Deep Learning for Automatic Optical Inspection and Quality Evaluation of Semiconductor and Optoelectronic Manufacturing

A thesis submitted for the degree of

Doctor of Philosophy

by

Abd Al Rahman M. Abu Ebayyeh

Department of Computer & Electronic Engineering
Brunel University London

February 2022

Declaration of Authorship

I, Abd Al Rahman M. Abu Ebayyeh, declare that this thesis entitled *Deep Learning for Automatic Optical Inspection and Quality Evaluation of Semiconductor and Optoelectronic Manufacturing* and the data presented in it are original and my own work.

I confirm that:

- No part of this work has previously been submitted for a degree at this or any other university.
- References to the work of others have been clearly acknowledged. Quotations from the work of others have been clearly indicated, and attributed to them.
- In cases where others have contributed to part of this work, such contribution has been clearly acknowledged and distinguished from my own work.
- None of this work has been previously published elsewhere, with the exception of the following:
  - **A. M. Abu Ebayyeh** and A. Mousavi, "A Review and Analysis of Automatic Optical Inspection and Quality Monitoring Methods in Electronics Industry", *IEEE Access*, vol. 8, pp. 183192–183271, 2020. doi: 10.1109/ACCESS.2020.3029127 Status: **Published**
  - **A. M. Abu Ebayyeh** , S. Danishvar and A. Mousavi, "An Improved Capsule Network (WaferCaps) for Wafer Bin Map Classification Based on DCGAN Data Upsampling", *IEEE Transactions on Semiconductor Manufacturing*, vol. 35, no. 1, 2022. doi: 10.1109/TSM.2021.3134625 Status: **Published**

– **A. M. Abu Ebayyeh** , A. Mousavi, S. Danishvar, T. Gresch, O. Landry and A. Müller, "A Waveguide Quality Inspection in Quantum Cascade Lasers: A Capsule Neural Network Approach", *Expert Systems with Applications* Status: **Under Review (2nd Round)**

Date: February 2022

Signature: Abdalrahman M. Abu Ebayyeh

Abstract

Electronics industry is one of the fastest evolving, innovative, and most competitive industries. In order to meet the high consumption demands on electronics components, quality standards of the products must be well-maintained. Automatic optical inspection (AOI) is one of the non-destructive techniques used in quality inspection of various products. This technique is considered robust and can replace human inspectors who are subjected to dull and fatigue in performing inspection tasks. A fully automated optical inspection system consists of hardware and software setups. Hardware setup include image sensor and illumination settings and is responsible to acquire the digital image, while the software part implements an inspection algorithm to extract the features of the acquired images and classify them into defected and non-defected based on the user requirements. This research aims to explore methods used in literature for quality and AOI of the basic building block of electronic and optoeletronic devices which are semiconductor and optoelectronic wafers. Deep Learning (DL) techniques has proven its ability in extracting the feature automatically within their hidden layers saving the export knowledge and efforts when compared to conventional image processing and feature extraction techniques. DL can also preform classification based on the examples learned by network, while conventional rule-based classification uses conditional statements to provide a decision. Based on the previous advantages, DL networks were used to be as a decision making system for classifying defect patterns and anomalies in semiconductor and optoelectronic wafers. Open source WM-811K were used in our investigation of semiconductor wafer bin maps (WBMs) patterns. It was observed that the dataset is imbalanced and some classes do not contain enough samples for training, therefore, Deep Convolutional Generative Adversarial Network (DCGAN) were used to upsample and increase the dataset. A novel improved capsule network WaferCaps were also proposed to classify the defect patterns according to eight classes. The performance of our proposed DCGAN

and WaferCaps was compared with different deep learning models such as the original Capsule Network (CapsNet), Convolutional Neural Network (CNN), and Multi-layer Perceptron (MLP). In all of our experiment, WM-811K dataset was used for the data upsampling and training. The proposed approach has shown an effective performance in generating new synthetic data and classify them with training accuracy of 99.59%, validation accuracy of 97.53% and test accuracy of 91.4%. For analysing optoelectronic wafer, we focused our attention on the waveguide of quantum cascade laser (QCL) in the wafer. Due to the lack of sufficient dirt and defect samples, data augmentation approach was used to increase the number of images. Decision fusion approach was used to integrate the classification performance of CNN and WaferCaps since they performed the best in different classes. The proposed approach was compared to similar DL algorithms and it achieved an overall accuracy of 98.5%.

Acknowledgment

First, I would like to express my sincerest appreciation for my principle supervisor Dr Alireza Mousavi for his guidance and support throughout this project. His dedication, motivation and pursuit of excellence along with his continuous stream of ideas has inspired me to do my best. Secondly, I would also like to thank SERG team for being there for me and for their support whenever I needed help and guidance. A special thanks goes to Dr Sebelan Danishvar for being a friend, mentor and co-author during my research journey.

Beside my supervisor and SERG team, I would like also to acknowledge the iQonic project funding opportunity which is provided from the European Union's Horizon 2020 Research and Innovation Program under Grant agreement No. 820677. Thanks to all the project partners for providing the necessary data to work with and special thanks goes to my colleagues in the project Dr Hang Geng and Nikolaos Markatos for being there in all the ups and downs.

Also, this could not have been possible without the outstanding support, love and encouragement from my brothers Farid, Ferass and Qais. They truly have been there for me whenever I needed their help. I am forever grateful.

Most importantly, I would like to extend my greatest appreciation to my wife Dima who has sacrificed so much for me to accomplish this work. Without her love and support, all would be lost. She has always been there for me and believed in me and gave her emotional support when things seemed tough. Thanks for being patient and understanding.

Lastly, I would like to thank my late mother Itidal for always being in my thoughts. Whenever the roads are dark, I always think of her to see the light. She was and always will be my motivator. Thank you Mama!

Abbreviations

ACO	Ant Colony Optimization
ADTree	Alternating Decision Tree
AE	Autoencoder
ANN	Artificial Neural Network
AOI	Automatic Optical Inspection
ART	Adaptive Resonance Theory
A-SVDD	Adaptive Support Vector Data Description
BDCT	Block Discrete Cosine Transform
BGA	Ball Grid Array
CAE	Convolutional Autoencoder
CapsNet	Capsule Network
CART	Classification and Regression Tree
CCD	Charge-coupled Device
CCL	Connected Component Labelling
CMOS	Complementary Metal-oxide Semiconductor
CMP	Chemical Mechanical Process
CNN	Convolutional Neural Network
COB	Chip-on-board
CR	Contour-based Registration
DAE	Denoising Autoencoder
DBSCAN	Density-based Spatial Clustering of Applications with Noise
DCGAN	Deep Convolutional Adversarial Network
DCT	Direct Cosine Transform

DDPFinder	Dominant Defective Patterns Finder
DIP	Dual-in-line Package
DL	Deep Learning
DNN	Deep Neural Network
DRAM	Dynamic-random Access Memory
DTSVM	Decision Tree Support Vector Machine
DTW	Dynamic Time Wrapping
DWT	Direct Wavelet Transform
ECS	Example Contribution
ESDAE	Enhanced Stacked Denoising Autoencoder
EWMA	Exponentially Weighted Moving Average
FCN	Fully Connected Network
FFT	Fast Fourier Transform
FOV	Field of View
FPD	Flat Panel Display
FR	Frame Rate
F-SVDD	Fast Support Vector Data Description
FVCP	Fuzzy Variable of Clustering Pattern
GA	Genetic Algorithm
GAN	Generative Adversarial Network
gHMT	Growing Wavelet Hidden Markov Tree
GLCM	Graylevel Co-occurrence Matrix
GRN	General Regression Network
HNN	Hopfield Neural Network
HP	High Power
IC	Integrated Circuit
ICA	Independent Component Analysis
IFCM	Improved Fuzzy c-means Clustering
INCC	Improved Normalized Cross Correlation
IR-CUT	Infrared Cut-off

k-NN	k-Nearest Neighbor
KPCA	Kernel Principle Component Analysis
LBF	Local Binary Fitting
LBP	Local Binary Patterns
LCD	Liquid Crystal Display
LDIM	Localized Defects Image Model
LED	Light Emitting Diode
LVQ	Learning Vector Quantization
MCV	Markov Concurrent Vision
MDC	Minimum Distance Classifier
MLP	Multi-layer Perceptron
MSCDAE	Multiscale Convolutional Denoising Autoencoder
MSE	Mean Square Error
MS-FCAE	Fully Convolutional Autoencoder
NCC	Normalized Cross Correlation
NDF	Neighboring Difference Filter
NSCT	Non-subsampled Contourlet Transform
OCSVM	One-class Support Vector Machine
OCT	Optical Coherence Tomography
OLED	Organic Light Emitting Diode
OPBC	Outlier prejudging-based Image Background Construction
OSC-TL	Online Sequential Classifier and Transfer Learning
PC	Principle Component
PCA	Principle Component Analysis
PCB	Printed Circuit Board
PCC	Pearson's Correlation Coefficient
PICC	Partial Information Correlation Coefficient
PLC	Programmable Logic Controller
PLED	Polymer Light Emitting diode
PLS	Partial Least Squares

PR	Pixel Resolution
PSO	Particle Swarm Optimization
QCL	Quantum Cascade Laser
QK-SVDD	Quasiconformal Kernel Support Vector Data Description
RBF	Radial-basis Function
RGRN	Randomized General Regression Network
RMI	Rotational Moment Invariants
ROI	Region of Interest
RPCA	Robust Principle Component Analysis
SAE	Sparse Autoencoder
SD	Source and Drain
SDAE	Stacked Denoising Autoencoder
SDC	Segmentation, Detection, and Cluster-Extraction
SEM	Scanning Electron Microscope
SMD	Surface-mounted Device
SMOTE	Synthetic Minority Over-sampling Technique
SMT	Surface Mount Technology
SOM	Self-organising Map
SURF	Speeded Up Robust Feature Extraction
SVC	Support Vector Clustering
SVD	Singular Value Decomposition
SVDD	Support Vector Data Description
SVM	Support Vector Machine
SVR	Support Vector Regression
SZ	Sensor Size
TFT	Thin-film Transistor
THT	Through Hole Technology
ViBe	Visual Background extraction
WaferCaps	Wafer Capsule Network
WBM	Wafer Bin Map

WD	Working Distance
WKDE	Weighted kernel density estimation
WM	Wafer Map
WMS	Wavelet-based Multivariate Statistical
WNN	Wavelet-based Neural Network
WTMS	Wavelet Tansform Modulus Sum

List of Publications

- **A. M. Abu Ebayyeh** and A. Mousavi, "A Review and Analysis of Automatic Optical Inspection and Quality Monitoring Methods in Electronics Industry", *IEEE Access*, vol. 8, pp. 183192–183271, 2020. doi: 10.1109/ACCESS.2020.3029127 Status: **Published**
- **A. M. Abu Ebayyeh** , S. Danishvar and A. Mousavi, "An Improved Capsule Network (WaferCaps) for Wafer Bin Map Classification Based on DCGAN Data Upsampling", *IEEE Transactions on Semiconductor Manufacturing*, vol. 35, no. 1, 2022. doi: 10.1109/TSM.2021.3134625 Status: **Published**
- **A. M. Abu Ebayyeh** , A. Mousavi, S. Danishvar, T. Gresch, O. Landry and A. Müller, "A Waveguide Quality Inspection in Quantum Cascade Lasers: A Capsule Neural Network Approach", *Expert Systems with Applications* Status: **Under Review (2nd Round)**

To my late mother and my beloved wife ...

Contents

Declaration of Original and Sole Authorship ii

Abstract iv

Acknowledgment vi

Abbreviations vii

List of Publications xii

Contents xiv

List of Figures xix

List of Tables xxiii

List of Algorithms xxv

1 Introduction 1

1.1 Overview . . . 1

1.2 Problem Statement . . . 1

1.3 Research Motivation . . . 2

1.4 Research Context . . . 3

1.5 Research Challenges & Questions . . . 6

1.6 Research Aim & Objectives . . . 7

1.7 Thesis Structure . . . 7

2 Literature Review 9
2.1 Overview . . . 9
2.2 Quality Monitoring Methods . . . 10
2.3 Automatic Optical Inspection . . . 12
2.4 AOI in Electronics Industry . . . 13
2.4.1 LED Defects . . . 14
2.4.2 FPD Defects . . . 15
2.4.3 PCB Defects . . . 17
2.4.4 Semiconductor Wafer Defects . . . 20
2.4.5 Miscellaneous Defects . . . 23
2.4.6 Hardware Setup . . . 23
2.4.6.1 Contrast and Illumination Setup . . . 23
2.4.6.2 Camera & Lens Selection and Positioning . . . 26
2.4.6.3 Auxiliary Systems and Other Image Acquisition Techniques . . . 32
2.5 Inspection Algorithm for Semiconductor Wafer Defects . . . 34
2.5.1 Preprocessing . . . 35
2.5.2 Feature Extraction . . . 36
2.5.2.1 Frequency Domain Analysis . . . 36
2.5.2.2 Segmentation . . . 38
2.5.2.3 Hough Transform . . . 41
2.5.2.4 Template Matching . . . 42
2.5.2.5 Gray-level Co-occurrence Matrix . . . 43
2.5.3 Classification . . . 45
2.5.3.1 Rule-based Classification . . . 45
2.5.3.2 Machine Learning Classifiers . . . 47
2.6 Deep Learning in AOI . . . 60
2.7 Conclusion . . . 70

3 Deep Learning for Computer Vision - An Overview & Evaluation 73
3.1 Overview . . . 73
3.2 Artificial Intelligence, Machine Learning & Deep Learning . . . 74

3.3 Perceptron . . . 76
3.3.1 Learning in Perceptron . . . 79
3.3.2 MNIST Hand Written Digits Dataset . . . 82
3.3.3 Evaluation Metrics . . . 82
3.3.4 Evaluating Perceptron using MNIST Dataset . . . 83
3.4 Multi-layer Perceptron . . . 85
3.4.1 Batch Training . . . 88
3.4.2 Evaluating MLP using MNIST Dataset . . . 89
3.4.3 Overfitting . . . 91
3.4.4 Data Imbalance . . . 92
3.5 Convolutional Neural Networks . . . 94
3.5.1 Evaluating CNN using MNIST Dataset . . . 95
3.5.2 Common CNN Architectures . . . 98
3.5.2.1 LeNet-5 . . . 100
3.5.2.2 AlexNet . . . 100
3.5.2.3 ZefNet . . . 100
3.5.2.4 VGG-16 . . . 101
3.5.2.5 GoogleNet . . . 101
3.5.2.6 Inception-V3 . . . 101
3.5.2.7 ResNet . . . 102
3.5.2.8 Xception . . . 102
3.5.2.9 ResNext . . . 103
3.6 Autoencoder . . . 103
3.6.1 Evaluating AE using MNIST Dataset . . . 103
3.6.2 Convolutional Autoencoder . . . 105
3.6.3 Evaluating CAE using MNIST Dataset . . . 106
3.7 Generative Adversarial Networks . . . 107
3.7.1 Evaluating GAN Using MNIST Dataset . . . 109
3.7.2 Deep Convolutional Generative Adversarial Networks . . . 110
3.7.3 Evaluating DCGAN Using MNIST Dataset . . . 110

3.8 Capsule Networks 112
3.8.1 Evaluating CapsNet using MNIST Dataset 116
3.9 Conclusion 118

4 Implementation of Deep Learning for Semiconductor Wafer Inspection 119
4.1 Overview 119
4.2 Motivation 120
4.3 WM-811K Dataset 121
4.4 Data Preprocessing 124
4.5 Data Augmentation & Upsampling 124
4.6 Classification using WaferCaps 127
4.7 Results 127
4.7.1 DCGAN 129
4.7.2 Experimental Data 129
4.7.3 Ablation Study: Parameter Influence on CapsNet 131
4.7.3.1 The Influence of Image Size 132
4.7.3.2 The Influence of Dropout 132
4.7.3.3 The Influence of Number of Layers 133
4.7.3.4 The Influence of Kernel Size 133
4.7.3.5 The Influence of Data Size 135
4.7.4 Comparison with Other Deep Learning Models 135
4.7.5 Results Discussion 136
4.8 Conclusion 140

5 Implementation of Deep Learning for Optoelectronic Wafer Inspection 142
5.1 Overview 142
5.2 Motivation 144
5.3 Investigated Anomalies 145
5.4 Image Acquisition 146
5.5 Data Augmentation & Upsampling 147
5.6 Classification with Parallel Decision Fusion: WaferCaps+CNN 149

5.7 Results . . . 151
5.7.1 Experimental Data . . . 152
5.7.2 Proposed method vs. standalone models . . . 152
5.7.3 Proposed method vs. other DL models . . . 153
5.8 Attempted Results . . . 153
5.8.1 CAE . . . 155
5.8.2 DCGAN . . . 158
5.9 Conclusion . . . 160

6 Conclusion and Future Work **161**
6.1 Overview . . . 161
6.2 Review of Main Findings . . . 161
6.3 Contributions . . . 163
6.4 Limitations . . . 164
6.5 Future Work . . . 164
6.5.1 Uncertainty in Deep Learning . . . 165
6.5.2 CapsuleGAN . . . 165
6.6 Conclusion . . . 166

Bibliography **167**

List of Figures

1.1 Electronic wafer 3
1.2 Semiconductor wafer fabrication processes. 4
1.3 Segments of optoelectronic wafers 5
2.1 Literature review structure. 10
2.2 AOI system 13
2.3 A map that shows the literature (2000 - 2019) according to the country and product considered. 14
2.4 Different LED types. 15
2.5 Mura defect types 17
2.6 The frequency of PCB-related defects. 19
2.7 Commonly investigated PCB defects 19
2.8 Different WM patterns (red squares indicate defective dies) 22
2.9 Illumination setup considered by Wu et al. 26
2.10 Illumination setup considered by Chiu et al. 26
2.11 Illumination setup considered by Liu et al. 28
2.12 Camera setups considered by Ong et al. 28
2.13 Illumination setup considered by Lu et al. 28
2.14 Illumination setup proposed 29
2.15 Imaging setup proposed by Tsai and Tsai 29
2.16 Illumination setup proposed by Deng et al. 29
2.17 Illumination setup proposed by Chang et al. 30
2.18 Factors considered in selecting camera and lens 31

2.19 OCT setup. 32
2.20 Various semiconductor wafer testing techniques 33
2.21 Various technologies used for solder ball inspection 33
2.22 Truncating lower frequency coefficient (C_{00}) 37
2.23 Binning of WM circle of raduis R 44
2.24 Optimal hyperplane that sperates the two classes 49

3.1 Deep learning is a subset of machine learning and machine learning is a subset of artificial intelligence. 76
3.2 Perceptron 77
3.3 Commonly used activation functions. 78
3.4 Gradient descent visualisation. 81
3.5 MNIST data sample. 82
3.6 Confusion Matrix. 83
3.7 Perceptron architecture for classifying MNIST dataset. 84
3.8 Perceptron's training and validation accuracy for MNIST dataset. 85
3.9 Confusion matrix for MNIST dataset classification using perceptron. 86
3.10 MLP. 87
3.11 MLP architecture for classifying MNIST dataset. 89
3.12 MLP's training and validation accuracy for MNIST dataset. 90
3.13 Confusion matrix for MNIST dataset classification using MLP. 90
3.14 MNIST training data count and distribution per class. 93
3.15 CNN architecture. 94
3.16 Convolution and pooling operations. 96
3.17 CNN architecture for classifying MNIST dataset. 97
3.18 CNN's training and validation accuracy for MNIST dataset. 98
3.19 Confusion matrix for MNIST dataset classification using CNN. 99
3.20 Evolution of common CNN architectures. 99
3.21 LeNet-5 architecture. 100
3.22 Residual layers concept. 102
3.23 AE architecture. 104

