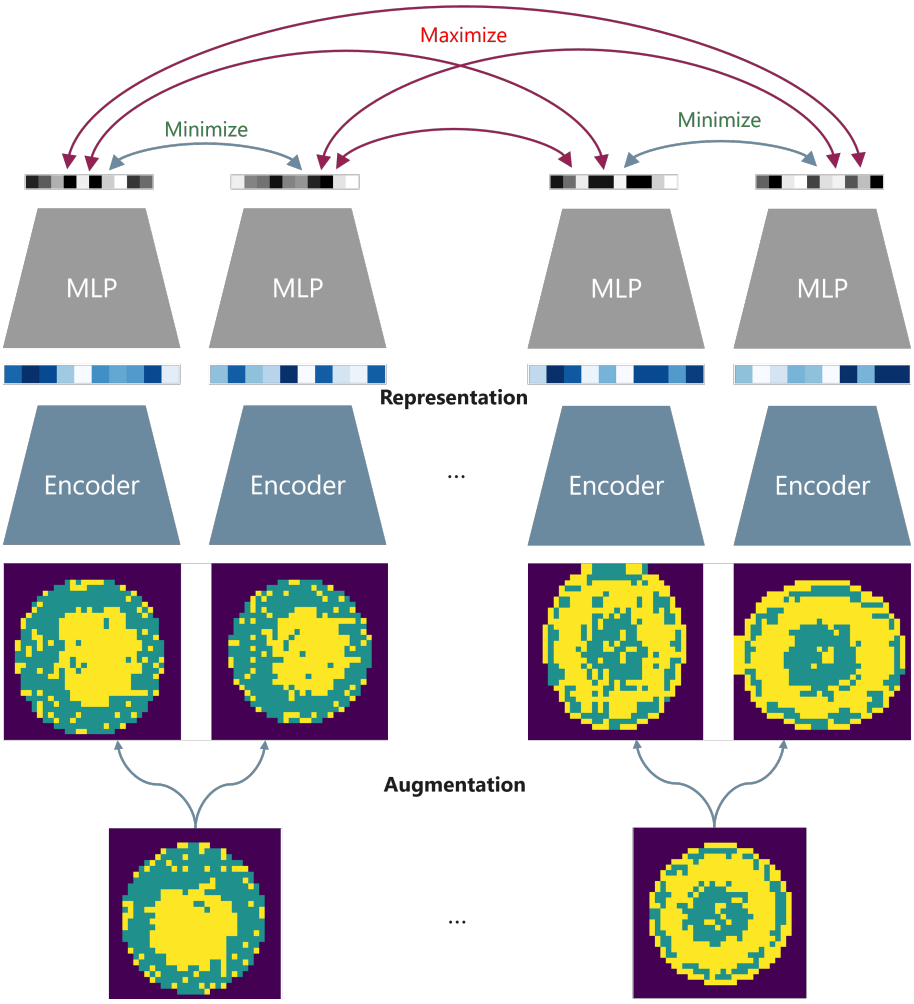


Figure 3.10: **Pre-training stage architecture.**

this hyperparameter can significantly impact the augmentation's results. If the ratio is set too low, there is a risk of cropping out the defect pattern and therefore losing its semantic feature. In our study, we have chosen a ratio range of [0.8, 1.0]. This choice aims to strike a balance by preserving the defect patterns especially edge patterns while introducing a certain level of perturbation to the wafer map, aligning with the specific requirements of WMFPR. We also conducted experiments with a broader ratio range of [0.6, 1.0]. However, the result indicated that the model encountered difficulties in achieving good performance on the contrastive prediction task (pretext task), and the classification results were also negatively affected. This outcome suggests that a broader cropping ratio range might introduce excessive variations that hinder the model's ability to learn meaningful feature representations.

(c) *Random die noise*: Adding Gaussian noise to images as a way of trans-

formation was shown to be ineffective for traditional image recognition tasks [5]. However, in the context of WMFPR, where it is crucial for the base encoder model to distinguish between defect areas and random die noise, adding random die noise emerges as a suitable domain-specific augmentation. It introduces a controlled degree of perturbation to the wafer map pattern while preserving its essential features. Specifically, the augmentation involves the random flipping of good and bad dice at positions where a valid die ($x_{ij} \neq 0$) is located. The transformation can be expressed as Equation 3.7, where p_n is set to 0.05 in our study:

$$x_{ij} = \begin{cases} 3 - x_{ij} & \text{with a probability of } p_n \\ x_{ij} & \text{with a probability of } 1 - p_n \end{cases} \tag{3.7}$$

We also evaluated the model's performance with a lower level of random cropping and die noise. We set p_n to 0.03 and the resized cropping ratio between [0.9, 1.0]. While these adjustments led to a significant improvement in the model's training performance on the pretext task, reaching a 91.34% top-5 accuracy in only 40 epochs, the linear evaluation result actually indicated a degradation in performance. This suggests that the augmentation strategy in this setting may have been too simple, failing to introduce sufficient variations to challenge the model to learn more essential feature representations.

(d) *Rotating*: Rotation augmentation stands out as a fundamental transformation in the domain of WMFPR tasks, primarily because WMFPR tasks can be regarded as rotation-invariant. Therefore, altering the orientation of a wafer map should not change its underlying semantic features. The study conducted by Hu et al. [14] compared the impact of different rotation strategies on wafer maps, including continuous rotations covering a range of angles from [0° to 360°] and discrete rotations where the wafer map is randomly rotated by angles from the set [0°, 90°, 180°, 270°]. Notably, their findings indicated that discrete rotations tend to yield superior results compared to continuous ones. Building upon these insights, our study adopts a strategy of rotating the original wafer map by angles within the discrete set [0°, 90°, 180°, 270°] to obtain WBMs of different orientations.

(e) *Cut-out*: Cut-out is an augmentation technique introduced in the YOLOv4 paper [3], which involves randomly covering a region of an input image by replacing its values with 0. The downside is that for smaller defect patterns, it runs the risk of entirely removing them. Therefore, it's important to maintain a relatively small size for the cut-out region compared to most defect areas. Another consideration is that during the manufacturing process, wafers may contain alignment markers designed to precisely position wafers. These markers can leave square or rectangular-shaped untested regions on the wafer maps. To ensure that the model is tolerant to variations in the size and location of these alignment markers, we introduced a maximum of two random cut-outs with scales of [0.001, 0.02] and [0.002, 0.04] relative to the size of the wafer map. Typically, the cropped region is square in shape, but we found that longer rectangular cut-outs are less

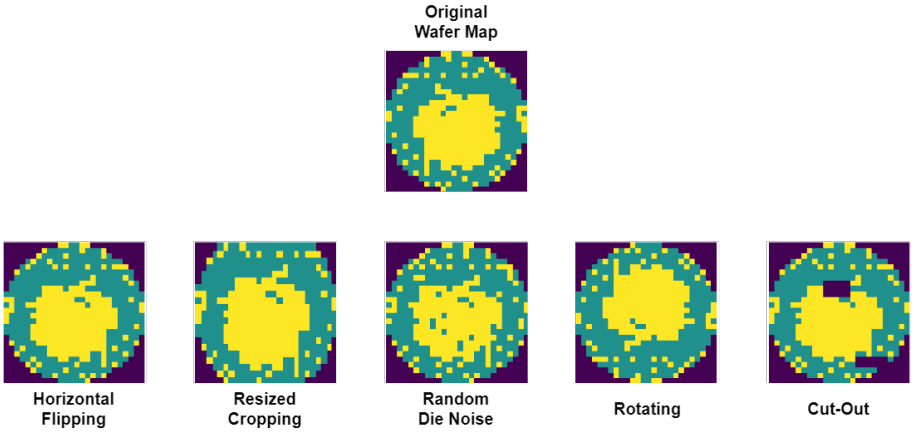


Figure 3.11: **Adopted Augmentations for wafer map pattern recognition.**

likely to completely remove the defect area. To encourage the model to learn from different parts of the image and become more robust to various defect sizes and alignment marker variations, we set the ratio of the cut-out region to be within the range of [0.1, 10].

Base Encoder

In the SimCLR contrastive learning paradigm, it has been suggested by Chen et al. [6], that larger models can yield substantial improvements even with fewer labeled examples. However, it's essential to consider that in the context of WMFPR, wafer maps inherently contain significantly less semantic information compared to general object data found in ImageNet. Therefore, we assert that utilizing an encoder model with a complexity similar to ResNet-18 should suffice as our Base Encoder.

To adapt the model to smaller wafer map data, we modified the first convolutional layer with a kernel size of (3, 3), stride, and padding set to 1. This adjustment allowed the model to better accommodate the smaller wafer map dimensions, and we configured the output dimension to be 512.

Projection Head Architecture

The intermediate feature representations, extracted by the base encoder, are further processed through a non-linear MLP known as the projection head. Our approach aligns with the setting outlined in SimCLRv2 [6], where the authors recommend employing three fully connected layers but discarding the last two layers for fine-tuning. However, my interpretation leans towards considering the first fully connected layer as part of the encoder. In this configuration, the projection head consists of two fully connected layers, as depicted in Figure 3.12. The first layer maintains the same dimensionality as the output layer of the base encoder, while the second layer reduces it to one-fourth of the original dimensionality. A batch normalization layer follows the first fully connected layer, and a ReLu activation layer is applied. The purpose of the projection

head is to project the representations into a lower-dimensional space. After the training phase, the projection head is discarded, and only the pre-trained encoder is retained for use as a feature extractor in downstream tasks.

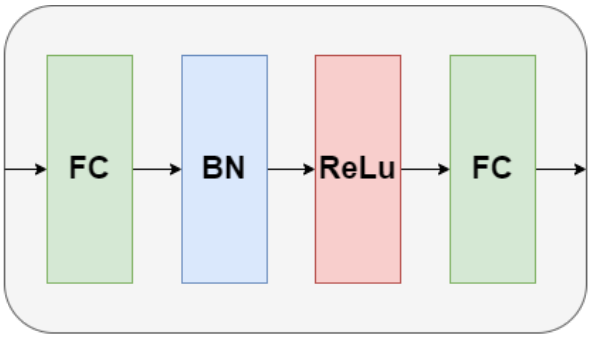


Figure 3.12: **Projection Head Architecture.**

A noteworthy discovery concerning the projection head is that wider projection heads tend to result in improved performance on both the pretext task and the classification task. Specifically, a broader projection head achieved a lower validation loss in the pretext task. In our experiments, we examined two projection head dimensionality configurations: 256-dimensional and 512-dimensional. The evaluation results are displayed in Figure 3.13.

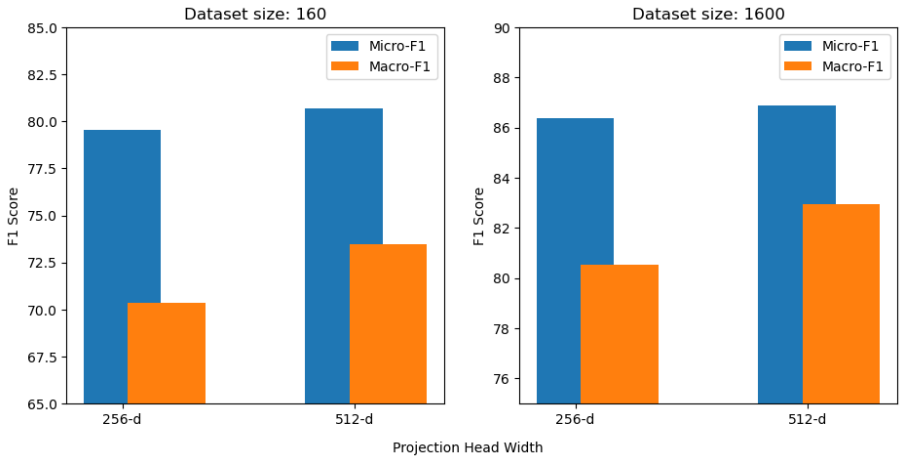


Figure 3.13: **Effect of projection head width on classification results with different amounts of training data.**

3.2.3 Supervised Fine tuning

The feature representations produced by the pre-trained encoder serve as input for the supervised linear classifier. A pivotal metric for the encoder's efficacy is the performance of this classifier. Given the linear classifier's inherent limited discriminative power, its classification accuracy heavily hinges on the quality

of the feature representations. Hence, the classifier's performance indirectly signifies the quality of the pre-trained encoder's outputs. This methodology draws inspiration from the works of Zhang et al. [42, 43].

Dataset for fine-tuning

Regarding the dataset, our primary goal is to train the linear classifier using a relatively small and balanced dataset. We are particularly interested in detecting various defect patterns, rather than determining the presence of a pattern. The latter task can be accomplished more easily using simple statistical methods like MLH.

To achieve our objective, a balanced dataset of 4000 data points was randomly sampled from the 8 labeled data, with 500 wafer maps in each pattern, for the evaluation using a small dataset. Additionally, we implemented a weighted random sampler to obtain a larger balanced dataset. This sampler extracted data exclusively from the initial 80% of the training dataset that contains defect patterns (20416 samples in total). Subsequently, we reserved the remaining 20% for testing and evaluation (5103 samples). There are several reasons for this choice:

(i) Utilizing a pre-trained encoder offers the capability to efficiently categorize wafer maps with distinct defect patterns based on their feature representations. By identifying representative defect patterns within dense clusters of samples in the feature space, wafer maps can be grouped by similar characteristics. Consequently, we have the ability to curate a customized and balanced training dataset by selecting and augmenting samples from within the same category. This approach allows us to tailor our dataset to meet our specific requirements effectively.

(ii) Practical considerations come into play. We operate with limited computational resources, and conducting extensive training and evaluation with a larger dataset would demand a substantial amount of time. By utilizing a maximum of 40,000 data points, we can train multiple linear classifiers within minutes, expediting the evaluation iteration process.

(iii) The results indicate that our model has achieved classification performance on par with other models trained on substantially larger datasets. This outcome underscores the model's efficiency and effectiveness in handling the given task with a more limited dataset.

Training process

In the linear evaluation phase, the pre-trained encoder's weights are retained or "frozen". The linear classifier is trained without altering the encoder, This approach ensures an unbiased assessment of the feature representations extracted by the pre-trained encoder. The resulting performance metrics from these classifiers, each corresponding to its specific pre-trained encoder, assist in evaluating and comparing various pre-trained encoders in their capacity to derive meaningful feature representations.

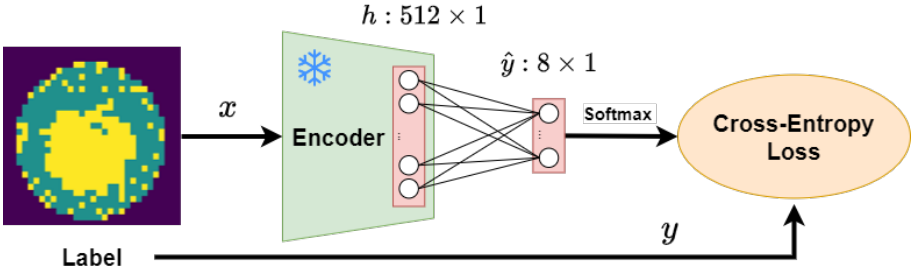


Figure 3.14: **Linear classifier architecture for supervised fine-tuning phase.**

Chapter 4

Evaluation

This chapter delineates the evaluation experiments and their corresponding results, based on the methodologies discussed in Chapter 3. The specific sections are as follows:

- Section 4.1 discusses the settings for the evaluation, including the selected dataset and its properties.
- Section 4.2 gives the hyperparameter setting, the quantitative result of the evaluation. including
- Section 4.3 employs t-SNE for the visualization of the feature space and showcases image retrieval outcomes.

4.1 Data

4.1.1 Data source and property

The data source is divided into two categories: (a) *Synthetic*, (b) *Real-World*.