3.24 AE architecture for reconstructing MNIST dataset. 104
3.25 AE's training and validation loss for MNIST dataset. 105
3.26 Original vs reconstructed results of AE. 105
3.27 CAE architecture. 106
3.28 CAE architecture for reconstructing MNIST dataset. 106
3.29 CAE's training and validation loss for MNIST dataset. 107
3.30 Original vs reconstructed results of CAE. 107
3.31 GAN architecture. 108
3.32 GAN loss in training MNIST dataset. 110
3.33 GAN epochs. 111
3.34 DCGAN loss in training MNIST dataset. 112
3.35 DCGAN epochs. 113
3.36 Originally proposed CapsNet for MNIST handwritten digit classification. 114
3.37 CapsNet's training and validation accuracy for MNIST dataset. 116
3.38 Confusion matrix for MNIST dataset classification using CapsNet. 117
3.39 Original vs reconstructed results of CapsNet. 117

4.1 Centre WBM patterns (yellow dots indicate defective dies). 122
4.2 Donut WBM patterns (yellow dots indicate defective dies). 122
4.3 Edge-Loc WBM patterns (yellow dots indicate defective dies). 122
4.4 Edge-Ring WBM patterns (yellow dots indicate defective dies). 122
4.5 Loc WBM patterns (yellow dots indicate defective dies). 122
4.6 Near-Full WBM patterns (yellow dots indicate defective dies). 122
4.7 Random WBM patterns (yellow dots indicate defective dies). 123
4.8 Scratch WBM patterns (yellow dots indicate defective dies). 123
4.9 WM-811k data availability 123
4.10 Architecture of the proposed DCGAN for generating synthetic WBMs. 126
4.11 Our proposed WaferCaps architecture. 128
4.12 Generated synthetic WBMs for DCGAN over multiple epochs. 130
4.13 Model loss for donut class. 130
4.14 Image upsampling procedure for Scratch class. 131

4.15 Experimental data used. 131
4.16 Confusion matrix for test data according to original CapsNet (image size of 28×28). 133
4.17 Confusion matrices for the test data with different parameter scenarios. 134
4.18 Confusion matrices for the test data with proposed WaferCaps. 136
4.19 Confusion matrices for the test data with CNN. 136
4.20 Confusion matrices for the test data with MLP. 137
4.21 CNN architecture used for comparison. 137
4.22 MLP architecture used for comparison. 137

5.1 QCL wafer. 143
5.2 Waveguide in QCL. 144
5.3 Framework 145
5.4 Samples for waveguide defects in QCL wafer. 146
5.5 Samples for dirt on waveguide in QCL wafer. 146
5.6 Normal waveguide samples in QCL wafer. 146
5.7 Image acquisition approach. 148
5.8 Manually augmented samples for defect and dirt anomalies. 149
5.9 CNN architecture. 150
5.10 WaferCaps proposed 150
5.11 Architecture of the combined WaferCaps+CNN proposed approach. 152
5.12 Confusion matrix for the validation dataset for different network setups. 153
5.13 Confusion matrix for the validation dataset for different network setups. 155
5.14 Classified vs. misclassified samples for proposed approach against other models. . 156
5.15 MNIST data sample. 157
5.16 WaferCaps proposed 157
5.17 Original images (left) and reconstructed images (right). 158
5.18 Manually augmented samples for defect and dirt anomalies. 159

List of Tables

2.1 Summary of quality monitoring approaches used in industrial inspection. 11

2.2 Summary of articles that used AOI system to investigate LED defects 16

2.3 Summary of articles that used AOI system to investigate FPD defects 18

2.4 Summary of articles that used AOI system to investigate PCB defects 20

2.5 Summary of articles that used semi-optical system (circuit probe) to produce WM 24

2.6 Summary of articles that used image acquisition optical system to inspect semiconductor wafers 25

2.7 Summary of articles that used AOI system to investigate miscellaneous electronic components defects 25

2.8 Summary for illumination systems used in AOI applications 27

2.9 Summary of articles that other image acquisition techniques and auxiliary systems 34

2.10 Inspection algorithm details for selected articles that investigated wafer defects . 66

2.11 Inspection algorithm details for selected articles that investigated PCB defects . . 67

2.12 Inspection algorithm details for selected articles that investigated FPD defects . . 68

2.13 Inspection algorithm details for selected articles that investigated LED defects . . 69

2.14 Inspection algorithm details for selected articles that investigated miscellaneous defects 69

3.1 Metrics for evaluating the validation dataset in percpetron. 85

3.2 Metrics for evaluating the validation dataset in MLP. 91

3.3 Suggested CNN network for MNIST classification. 97

3.4 Metrics for evaluating the validation dataset in CNN. 98

3.5 Metrics for evaluating the test data of mixed (DCGAN & Original) dataset for different deep learning models 118

4.1 Generator network 125
4.2 Discriminator network 125
4.3 Layers of the prposed WaferCaps 128
4.4 Overall training, validation and test accuracies using mixed dataset 138
4.5 Overall training, validation and test accuracies using original dataset 138
4.6 Metrics for evaluating the test data of mixed (DCGAN & Original) dataset for different deep learning models 138
4.7 Metrices for evaluating the test data of original only dataset for different deep learning models 139

5.1 Metrics for evaluating the validation data against different DL networks. 154

List of Algorithms

1 Back-propagation with gradient descent algorithm. 80
2 Routing algorithm. 116
3 Selection rules. 151

Chapter 1

Introduction

> "*Computers are able to see, hear and learn. Welcome to the future*"
>
> Dave Walters

1.1 Overview

This chapter introduces the use of deep learning (DL) techniques as a viable and efficient solution for performing automatic optical inspection on semiconductor and optoelectronic wafers. In addition, it introduces the reason of choosing DL techniques in this application and what makes it preferable than conventional optical inspection methods. The problem statement for this research is first defined, then the motivation for conducting it is introduced. Research context and the framework of conducting the research is then discussed. The challenges and questions of research is then defined and discussed. The aim and objectives are then set based on the motivation and problem statement. Finally, the work in each chapter is explained briefly.

1.2 Problem Statement

Product defects in industries cause severe drawbacks on the product competitive advantage. Furthermore, it increases costs and deteriorates manufacturing processes [1]. Early and accurate detection for these defects can help industries to conduct quality management and enhance product competitive advantage [2].

Rapid development in processors and image capturing devices in the recent years opened the eyes for various industries to use machine vision techniques for detecting defects as its considered one of the non-destructive testing techniques [3]. The main goal for using such techniques is to achieve 100% quality assurance of the finished products [4].

The purpose of this thesis is to explore how machine vision techniques can be applied and integrated with the latest artificial intelligence (AI) algorithms such as DL in order to identify defects and flaws in electronic products. In the investigation of solving this problem, the open source dataset WM-811K that contains different semiconductor wafer defect patterns will be used for analysis, as well as samples of optoelectronic wafers that are taken from an actual industry in Europe as a part of a project called iQonic (H2020 -DT-FOF-03-2018 – project No: 820673).

1.3 Research Motivation

Electronics is one of the fastest evolving, most innovative, and most competitive industries [5]. The past five years, have been characterized by growth in emerging markets and introduction of new products, leading more people to buy consumer electronics. The global consumer electronics industry was valued at $1 trillion in 2020 [6]. Production of electronics involves many manufacturing processes from automated assembly lines to testing and final manual assembly [7]. The electronics product sizes are getting larger for home display devices such as the LCD TV sets. Or, the product sizes are dramatically reduced for handheld or wearable devices such as smart phones and smart watches, as a result the need for an Automatic Optical Inspection (AOI) system to the various sizes of electronics products were not an exception from any other industry [8]. The challenges facing the electronics industry is much severe compared to others, as the electronics products and manufacturing plans evolve rapidly and the electronics products tend to have short life cycles [9]. These challenges require to change the process parameters accordingly. All of these changes (whether they are sudden or long-term) cause defective products, and therefore quality assurance and process improvement is very essential in electronics industry. According to Xie et al in [10], in electronic manufacturing industry, a defect is a condition that may be insufficient to ensure the form, fit or function of the end product.

One of the key building blocks of producing electronic devices is electronic wafer. Electronic wafer is a thin slice of semiconductor that consists of repeated dies (also called chips) as shown

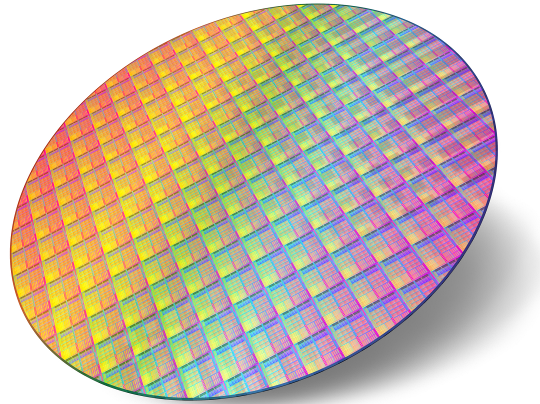


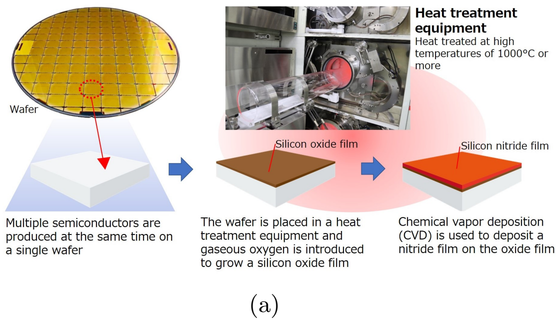
Figure 1.1: Electronic wafer

Figure 1.1. They can be made of different types of semiconductor materials such as crystalline silicon (c-Si) and Gallium arsenide (GaAs). The process of fabrication these wafers is a complex, long and costly as well as it requires monitoring a large number of key process parameters. It involves hundreds of chemical steps that must take place in a clean room environment such as oxidation 1.2(a), photolithography 1.2(b), etching 1.2(c), ion implementation 1.2(d), electrode layer formation (metallisation) 1.2(e) and slicing and packaging 1.2(f). Due to the complexity of the fabrication process, electronic wafers are prone to many types of defects and anomalies.

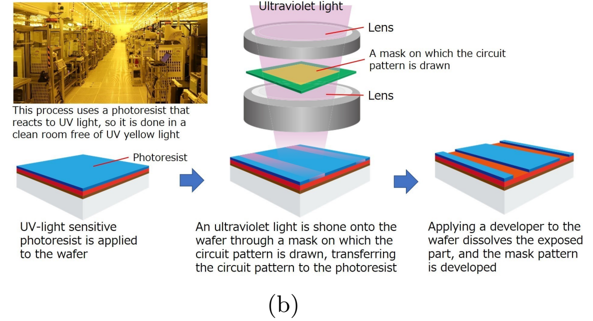
In this research, the attention will be focused on inspecting defects and flaws in electronics industry; mainly opto-electronic and semiconductor wafers by deploying computer vision and investigate the use of state-of-the-art DL techniques for potentially solving the problem.

1.4 Research Context

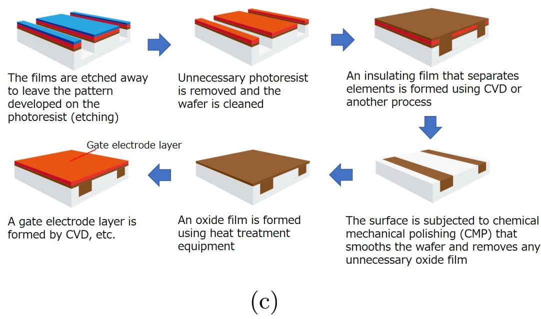
Identifying and localising objects is a vital aspects in many computer vision, pattern recognition and industrial applications [12]. If we take a closer look on the electronics industry for instance, a common application is to detect defects and flaws automatically using a digital image taken from a camera during the manufacturing process. Figure 1.3 shows three segments of an opt-electronic wafer, where two of them belongs to the same category and the other represent a defected sample. At the first glance one may think that Figures 1.3(b) and 1.3(c) belong to the same category (class), while Figure 1.3(a) belongs to different class. However, in these particular



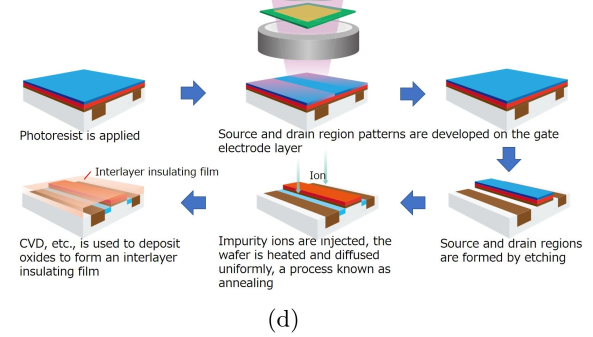
(a)



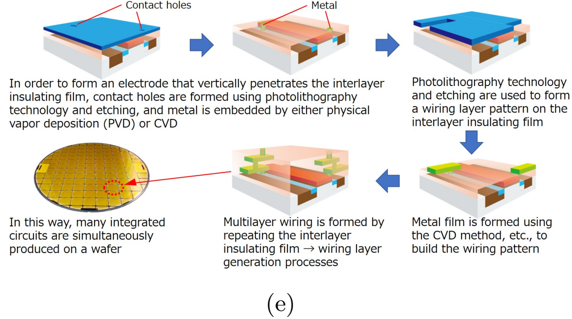
(b)



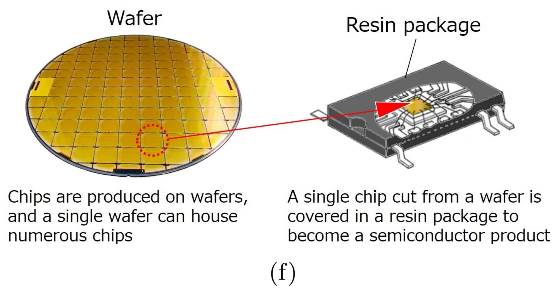
(c)



(d)



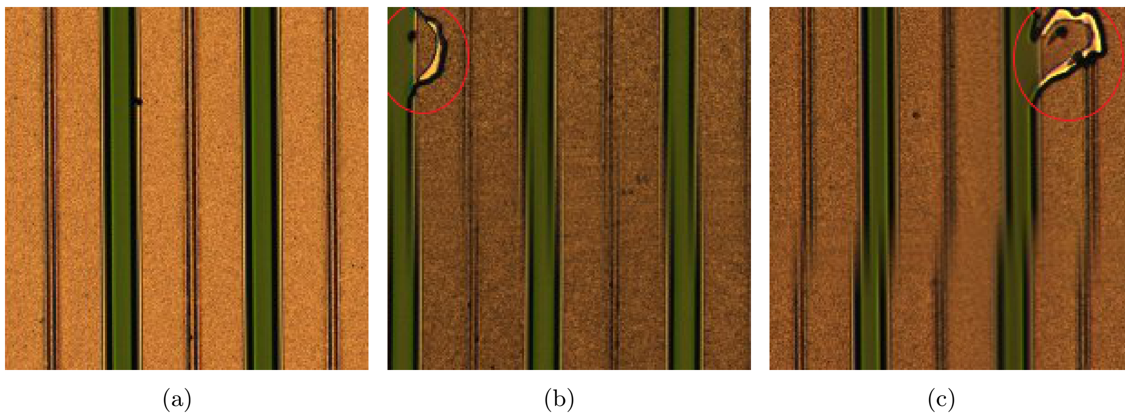
(e)



(f)

Figure 1.2: Semiconductor wafer fabrication processes [11].

segments, a defect is defined when an interruption occurs on the black lines in the middle (known as waveguides). Therefore, since there are no interruption in the waveguides in Figures 1.3(a) and 1.3(b), these segments are considered normal while the sample in Figure 1.3(c) is considered a defect. A conventional image processing tool such as template matching can easily identify the features circled in red; however, it is becomes much harder to classify Figures 1.3(a) and 1.3(b) to the same class. This difficulty arises when two images that belong to same class share similar features together. This does not necessarily apply to industrial application only, it can also applies in general computer vision applications such as distinguishing between two numbers that are written in similar way e.g. "3" from "8" or "4" from "9" as will explore in Chapter 3.



(a) (b) (c)

Figure 1.3: Segments of optoelectronic wafers

In electronics industry, hand-crafting feature extraction and customised image processing tools for real-time inspection purposes has been implemented for many electronic products as will be seen in Chapter 2. Unfortunately, these conventional techniques are done by trial and error which may take a huge amount of time. Furthermore, it requires an extensive expert knowledge to tailor the algorithm in a way to detect defects and flaws in the products. Moreover, conventional computer vision algorithm are usually rule-based, which means they obey the conditions of an if-else statement, a small variation or a false-alarm feature in a sample could produce misclassification errors.

Given all the previous considerations, it has now become a must to investigate alternative routs and solutions to overcome these limitations. The rise of computational power and graphical processing units (GPUs) brought the application of machine learning (ML), and in particular DL techniques to closer to solving actual and real manufacturing process applications and replace

conventional computer vision methods. DL networks allows to process the raw data and perform feature extraction automatically due to the multiple and complex layers that they have making it capable of learning low- mid- and high-level features for recognition [13, 14]. DL techniques are applied in many fields such as medical diagnostics, autonomous driving and industrial defect inspection. For the context of this thesis, various DL approaches will be investigated and implemented for better identification of defects and anomalies of semiconductor and optoelectronic wafers.

1.5 Research Challenges & Questions

Despite the remarkable advantages of DL techniques for industrial applications, it still cannot be considered as a walk-in-the-park method that will apply itself without requirements. A major challenge of a DL algorithm is the availability of large number of examples (data). This is because DL needs large amount of examples to update their weights within the hidden layers in order to learn the features of an image and classify it in the right class. A minor challenge would be the data labelling, which requires the expert knowledge in providing the label of each example (image). Fortunately, DL techniques can also perform data augmentation in order to increase the library of the dataset, and therefore increase the accuracy of classification and detection. One of the common DL algorithms to perform data augmentation is Generative Adversarial Network (GAN) as will be explored in Chapters 3 and 4.

The previous discussion will lead us to the gap of previous research works which initiates from the limitations concerning (i) data availability and (ii) classification of defect methods in semiconductor and optoelectronic wafer industries. In this research, these two points will be explored by providing state-of-the art solutions using latest technologies and DL techniques. Based on the limitations and research gaps outlined above, the research questions of this thesis are:

1. "*How to develop and efficient defect inspection technique according to the investigated defects?* "
2. "*How can we use technology and DL for automating the inspection process?*"

3. "*How to mitigate the challenges of using DL techniques such as the lack of data for training?*"

1.6 Research Aim & Objectives

The aim of this research is to propose framework architecture that automates the defect inspection process for semiconductor and optoelectronic wafer industries and solve the lack of data samples problem that exists in this type of architectures. To address the aim of this research, the following objectives are established:

1. To review the existing methods and techniques in literature that are used for defect assessment and inspection in electronics industry.

2. To explore different DL techniques that are used in computer vision and industrial inspection applications.

3. To investigate optoelectronic and semiconductor wafer anomalies and defect features using image samples.

4. To increase the image library dataset for the samples when necessary using automatic and manual image augmentation techniques.

5. To propose a customised novel DL architecture for classifying defect samples and patterns.

1.7 Thesis Structure

The chapters in this thesis are structured as follows:

- Chapter 2 provides an in-depth review of automatic optical inspection methods in electronics industry. The chapter starts by comparing different quality motoring methods with AOI. It also reviews the hardware and software requirements for establishing an efficient AOI systems. Finally, it reviews the latest DL algorithms used to aid the AOI techniques.

- Chapter 3 discusses the background theory behind DL networks. The discussion starts by the basic building block of any DL algorithm (the perceptron) until we reach more complex

and sophisticated architectures (e.g. capsule networks) that can automatically capture the features of the target image. The chapter also discusses about GAN which considered a powerful tool in conducting data augmentation. In this chapter, the well-known MNIST dataset was used to evaluate the performance of each architecture.

- Chapter 4 introduces the open source dataset for semiconductor wafer maps WM-811K. The analysis starts by performing data augmentation using DCGAN to upsample and increase the dataset for classification. Then a novel and improved cpasule network called WaferCaps is proposed to classify the defect patterns in the dataset.

- Chapter 5 deals with optoelctronics wafer image samples that are collected from iQonic industrial partner. Our analysis focused on the waveguide anomalies in the quantum cascade lasers in the wafer. Two type of anomalies were investigated, namely, dirt and defect. Manual and automatic augmentation techniques were used in this chapter to increase the dataset. A combination of WaferCaps and CNN were also proposed to classify the samples.

- Chapter 6 discusses the contributions of this research. It also discusses the limitations of the present study and possible future work.

Chapter 2

Literature Review [1]

"If I have seen further than others, it is by standing upon the shoulders of giants"

Isaac Newton

2.1 Overview

The previous chapter presented a detailed view of the research background and identified the research problem, motivation, context, challenges and objectives. This chapter reviews relevant researches in the area of computer vision in general in the domain of optical inspection in electronics industry. Section 2.2 introduces quality monitoring methods in general and nondestructive techniques in performing industrial inspection. In Section 2.3 the discussion is narrowed down on the use of automatic optical inspection and compare it with conventional manual optical inspection in industry, it also explains why automating the process can be more efficient, accurate and cost effective for industrial use. In Section 2.4 the discussion is narrowed down by considering AOI in electronics industry in terms of inspected defects, hardware setup and the use of auxiliary systems. Conventional inspection algorithms is then discussed in Section 2.5. Since too many papers was found in literature that conducted research on electronics component, the discussion in this section was narrowed down to inspection algorithms for semiconductor wafer inspection. Finally, Section 2.6 discusses the use of DL in AOI of semiconductor wafers in literature. Figure

[1] The contents of this chapter were published in the following article: **A. M. Abu Ebayyeh** and A. Mousavi, "A Review and Analysis of Automatic Optical Inspection and Quality Monitoring Methods in Electronics Industry", *IEEE Access*, vol. 8, pp. 183192–183271, 2020. doi: 10.1109/ACCESS.2020.3029127

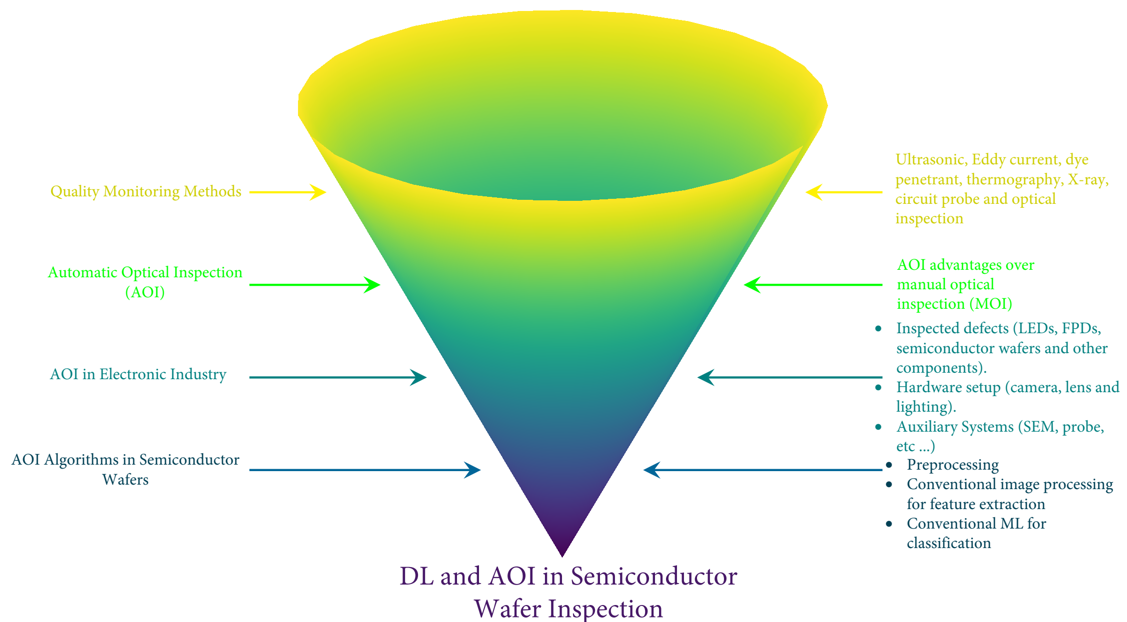


Figure 2.1: Literature review structure.

2.1 summarises the literature review structure of this chapter.

2.2 Quality Monitoring Methods

One of the primary goals of manufacturers is to ensure product quality, that is, to ensure that the product can perform its intended function for an extended period of time. Ideally, a product must be free of any defects or damages to meet this criterion. Where a defect is defined as an imperfection in a product that can restrict its life by impeding its performance, eventually causing it to fail, even during operation under ordinary conditions [15]. Several quality monitoring approaches can be used to inspect products. These methods can be divided into two categories, destructive and non-destructive. Nondestructive techniques (NDT) are found to be more appropriate in assessing the defects since they do not alter the original attributes or harm the object being tested. Therefore, they provide cost effective and efficient quality monitoring system [16]. Ultrasonic testing, Eddy current, dye penetrant testing, theromography, X-ray, circuit probe testing and optical inspection are examples of different NDT used for industrial inspection. Table 2.1 provide a description for each of these methods as well as lists of their advantages and limitations. Among these techniques, optical inspection approach for defect detection is one of the most common procedures used in industry [3].

Table 2.1: Summary of quality monitoring approaches used in industrial inspection.

Quality Monitoring Approach	Description	Advantages	Limitations
Ultrasonic Testing	A typical ultrasonic-based inspection system consists of several functional units such as pulser, receiver, transmitter, and display device. Ultrasonic inspection uses the electrical voltage generated by the pulser to activate the transmitter. Transmitter generates sound signals at frequencies beyond human hearing (more than 20 kHz), to estimate some properties of the irradiated inspected component by analyzing either the transmitted or received signals detected by the receiver [17].	Speed of scan. Good resolution and flaw detecting capabilities. Suitable of use in the field and assembly line where the same part design must test repeatedly [18].	It is necessary to align signals that relate to materials of different thicknesses. High skill is needed to scan a part accurately. Need of test sample to insure accurate testing [17, 18].
Eddy Current	A typical Eddy current inspection system consists of magnetic coil that induce magnetic field and electromagnetic sensors. The interaction between the subjected magnetic field and the component under inspection induces eddy current which can be measured using electromagnetic sensors [19, 20].	Highly sensitive to material conductivity. Very simple and easy to implement. Inspection can be implemented without any direct physical contact between the sensor and the inspected component.	Electric signal obtained is low; thus, a high-performance measurement technique is required to improve signal-to-noise ratio. Scanning speed has to be improved when considering PCB testing. Inspection is limited to conductive materials (ferromagnetic or non-ferromagnetic) [19, 21].
Dye Penetrant Testing	coloured fluid penetrant is applied to the surface being inspected to detect any discontinues. Light source is usually used by inspector to highlight the defective features on the surface being inspected.	Suitable to reveal surface defects such as cracks. Considered simple and low-cost method.	This method was a subject of research till the 90's but no further significant developments have been published there after. The capillarity and the viscosity of the fluid used for inspection can affect the accuracy of this method [22]. Requires human inspector interference.
Thermography	Thermal sensor is used to measure the infrared radiation from the inspected component and convert the radiation flux to temperature. The temperature distribution can be then illustrated in a form of thermal images.	Suitable for surface and inner inspections. Suitable for detecting large voids and crack defects.	Thermal noise can affect the accuracy of results. The ability of detecting inner defects is limited to certain inspection range. Overheating problems may occur [23].
X-ray	A typical X-ray inspection system consists of three components: X-ray source (tube), X-ray detector and a fixture to hold and control the position of the inspected component. Transmission of X-ray by a source through an object and a receiver receives the transmitted energy. From the transmitted energy, the internal condition of the inspected component can be assessed.	Suitable for surface and inner inspections. Suitable for detecting large voids and crack defects.	Conventional X-ray methods can be destructive. Relatively low resolution in micro level inspection. Long processing time (in the order of hours) [23]. High cost.
Circuit Probe testing	Widely used for semiconductor wafers inspection by establishing a temporary electrical contact between test equipment and each individual die on a wafer to determine the state of a die (defected or non-defected). The major components of a circuit probe testing system include probes, probe cards, probe stations, and test equipment.	Circuit probe testing procedures are the simplest and most direct method of testing unencapsulated bad dies in semiconductor wafers.	Considered destructive testing. Probing errors may arise which cause good dies in the semiconductor wafer to be faulty [24].
Optical Inspection	Detects surface flaws and defects according human's visual perception. Optical inspection is performed using human inspector (manual inspection) or image sensor and processor (automatic inspection). A typical automatic optical inspection system is shown in Figure 2.2.	Most basic, low-cost and frequently used type among all quality monitoring approaches. Considered non-contact and non-destructive. Can detect surface defects and flaws. Automatic inspection can save time and enhances detection's accuracy.	Not efficient for inner defects. Manual optical inspection considered costly and can cause fatigue for human inspector.

2.3 Automatic Optical Inspection

Optical inspection is the process of determining if a product deviates from given set of specifications or standards visually. It usually involves an assessment of a specific part or feature in the product such as surface finish, geometric dimension, colour deviations and unusual patterns [25]. Optical inspection techniques can be subdivided into manual optical inspection (MOI), which is performed by a human inspector, and automatic optical inspection (AOI) which is performed by a machine vision system (image sensor and processor). Compared to MOI, AOI techniques are preferable in industrial inspection for several reasons. First, the cost of inspection can be reduced by saving training and human labour cost. Secondly, human inspectors are subject to fatigue, dull and routine [26]. Thirdly, accuracy and speed of inspection can be increased as modern capturing devices are capable of detecting tiny features with low intensity and contrast that even the most experienced inspectors cannot detect them by using their naked eye. Recent studies have also showed that humans cannot handle identifying colour defects as colour is psychological perception although it is triggered by physical radiation. Therefore, there is no guarantee that the colours perceived by different workers are the same [27, 28]. Fourthly, AOI can replace human inspector in unfavourable and/or unsafe environments such as nuclear plants and small workstations. Lastly, AOI systems can also help in gathering statistical information for the defected products in order to provide necessary feedback for further quality enhancement and control in the future [4, 29, 30]. Ideally, an AOI system must satisfy two major requirements, 100% detection rate for defects and 0% false alarm rates [31].

According to Chin Harlow in [29], a standard AOI system consists of image sensor (e.g. camera), lightning setup (illumination), computer (processor), conveyor and sorting mechanism as shown in Figure 2.2. The image sensor inside a camera is responsible in acquiring the image. The illumination is necessary for providing constant or customised lighting conditions. The computer or processor applies the inspection algorithm that is usually follow three stages, preprocessing, feature extraction and classification. According to the previous steps, a decision should be taken to consider or disqualify the product being inspected. The sorting mechanism includes conveyor and programmable logic controller (PLC) to separate the unwanted products from the qualified ones.

The nature of the inspected product plays a vital role in choosing the right image acquisition

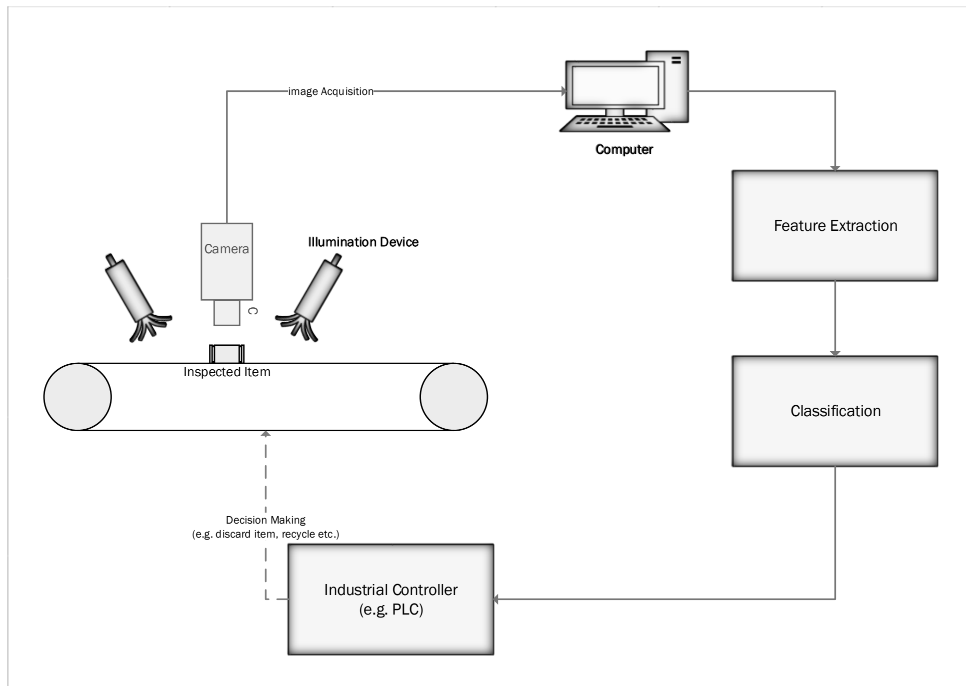


Figure 2.2: AOI system

setup, whether it is hardware (camera, illumination) or software (inspection algorithm). Therefore, in the next section we will mention different electronic products in literature that were inspected using AOI system.

2.4 AOI in Electronics Industry

In literature, it was found that the AOI has been mainly conducted to four electronic products, light emitting diodes (LEDs), semiconductor wafers, printed circuit boards (PCBs) and flat panel displays (FPDs) (e.g. LCD and OLED). It was also found that some literature considered AOI for miscellaneous products such as camera modules, fuses, and passive electronic components [32]. Figure 2.3 summarises the research article (from 2000 to 2019) that used AOI for different electronic components according to the country. The highest majority (around 32.98%) of the reviewed articles investigated semiconductor wafer defects, 29.43% for FPDs, 26.2% for PCBs, 6.74% for LEDs and 4.26% for miscellaneous electronic component defects. According to Figure 2.3, the highest concentration for the AOI research articles in electronics industry was from Taiwan with a percentage of 41.84%, followed by China with 22.7%, and South Korea with 12.8%. That result was expected since most of the world's exports of electronic components

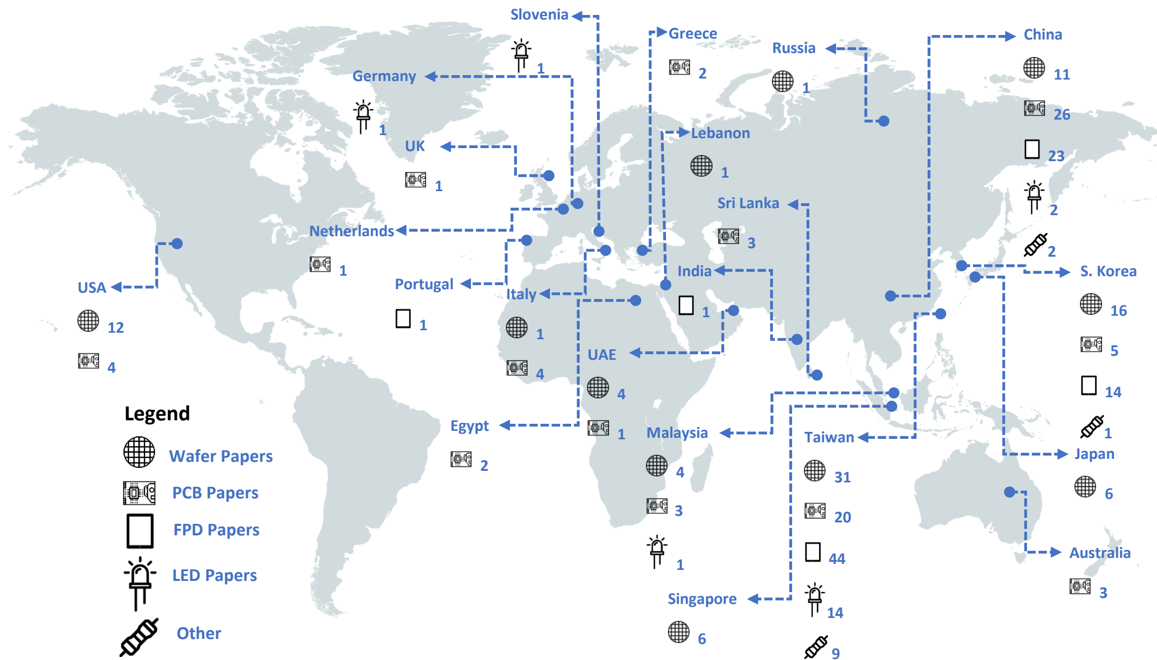


Figure 2.3: A map that shows the literature (2000 - 2019) according to the country and product considered [32].

and products are produced in Asia. Moreover, Asia is the major end market for many of these products [33]. For instance, in 2018, 40% of all personal computers sold worldwide, were sold only in China. According to the same source [33], electronic exports in Taiwan alone accounts 45% of the country's total exports, while China and South Korea account 35% each.

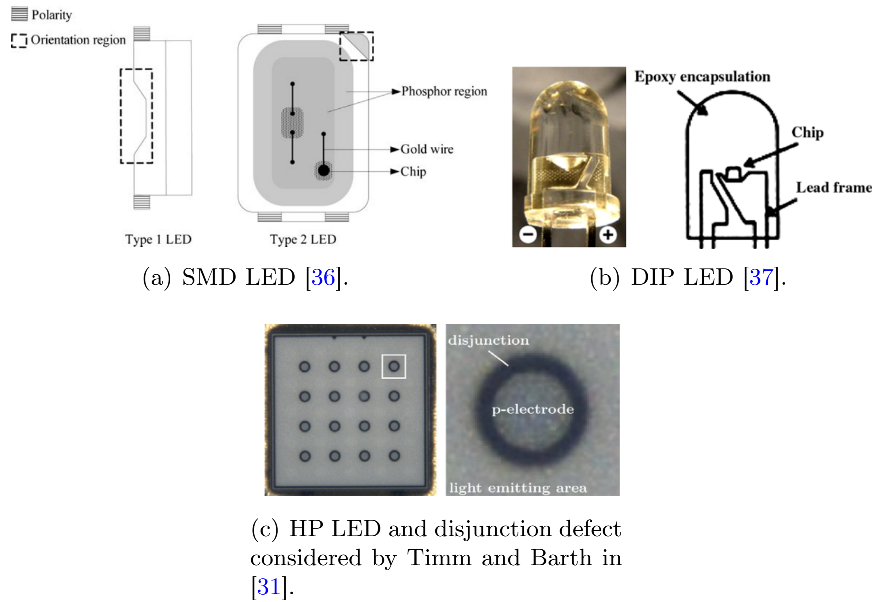
In the next sections, the investigated articles will be reviewed with more details according to the product type and defects considered.

2.4.1 LED Defects

According to the application there are various types of LEDs such as surface-mounted device (SMD) LEDs (Figure 2.4(a)), dual-in-line package (DIP) LED (Figure 2.4(b)), and High Power (HP) LEDs (also called chip-on-board COB LEDs) (Figure 2.4(c)). To meet consumer and industry needs, LED products are being made in smaller sizes, which increase the difficulties of product inspection [34]. Mainly there are two types of inspection, electrical and optical. The electrical inspection ensures correct functionality, but since an extensive stress test cannot be applied for all LEDs, defects that might cause malfunction after a period of time cannot be detected accurately [31]. Therefore, many researchers have conducted studies to detect the

different defects of LED types using AOI. In some studies such as in [35], AOI was used to evaluate physical properties of the LED light such as intensity, mean colour, colour variation, viewing angle and divergence of the optical axis by analysing the projection of the LED light.