The *Synthetic data* are collected through generative models, which capture the probability distribution of the data. It is commonly used as a data augmentation tool to tackle class imbalance. However, with synthetic data, training generative models to produce realistic wafer maps that are up-to-date on present design standards (i.e. wafer size, IC node size), and similar to real-world defect patterns is difficult and relatively time-costly[21].

Real-world data consists of publicly available datasets and our private data sets. For example, the WM-811K data is a prominently used public dataset. It consists of 811,457 wafer bin maps from real-world fabrication lots and other manufacturing process data, including die size, lot name, and wafer index[38]. Another publicly available dataset is the Mixed WM-38, which has 38015 WBM with 38 defect classes, including 29 mixed-type defects of 2-,3-,4-mixed types, 8 single-type defects and non-defect WM[34].

Imbalanced Distribution

The imbalanced distribution of real-world wafer map dataset is primarily characterized by two facets: wafer size and pattern.

Table 4.1: **Sample data from WM-811k dataset**

	waferMap	diceize	lotName	waferIndex	trianTestLabel	failureType
811452	[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 1, 1,...	600.0	lot47542	23.0	[Test]	[Edge-Ring]
811453	[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 1, 1,...	600.0	lot47542	24.0	[Test]	[Edge-Loc]
811454	[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 1, 1,...	600.0	lot47542	25.0	[Test]	[Edge-Ring]
811455	[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1,...	600.0	lot47543	1.0	[]	[]
811456	[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 1, 1,...	600.0	lot47543	2.0	[]	[]

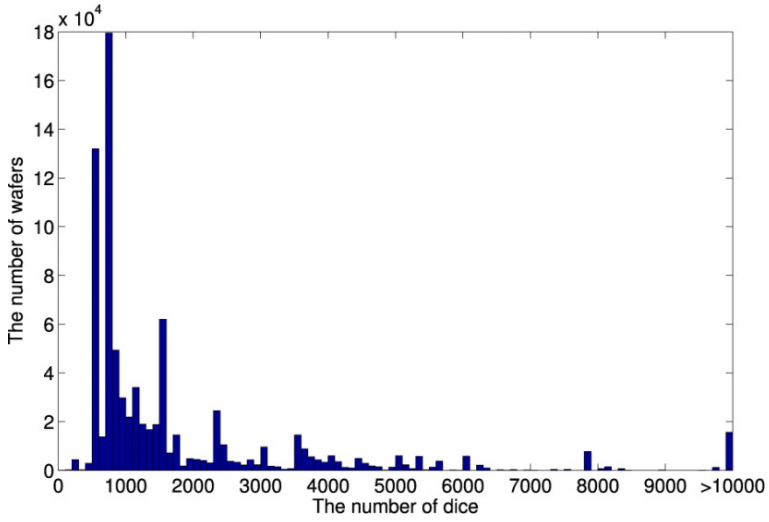


Figure 4.1: **Histogram of the number of dice for the wafer maps in the WM-811K data set.**

Wafer size: The number of dice on each wafer is contingent on the product type. This number can range from a mere few hundred to several thousands, as illustrated in Figure 4.1.

Pattern: In a typical production environment, the majority of products should be of high quality, which implies that most wafer maps exhibit no defect patterns. However, among the few defective wafer maps, the occurrence of wafer defect patterns varies considerably. The distribution of these defective wafer maps in the WM-811K dataset is depicted in Figure 4.2. Such an imbalanced data distribution profoundly influences the classifier's performance.

While oversampling and re-weighting have proved to be effective in supervised classification on Wafer Bin Maps, these strategies aren't feasible for unlabeled datasets, especially with unknown distributions. Considering that products from different semiconductor manufacturing companies should be distinct, their wafer test data will also differ. Moreover, using sampling tricks could potentially compromise the model's ability to learn meaningful representations.

4.1.2 Selected Data

Given these considerations, we refrained from employing any sampling strategies to generate an "ideal" balanced dataset. Instead, we curated a subset of the WM-811K dataset for pre-training purposes. The specifics of this dataset are detailed in Table 4.2.

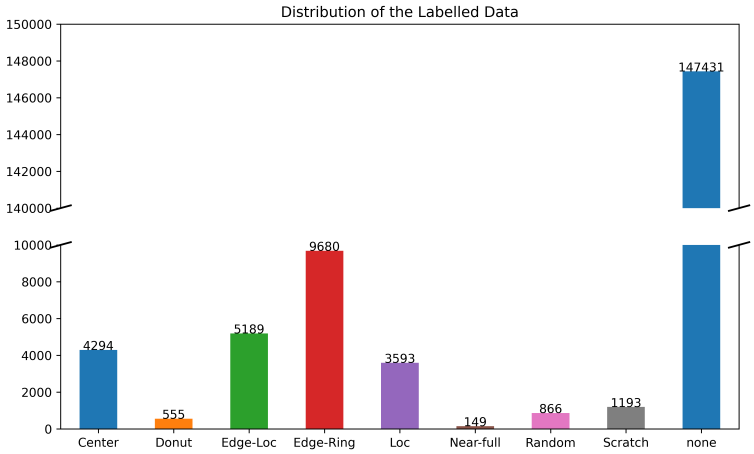


Figure 4.2: **Label distribution of WM-811K data set. Over 85% of data does not contain any defect pattern, and the occurrence of defect pattern also varies significantly: 42% were labeled 'Edge-Ring', only 0.6% were labeled 'Near-full'.**

From the entire dataset, all labeled wafer maps exhibiting a defect pattern were filtered and selected, totaling 25,519 wafer maps. Subsequently, an estimation strategy was employed to select normal wafer maps. Utilizing MLH (Maverick Lot Handing) rate, we were able to manually set a desired failure rate limit, thereby selecting the wafer maps most likely to exhibit a defect pattern. Our conservative estimation led to a split of 40% defective and 60% normal, resulting in the inclusion of 38,278 normal wafer maps.

The defective and normal wafer maps were aggregated, resulting in a comprehensive dataset of 63,797 wafer maps designated for the pre-training stage. Subsequently, this dataset was partitioned based on a 90-10 split, allocated for training and validation respectively.

During the evaluation phase, the performance of the learned representation was assessed using a single-layer linear classifier, primarily focusing on wafer maps exhibiting defect patterns. This is due to the fact that classifying all test data would be redundant since the majority is normal and can be filtered out using MLH. Our objective targets the data subset traditionally subjected to manual inspection, namely the wafer maps identified as highly suspicious by MLH with a stringent limit (e.g., 1-2% of the data). With our model, we could expand this limit (e.g., 5-10% of the data) for inspection, aided by our classification model, while still primarily addressing wafer maps with defect patterns. Therefore, a comprehensive evaluation was conducted on all 25,519 labeled defective wafer maps in WM-811k dataset.

Table 4.2: **Statistics of the Dataset**

Label	Numbers	Pre-Training (90%)	Validation (10%)	Fine-Tuning (80%)	Evaluation (20%)
Center	4294	3865	429	3474	820
Donut	555	500	56	436	119
Edge-Loc	5189	4670	519	4118	1071
Edge-Ring	9680	8712	968	7740	1940
Local	3593	3234	359	2879	714
Near-full	149	134	15	684	182
Random	866	779	87	961	232
Scratch	1193	1074	119	124	25
None	38278	34450	3828	0	0
Sum	63797	57417	6380	20416	5103

4.2 Quantitative Evaluation - Classification Performance

4.2.1 Experiment setting

In summary, the detailed hyperparameter setting of our experiment is listed below:

- Wafer Map Preprocess: All wafer maps were reshaped to (64,64) and expanded to 3 channels using one-hot encoding.
- Augmentations and setting:
  - Random Discrete Rotation. Rotation degrees chosen from 0, 90, 180, 270.
  - Random Cut Out: The first with a probability of 0.125, scale=(0.002,0.04), and ratio=(0.1,10). The second with a probability of 0.33, scale=(0.001,0.02), and ratio=(0.1,10).
  - Random Horizontal Flip.
  - Random Resized Crop with a ratio range of [0.8, 1.0].
- Base Encoder: Modified ResNet-18. The first convolution layer's kernel size was changed from (7,7) to (3,3), with stride and padding adjusted to 1. The output layer produced a 512-dimensional feature vector.
- Projection head: Two fully connected layers with dimensions of 512 and 128. A batch normalization layer and a ReLu activation layer were included in between.
- Contrastive Loss: NT-Xent loss with a temperature of 0.2 for the contrastive learning objective.
- Pre-training: Training was performed for a maximum of 100 epochs with a batch size of 512 samples. Optimizer is Adam with an initial learning rate of 2e-3 and a weight decay of 1e-4. The learning rate is scheduled by a cosine annealing scheduler with a minimum learning rate of 4e-5.