Table 2.2 summarises the LED defects studied in literature according to the LED type and part.



(a) SMD LED [36].

(b) DIP LED [37].

(c) HP LED and disjunction defect considered by Timm and Barth in [31].

Figure 2.4: Different LED types.

2.4.2 FPD Defects

Flat Panel Displays (FPD) including liquid crystal display, organic LED (OLED), active-matrix OLED (AMOLED), and polymer LED (PLED) are widely used in various applications ranging from smartphones, tablets, computer monitors, televisions (TVs), to data projectors [32]. The recent and complex technologies used in fabrication of FPD displays (e.g. thin-film transistor (TFT)) made it possible for the products to be subjected for many defect types. Many of the recent articles investigated defects in TFT layers technology used in LCD fabrication (see Table 2.3). Mura (derived from Japanese which means blemish) defect is one of the common defect types investigated among these studies. It is defined as a local lightness variation on a surface without clear contours and causes an unpleasant sensation to the human vision. Mura assessment is performed by observing any imperfections present visually on the scale of a few pixels to usually less than 20% of the screen [53]. Figure 2.5 show different types of Mura defects.

Table 2.2: Summary of articles that used AOI system to investigate LED defects

LED Part	Reference	Inspected Defects
Exterior parts	[38]	Defects occur on the epoxy dome of the DIP LED such as contamination, scratches and blister/blemish and fuzzy defects.
	[39]	Tiny surface flaws (flaws that occupies 0.0015 – 0.0229% of the total surface area) on the transparent epoxy dome-shape encapsulation of DIP LEDs
	[34]	Blemish defects in curved LED lenses that occur on the external surface of the lens
	[40]	Aperture defects in LED bulb cups
SMD LED	[36] [41]	Two types of LEDs were studied, and the defects investigated are: missing component, wrong orientation, inverse polarity, mouse bites and surface defects
	[42]	Missing component, no chip, wire shift/defect and foreign material presence
LED chips	[43] [44]	Inspection of LED die defects geometrically (in terms of die size, electrode size) and heuristically in terms of other defect types in light-emitting regions and electrode region such as empty points in light-emitting region
	[37] [45]	Water-drop defects that causes blemishes on LED chip surface
	[46]	Overall inspection of LED dies in a wafer in terms of the quality of light-emitted area, N-electrode, P-electrode and probe marks
	[46]	Overall inspection of LED dies in a wafer in terms of the quality of light-emitted area, N-electrode, P-electrode and probe marks
	[31]	Discontinuity and erosion defects located at the area of p-electrodes and disjunction region of LED die.
	[47]	LED chip defects that include: fragment chips, scratch marks and remained gold on the pad area, scratch marks on the luminous zone and missing luminous zone
	[48]	LED wafer luminance test
	[49]	Polycrystalline and fragmentary defects
	[50]	Electrode area defects such as contamination, scrap, and non-probe defects. Light area defects such as breakdown and colour aberration defects
	[51]	Line blemishes and scratch marks in two types of chips
LED optical properties	[35]	Intensity, mean colour, colour variation, viewing angle and divergence of the optical axis.
	[52]	Luminance and forward voltage inspection

Figure 2.5: Mura defect types

Other parts such as glass substrates, colour filters and backlight modules have been also studied in literature. Table 2.3 summarises research articles that used AOI in detecting FPD defects.

2.4.3 PCB Defects

Electric inspection and MOI are still used in PCB evaluation; however, introducing surface mount technology (SMT) to the PCB assembly process made the PCB circuits much finer and more complex, which highlights the convenience for using an AOI system [125]. Moreover, around 80% of the optically recognisable defects could not be recognised electrically [126].

The main steps in manufacturing PCBs are the following: applying solder paste to PCB, placing integrated circuit (IC) on the board at correct positions, and placing the board in an oven in order to solder the ICs (or other components) to pads [127]. During applying these steps, several kinds of defects may occur at each stage, most of these defects take place at soldering and component placement stages as shown in the pie chart of Figure 2.6.

For instance, at the IC placing stage, defect types of missing, wrong or doubled components may occur. On the other hand , most of the soldering defects occur applying solder paste, such as defects at the non-IC components (raised components, tombstoned components, pseudo joint, insufficient solder, excess solder, shifting, solder bridge, side termination) and defects at the IC package components (pseudo joint, insufficient solder, excess solder, shifting Solder and bridge defects) [129]. These defects can severely damage the functionality of the PCB. For example,

Table 2.3: Summary of articles that used AOI system to investigate FPD defects

FPD Type	Reference	Inspected Defects
LCD	[54–69]	Mura defects
	[70–75]	TFT-LCD panel defects
	[76–78]	TFT-LCD panel micro-defects such as pinholes, scratches, particles and fingerprints
	[8, 79–81]	Polarising film defects
	[82–84]	Glass substates defects in TFT-LCD
	[85]	Defects during photolithography process
	[86–90]	Backlight defects
	[91–94]	Gate electrode operation defects during TFT array process
	[95, 96]	Source and drain operation defects during TFT array process
	[97, 98]	colour filter defects
	[99]	TFT array defects such as fibre defect, particle defect, pattern damage, pattern residual and pattern scratch
	[100–103]	Anisotropic Conductive Film defects
	[104]	Optical thin film defects
	[105]	TFT-LCD pad area defects
	[106]	LCD surface deformation for smartphones
	[107, 108, 108–110]	Polariser transparent microdefect
	[111]	Liquid resin defects
	[112]	Subpixel (dots) functional defects
OLED	[113, 114]	Directional textured surface defects in OLED and PLED
	[115]	Ambiguous surface defects of AMLOED such as scratch, long dust, circle dust, pit and stain
	[116]	Macro defects such as film tear and pit and micro defects such as scratch and spot
	[117]	Salt-and-pepper defects
Mobile and Touch Panels	[118–122]	Mobile screen defects
	[123, 124]	Touch panel defects
All Types	[28]	Chromatic (colour) defects
	[61]	Mura defects

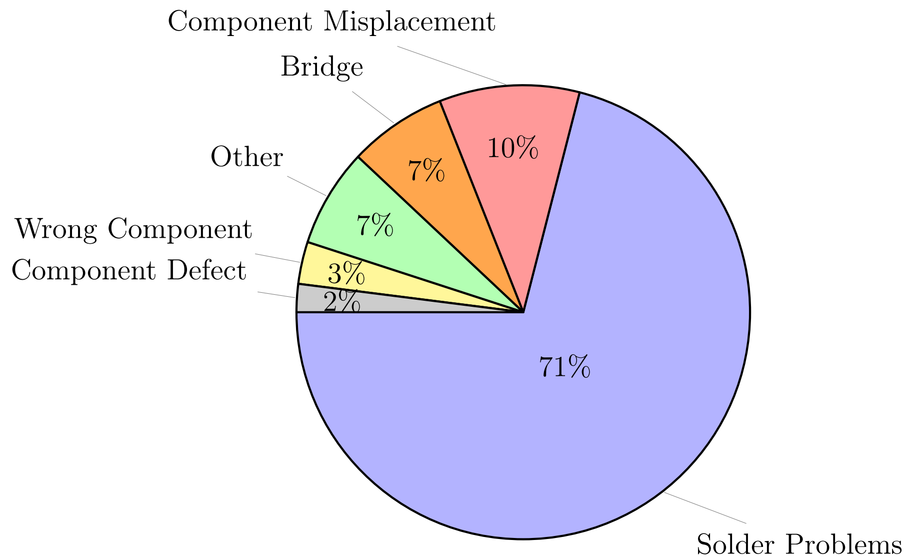
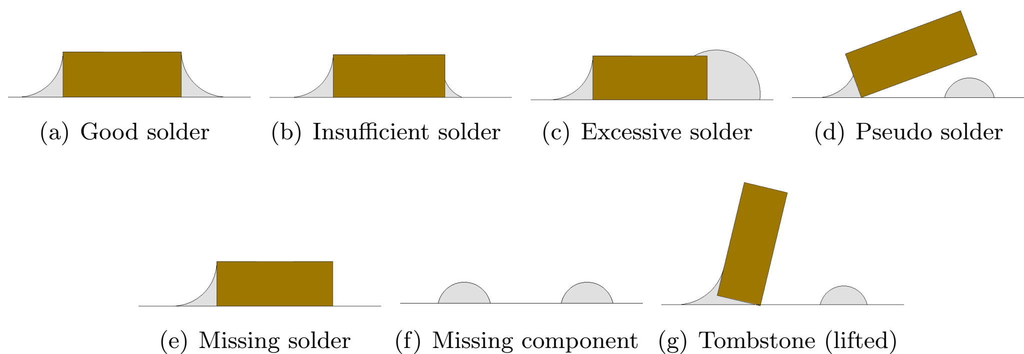


Figure 2.6: The frequency of PCB-related defects [128].

an excess solder joint can cause bridging with other PCB solder joints which can lead to a short circuit. A missing solder joint or insufficient solder joint can cause an open circuit of the PCB and thus the overall functionality of the circuit will be affected. Pseudo solder (also known as cold solder) is considered a complex defect for detection [130]; this defect occurs when the solder joint terminals are seemed to be connected with the electronic component; however, there is no physical connection occurring which may lead to open circuit contact. A pseudo joint is formed when insufficient heat is applied to completely melt the solder [131]. Figure 2.7 shows commonly investigated defects in PCB industry.



(a) Good solder (b) Insufficient solder (c) Excessive solder (d) Pseudo solder

(e) Missing solder (f) Missing component (g) Tombstone (lifted)

Figure 2.7: Commonly investigated PCB defects

Inspecting the IC marking is also important and it is one of the topics studied in literature. A typical industrial version of an inspection system has to check about 7,300 – 7,500 IC chips

per hour, which justifies the need for an AOI system [132]. In IC marking inspection, incorrect direction or marking will lead to incorrect placement of an IC on a PCB [133]. The inspection has to identify print errors such as illegible characters, missing characters and upside down printing which frequently occur due to the malfunction of the machinery [132, 134]. Table 2.4 summarises research articles that used AOI system for defect detection according to the PCB part and defect type.

Table 2.4: Summary of articles that used AOI system to investigate PCB defects

PCB Part	Defect Type	Reference
Solder Joint	Missing Solder	[130, 135–140]
	Insufficient or open solder	[126, 135–149]
	Excess or bridged solder	[126, 130, 135–147, 150]
	Pseudo Solder (cold solder)	[130, 136, 137, 140, 148, 149]
	Solder Joint Overall Quality	[151–154]
	Solder Paste Inspection	[125, 155, 156]
Electronic Components	Missing Component	[127, 130, 140, 157–159]
	Wrong Component	[130, 148, 149, 158]
	Shifted or rotated component	[130, 136, 137, 140, 148, 149, 157, 158]
	Component lifted (tombstone)	[130, 136, 140, 148, 149, 158]
	Component misplacement	[160–164]
	IC molding surface	[165]
	Electronic Component Overall Quality	[8, 127, 166]
PCB Holes	Via holes	[150, 167]
	Microdrill bits	[168, 169]
Other	Golden fingers	[170–172]
	Traces & Inner layers	[173–176]
	Marking inspection	[132–134, 159]
	Cosmetic & small defects	[10, 176]
	Scratches and improper etching	[177]
	Glue quality	[178]
	Flux Cutting	[159]
	Alignment and position inspection of PCB	[8, 179]
	PCB Overall quality	[166, 180–182]
	Ball Grid Array (BGA)	[166, 183–193]

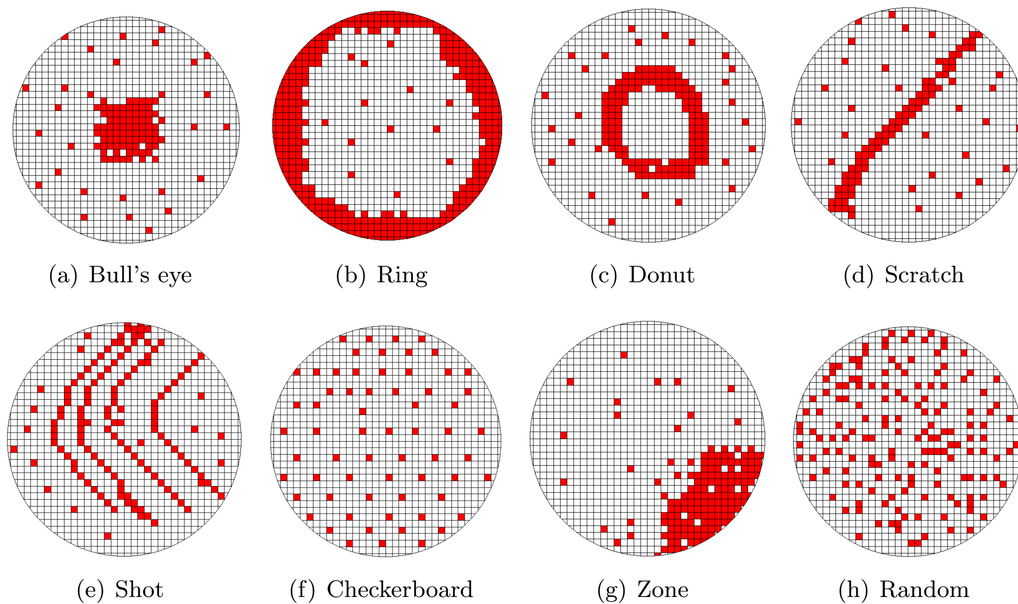
2.4.4 Semiconductor Wafer Defects

Advances in semiconductor technology and design have been the driving forces behind the successful progress of electronics industry [194]. The majority of ICs – that are used in microelectronic devices – are manufactured using semiconductor wafers on their surface [195]. Semicon-

ductor wafers consist of repeated dies which have the same structure and results in large numbers of ICs and devices. Therefore, the process of fabrication is a complex, long and costly which involves hundreds of chemical steps such as oxidation, photolithography, etching, ion implementation, and metallisation and requires monitoring a large number of key process parameters [196, 197]. After the fabrication process, wafer testing on each fabricated die is performed using test equipment called "wafer probe" to verify whether all dies meet the product specifications and ensure that only good dies are sent to the next manufacturing process [196]. To visualise the defective dies in a wafer, a Wafer Map (WM) is created. WM allows to compare neighbour dies with each other and locate how many dies are defected visually using a map image [198]. WMs are also called Wafer Bin Map (WBM) when the dies are represented in binary form such as the defective dies has the logic '1' and the non-defects has the logic '0'.

Wafer defects cannot be avoided even when modern highly automated and precise equipment in a nearly dust- free clean room are used. Defects do not only decrease yield performance but also generate reliability problems. Therefore, the immediate identification of the root cause of any defect is an important task [199]. Normally three types of wafer defects occur; Random, Systematic, and Mixed defects. Random defects result from random manufacturing environmental factors, such as particles in the clean room; these can become randomly scattered all over the wafer (Figure 2.8(h)). Such defects are long-term and expensive to correct. On the other hand, systematic defects are normally generated by an assignable cause such as a human mistake, particles from equipment, or chemical staining [199–201]. Systematic defects usually follow a specific pattern, for instance, a centre pattern (also known as Bull's eye) concentrated in the centre of a wafer typically occurs when there are uniformity variations caused by a chemical mechanical process (CMP), the distance between the wafer centre and the centre of the region is less than four dies (Figure 2.8(a)). A ring pattern appears along the wafer edge when there is a layer-to-layer misalignment in the storage-node process, which results in a write recovery time failure that is due to the decreased size of the contact holes, most of the failed dies occur along the wafer edge and encompass more than four-fifths of the wafer perimeter (Figure 2.8(b)). A scratch pattern (also known as line pattern) is caused by agglomerated particles and the hardening of the pad during the CMP, where most of the failed dies on the wafer form a line, and the length of the line is five or more dies (Figure 2.8(d)). A shot pattern is caused

by a probe-card problem when multiple dies are simultaneously tested to reduce the test cost (Figure 2.8(e)). A ckeckerboard pattern is generated because of the mask misalignment during the lithographic process (Figure 2.8(f)). A zone pattern (also known as blob, spot, or cluster) at a specific location on the wafer is caused by non-uniformity or uneven cleaning, where most of the failed dies on the wafer occur as an arbitrary shape, and the distance between the wafer centre and the centre of the region is more than four dies (Figure 2.8(g)). Mixed defects consists of a random defect and a systematic defects in one WM. Most WMs are of this type. Hence, it is important to separate random and systematic defects in the WBM since the systematic defect's signature can reveal the process problem. Therefore, it is important to detect and classify these defects, in order to identify the root causes of failure and to take appropriate actions for quality and yield enhancement [196, 199, 200, 202].



(a) Bull's eye (b) Ring (c) Donut (d) Scratch

(e) Shot (f) Checkerboard (g) Zone (h) Random

Figure 2.8: Different WM patterns (red squares indicate defective dies)

Even though, after generating the WM the defective dies can be detected by an experienced human inspector; however, as this process can be costly and time consuming, as a result many researchers evaluated the dies' quality using AOI techniques [203]. Most of the AOI techniques used in semiconductor wafers can be considered semi-optical since most of them do not require establishing hardware optical system (e.g. CCD device and illumination setup). Hence, the WM image is constructed from the circuit probe scanning. However, some paper still considered

conducting a hardware optical system for the inspection purposes of semiconductor wafers. In order to distinguish between both used methods, research articles that used circuit probe to produce the WM are labelled as "semi-optical" in Table 2.5. On the other hand, research articles that used optical inspection setup are labelled as "optical" in Table 2.6.

2.4.5 Miscellaneous Defects

The previous sections discussed about common inspected components in literature in a categorised manner. Nevertheless, some articles have investigated other electronic component defects such as camera modules, passive components and thermal fuses. Table 2.7 summarises miscellaneous defects that were investigated using AOI techniques.

2.4.6 Hardware Setup

Choosing the appropriate image acquisition system including camera, illumination and auxiliary setup will highly depend on the component being inspected as stated in the next sections. In this Section different hardware setups that used in literature for electronic components inspection are reviewed.