Table 4.3: **Linear evaluation results of SimCLR obtained over different values of τ. Bold value represents the best performance**

τ	0.07	0.1	0.12	0.2	0.5
Marco F1	80.37	82.27	83.12	**83.78**	82.54

- Fine-tuning: A linear layer was added to the base encoder with an input dimension of 512 and an output dimension of 8. The linear layer was trained for a maximum of 100 epochs using the Adam optimizer with a learning rate of 1e-3 and a weight decay of 1e-3. A multistep scheduler was used for fine-tuning with milestones of 60 and 80 epochs. Cross-entropy loss was used as the supervised loss during fine-tuning.

During the evaluation process, we conducted experiments to assess the impact of various augmentation settings and hyperparameters, aiming to gain insights into the optimal configuration for SSCL in the context of WMFPR.

4.2.2 Hyperparameter selection

The temperature parameter (τ) plays a crucial role in enhancing the quality of representations in contrastive learning. To determine the most suitable value of τ, we conducted an experiment in which we trained SimCLR with various τ values, specifically 0.07, 0.1, 0.12, 0.2, and 0.5. We then evaluated the model's performance using the linear evaluation protocol. The results, as shown in Table 4.3 indicate that the best performance was achieved when $\tau = 0.2$. Therefore, we decided to keep this value fixed at 0.2 for all subsequent experiments.

We also conducted tests to assess the influence of batch sizes, including 256, 512, and 1024. However, our analysis did not reveal any statistically significant differences among them. Therefore, we decided to proceed with the largest batch size that could comfortably fit within the memory of our graphics card.

In terms of the pre-training process, our findings indicated that the best-performing model was achieved when trained for a duration of 40 to 80 epochs. Extending the training beyond this range tended to improve the model's performance on the pretext task and with a smaller dataset. However, these two aspects were not correlated, and the performance in the linear evaluation task either remained unchanged or slightly decreased when training for longer epochs.

4.2.3 Unsupervised Baseline

To assess the effectiveness of unsupervised pre-training on classification tasks, we compared the classification results with another method of unsupervised representation learning: Convolutional AutoEncoder (CAE).

For the approach employing SimCLR contrastive pertaining: The model was pre-trained on 57,417 unlabeled WBMs and fine-tuned on a small balanced training set, which was randomly selected from eight labeled defect patterns. The classification accuracy was subsequently evaluated on the evaluation set with 5,103 data from the WM-811K dataset that has a defect pattern.

For the CAE model, it was based on CNN and served as a generative model baseline for comparison. The CAE model's architecture is illustrated in Figure 4.3. To assess the quality of the learned features, we applied a similar linear

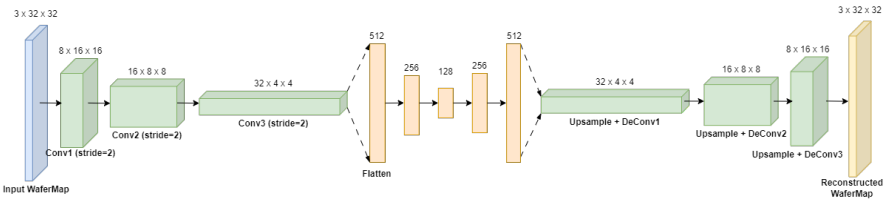


Figure 4.3: **CNN-based Convolutional AutoEncoder learns feature representations through reconstructing the input wafer map.**

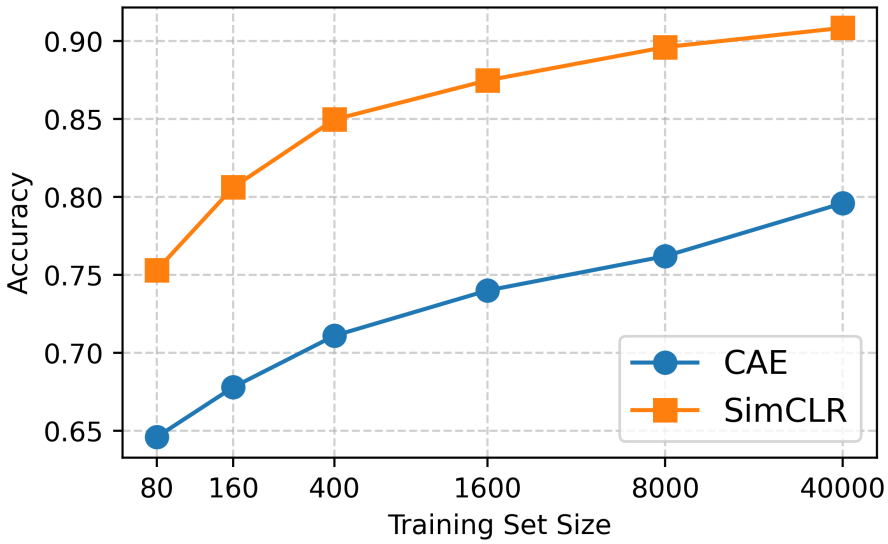


Figure 4.4: **Classification accuracy of CAE and SimCLR model.**

evaluation protocol, utilizing the bottleneck of the encoder-decoder structure as features and classifying them using a single Linear layer. The comparison between the CAE model's performance and SimCLR is depicted in Figure4.4. We can observe that SimCLR outperforms the CAE model by achieving an accuracy improvement of over 10%.

A limitation of CAE in that it primarily learns the distribution of the data and is evaluated based on pixel-wise Mean Square Error (MSE) loss. Consequently, increasing the model's complexity, such as adding more layers or widening the bottleneck dimension, does not necessarily result in improved classification performance, as it primarily aims to accurately reconstruct the original image, including irrelevant details for distinguishing defect patterns. Despite this limitation, CAE can still be valuable for wafer map analysis as a data augmentation tool, generating additional data similar to the original by introducing Gaussian noise to the bottleneck layer.

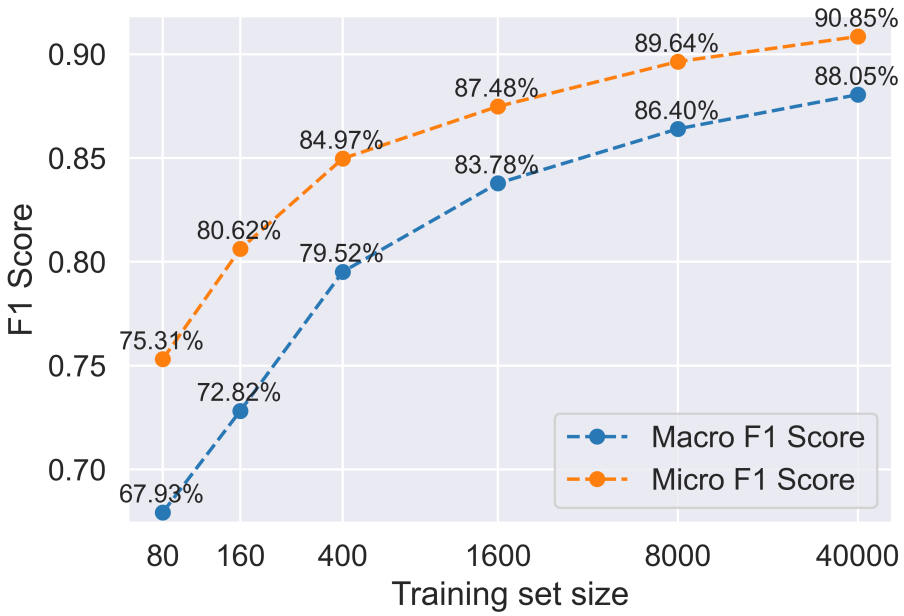


Figure 4.5: **The evaluation results on various training set sizes demonstrate the label-efficiency of our approach, with a Marco-F1 score of over 80% achieved with just a few hundred data samples.**

4.2.4 Classification Result and Comparison

To assess the quality of the extracted features, we conducted a series of linear evaluations using small, balanced datasets. This evaluation approach is specifically designed to evaluate the representativeness of the extracted features.