2.4.6.1 Contrast and Illumination Setup

Contrast is one of the key factors that contributes to image quality, it defines the variations in intensity values between the inspected feature and the background [288]. Therefore, it is very important to select background that has unique intensity levels, so that the inspected components can be distinguished when applying the inspection algorithm. Contrast and other factors that contribute to image quality can be also influenced by illumination and lightning settings, since the camera does not see the object; it sees the light reflected by the object. A good illumination system can reduce shadow, noise, and reflection and increase image contrast, as a result shortening the image processing time and increasing the accuracy of inspection [289]. In AOI applications, the environment light (e.g. sunlight) is avoided since it changes with the change of environmental conditions and this can affect the image quality and thus the detection algorithm. Therefore, non-varying illumination sources are used in such applications [290]. Inspected component size, colour, surface feature, geometry, material, inspection environment and system

Table 2.5: Summary of articles that used semi-optical system (circuit probe) to produce WM

Inspection Type	Reference	Defect / Pattern Type considered
Semi-optical	[204]	Ring, scratch, zone and repeating types
	[205]	Ring, scratch, random and new patterns
	[206]	Systematic and random patterns
	[207]	Circle, cluster, scratch and spots
	[208] [209]	Bull's Eye, Edge ring, scratch, random, multiple zones, multiple scratches, ring-zone mixed pattern and ring-scratch mixed pattern
	[210] [211]	Multiple zones, multiple scratches, ring-zone mixed pattern and ring-scratch mixed pattern
	[212] [213]	Cluster defects such as scratch, strains and localised failures
	[200]	Checkerboard, ring, right-down edge, composite and random patterns
	[214]	Spatially homogeneous Bernoulli process, cluster, circle, spot, repetitive and mixed pattern
	[215]	Scratch, center and edge
	[216]	Quarter ring, up and left, Quarter ring, up and right, Edge effects, Ring effects, Semi-ring, up, Semi-ring, up Edge effects, up and bottom Cluster
	[217]	Annulus, half-annulus, band and half-ring
	[218–222]	Curvilinear, amorphous, and ring
	[223]	Linear and circular patterns
	[224] [225]	Bull's eye, Bottom, Crescent moon, edge and random
	[226]	Random, ring, curvilinear and ellipsoid
	[199]	Line, edge, ring, blob and bull's eye
	[202] [194]	Bull's eye, blob, line, edge, hat and ring
	[227]	Multiple patterns including ring, checkerboard and five radial zones
	[228]	Random, systematic and ,mixed patterns
	[198] [229]	Circle, cluster, repetitive and spot
	[197, 201, 203, 230–235]	Center, donut, edge-local, edge-ring, local, random, scratch, near-full and non-pattern
	[236]	Center, edge, repeated scratch, C-shape and donut, center+edge and mask+local
	[237]	Rings, semicircles, clusters, and scratches
	[238]	One-side, center, and edge
	[239]	Random patterns
	[240]	DRAM wafer failures
	[241] [242]	Random, cluster, circle, and repetitive
	[243] [244]	General patterns
	[196] [245]	Circle, ring, scratch, shot and zone
	[246]	Systematic patterns
	[247] [248]	Random, edge ring, line scratch, curved scratch, non-random cluster, gross defect at entire wafer and each half.
	[249]	Circle, spot, cluster, scratch, circle-spot, cluster-scratch and circle-scratch
	[250]	Donut, moon, reticle, scratch, center, and edge

Table 2.6: Summary of articles that used image acquisition optical system to inspect semiconductor wafers

Inspection Type	**Reference**	**Defect / Pattern Type considered**
Optical	[251–259]	Overall quality inspection of wafer
	[260]	IC wafer contamination
	[261]	Micropipes defects
	[262–264]	Chip-out, bridging, metal lifting, glassiviation and peel off
	[265]	Wafer topside scratch, foreign material, ink residue, pad damage, passivation/metal damage, ink smeary, and passivation covering
	[266]	Pinhole defects
	[267]	Protrusion, dent, flat and bumpy defects
	[268]	Hole, Protruding and flat patterns
	[269]	Particles, contamination and scratches
	[270]	Defects between line edges
	[271]	Hole, flaw and scratch defects
	[272]	Alignment, probe marks and bump defects for in-tray semiconductor chip
	[273]	Spots, scratches, and bruises
	[274]	Bond pad discolouration
	[195]	Die edge, die street and determination of chipping size and shape
	[275]	Spot, rock-shaped particle, ring-shaped particle, misalignment and scratch
	[276]	Defects are classified as small, medium and large

Table 2.7: Summary of articles that used AOI system to investigate miscellaneous electronic components defects

Component	**Reference**	**Inspected Defects**
Camera Modules	[277]	Black and white defect, dim defect, colour defect, and line defects in manufacturing process of CMOS compact camera module
	[278, 279]	Six major defects: solid white dot, gray dots, black spots halo, strip defect, bubble defect, and solid black spots. Minor defects were also included in separate category.
	[280]	IR-CUT filter defects such as stain, scratch, and edge crack
	[281]	Surface defects in micro multi-layer non-spherical lens module of CMOS such as bright spot, dark spot, scratch, foreign material and hole
	[282]	Compact camera lens and spacer ring defects such as stain, bright dot, scratch, pit, and scar
Passive Components	[283, 284]	Ripple defects in the surface barrier Layer chips of ceramic capacitors
	[285]	Tiny surface defects in the surface barrier Layer chips of passive electronic components
	[286]	Surface defects of film capacitors
Thermal Fuse	[287]	Bur, black dot, small-head, and flake defects

needs are all important factors to be considered in selecting the right illumination source. LED, fluorescent lights and quartz halogen light with fibre optics are commonly used as illumination sources for AOI applications. Moreover, positioning of the illumination source plays vital role as well especially when inspecting polished and shiny objects (e.g. metal parts and solder joints) that reflects light with specular reflections. On the other hand, dull surfaces such as plastic diffuse light in several directions [291]. Therefore, the positioning of the illumination source highly depends on the applications as well [289]. Table 2.8 and Figures 2.9 - 2.17 show the different illumination settings used in litreature.

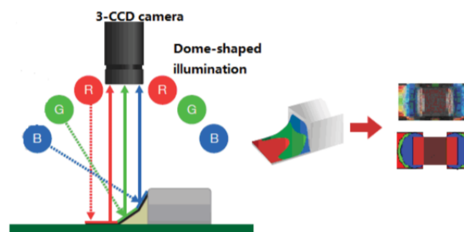


Figure 2.9: Illumination setup considered by Wu et al. in [148]

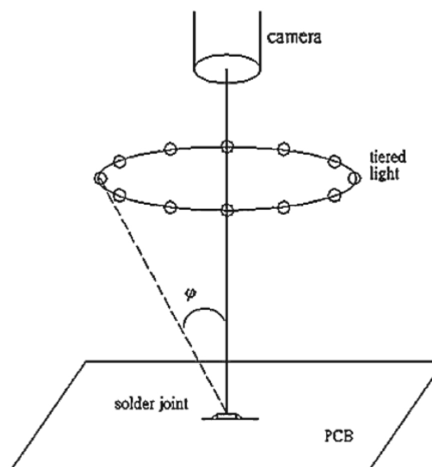


Figure 2.10: Illumination setup considered by Chiu et al. in [145]

2.4.6.2 Camera & Lens Selection and Positioning

According to National Instruments in [298], to set up an AOI system using imagery modus operandi, a number of factors need to be considered. These factors are: the working distance (WD) (distance from the camera lens to the object under inspection), resolution (smallest feature

Table 2.8: Summary for illumination systems used in AOI applications

Illumination Setup	Reference	Description	Advantages	Limitations
Three-layer tiered light (Figure 2.9)	[130, 136, 137, 140, 141, 148, 149, 151–154, 160, 292–294]	This setup was used for solder joints inspection in PCBs. The setup consisted of 3 (red, green and blue) circular ring-shaped lamps at 3levels constructed on hemispherical way with different light reflection angles that can be fixed or controlled by host computer.	The 3D nature of the inspected component is converted into 2D image. Each kind of defect is related to unique arrangement of multi-coloured image and intensities, which make it easy to classify the defect. Suitable for solder joint inspection	System Configuration is very complex [126] The processing of colour images require higher computation time and complex processing [146]. Precise geometric calibration of the light source position is required to yield accurate results [146]
Single-layer tiered light (Figure 2.10)	[145]	This setup was used for solder joints inspection in PCBs. The setup consisted of tiered LED light at one level instead of three as in the previous settings.	The 3D nature of the inspected component is converted into 2D image. Each kind of defect is related to unique arrangement of gray-scale image and intensities, which make it easy to classify the defect. Low computational time and more convenient system configuration compared with the multi-layer adjustment Suitable for solder joint inspection	Angle φ must be carefully selected and maintained for good results.
Multi-layer infrared light (Figure 2.11)	[145]	This setup was used for IR-CUT filter surface defect inspection. The first layer omits a beam with an angle of 30°, which is intended to reflect a perpendicular beam from major surface defects detected. The 60° tilted beam is intended to reflect the light from the minor surface defects perpendicularly to the camera. In case of no defect has been detected, the light will be scattered such that it will not be detected by the camera.	As the infrared beam cannot pass through the IR-CUT filter, therefore the nature of the reflected light will be an indication for a defect detected.	The angle for the infrared beams should be carefully chosen by trial and error, such that two layers of infrared lights were used.
Two lights (Figure 2.12)	[146]	This setup was used for solder joints inspection in PCBs. The setup consisted of two different light sources, namely, direct-top and diffuse-wide were used. Direct-top is the light source coming directly from the distant top, whereas in diffuse- wide, the light source is diffused evenly before being directed to the solder joint.	The computational time required to process the images is low compared with the tired illumination sources.	The system is susceptible to error in lighting condition variations.
Blue co-axial light (Figure 2.13)	[106]	This setup was used to detect surface deformation in LCD of smartphones with the aid of precision plate. By the illumination from the blue coaxial light, the images of the standard circle holes on the precision plate are reflected on the surface of the measured LCD, and then the images are further reflected onto the surface of the CCD camera by the semi-reflecting mirror, thus the circle hole images from the measured reflected LCD	Provide high reflection rate of the measured LCD surface	Limited in application
Sodium light (Figure 2.14)	[295, 296]	This setup was used to detect gap Mura transparent defect in TFT-LCD, where a sodium point light source is projected onto the panel	Light band appears as dark and light instead of coloured for easier defect identification	Applicable for transparent defects only.
Two LED lights (Figure 2.15)	[67]	This setup was used to detect Mura transparent defects in LCD. Two LED movable, lights one parallel to flow direction of the flow and the other is perpendicular. Both are tilted with an angle of 45°.	Highlighting low contrast Mura defects in LCDs	Require special configuration and limited in application
Structured light (Figure 2.16)	[107–110, 297]	This setup was used to detect aesthetic defects in LCD. The setup consisted of white and black stripes that is positioned under the inspected polymeric polariser.	Enhance the image contrast of the aesthetic defects. Black and white stripes sizes can be adjusted.	Detecting tiny defects in illuminated stripes is difficult [109]
Two fan-shaped LED lights (Figure 2.17)	[124]	This setup was used to detect touch panel glass defects. The setup consisted of white and black stripes that is positioned under the inspected polymeric polariser.	Defects can be easily distinguished such as dark images correspond with no defects and bright regions in the image represent the defect	Carrier of the inspected items must be maintained coated with high reflective substance to avoid false alarm errors

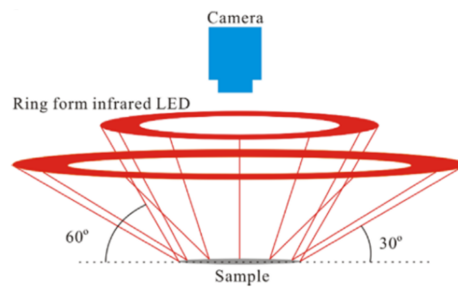
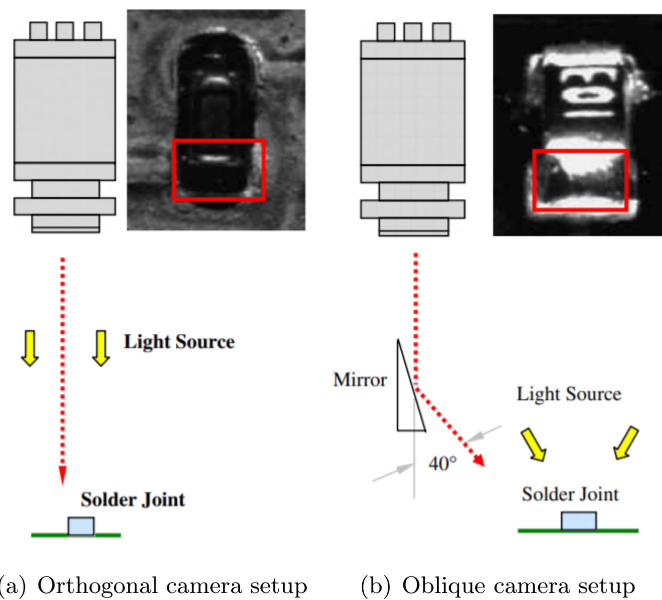


Figure 2.11: Illumination setup considered by Liu et al. in [280]



(a) Orthogonal camera setup (b) Oblique camera setup

Figure 2.12: Camera setups considered by Ong et al. in [146]

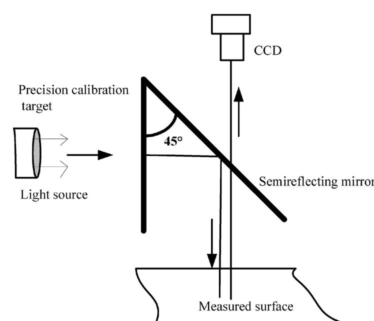
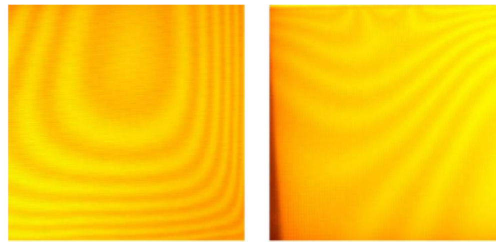


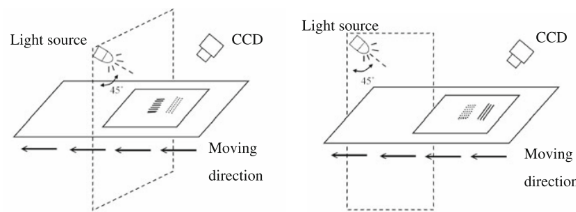
Figure 2.13: Illumination setup considered by Lu et al. in [106]



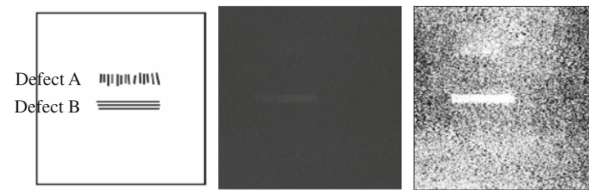
(a) Normal interference patterns represented by connected rings

(b) Abnormal interference patterns showing unconnected fringes

Figure 2.14: Illumination setup proposed in [295, 296]

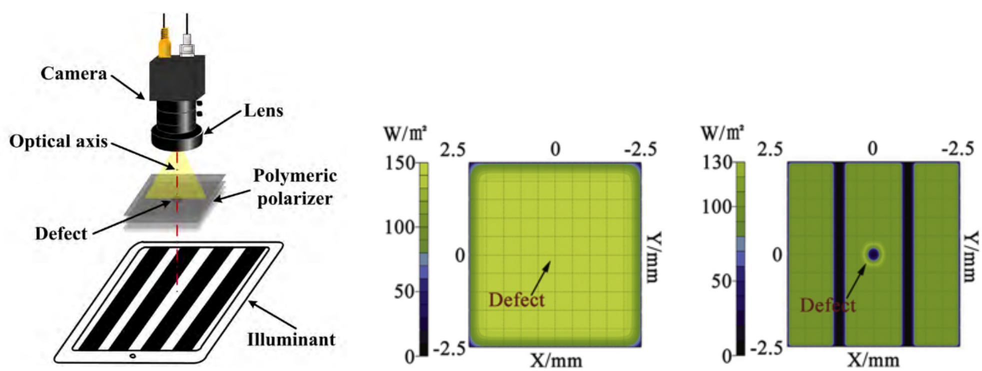


(a) Image acquisition configuration



(b) Difference in highlighting two types of defects A and B (left) between conventional setup (middle) and proposed system (right)

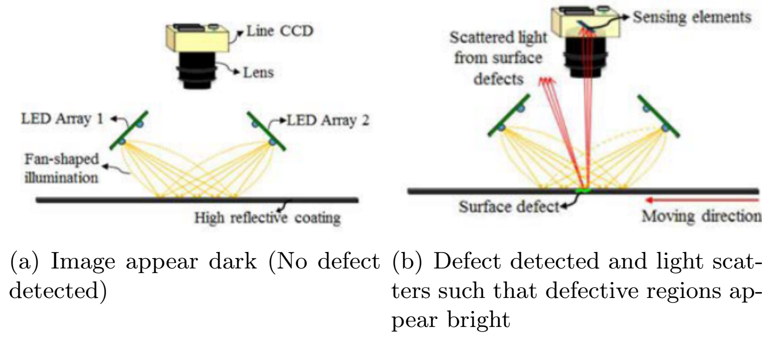
Figure 2.15: Imaging setup proposed by Tsai and Tsai in [67]



(a) Structural light configuration using black and white stripes

(b) Conventional light configuration in highlighting defect (left), proposed structural light configuration in highlighting defect (right)

Figure 2.16: Illumination setup proposed by Deng et al. in [110]



(a) Image appear dark (No defect detected)

(b) Defect detected and light scatters such that defective regions appear bright

Figure 2.17: Illumination setup proposed by Chang et al. in [124]

to be inspected), pixel resolution (PR) (the minimum number of pixels needed to represent the object under inspection), depth of field (maximum object depth that remains in focus), image sensor size (SZ), the size of sensor active area, field of view (FOV) (the area of the object under inspection that the camera can acquire), and frame rate (FR) as shown in Figure 2.18. The common image sensors used in inspection cameras are CCD, and CMOS which mimics human's perception in vision [299]. In addition to the previous factors several criteria are considered in determining the suitable image sensor and its size such as responsivity, dynamic range, uniformity, speed of operation and reliability. Using the field of view (FOV), the pixel resolution can be obtained by equation 2.1

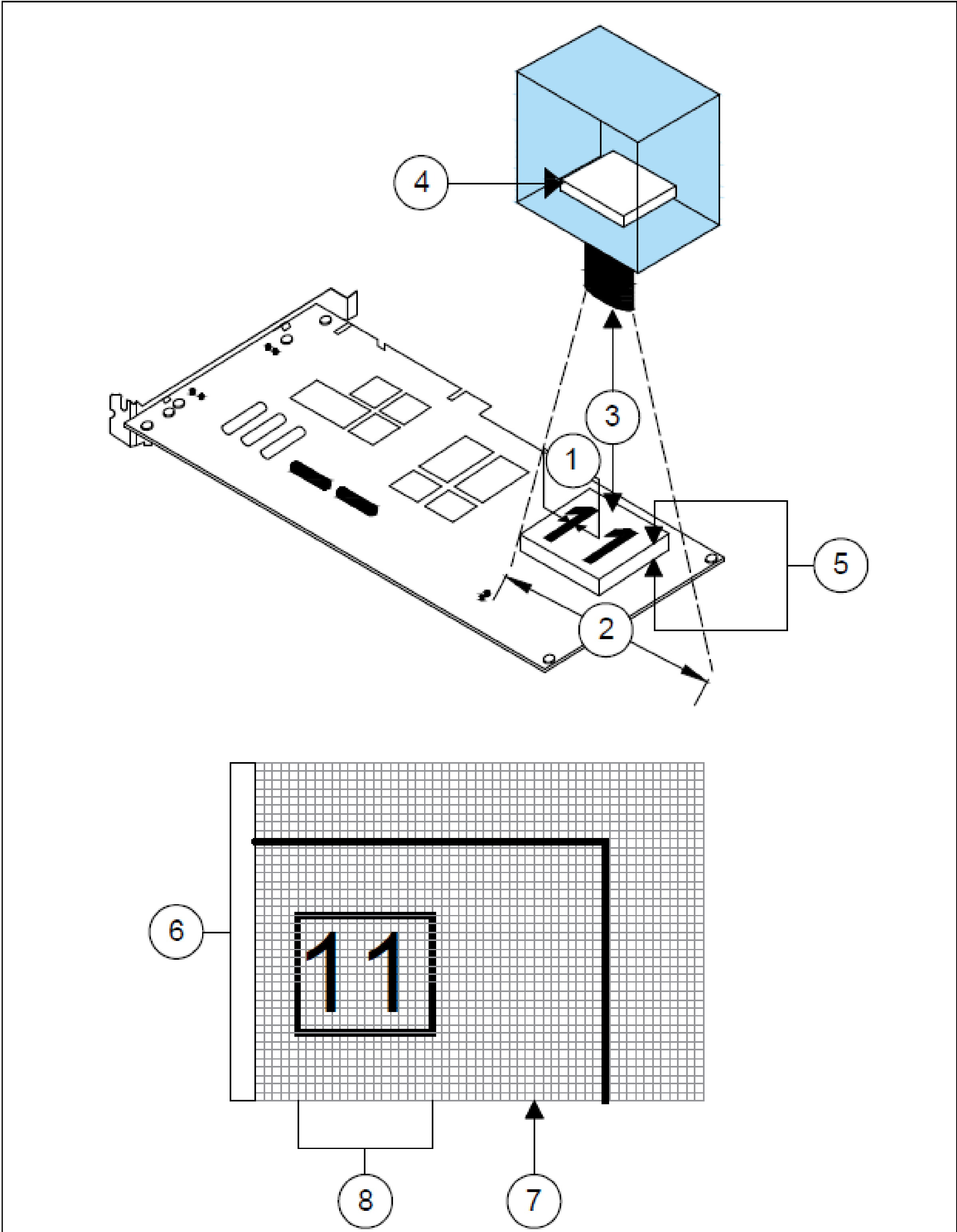
$$PR = 2 \cdot \left(\frac{FOV}{Resolution} \right) \tag{2.1}$$

And the focal length, which is used to determine the lens specifications, can be obtained using equation 2.2

$$FL = \frac{SZ \cdot WD}{FOV} \tag{2.2}$$

Focal length is important for selecting the right lens for application, as can be seen from equation 2.2, the three factors that affect selection of right focal length are sensor size, working distance and field of view. Frame rate is also important factor to consider when selecting a camera especially in batch inspection in industries that have high production rate [291].

Spatial calibration techniques are commonly used in order to set the camera in optimal position for the application. In most optical inspection applications, the position of the camera is fixed [300] and positioned in 90° with the plane of inspected components to avoid image distortion and reduce perspective errors as shown in Figure 2.18 [298]. However, in some cases such as in



1. Resolution	3. Working Distance	5. Depth of Field	7. Pixel
2. Field of View	4. Sensor Size	6. Image	8. Pixel Resolution

Figure 2.18: Factors considered in selecting camera and lens [298]

[84, 286] the camera can be aligned in different angles according to the application.

2.4.6.3 Auxiliary Systems and Other Image Acquisition Techniques

Scanning Electron Microscope SEM and Optical Coherence Technology OCT are two examples of auxiliary systems that still need visible light sensor devices to capture the image. OCT is a non-invasive and non-destructive optical imaging technique that uses a low-coherence Michelson interferometer to generate high-resolution cross-sectional imaging of samples. The OCT technique shows the structure of a material by measuring small changes in back-scattered light at various depths. The OCT- scanning depth is a function of absorption and scattering with a resolution on the order of microns and a depth range of 3 – 4mm [111, 301]. The interference signal from the interferometer can be acquired using CMOS or CCD line scan camera. The OCT technique has been successfully applied to the early diagnosis of many diseases originating under superficial areas, including cancers [302]. For the scope of this thesis, many researchers used OCT to inspect various FPD defects such as optical thin film, industrial resin, and minute defects in LCDs as shown in Figure 2.19. SEM is usually used to detect micro defects in semiconductor wafer (shown in Figure 2.20(b)) and LED chips with the aid of image sensing device, it can be used also to detect small defects in FPD as in [303].

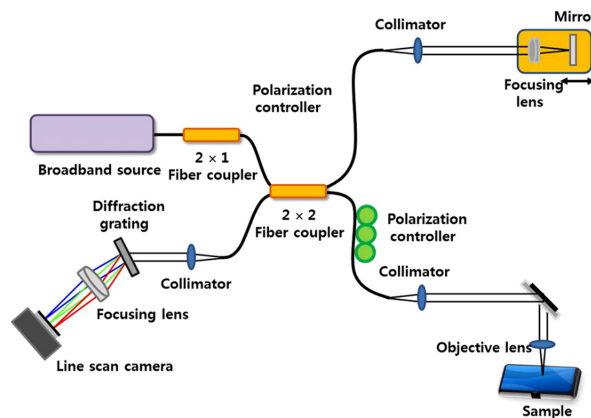
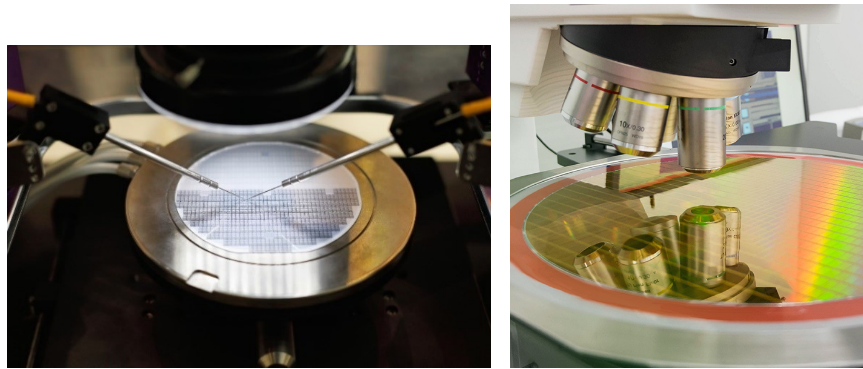


Figure 2.19: OCT setup used in [301]

The use of alternative methods of imagery alongside the traditional surface optical systems such as thermography and X-ray are also gaining momentum. For example, X-ray scanning for alignment and defect inspection of IC components and inner layers of PCBs as shown in Figure 2.21(a). However, X-ray technology is not efficient in detecting fine cracks and open solder joints,

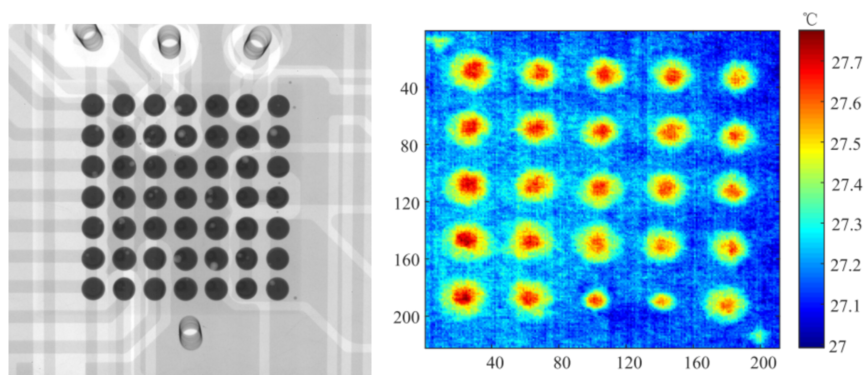


(a) Semiconductor wafer testing using circuit probe [304]

(b) Semiconductor wafer testing with the aid of SEM [305]

Figure 2.20: Various semiconductor wafer testing techniques

as the small air gap in the defects do not attenuate the X-rays. Although 3D X-ray system (e.g. CT) can solve some of these problems, but data processing and image resolutions could reduce their deployment in embedded in process production systems [193]. In thermography, a camera with a thermal sensor is used to measure the infrared radiation from the sample and convert the radiation flux to temperature. The temperature distribution can be then illustrated in a form of thermal images. For inspection applications, some defects such as solder ball defects will change the heat flow resulting in abnormal thermal behaviours, which is dependent on the defect size, location, and thermal physical properties of the material as shown in Figure 2.21(b) [193]. In such cases thermography inspection is considered an efficient tool to highlight these defects for further analysis and classification. Unlike previously mentioned technologies, circuit probe



(a) Solder ball inspection using X-ray [191]

(b) Solder ball inspection using thermography [193]

Figure 2.21: Various technologies used for solder ball inspection

does not need image sensor to operate as the defect map (WM) is generated according to the data provided by circuit probe. As mentioned in section 2.4.4, this technology is widely used for semiconductor wafer inspection as shown in Figure 2.20(a). Table 2.9 summarises the other image acquisition techniques and auxiliary systems used for AOI in literature.

Table 2.9: Summary of articles that other image acquisition techniques and auxiliary systems

Technology	Reference	Defects considered			
		LED	Wafer	PCB	FPD
Circuit Probe	Articles in Table 2.5		✓		
SEM	[252, 254, 258, 267, 268, 275, 276, 306]		✓		
	[43]	✓			
	[303]				✓
OCT	[104, 111, 301, 302]				✓
Thermo-graphy	[193, 307]			✓	
X-ray	[179, 186, 187, 191, 307–309]			✓	

2.5 Inspection Algorithm for Semiconductor Wafer Defects

After acquiring the image using the hardware setup mentioned earlier, the image will undergo through inspection algorithm to decide if it represent an anomaly or normal sample. Most traditional inspection systems use subtraction or template matching technique to compare the inspected component with the reference template image [32]. Choosing a suitable inspection algorithm can enhance the classification accuracy and minimise false alarm rates. Usually in AOI application, the collected images have to go through some enhancement (preprocessing) before the application of inspection algorithms. Feature extraction and selection techniques are then used to segment the defective regions and to discover the important defect features. The final stage is to provide the processed information to the classifier algorithm as shown in Figure 2.2. In the next sections, the discussion will be limited to the feature extraction and classification methods used in semiconductor wafer and microelectronic chips inspection.

2.5.1 Preprocessing

Isolating the part of interest in AOI application can be an effective way to decrease the computation time as well as enhancing the background contrast in some applications. It is considered a preprocessing stage in analysing the image for inspection. Image mask is used for this purpose; which can be defined as an 8-bit binary image that is the same size or smaller than the inspected image [298]. The value of mask's pixel can determine whether the corresponding pixel of the inspected image can be processed or not. If a pixel in the image mask has a 0 value, the corresponding pixel of the inspected image will be masked. Otherwise if the value is 1, the corresponding pixels of the inspected image will be visible for inspection. The process of applying an image size can be implemented manually or automatically depending on the application. The resulting area of the inspected image after applying the mask is called region of interest (ROI). ROI can have the shape of a circle, oval, polygon, rectangle, or any customised shape. The most commonly shape used for defining the ROI in AOI applications is rectangle (block). After specifying the ROI, the resulting image can be subjected to geometric transformation or/and filtering for further enhancement before applying the inspection algorithm. Geometric transformation involves calculating the projection of each pixel in the ROI onto another space. This method can help in image restoration and correction in case of any presence of geometric distortion. The method of applying geometric transformation include scaling, rotating and translation of the image [167].

Applying filters is considered as an important preprocessing steps. Filters serve a key role in enhancing the image for inspection and highlighting the important features. They can smoothen, sharpen, transform, and remove noise from the image, so that the inspection algorithm can do the feature extraction task easily. Filters can be subdivided as linear and non-linear [298]. In linear filters, a convolution kernel has to be defined, it can be square (e.g. 3×3), or a rectangle (e.g. 3×5). Each value of the convolution kernel is selected according to the purpose of the filter as mentioned before, which can be multiplied by the value of the corresponding pixel and neighbour pixels in order to recalculate a new value of the corresponding pixel. Examples of linear filters are Laplacian, and Gaussian filters. Non-linear filters use non-linear functions for the parameters of the kernel, the process of recalculating the pixel is similar as in the linear filters. Some examples of nonlinear filters are median filters and Prewitt filter. Median filters

are very common pre-processing tool in defect inspection applications. For instance, Huang et al. in [212] relied on median filter to inspect semiconductor wafer dies.

2.5.2 Feature Extraction

Feature extraction process involves applying one or more of image processing techniques (e.g. frequency analysis and segmentation) in order to describe the characteristics of the studied regions (e.g. defects and abnormalities). These characteristics are usually described using a set of values called *Feature Values* that can be represented in vector or matrix form. The purpose of feature selection step is to consider the important feature values only that can contribute to the classification process and discard the redundant ones. This step is crucial in reducing the computational time for the inspection algorithm. Principle component analysis (PCA) [310] is very popular technique used in feature selection. Other algorithms such as along genetic algorithm (GA) [157], particle swarm optimization (PSO) [162], Adaboost [136] and neural networks [149] are also used for this purpose.

2.5.2.1 Frequency Domain Analysis

Frequency domain analysis can be used to convert pictures from the spatial domain to the frequency domain. Such transformation is helpful for capturing the global structure of the image and minimising reconstruction errors. In frequency analysis, high pass and low pass frequency filters can be used to attenuate the unnecessary noise in the image. After attenuation, the picture can be transformed back to the spatial domain for further processing. Frequency analysis is preferably used in investigating defects that have low contrast and high illumination noise such that spatial domain cannot capture the features of the defect (e.g. Mura defects).

The frequency representation can be obtained by various transform functions such as Fast Fourier Transform (FFT), Discrete Wavelet Transform (DWT), and Discreet Cosine Transform (DCT). Transform functions allows for a noninvertible transformation from spatial domain to a reduced dimensionality feature space. Thus, facilitating classification with substantially less features and manageable classification error.

DCT is mainly applied to perform image reconstruction and dimension reduction by converting the image to a frequency spectrum and attenuates certain coefficients that lays into two

major bands, namely, low frequency, and high frequency [266]. In most AOI applications, since illumination variations mainly lie in the low-frequency band (in upper-left corner of the frequency spectrum), an appropriate number of DCT coefficients in low-frequency band are truncated to minimise variations under different lighting conditions as shown in Figure 2.22 [138].

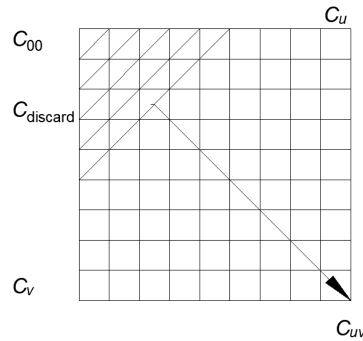


Figure 2.22: Truncating lower frequency coefficient (C_{00})

Lin and Ho in [266] discovered that after transforming the digital image of a chip in a wafer to the DCT domain, the frequencies of the pinhole defects spread around the middle and high frequency regions. Therefore, to highlight pinhole defects features, high pass filtering where used to attenuate the frequency components of the non-defect regions by setting their values to zero. In order to adapt the background for DCT implementing, the proposed method were insensitive to light variations since the low frequency band has been eliminated. However, this method has some limitations such as defect size, background property and processing time. Defects of large size features can be lost after the attenuation of low frequency band. Moreover, defects embedded in structure textures cannot be detected because this method is suitably applied to identify defects in random textures. Finally, the proposed method takes four seconds to process a forward and an inverse DCT transformation, which could not meet the requirements of an on-line inspections system.

As mentioned earlier that DWT is one of the frequency domain analysis methods for feature extraction. DWT has the advantage of providing an easy way for multi-resolution representation, from which defect texture features can be easily extracted. The merits of using DWT include local image processing, simple calculations, high speed processing and multiple image information. Lin in [37] considered one type of wavelet transform called Haar transform to highlight water-drop defects on LED chip surface. Haar transform is considered the simplest approach among wavelet

transforms. The transform is performed to each row of pixel values and then performing another wavelet transform to each column. Four wavelet transforms were used that generate four sub-images; one level of wavelet decomposition generates one smooth sub-image and three detail sub-images that contain fine structures with horizontal, vertical and diagonal orientations. Each sub-image represents certain characteristic that has half the number of columns and rows of the original image.

2.5.2.2 Segmentation

Segmentation is a process of grouping an image into units that are homogeneous with respect to one or more characteristics; it is an important step in image processing and feature extraction [311]. Segmentation can be divided into three classes, thresholding, edge detection, and region extraction [312]. Morphological operations are used to aid segmentation process by removing the noise or unwanted regions. Morphological operations contain erosion, dilation, opening and closing processes.

2.5.2.2.1 Thresholding

Thresholding techniques are mainly effective when the contrast between various features are well established. Thresholding is considered one of the segmentation techniques that split the image into two regions; a particle region (e.g. feature or component) and background region with the aid of gray-level threshold value [298]. In this manner, all the gray-level pixel values that belong to the particle region (above threshold value) are set to logic 1 and the rest of pixels that represent background (below threshold-value) are set to 0; in other words the image is converted into binary image. The threshold value can be selected automatically with the aid of a histogram [313]. The x-axis of the histogram represents the gray-scale values of the image (in case of 8-bit image it will be 0 - 255) and the y-axis represent the number of pixels for each intensity value. Choosing the right thresholding value is considered a key factor in the thresholding process. In general, thresholding techniques can be subdivided into two categories Global and Local. Global thresholding performs the same mentioned in the previous discussion, in which a single threshold value from the histogram of the entire image is selected. However, local thresholding (also called adaptive thresholding) uses localised gray-level information to choose multiple threshold values. Local thresholding methods are superior to the global ones when ununiform illumination settings

are applied. On the other hand, global thresholding is considered much simpler and require less computational time. For defect inspection applications, since the illumination settings are well-controlled and uniform, global thresholding is preferable [313, 314]. Otsu-thresholding is considered one of the widely used global thresholding techniques. This method selects threshold value that maximise the between-class variance of the histogram [313, 315]. Otsu-thresholding techniques are preferably used when the feature of the defect has high contrast when compared to the background. Example of these defects are missing component and missing solder defects in PCBs.

2.5.2.2.2 Region Extraction

Connected Component algorithm is a commonly used region extraction technique which regard each pixel as a node in a graph. Connected Component Labelling (CCL) can be used after this step in defect inspection tasks to label the regions found (e.g. defective regions) into certain classes. Chang et al. in [43] applied CCL algorithm to identify the number of LED dies from the binary image map. The algorithm were applied after segmenting the die region pixels as 1s and non-die region as 0s in a binary image that have the same size as the original image using Hopfield neural network (HNN). The CCL algorithm examine each die region pixel to give it a label and compare this pixel to neighbor pixel so that all neighbor die region pixels are considered as one die. This procedure is repeated until all the dies are recognised. Ooi et al. in [202] proposed a comprehensive data-mining process using a Segmentation, Detection, and Cluster-Extraction (SDC) algorithm to extract common defective wafer patterns from the raw production test data. The proposed SDC algorithm has been developed to extract meaningful cluster features from a database of manufacturing test results accurately and automatically. It can be implemented either in an online or offline mode. CCL algorithm were used to simply pairing adjacent 1s (that represent defective dies) and 0s (that represent normal dies) in a binary image. Finally, morphological operations were used to remove the noisy parts. This study has achieved a 90% accuracy rates in detection. Furthermore, SDC algorithm have helped in reducing the false alarm rates by more than 90% when compared to using segmentation algorithm alone. A similar study was also conducted by Xie et al. in [237], where four defect patterns of WMs were generated by connecting-components and morphological operation approaches. Wijesinghe et al. in [111] considered similar approach; however, in inspecting liquid resin defects in LCDs. The acquired

images from the OCT are binarised and pixels were classified as defective and non-defective according to certain threshold related to the intensity. Next, CCL technique were used to define the boundary of the defective regions and non-defective regions by grouping each pixel's class together. This technique is repeated until all regions are classified. Using this approach does not only evaluates the particular defective location; however, it defines the magnitude of each defect, and the number of defects in a unit sample. Kubota et al. in [261] used Region Growing to extract the shape feature of the micropipe wafer defect. Their algorithm procedure starts by extracting points of local minimum within a square region of pixels centred at the pixel of question called neighborhood region. If the local minimum point value is within predefined range, then this point belongs to the specified region. Furthermore, they extracted the surface curvature features of the defected. Then they created a feature vector out of the extracted features and fed it to a neural network for classification. Hsieh and Chen in [204] used multiple feature extraction approaches based on sub-region analysis for defining the possible defective point of semiconductor wafers. Then they applied Minimum Rectangle Area approach to combine the defective points into clusters. However, they did not mention how to specify the number of clusters or and how to use clustering features to separate various types of defect patterns [208]. Liao et al. in [236] proposed morphology-based similarity approach to generate simple and complex wafer sample patterns with certain degrees of similarity, as compared to the objective target WMs. Several morphological operations were used in this process including dilation, erosion, opening, closing, position shift, density change and rotation with variations. The generated data were then sent to classifier to specify the type of the patterned defect.