For a practical application on Nexperia's private dataset, our objective is to facilitate the creation of a tailored dataset and initiate classification with a smaller dataset while ensuring acceptable performance. As we accumulate more data, we can further refine a classifier optimized for our specific dataset. This goal may not be represented by evaluating performance solely based on the WM-811K dataset, given the potential differences in class definitions and distributions. By employing a balanced dataset, we mitigate the bias introduced by varying class weights, providing a more accurate assessment of the discrimination quality of the features.

The evaluation result using different amounts of data in the fine-tuning phase is shown in Figure 4.5. Performance gains exhibit a consistent logarithmic relationship with increasing training dataset size.

To gain a deeper understanding of the classification results for each defect pattern, we conducted a comparative analysis of our evaluation results with two recent state-of-the-art contrastive learning approaches: MOCO4LD [36], SCL [1], and the well-known WMFPR [38] as one of the best-performing methods that have been applied in real-world wafer fabs. As Table 4.4 shows, our SimCLR approach has demonstrated remarkable performance comparable to state-of-the-art methods with just a single linear layer as classifier. This outcome highlights the quality of the extracted features, demonstrating their effective-

Table 4.4: **Comparison of classification accuracy on each pattern with other Contrastive Learning algorithms and classical Machine Learning algorithm. Bold represents the best performance**

Defect Pattern	WMFPR	MOCO4LD	SCL(N=5000)	SimCLR
Center	84.9	90	91	**96**
Donut	74	80	75.17	**96.6**
Edge-Loc	85.1	**88**	77.63	86.2
Edge-Ring	79.7	**98**	96.09	97.1
Local	68.5	74	71.83	**77.2**
Near-full	97.9	97	81.49	**100**
Random	79.8	94	84.17	**94.5**
Scratch	82.4	**85**	43.64	77.2
Average	81.5375	88.25	77.6275	**90.6**

ness in distinguishing between different defect patterns. Moreover, our approach doesn't demand extensive memory or a large labeled dataset to achieve a robust, satisfactory results.

4.2.5 Confusion Matrix

To visualize the quality of the extracted representation for identifying each defect type, we generated a Confusion Matrix, as depicted in Figure 4.6. In this matrix, the diagonal cells represent correct predictions, and the orange-colored numbers within the squares indicate the number of test samples belonging to this cell.

The classification results are notably high for *Center, Donut, Edge-Ring, Random*, and *Near-full*, with an average accuracy of over 95%. However, the classification performance for *Edge-Loc, Loc*, and *Scratch* in our proposed method is relatively lower. This is partly because they all share a certain level of geometric features. Additionally, *Scratch* and *Loc* are frequently confused because these two defect types often appear together on a single Wafer Map, leading to mixed-type problems.

Upon closer examination of the dataset, we noticed that mixed-type defects frequently occur on a single Wafer Map, yet only one label is assigned to each Wafer Map, causing confusion for the model and a decrease in classification performance. Furthermore, labeling bias is evident, particularly with *Loc* and *Edge-Loc*, as there is no clear definition of how much contact a defect pattern must have with the edge to be categorized as *Edge-Loc*. This observation is illustrated in Figure 4.10 and can be further supported by the results of other supervised deep convolutional neural network-based methods, such as in [32], where the classification accuracy for *Local* and *Scratch* patterns are also the lowest among the eight defect types.

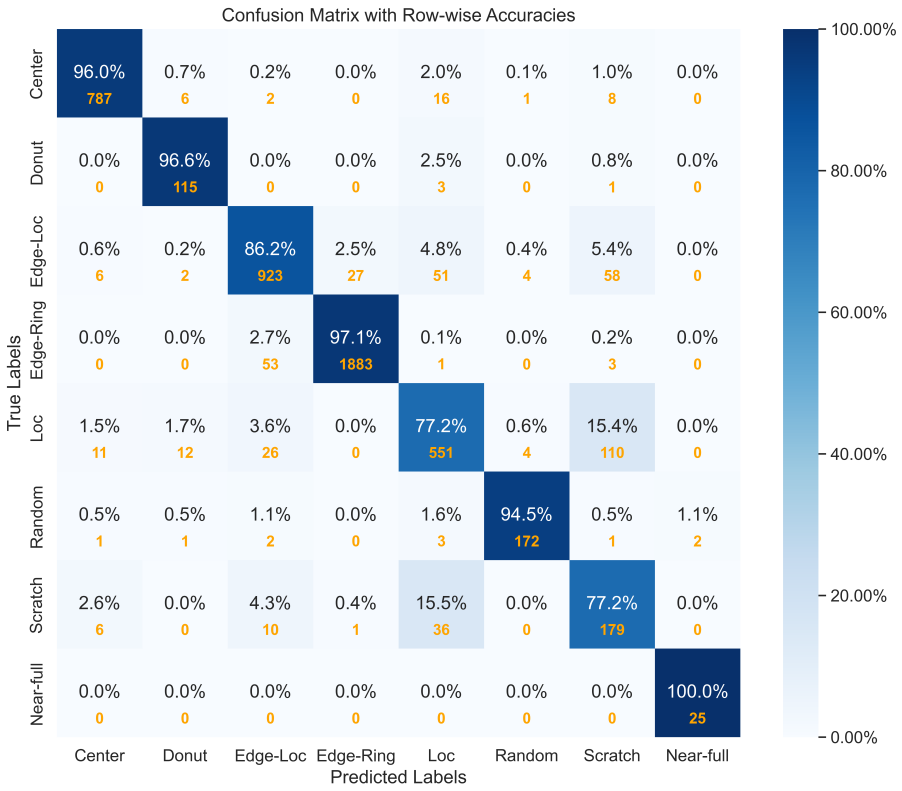


Figure 4.6: **Confusion Matrix of eight defect types.**

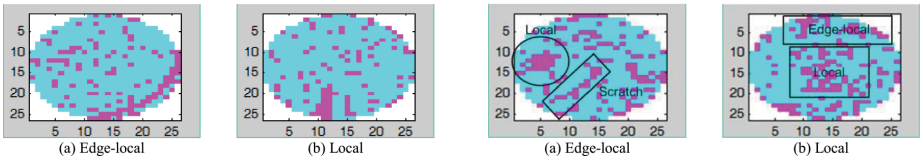
4.3 Qualitative Evaluation

4.3.1 t-SNE Visualization

To get a better understanding of the overall quality of features learned by the models via contrastive learning, we utilize t-distributed Stochastic Neighbor Embedding (t-SNE) [33]. This method allows for the visualization of the high-dimensional feature maps obtained from the pre-trained encoder. For this visualization, we randomly selected 500 images from each pattern and extracted their feature encodings. The models' output dimensions are set at 512.

We initially applied t-SNE directly to the raw wafer maps, as depicted in Figure 4.8, resulting in a two-dimensional visualization of the data. The outcomes revealed some clusters corresponding to similar wafer map patterns, particularly for *Edge-Ring*, *Near-full*, and *Center* patterns. However, these clusters were not distinctly separated, and patterns such as *Edge-Loc*, *Loc*, and *Scratch* appeared more intertwined. This underscores the complexity of raw wafer maps as high-dimensional data, where t-SNE without feature extraction struggles to effectively disentangle defect patterns.