2.5.2.2.3 Edge Detection

One of the used segmentation techniques to extract used in machine vision applications is edge detection. This technique is very useful in applications that requires locating a feature, alignment or for gauge inspection and measurements. As the name says, edge detection is used to find boundaries in the image and sharp edges, this can be performed by locating the discontinuities in pixel intensities in the image using several filters and operators such as Prewitt, Canny, and Sobel. Pixels that are found on the boundary using this technique are called edge pixels. An edge pixel can be described using two parameters; edge strength (also called edge contrast) which defines the minimum difference in the grayscale value between the edge and background, and

edge direction, which can be defined by the angle of the edge direction [298].

2.5.2.3 Hough Transform

Hough transform (HT) is also one of the techniques used for feature extraction and defect detection applications, due to its ability of detecting various shapes (e.g. linear, circular and elliptic shapes), given a parameterised description of the shape. Edge detection operations are used as preprocessing procedures before applying HT. This technique is appropriate for detecting wafer defects patterns in WBM as they tend to follow a specific pattern. Since WBM are considered binary images, HT transforms the binary image into a parameter space and tries to detect the parameterised pattern through a voting process in which each point votes for all the possible patterns passing through it. Patterns with a higher number of votes indicate a higher probability of the occurrence of this pattern on the map. As long as a parameterised model can be established for the spatial pattern, this method can be adopted [199]. White et al. in [215] used HT based on linear parameterisation to detect defective dies patters on WBM. The patters detected in this study were of linear shapes such as scratches and edge patterns. The main advantage of this study is its simplicity and the ability of being fully automated. However, the study was not useful in detecting commonly defective WBM patterns such as ring patterns. Zhou et al. in [223] presented a control chart technique to detect line and circular patterns through statistical evaluation using HT. This statistical evaluation calculates the number of votes for the line and circular defect maps. When the number of votes is larger than the control limit, an alarm is raised indicating that a pattern is detected. However, this method cannot distinguish specific definite cluster patterns such as blob, bull's eye, edge, and ring. Chang et al. in [199] applied HT to detect WM defect patterns and were able to overcome the limitations in previous studies by detecting all the patters that were missed by them.

Despite of HT's ability of detecting various defect patterns, this technique is considered insensitive to gaps (missing pixels), which make it difficult to detect random or mixed pattern defects in WMs since it may group them as one pattern instead of multiple ones [215]. HTs performs poorly when it comes to low contrast images, this may not be a problem in WMs since they are generated using circuit probe techniques. However, it may become serious problem in other cases that utilise illumination to highlight small defects such as Mura defects in LCDs.

Furthermore, HT is considered costly in terms of computational time [66].

2.5.2.4 Template Matching

Template matching is considered one of the simplest and earliest pattern recognition techniques [316]. In AOI applications, template matching algorithm works by first identifying a reference template which usually represent the non-defected case (also known as *golden template*) that can be used for comparison. The selected template can be compared to the target samples using various kind of correlation functions. Template matching is preferable in investigating systematic defects that have common patterns. The reason behind that is the defect is always compared with a reference template so if the defect occurs is not systematic, the comparison with the reference template will fail and high false alarm rates will occur. Template matching techniques can be used in most application; however, it is not preferable in inspecting WM defects, since most of these defects are of mixed patterns (systematic and random).

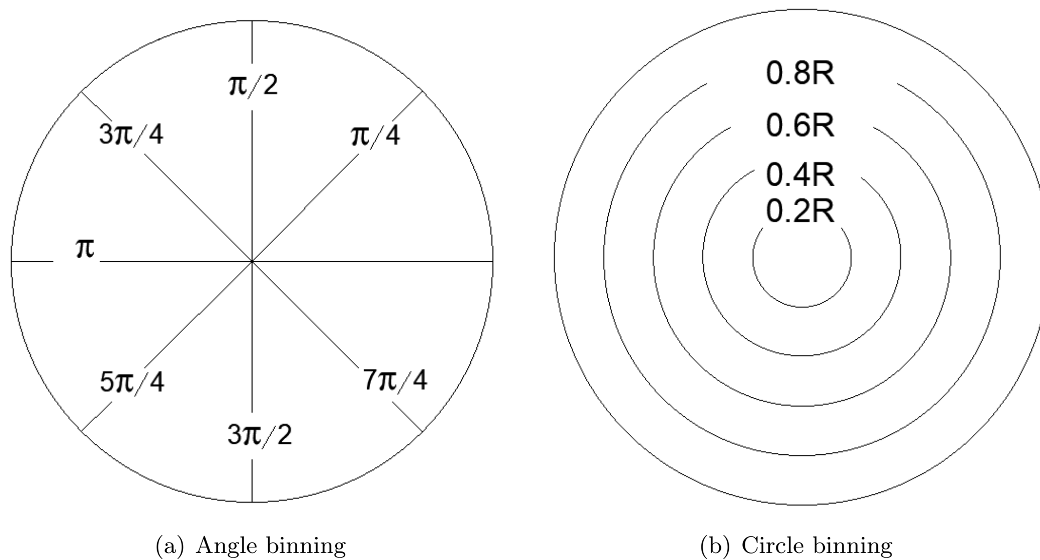
Normalised cross-correlation (NCC) function is one of the most commonly used template matching functions, it involves sliding the sub-image over the larger image pixel by pixel and calculating normalised correlation to estimate the degree of similarity between the sub-image and the large image regions [127]. It has been proven to greatly reduce the data storage and also reduces the sensitivity to the acquired images when compared with traditional image subtraction [317]. Zhong et al. in [49] proposed blob analysation and template matching approach based on NCC to inspect polycrystalline and fragmentary defects on LED chips. Regional image segmentation was first performed to locate the blob defect features and exclude them. NCC approach were also used to locate LED chips at pixel accuracy. A certain threshold was used to classify the abnormal chips from the normal ones. The study showed good accuracy in detecting normal chips with zero false alarm rate. However, a false alarm rate is presented in detecting defective chips because of the NCC threshold value selected. Despite of NCC simplicity, this method is time-consuming since it is based on 2D summation and multiplication operations to compute the correlation, and is often combined with the image pyramid method to compress image size to alleviate the computation load of the correlation coefficient method [47, 157]. Furthermore, this method is very sensitive to illumination variations and image shifts, which can increase false alarm rates [166]. In order to reduce the time complexity of this method, Crispin and Rankov in

[157] proposed a modified NCC based template matching approach using the generalised gray-model template for inspecting component placement errors in PCBs such as missing, misaligned or incorrectly rotated components. The gray-model template can extract a vector of all edge positions of the inspected component. By only searching the image at edge locations, the number of NCC calculations that are needed to be performed can be significantly reduced. Genetic algorithm was also proposed to replace the sliding sub-image process involved in NCC operation. Despite their efforts in reducing the computational time of the NCC approach, this method did not mention the classification criteria for the investigated errors (missing and misaligned component). Tsai and Huang in [182] proposed a method, that can be used to investigate PCB and IC dies defects, based on the comparison of the whole Fourier transformed image between the template and the sample inspected images. It retains only the suspicious frequency components in the Fourier domain of the test image and discards the common frequency components. After the anomalies are detected and the background has been removed, the inverse Fourier transform is then applied to restore the sample image. A simple statistical control limit is finally used as the adaptive threshold to segment the local defect for classification purposes. Unlike conventional template matching approaches, this method is considered invariant to translation and illumination variations in the sample images. Moreover, the proposed approach can detect subtle defects as small as 1-pixel wide. Liu et al. in [255, 256] proposed similar approach to detect IC wafer defects. In their studies they used 2D and 1D DWT respectively to extract a standard template image form three defected images using comparison and subtraction approaches. In [256], they showed that the accuracy of detection does not differ between 1D and 2DWT, while the computational time in 1D DWT is less than 2D. The advantage of this method is its robustness to illumination. However, the accuracy measures for these methods were not mentioned in both studies.

2.5.2.5 Gray-level Co-occurrence Matrix

Various statistical features can be extracted from (Gray-level Co-occurrence Matrix) GLCM such as energy, entropy, contrast, variance, correlation, and inverse difference moment. Considering large number of features for extraction will increase the quality of the assessment; however, it will increase the computational time in return [318, 319]. Li and Huang in [216] used GLCM to

extract eight features: entropy, energy, contrast, local Homogeneity, mass, centroid, geometric moments, and central moments along with seven moment invariant features (translation, scaling and rotation) for WBMs. These features were fed to classifier algorithm to classify detected patterns. Wang and Chen [250] noticed that the WM defect pattern features are rotation invariant because of the round shape of the wafer's image. This means that the output of the features is not affected by the rotation of the input image. Therefore, they proposed three rotatable weight masks of the same size of the circular area of WM to extract the defected features. They named these masks polar masks, line masks and arc masks. Rotation invariance is achieved by making several rotated copies of each mask and only the max feature value is retained for each master mask. Polar masks aim to extract features of concentric patterns, while line and arc masks are designed to mainly deal with eccentric patterns such as scratches. For polar mask the area of the mask is divided into several zones using two methods: angle binning and circle binning as show in Figure 2.23. Angle binning (shown in Figure 2.23(a)) divides WM circle into equally spaced circular sectors, while circle binning (shown in Figure 2.23(b)) draws concentric circles of the WM, which separate the circle into annuli. The features are extracted using these zones, for example edge defect pattern appears mostly at interval $[0.8R, R]$, so circle binning at $0.8R$ could help detect "Edge" pattern.



(a) Angle binning

(b) Circle binning

Figure 2.23: Binning of WM circle of raduis R

2.5.3 Classification

Classification is considered the last stage of the inspection algorithm. In this stage the inspection algorithm uses the extracted features as an input in order to produce a an output of categorised classes. In terms of the nature of the output, classification can be subdivided into binary classification and multi-class classification. In binary classification, the outputs are categorised into two groups (e.g. pass/fail, defect/non-defect). In multi-class classification, the outputs are categorised into more than two groups. Classification can be also subdivided according to the algorithm used in this process into rule-based classification and learning-based classification (machine learning).

2.5.3.1 Rule-based Classification

As a simple and easy way to classify defective and non-defective products, a lot of researchers used simple logic classifiers such as conditional statement (if-else) and Boolean rules. Hence, these statements are considered pre-programmed and has no ability to learn as the case in machine learning tools that are discussed in Section 2.5.3.2. These classifiers are usually used after implementing a relevant image processing technique to highlight the features for analysis, therefore the accuracy of them is highly dependent on the image processing tool for feature extraction. A certain threshold is then used for the assessment process. For instance, in template matching this threshold is called the matching score. On the other hand, if thresholding and segmentation techniques were used, then the threshold could be the number of bright pixels. Human assessor can define a threshold value for the previous thresholds such that if the obtained value exceeds threshold value, an action must be made (e.g. consider the sample defective).

Shankar and Zhong in [263] set up a series of five logic rules based on the energy features of the sample images to classify the wafer defects. Such that, if the energy features of error image exceeds 1μm, this means the sample image suggests a defect (rule 1). Further on, if the energy features of the chip-out region is greater than energy features of scratch, bridging, and metal lifting then the defect is caused by chip-out (rule 2) and so until all the investigated defects are classified. The previous study suggested a simple and less computational time consuming algorithm. However, there was no discussion about the accuracy rate of the classification. Yeh et al. in [269] classified semiconductor wafer dies defects such as particle contamination and scratch

based on conditional logic algorithm. The features are extracted using a recent approach called wavelet transform modulus sum (WTMS). A golden sub-image (which represent a defect free sample) were also used for comparison with sample images. The WTMS approach is modified in this paper to detect abnormalities in each pixel of the sample image that may indicate a possible defect. Various coefficients are investigated (e.g. vertical wavelet coefficient and horizontal wavelet coefficient), were these coefficients are called “wavelet energy”. The investigated object (wafer) preserve more energy than background does. Furthermore, pixels on corners clustering noisy spots or irregular edges maintain much more wavelet energy than pixels on small portions of an object do. A ratio between a pixel and its neighbor can describe the status of this pixel. The ratio is treated as threshold, such that if it is more than or equal to zero, the pixel is considered as a defect otherwise the sample goes to another check which compares the sample pixel with the golden sample pixel to see if they belong to the same coordinate. If the comparison failed to show that they belong to the same coordinate, the pixel is considered a defect, otherwise it will be considered as defect-free.

Fuzzy Logic which is a form of multi-valued logic was also one of the classification techniques used in literature. Fuzzy logic is capable in handling the problem of blurred uncertainty phenomenon that cannot be described by binary logic (0s and 1s). Therefore, an appropriate value is taken between 0 and 1 to represent the degree of one element belonging to a set (called fuzzy set). A set of rules is used to represent the fuzzy values based on if-then statements, the process of converting an input value to a fuzzy value is called fuzzification and is used in fuzzy logic controllers. Lin in [224] noticed the fuzziness of WBM patterns classification, as not all maps should belong to one pattern only. Therefore, the study proposed new fuzzy variable of clustering pattern (FVCP) by using fuzzy logic control. Where FVCP represents a numerical value ranges from 0 to 5 according to the five patterns considered in this study. Hence, the FVCP can be a decimal number, for instance if the value of 4 represents bottom pattern and a value of 5 represents crescent moon pattern, then an FVCP of 4.7 implies that the recognised clustering pattern has 70% degree belonging to crescent moon pattern and 30% degree belonging to bottom pattern.

2.5.3.2 Machine Learning Classifiers

Unlike rule-based classification approaches mentioned in the previous section, machine learning provides the ability of the algorithm to learn from training data. In manufacturing and optical inspection applications, supervised machine learning techniques are mostly applied among the other categories [320]. Multi-layer perceptron (MLP), convolutional neural network (CNN), support vector machines (SVM), decision trees and k-nearest neighbor (k-NN) are all examples of supervised learning algorithms. Unsupervised learning involves the process of developing a model or function without predefining the outputs. This method is typically used for finding meaningful patterns (e.g. WBM defect patterns) or classifications within a large data set [321]. Clustering, adaptive resonance theory network (ART), Hopfield neural network (HNN), Cellular Neural Network, and self-organizing map (SOM) are all examples of unsupervised learning algorithms.

2.5.3.2.1 Decision Tree Classifier

A decision tree is a decision support tool that uses a tree-like graph or model to describe relationships among different variables and makes decisions [322]. Decision trees are considered one of the supervised learning classifiers. They share some similarities with rule-based classification; additionally, they have the ability to learn by setting cost-function-like measures such as information gain and Gini impurity.

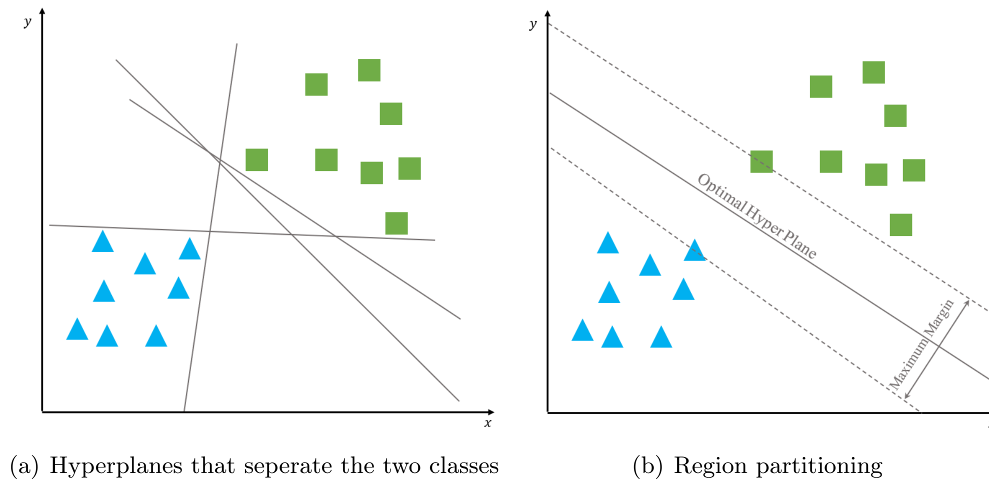
C4.5 (which is one of the modified decision tree algorithm) has achieved best accuracy classification results compared to others for actual manufacturing data sets of WM pattern defects in the study conducted by Chang et al. in [199]. In this study the training process for the classifiers used artificially generated defect data along with actual manufacturing data. In this study, circular HTs, linear HTs and zone ratio were used to identify bull's eye, line, and ring patterns respectively. Logitboost has achieved the best classification accuracy in terms to the artificial manufacturing data. Even though that this study achieved a considerably accurate classification results; however, if two or more defect clusters partially overlap, the proposed method was unable to classify them accurately. Ooi et al. in [194] investigated WM pattern defects using a modified decision tree called "alternating decision tree" (ADTree) to overcome the main disadvantage of conventional decision tree algorithm which is empty or null-leaf phenomena. This phenomena occurs when there is a valid path with no corresponding learning example, which

results in an unclassified instance. ADTree is considered a combination of boosting algorithm and decision tree which generates classification rule that are relatively simpler and easier [323]. The boosting process combined with decision tree involves training the weak classifiers of the decision tree, which are reweighted according to the mistakes occurred in the classifier. In their approach, they have considered that the dies in the wafer tend to have circular shape rather than square, therefore they have used rotational moment invariants (RMI) method for feature extraction to feed the ADTree. They have achieved classification accuracy of 95%. Even though that their system was trained to recognise a limited set of defects including bull's eye, blob, line, edge, ring and hat. However, the system could be trained to recognise new defect types by specifying their geometry and simulating it. Piao et al. in [231] proposed a decision tree ensemble-based WM failure pattern recognition method based on the radon transform features. The radon transform is the projection of image angle and were used for feature extraction. Four features were extracted; max, min, average and standard deviation of projections from the radon transform and used to build the decision tree ensemble. The classifier has the ability to classify the defect according to eight failure patterns. Even though the accuracy of the classifier achieved relatively good results for all the selected patterns (90.5%); however, the proposed method failed to efficiently recognise several pattern types, which may indicate that using the four previously mentioned features of projections are not enough to present the geometric and spatial information of defects in WM. Kim et al. in [242] proposed a generalised decision tree classifier to classify dynamic-random access memory (DRAM) semiconductor wafer based on WM. The WM they used shows the severity of defected chips on the wafer according to colour code, then the WM is binarised to form WBM. The uncertain features of the WBM is then classified according to the generalised decision tree approach according to shape patterns and location of defects. Based on their proposed algorithm, they achieved an average accuracy of 95.6%. Despite of the good results, their study considered only four patterns, rather than considering all possible patterns. Furthermore, the defect size of the pattern was not an important factor in this study.