In contrast, the feature space learned by SimCLR exhibits a more fine-grained and separable structure. As shown in Figure 4.9, we can observe well-separated clusters corresponding to *Random*, *Donut*, *Edge-Ring*, *Near-full*, and *Center* patterns. Overall, it is obvious that the interclass distance has increased com-



(I) **Labeling Bias** (II) **Mixed-type pattern**

Figure 4.7: **Examples of Labeling Bias and Mixed-Type Patterns: In Figure (I), wafer map (a) exhibits features that could be interpreted as both a Scratch and an Edge-Local defect. Similarly, wafer map (b) could be categorized as either Local or Edge-Local. In Figure (II), each wafer map contains multiple defect patterns, as indicated within the black boxes, yet they are labeled with only one specific pattern.**

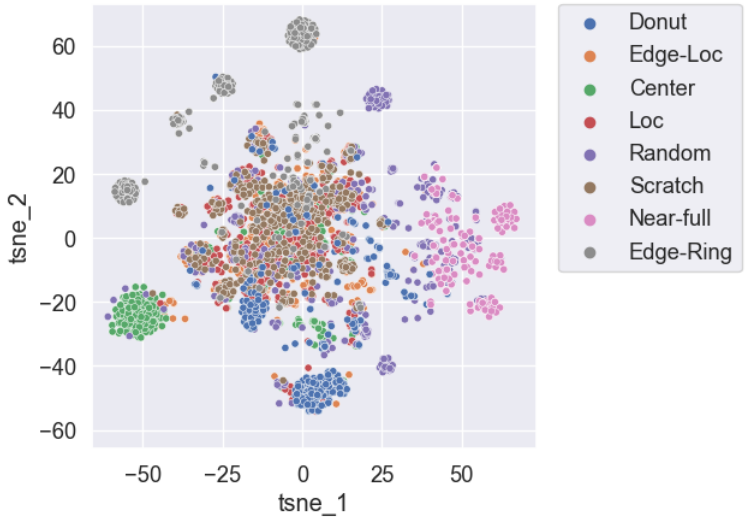


Figure 4.8: **Result of directly applying t-SNE to raw wafer maps.**

pared to the previous result, forming a more fine-grained distribution in feature space. However, the remaining three patterns still display some degree of overlap, which is consistent with the results discussed earlier in Section 4.2.5. This visualization result has also been developed as an interactive map using Plotly, allowing users to view the wafer map of selected data points by hovering the mouse over them. This tool has received positive feedback from experts for efficiently identifying similar wafer maps.

4.3.2 Similar Image Retrieval

To assess the model's capacity to generalize to novel, unknown data—simulating the processing of newly generated wafer maps from wafer fabs with potentially new products and defect patterns—we conducted a Similar Image Retrieval experiment on the dataset obtained from Nexperia's cloud server. The procedure involved the following steps:

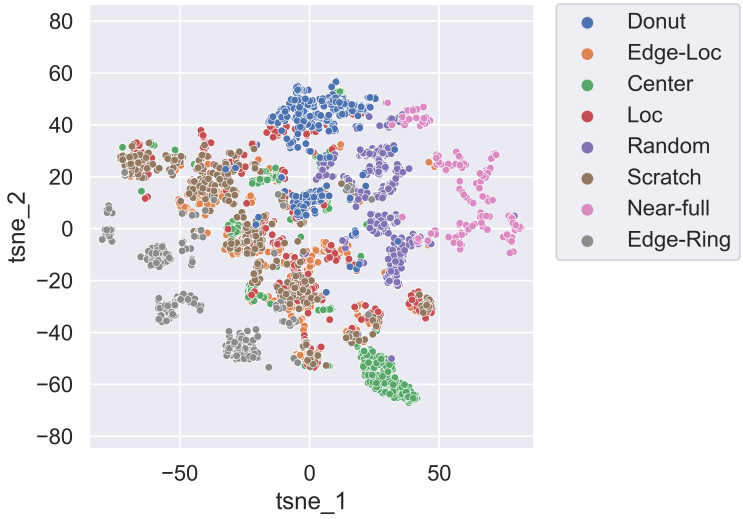
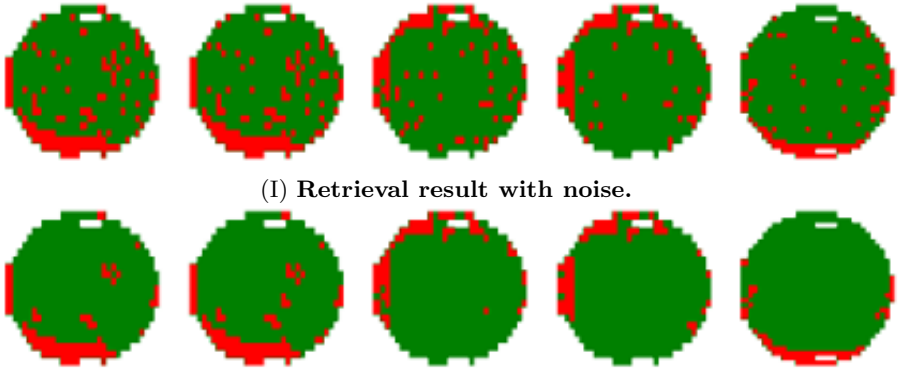


Figure 4.9: **Feature map visualization based on output 512-D feature vectors of the pre-trained encoder and t-SNE.**

1. Collect a database of historical wafer maps and their representations extracted by the Encoder model.
2. Randomly selecting a wafer map as the query image.
3. Utilizing the encoder model to obtain the query wafer map's 512-dimensional feature representation.
4. Applying cosine similarity to identify similar wafer maps within the database and listing the top-N results.

This process yielded several noteworthy findings:

1. **Rotation-Invariance** and **Noise-Resistance**: The first finding may be evident, given that we applied random discrete rotation in our augmentation pipeline. To test the model's ability to distinguish defect regions from noise, we conducted an experiment. We performed image retrieval on the same wafer map twice: once using the raw data and a second time by applying a c-means filter to all the data. Surprisingly, the top-5 retrieval results remained the same, demonstrating that noise has minimal impact on the wafer map's representation.
2. Detection on **mixed-type** pattern: As shown in Figure 4.10 the wafer map contains multiple small patterns, they can be captured by the features. However, if one defect pattern is much larger than another, it would be hard to detect the small one.
3. Generalization to **Unknown Patterns**: In Figure 4.12, we encountered a new pattern in Nexperia's data that did not correspond to any previously defined defect pattern in the WM-811K dataset. This pattern, resembling a random distribution, cut through the wafer by its diameter instead of



(I) **Retrieval result with noise.**

(II) **Retrieval result on filtered data.**

Figure 4.10: **Comparison of image retrieval results with and without noise. The first wafer map is the query wafer map, the rest are ranked by their similarity score.**

Figure 4.11: **Mixed-type wafer map with an Edge-Local and a Local pattern. The first wafer map is the query wafer map; we can see from the rest of the wafer maps that they all exhibit a similar mixed-type defect.**

being randomly spread. Despite its novelty and absence from the pre-training dataset, our retrieval results clearly identified similar examples from the new dataset. This discovery highlights the model's capability to generalize to previously unseen defect patterns.

These findings underscore the model's robustness and its potential applicability to real-world scenarios where variations and unknown patterns are common.

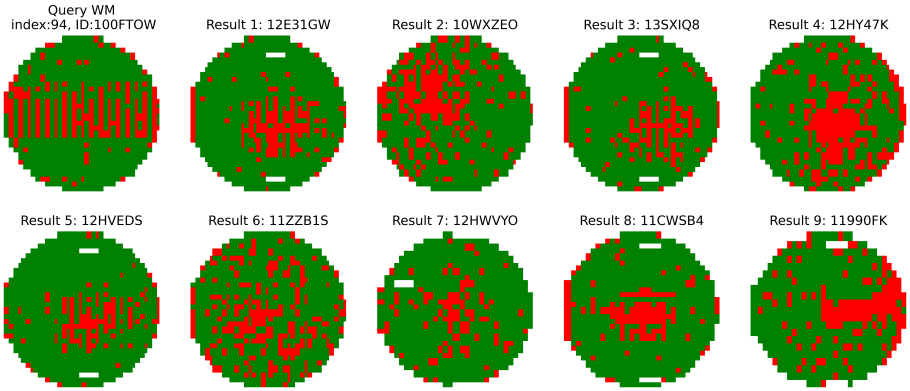


Figure 4.12: **Retrieval result on undefined pattern.**

Chapter 5

Discussion

5.1 Conclusion

In this collaborative thesis project between ITEC and TU Delft, we explored the potential of using Self-Supervised Contrastive Learning algorithms, particularly SimCLR as a generalized approach to extract meaningful feature representations from Wafer Bin Maps. And the application of the learned features for automated RCA and efficient detection of out-of-control manufacturing processes.

The motivation of this project is the demand for automated visual inspection of Wafer Maps, given the limitations of the current MLH method for anomaly detection:

- In practice, due to resource and time constraints, only a small fraction of the generated wafer maps can be inspected, leaving gaps in our understanding of the manufacturing process and potentially hidden issues. Increasing the number of inspected wafer maps could facilitate the early detection of out-of-control processes, preventing further escalation of the issue and reducing costs.

- While MLH is effective in identifying suspicious data based on a statistical analysis of historical data, its adaptability to new situations is limited. For instance, if a manufacturing disruption leads to a temporary spike in failures in a specific region that cannot be immediately removed, MLH would flag all data as suspicious.

- MLH could not provide any information to assist root-cause analysis. The decision-making process is heavily dependent on the experience level of the engineers inspecting it.

We optimized the performance of the SSCL algorithm in extracting meaningful feature representations by employing domain-specific transformations and fine-tuning hyperparameter settings on unlabeled wafer data. Subsequently, we evaluated the algorithm on a small labeled dataset. Our experiments confirmed that these learned features significantly enhance label efficiency, enabling better recognition of defect patterns and improved identification of similar data for a given wafer map.

Table 5.1: **Estimation on the cost of Manual Wafer Inspection at Nexperia**

Factory	Estimated Capacity(Wafer)	Actual Averaged Utilization Rate	Number of Wafers for Inspection(1%)	Inspection Time(hours)/Cost($)
Hamburg Fab	1,000,000(2023)	0.8	8000	667
Manchester Fab	316,800(2022)	0.8	2535	212
New Port Fab	420,000(2021)	0.5	2100	175
Lingang Fab	400,000(2022)	0.4	1600	134
Sum	2,136,800		14235	1188/59400

5.1.1 Business Value

In terms of the business value of this project, to provide a concrete estimation of the potential benefits, I collected data and conducted a preliminary calculation:

Table 5.1 presents data on Nexperia's factory production capacity from various sources. To estimate the average annual wafer (8-inch-equivalent) production, we introduced a factor called the Actual Average Utilization Rate. Based on this, we calculated the Number of Wafers for inspection, assuming a low inspection rate of 1%, and determined the Inspection time, estimating that inspecting one wafer map takes an average of 5 minutes. This allowed us to calculate the annual cost of time for manual wafer inspection. With an assumption of a 50$ per hour labor cost, we calculated the annual cost of this process.

The application scenarios for our approach can be summarized as follows:

1. Automating manual inspection process. Achieving a classification accuracy of approximately 90% would result in a reduction of over 1000 hours of work for process engineers. Alternatively, it would allow the inspection of 128,115 more wafers without introducing additional work.

2. Optimizing screening process for known defects. In situations where there is a known issue with current production that cannot be mitigated, our approach can determine the existence of the pattern and assign different limits, thus avoiding missing other patterns due to a wide limit.

3. Discovering more/unknown patterns early. If an engineer identifies an unknown wafer defect pattern and wishes to find similar cases for analysis, our similar image retrieval technique can be applied to analyze the historical dataset. Alternatively, they could use the 2-D visualization to gain an overview of the characteristics of the test data and find clusters of similar wafer maps.

5.2 Limitations and Challenges

- Mismatch of Pretext Task Performance and Model's Performance: There may be situations where the performance on the pretext task (contrastive learning) does not directly correlate with the model's performance on the downstream task (defect pattern classification). This discrepancy can make it challenging to predict the model's actual performance accurately.

However, generally linear evaluation with a limited amount of data would benefit from longer training.

- Computation Intensive: The approach, particularly when using larger models or extensive augmentations, can be computationally intensive. This could limit its deployment in scenarios with limited computational resources.

- Inability to Evaluate Unknown Patterns: The model's ability to detect unknown defect patterns, i.e., patterns not seen during training, is not evaluated. We could not test the model's performance to generalize to entirely new defect types that are not present in the training data. In our project, we process data from the web server rather than monitor it in real-time on test machines. Therefore, computation resource and response time is not the major concern.

- Requirement for Resizing Wafer Maps: To fit the encoder, all wafer maps must be resized to the same dimensions. This resizing process can potentially lead to information loss in wafer maps with higher original resolutions, affecting the model's ability to capture fine-grained details.

These limitations highlight areas where further research and improvement are needed to enhance the effectiveness and practicality of the proposed approach.

5.3 Future Work

In our pursuit of enhancing the usability and generalizability of the SimCLR contrastive learning model for real-world manufacturing applications, particularly in the field of wafer map defect pattern recognition, we propose the following directions for future research:

1. Detection of **Mix-Type Defect Patterns**: Investigate the model's ability to detect and classify mix-type defect patterns. This could involve generating synthetic mixed-type defect patterns by combining two different defect wafer maps or utilizing publicly available datasets containing mixed patterns. Quantify the impact of classification accuracy on these mixed patterns to better understand the model's capabilities.

2. **Incorporate geographical information**. To enhance the model's ability to differentiate between similar morphological patterns, such as *Local*, *Edge-Local*, and *Center*, which primarily differ in their locations (near the center or on the edge), it may be advantageous to incorporate geographical information during the training process, for example using attention mechanism or adding an extra information channel. This additional information could help the model better understand and distinguish patterns based on their spatial characteristics.

3. **Applying similar paradigm to raw wafer test data**: Integrating findings from raw wafer test data, which may contain patterns indicating potential failures even when devices are within test specifications, with wafer bin map data has the potential to enhance our understanding and

control over the manufacturing process. This integrated approach can help identify issues early in the manufacturing process, preventing potential failures in subsequent procedures.

4. **Self-Training/Knowledge Distillation**: To facilitate model deployment on computation-limited devices, we could consider exploring knowledge distillation as a technique. Researchers have suggested that knowledge distillation, as seen in SimCLRv2 [6], can enhance model efficiency by transferring task-specific knowledge from a larger teacher model to a smaller student model. Additionally, self-distillation may improve the performance of semi-supervised learning.

Bibliography

[1] Youngjae Bae and Seokho Kang. Supervised contrastive learning for wafer map pattern classification. *Engineering Applications of Artificial Intelligence*, 126:107154, 2023.

[2] Uzma Batool, Uzma Batool, Mohd Ibrahim Shapiai, Muhammad Tahir, Zool Hilmi Ismail, Noor Jannah Zakaria, and Ahmed Elfakharany. A Systematic Review of Deep Learning for Silicon Wafer Defect Recognition. *IEEE Access*, 2021.

[3] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. YOLOv4: Optimal Speed and Accuracy of Object Detection, 2020.

[4] Shouhong Chen, Mulan Yi, Yuxuan Zhang, Xingna Hou, Yuling Shang, and Ping Yang. A self-adaptive DBSCAN-based method for wafer bin map defect pattern classification. *Microelectronics Reliability*, 123:114183, 2021.

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations, 2020.

[6] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big Self-Supervised Models are Strong Semi-Supervised Learners. *CoRR*, abs/2006.10029, 2020.

[7] Xiaoyan Chen, Chundong Zhao, Jianyong Chen, Dongyang Zhang, Kuifeng Zhu, and Yanjie Su. K-means clustering with morphological filtering for silicon wafer grain defect detection. In *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, volume 1, pages 1251–1255, 2020.

[8] Deval Shah. Self-Supervised Learning and Its Applications, 8 2023.

[9] Abd Al Rahman M Abu Ebayyeh, Alireza Mousavi, and Alireza Mousavi. A Review and Analysis of Automatic Optical Inspection and Quality Monitoring Methods in Electronics Industry. *IEEE Access*, 2020.

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[11] Jean-Bastien Grill, Florian Strub, Florent Altch'e, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot,

Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap Your Own Latent: A New Approach to Self-Supervised Learning. *ArXiv*, abs/2006.07733, 2020.

[12] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B Girshick. Momentum Contrast for Unsupervised Visual Representation Learning. *CoRR*, abs/1911.05722, 2019.

[13] Shao Chung Hsu and Chen Fu Chien. Hybrid data mining approach for pattern extraction from wafer bin map to improve yield in semiconductor manufacturing. *International Journal of Production Economics*, 107(1):88–103, 5 2007.

[14] Hanbin Hu, Chen He, and Peng Li. Semi-supervised Wafer Map Pattern Recognition using Domain-Specific Data Augmentation and Contrastive Learning. In *2021 IEEE International Test Conference (ITC)*, pages 113–122, 2021.

[15] Young Seon Jeong, Seong Jun Kim, and Myong K. Jeong. Automatic identification of defect patterns in semiconductor wafer maps using spatial correlogram and dynamic time warping. *IEEE Transactions on Semiconductor Manufacturing*, 21(4):625–637, 11 2008.

[16] Marc-Oliver Otto Jeonghoon Choi Dongjun Suh. Boosted Stacking Ensemble Machine Learning Method for Wafer Map Pattern Classification. *Computers, Materials & Continua*, 74(2):2945–2966, 2023.

[17] Cheng Hao Jin, Hyuk Jun Na, Minghao Piao, Gouchol Pok, and Keun Ho Ryu. A Novel DBSCAN-Based Defect Pattern Detection and Classification Framework for Wafer Bin Map. *IEEE Transactions on Semiconductor Manufacturing*, 32(3):286–292, 2019.

[18] Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding Dimensional Collapse in Contrastive Self-supervised Learning, 2022.

[19] Hyungu Kahng and Seoung Bum Kim. Self-Supervised Representation Learning for Wafer Bin Map Defect Pattern Classification. *IEEE Transactions on Semiconductor Manufacturing*, 34(1):74–86, 2021.

[20] Jinho Kim, Youngmin Lee, and Heeyoung Kim. Detection and Clustering of Mixed-type Defect Patterns in Wafer Bin Maps. *IISE Transactions*, 50, 10 2017.

[21] Tongwha Kim and Kamran Behdinan. Advances in machine learning and deep learning applications towards wafer map defect recognition and classification: a review. *Journal of Intelligent Manufacturing*, 2022.

[22] Yuting Kong and Dong Ni. A Semi-Supervised and Incremental Modeling Framework for Wafer Map Classification. *IEEE Transactions on Semiconductor Manufacturing*, 33(1):62–71, 2 2020.

[23] Min Gu Kwak, Young Jae Lee, and Seoung Bum Kim. SWaCo: Safe Wafer Bin Map Classification With Self-Supervised Contrastive Learning. *IEEE Transactions on Semiconductor Manufacturing*, 36(3):416–424, 2023.

[24] Katherine Shu-Min Li, Xu-Hao Jiang, Leon Li-Yang Chen, Sying-Jyan Wang, Andrew Yi-Ann Huang, Jwu E Chen, Hsing-Chung Liang, and Chun-Lung Hsu. Wafer Defect Pattern Labeling and Recognition Using Semi-Supervised Learning. *IEEE Transactions on Semiconductor Manufacturing*, 35(2):291–299, 2022.

[25] Ishan Misra and Laurens van der Maaten. Self-Supervised Learning of Pretext-Invariant Representations. *CoRR*, abs/1912.01991, 2019.

[26] Lars Mönch, J W Fowler, and Scott Mason. *Production Planning and Control for Semiconductor Wafer Fabrication Facilities: Modeling, Analysis, and Systems*. 1 2013.

[27] Takeshi Nakazawa and Deepak V Kulkarni. Wafer Map Defect Pattern Classification and Image Retrieval Using Convolutional Neural Network. *IEEE Transactions on Semiconductor Manufacturing*, 2018.

[28] Minghao Piao, Cheng Hao Jin, Cheng Hao Jin, Jong Yun Lee, Jong Yun Lee, and Jeong-Yong Byun. Decision Tree Ensemble-Based Wafer Map Failure Pattern Recognition Based on Radon Transform-Based Features. *IEEE Transactions on Semiconductor Manufacturing*, 2018.

[29] Muhammad Saqlain, Bilguun Jargalsaikhan, and Jong Yun Lee. A voting ensemble classifier for wafer map defect patterns identification in semiconductor manufacturing. *IEEE Transactions on Semiconductor Manufacturing*, 32(2):171–182, 5 2019.

[30] Thalles Santos Silva. A Few Words on Representation Learning. *https://sthalles.github.io*, 2021.

[31] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What Makes for Good Views for Contrastive Learning?, 2020.

[32] Theodoros Tziolas, Theodosis Theodosiou, Konstantinos Papageorgiou, Aikaterini Rapti, Nikolaos Dimitriou, Dimitrios Tzovaras, and Elpiniki Papageorgiou. Wafer Map Defect Pattern Recognition using Imbalanced Datasets. In *2022 13th International Conference on Information, Intelligence, Systems & Applications (IISA)*, pages 1–8, 2022.

[33] Laurens van der Maaten and Geoffrey Hinton. Viualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 10 2008.

[34] Junliang Wang, Junliang Wang, Chuqiao Xu, Zhengliang Yang, Jie Zhang, Jie Zhang, Xiaoou Li, and Xiaoou Li. Deformable Convolutional Networks for Efficient Mixed-Type Wafer Defect Pattern Recognition. *IEEE Transactions on Semiconductor Manufacturing*, 2020.

[35] Rui Wang and Nan Chen. Defect pattern recognition on wafers using convolutional neural networks. *Quality and Reliability Engineering International*, 36(4):1245–1257, 2 2020.

[36] Yi Wang, Dong Ni, and Zhenyu Huang. A Momentum Contrastive Learning Framework for Low-Data Wafer Defect Classification in Semiconductor Manufacturing. *Applied Sciences*, 13(10), 2023.

[37] Lilian Weng. Self-Supervised Representation Learning. *lilianweng.github.io*, 2019.

[38] Ming-Ju Wu, Ming-Ju Wu, Jyh-Shing R Jang, Jyh-Shing Roger Jang, and Jui-Long Chen. Wafer Map Failure Pattern Recognition and Similarity Ranking for Large-Scale Data Sets. *IEEE Transactions on Semiconductor Manufacturing*, 2015.

[39] Qiao Xu, Naigong Yu, and Firdaous Essaf. Improved Wafer Map Inspection Using Attention Mechanism and Cosine Normalization. *Machines*, 10(2), 2022.

[40] Nai-gong Yu, Qiao Xu, Hong-lu Wang, and Jia Lin. Wafer bin map inspection based on DenseNet. *Journal of Central South University*, 28(8):2436–2450, 2021.

[41] Naigong Yu, Qiao Xu, and Honglu Wang. Wafer Defect Pattern Recognition and Analysis Based on Convolutional Neural Network. *IEEE Transactions on Semiconductor Manufacturing*, 32(4):566–573, 10 2019.

[42] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful Image Colorization, 2016.

[43] Richard Zhang, Phillip Isola, and Alexei A Efros. Split-Brain Autoencoders: Unsupervised Learning by Cross-Channel Prediction. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 645–654, 2017.

[44] Huilin Zheng, Syed Waseem Abbas Sherazi, Sang Hyeok Son, and Jong Yun Lee. A Deep Convolutional Neural Network-Based Multi-Class Image Classification for Automatic Wafer Map Failure Recognition in Semiconductor Manufacturing. *Applied Sciences*, 11(20):9769, 10 2021.