2.5.3.2.2 Support Vector Machine

Support Vector Machines (SVMs) are a family of supervised machine learning algorithms used for binary classification problems [324]; however, it can be modified for multi-class classification [325]. Its objective to find a hyperplane (also known as kernel function) in two or higher dimensional

problems to separate the two classes as shown in Figure 2.24(a). If the two classes are non-separable, we can still look for the hyperplane that maximise the margin between the classes such that the misclassification error can be minimised as shown in Figure 2.24(b) [326]. SVMs



(a) Hyperplanes that seperate the two classes (b) Region partitioning

Figure 2.24: Optimal hyperplane that sperates the two classes

have been found to be an excellent tool in terms of the computational time requirement, high classification accuracy and stability. A key feature of SVMs classification method is its ability to use high-dimensional data without the need of feature selection step to reduce the dimensionality of the data [327]. However, SVM classifier is not very effective against data imbalance problem because the error penalties for positive and negative classes are the same. This will make the optimal separating hyperplane move toward the smaller class. In other words, if the positive class is smaller than the negative class, then the hyperplane will move toward the positive class, which will further result in numerous false negative errors [93]. Kuo et al. in [50] investigated two types of defects in the light area and three types of defects in the electrode area of LED chip using several SVM-based algorithms. Since SVM is optimised for binary classification, a standalone SVM algorithm were used to classify light area defects into two defect classes: breakdown and colour aberration. However, to investigate the three defects in electrode area, a multi-class classification scheme must be applied. Therefore, a combination of decision tree and SVM algorithms called DTSVM were used for the multi-classification task. Baly and Hajj in [257] suggested SVM classifier due to its ability for efficient classification of multi-modal, multivariate, and inseparable wafer data points. Their proposed model applied multidimensional hyperplanes for separating and classifying wafer data into high-yield and low-yield classes. They

evaluated the accuracy of their model along with other models such as partial least squares (PLS), General Regression Network (GRN), C4.5 and k-NN. The results of their experiments illustrate that the SVM model outperforms all other models in term of classification accuracy. However, their classification approach classified wafer defects as good or bad without specifying the cause of defect. After generating WMs using morphological operation (based real defective wafers data), Liao et al. in [236] used SVM classifier to specify the type of the defect according to the pattern generated. The experimental results showed that the proposed method achieved an overall catching rate of 95% with only 5% false-alarm rate. However, this study faced difficulties in detecting some patterns such as donut and repeated scratches, such that low catching rates of 72% and 73% were achieved respectively for detecting these patterns.

Another classification algorithm that share a lot of similarities with SVM called Support Vector Clustering (SVC). SVC acts the same as SVM classifier; however in the final step of this algorithm, the classifier tries to search for an enclosed hypersphere instead of hyperplane in the feature space to cover all data samples as tightly as possible [211]. Wang et al. in [211] used SVC classifier to identify and classify multiple zones, multiple scratches, multiple rings and ring-zone mixed type pattern defects in WBM. To further categorise the unclassified defect patterns from the SVC classifier, a decision tree approach where used for this purpose. SVC were also used in the study conducted by Yuan et al. in [221] to separate random defects from systematic defects in WMs, which improve both classification accuracy and computational accuracy for further classification of defect patterns using Bayesian classifier. Despite the good results that were achieved, the speed of the clustering significantly depends on the number of defects and the number of clusters. If a lot of clusters exist on the wafer, efficiency of the proposed algorithm may be suffered from computationally intensive simulation.

2.5.3.2.3 Clustering

Clustering is considered one of the unsupervised machine learning algorithms. Hence, it does not require training set for performing classification problems. Hierarchical and K-means clustering are one of the commonly used clustering algorithms [328]. In optical inspection applications hierarchical clustering is suitable with small datasets. It creates a hierarchy of clusters that can be represented in a tree structure where the root of the tree consists of single cluster that contains all the dataset and the leaves correspond to individual dataset. Hierarchical clustering

methods are either agglomerative, in which the algorithm starts with the leaves and merge the clusters together, or divisive, in which the algorithm starts with the root and gradually split the clusters. The linkage criterion is the deciding factor of which hierarchical clustering method should be used, which is a function of the pairwise distances between observations. K-means clustering defines a K number of points (also known as centroids) that indicates the number of groups which is established a priori by expert. These centroids are initialised randomly within the dataset, in which the algorithm tries to assign the K centroid to the nearest neighbor of datasets in order to group the near datasets into one cluster. To calculate the degree of homogeneity and heterogeneity, the K-means clustering method employs the Euclidean distance as a measure of the similarity between observations and groups. The heart of this algorithm is the for-loop which keep updating the centroids' positions until an optimal position of the centroid is reached [329]. K-means clustering were used in the study conducted by Kuo et al. in [47] to classify the LED chip regions into; pad area, luminous zone and background. The average gray level of LED appearance structures serves as the cluster center of the K-means clustering method, where the number of clusters K is set to three (which corresponds to the number of regions). Beside using Gaussian EM algorithm to classify both linear patterns and elliptic patterns, and spherical shell algorithm to classify ring patterns in WMs, Wang et al. in [210] combined K-means and hierarchical clustering techniques to identify the different defect patterns when both convex and non-convex clusters simultaneously occur on the wafer. The combination of the two clustering techniques can overcome the problem of specifying the number of clusters to execute the algorithm.

Clustering algorithms can also be developed based on probability models. The term model is often used to represent the type of restrictions and geometric properties of the covariance matrices. Unlike previously mentioned clustering techniques, model-based clustering algorithms tries to optimise the fit between the data and models used for clusters, where data are viewed as generated by a mixture of probability distributions in which each component represents a different cluster [330]. Model-based clustering has many advantages in classifying wafer defects over other clustering approaches (that cluster aggregated local defects), such as identifying defect clusters simultaneously and obtaining spatial pattern information in the ICs yield model [222]. Hwang and Kuo in [218] used model-based clustering along with a spatial nonhomoge-

neous Poisson process, the bivariate normal distribution and the principal curve to classify WM defects according to their defect generation mechanisms. Their proposed algorithm was able to identify complicated defect patterns with fewer parameters. Yuan et al. in [219, 220] investigated amorphous/linear and curvilinear WM defect patterns simultaneously, and modelled them using multivariate normal distributions and PCs, respectively, extending the traditional model-based clustering approach by considering the mixture of two different probability densities. However, their approach is mainly based on simulated results and lacks the capability to detect closed-ring shaped patterns that have been widely observed in WM defects. Furthermore, their approach is computationally intensive when the number of defect clusters is relatively large [222]. Yuan et al. in [222] used clustering for multi-stage classification of spatial defect patterns in WMs. First, they used k-NN to classify the local defect from the global defects and for noise removal. The local defects filtered from the previous step is then grouped into different clusters using similarity-based clustering method, which is well-structured procedure based on a simple total similarity objective function and is considered a robust method in terms of its initialization, outliers and the ability of detecting clusters with different shapes. Finally, pattern identification along with fine tuning were then used for each of the local defect clusters (e.g. linear, curvilinear) via various model-based techniques. The proposed method for classification were compared to the model-based clustering approach used in [219] and it has been found that they were able to detect more clusters for three chosen WM samples. However, in [219] they were able to detect more clusters for one of the samples. Despite of the good results achieved, their proposed approach faced some limitation such as it uses location information only to analyse defect clusters. Furthermore, if two or more clusters are close to each other or polarity overlap, the proposed algorithm will not be able to accurately distinguish between them. Finally, the k-NN noise removal approach may not perform well in some situation, as when multiple local defect clusters on the same wafer have different defect densities. Nakata et al. in [243] have also used multi-stage approach to identify failure in WM patterns, identify the causes of the failure, and monitor the failure recurrence. To identify failure in WM patterns, they used K-means++, which is proven to be faster by conventional K-means algorithm for clustering wafer patterns. A pattern mining approach called FPGrowth were then used to identify responsible devices of the failure patterns. FPGrowth is an efficient algorithm for association rule mining. It uses a data

structure called frequent pattern tree (FP-tree) to store compressed information about frequent patterns [331]. Finally, a CNN were used to monitor recurrences of failures. This study has used various methods to reduce the computation time so that it can be applicable for practical application.

A less commonly used clustering algorithm called Dynamic Time Wrapping (DTW). DTW is frequently used in speech recognition applications but can also be used for different classification problems. DTW is able to find optimal global alignment between sequences based on the Levenshtein distance (also called edit distance), it also provides an overall real number that quantifies similarity. Furthermore, DTW is able to correctly re-align one sequence with the other, a process which highlights similarities that Euclidean distance is unable to capture [332]. Jeong et al. in [214] used DTW algorithm to detect anomaly defect patterns of WBM and compared it with nearest neighbor classifier that is based on Euclidean distance. First, they presented the spatial pattern on the WBM using a spatial correlogram, where a spatial correlogram represents the correlation between values of the same variable at different locations. Spatial correlogram gives more useful information for the monitoring of defect patterns that appear on WMs because this can describe spatial dependence, a phenomenon known as spatial autocorrelation. Then they calculated the distance using DTW approach to identify the different defect patterns. However, the proposed approach did not show robustness to some defect patterns such the rotation-variant defect patterns [237]. Furthermore, it could not identify the geometric shape of the defect pattern [333].

Another clustering techniques used to classify defect patterns in WMs called density-based spatial clustering of applications with noise (DBSCAN). In this algorithm, the density associated with certain data point is obtained by measuring the number of neighbor points along predefined radius, where neighbor points with a density above a specific threshold are considered as clusters. DBSCAN have several advantages such as detecting clusters of arbitrary shapes and patterns and the ability of detecting clusters without specifying the number of clusters in advance [334]. Jin et al. in [197] used this algorithm to classify wafer defects, where the Cartesian and polar coordinates of all defective regions and edge die were extracted to be used for calculating DBSCAN parameters. These parameters are the radius of the neighborhood points and the minimum number of points in the neighborhood. In this procedure, the outlier detection and defect pattern

detection can be done simultaneously. Taha et al. in [244] proposed a clustering-based algorithm called Dominant Defective Patterns Finder (DDPfinder) that clusters the patterns of defective chips on wafers based on their spatial dependence across WMs. The algorithm begins by selecting a number of chips randomly across a WM, where the intensity of these chips is greatest at the edges and lowest at the center. This is due to two reasons: (1) the yield in the near-edge region is usually as much as 50% less than the yield in the center region, and (2) the high yield loss in the near-edge region can have a significant impact on the overall wafer yield and fab profit. The chips selected are then used to construct Voronoi regions with the chips selected are the centroids of these regions. The region is considered defected or non-defected based on how dependent are the spatial patterns of their centroid points on the dominant defective centroid points across WMs. By using this technique, the overall time complexity for classifying defects can be reduced significantly, because the centroid point of each Voronoi region will be used as a representative of all chips within the region. This will cause the size of the processed data to be significantly reduced. This technique was compared to other algorithms such as RGRN, MLP, and RBF. It was found that DDPfinder outperformed all the other mentioned algorithms in terms of reducing computation time only.

2.5.3.2.4 Supervised Neural Networks

Artificial Neural Networks (ANNs) can be classified according to the data processing criteria as feed forward and recurrent networks. Furthermore, ANNs can be classified according to the necessity of a training set as supervised and unsupervised networks [335].

One of the most commonly used supervised ANNs are Multilayer perceptron (MLP) networks (also known as back-propagation neural network BPNN), they are also considered as feed-forward networks. The development of back-propagation learning algorithm for determining weights and biases was the main reason behind the popularity of these networks among researchers. MLPs are usually described according to the number of neurons and layers used. For example, a 3-5-4 MLP means that this is a two-layer network that consists of three layers; input layer with three inputs, hidden layer with five neurons and output layer with four outputs.

Another type of supervised neural networks that is widely used called Learning Vector Quantization (LVQ). LVQ are also formed using input, hidden and output layers; however, the hidden layers of these networks are not fully connected to the outputs, which can reduce the computa-

tion time. A modified versions of LVQ, namely, LVQ1 and LVQ2 were also proposed by Kohonen in [336], which further reduce the computation time and increase the stability of the learning process.

Kuo et al. in [47] used two MLPs to investigate five defect types in LED chip. K-means clustering with the aid of NCC was first used to divide the area of the chip into three regions: pad area, luminous zone, and background. The reason behind this step is to classify the defected features according the region. Otsu auto-thresholding method were then used to highlight four features for each selected region and the overall chip; these features are area, perimeter, tightness, and defect rate. The extracted features were then fed to the first MLP that is responsible to decide whether the sample contain fragment chip defect are not. Another 15-10-5 MLP were used to classify the samples according to the remaining four defect types considered. The total percentage of defect recognition according to this method was 97.83%. Lin in [37] used two approaches to classify water-drop defects on LED chip surfaces: wavelet-based multivariate statistical (WMS) and MLP network called wavelet-based neural-network (WNN). In WMS, wavelet features were used to obtain the Hotelling T^2 value such that the distance between defective features can be estimated. The T^2 value is then classified according to upper and lower limits, such that if not in the range this means a defect has been detected. WNN approach uses four wavelet characteristics as the input values of MLP neural network which has two neuron outputs that represent two classes: in-control and out-of-control. The detection rates using WMS and WNN approaches were 92% and 95% with false alarm rates of 5.8% and 7.5% respectively. However, these results were contradicted by another study conducted by the same author in [45] that used the same exact approach with different results obtained. The detection rates for this study using WMS and WNN approaches were 92.4% and 90.8% with false alarm rates of 6.1% and 4.4% respectively. The reason for this contradiction is not explicit; however, since the major difference was in the WNN results, it could be due to different sample images used in both studies for the training process. Su et al. in [252] used three types of ANNs to inspect semiconductor wafer post-sawing quality, which are BPNN, RBF and LVQ. Where the RBF network can be considered similar to BPNN; however, it only consists of one hidden layer and it lack of connection weights between input layer and hidden layer. They achieved optimum results in their study upon using 360 image mask size and 224 input nodes for all the neural network used. The number of hidden

layer nodes in the three ANNs to achieve optimum results were; 20 in BPNN with accuracy of 100%, 15 in RBF with accuracy of 90% and 18 in LVQ with accuracy of 100%. It is clear in this study that BPNN and LVQ showed an outperformance in the inspection when compared to RBF. However, the proposed study is not suitable for high variety of defects. In the study conducted by Aghaeizadeh Zoroofi et al. in [260], BPNN have also showed an outperformance in classifying IC wafer contamination defects when compared to maximum likelihood and maximum distance classifiers. However, these algorithms were used as binary classifiers and did not give many details about the defect. Adly et al. in [229] proposed a novel algorithm that combines a general regression network-based (GRN) consensus learning model with randomization technique to detect defective patterns in semiconductor wafers, where the combination results in randomised general regression network (RGRN). GRN are single-pass associative memory feed-forward type ANNs which use normalised Gaussian kernels in the hidden layer as activation functions [337]. The randomization technique was applied by implementing randomised bootstrap to the original data. Randomised bootstrap technique creates random new subset of data by sampling from the original dataset (with replacement). This method of applying RGRN with randomization technique were compared to other techniques such as MLP and it showed an outperformance compared to them with an average accuracy of 99.8%. Adly et al. in [198] conducted the same previous approach; however, they proposed a data reduction technique based on data partitioning and clustering to simplify the overall algorithm. Voronoi Diagram were used for the portioning of data, where it clusters the whole vector space into smaller Voronoi regions. k-means clustering is used to fetch the centroid of each Voronoi region to be used as representative of all original vectors. By using the centroids, the size of the data is reduced and therefore the computation time of the algorithm is also reduced. Their overall accuracy using this technique were slightly improved compared with the previous method, as they have achieved an accuracy of 99.884%. RGRN algorithm were used again in the study conducted by Tello et al. in [249]; however, they added a CNN model in order to identify and classify both single and mixed defect patterns. This study suggested using a *splitter* based on information gain concept to classify single and mixed patterns separately. The proposed approach achieved overall accuracy of 86.17%. Even though the accuracy achieved were lower than the one conducted by Adly et al.in [198, 229] that used similar methods; however, these two studies considered single defect patterns only and did not

consider mixed defect patters as in this study.

2.5.3.2.5 Unsupervised Neural Networks

Unsupervised neural networks are mostly used for identifying defect patterns in WMs. Recurrent networks are example of unsupervised learning networks. They are called recurrent because inputs to the neurons of these networks come from external input, as well as from the internal neurons, consisting of both feed-forward and feedback connections between layers and neurons. Hopfield neural network (HNN), Self-organising map (SOM) and adaptive resonance theory network (ART) are all examples of recurrent neural networks [338]. HNN can be used as an associative memory and can also help in optimization problems. Chang et al. in [43] used two HNNs to investigate LED die defects. The first one was called Contextual HNN and was used to locate the dies in the sample sub-images. The number of the input neurons for this network were equal to the number of pixels for the sample sub-image. The network detecting criteria must satisfy the fact that intensity distribution of pixels in die region in the sample sub-image must be similar to that of template image. Since the network is used to identify the die locations, the output neurons of this network form a die map of individual dies. Each neuron node represents the state of the pixel digitally, such as if the pixel is located within a die region the neuron state is 1 otherwise it will be 0. Therefore, the die map represent a binary image that has the same dimension as sample sub-images. The regions that represent a die location are considered 1's, and the ones that are not considered regions are considered 0's. The die region are then identified using CCL algorithm. The second HNN is called competitive HNN were used to classify the die image into three classes; light emitting region, p-electrode and background in an unsupervised approach. The number of neurons of this network depends on the gray intensity values and the number of classes. In competitive HNN, winner take all (WTA) approach were applied. The WTA ensure that no two neurons are categorised with two different classes and also guarantees that all gray scales are classified. The die defect inspection algorithm were then applied to compare geometrical and heuristical defects against the selected template. 10% threshold margin were used to compare the sample sub-images with the template image. The method achieved an accuracy of 95% in detecting defective dies. Chen and Liu in [205] used a modified ART called ART1 to recognise pattern wafer defects. Unlike other types of ANNs that are less likely to learn new information without damaging what was previously learnt, ART network has the advantage

of not forgetting after learning [335]. ART1 is considered an unsupervised binary network that accepts binary inputs. In this study, an unsupervised technique were selected as they found it difficult to decide how many clusters of defect spatial pattern to be selected, therefore the learning was accomplished using input data alone. They also gave detailed explanation behind the reason of choosing this algorithm as:

- If the characteristic of a new pattern is quite similar to a previously stored pattern (vigilance test passed), only a slight modification of the knowledge contained in the old patterns will be executed. The characteristics of the old and new patterns can be satisfied and the old knowledge can be properly retained. Stability of the system can be maintained.

- If the characteristics of a new pattern are not similar to all of the previously stored patterns (vigilance test failed), new knowledge for the new pattern will be created. This implies quick learning of a new pattern, or the so-called plasticity.

The network was able to recognise both ring and scratch defects using matching values. They compared the performance of ART1 to SOM. They showed that ART1 outperformed SOM in the comparison. Even though, they did not achieve high matching scores for the detected pattern; however, this study over comes others in predicting the possible defect patterns rather than assuming a pre-defined scenario. Unlike the outcomes achieved in the previous study, Di Palma et al. in [207] showed that ART1 is not adequate due to AND logic usage, whereas SOM provide completely satisfactory results including visually effective representation of spatial failure probability of the pattern classes. However, their method cannot separate ring patterns from the other types of defect clusters [227]. Liu et al. in [206] implemented the same approach in [205], in which they used ART1 to detect the defect patterns in WBM. They were able to detect more patterns than the previous study such as bull's eye pattern and they reached a recognition rate of 95%. However, the low number of WBM provided to them limited their ability to identify further patterns. Choi et al. in [226] conducted a study similar to what have been done in [205, 206]. However, they proposed an advanced ART1 algorithm called "multi-step ART1", where it sequentially uses the modules to classify each pattern separately, instead of training all patterns at once. This algorithm has a specific learning procedure according to the characteristics of each pattern by changing and re-learning the preprocessing of the input data, the threshold decision method, the type of similarity and the vigilance parameter during module

generation. Their proposed algorithm was able to detect more type of defect pattern including mixed pattern compared to the similar studies. Hsu and Chien in [200] developed a hybrid data mining approach which combined ART1 and spatial statistics to quickly extract systematic and random patterns from WBM. Furthermore, a decision tree algorithm was constructed for identifying the root causes of specific patterns. However, due to the vigilance parameter settings used in ART1, some systematic patters were misclassified as random patterns in this study. Hence, the lower the vigilance threshold value, the more patterns are extracted from the maps. On the other hand, a low vigilance threshold value may also cause dissimilar maps to group in the same cluster. Huang et al. in [213] conducted similar study to what have been investigated in [212]. However, cellular neural networks combined with genetic algorithm were used to classify wafer patterns instead. A cellular neural network (not to be confused with convolutional neural network) is an unsupervised neural network that is composed of a massive aggregate of analog circuit components called cells. Setting up a cellular neural network needs a proper selection of circuit parameters of cells. The dynamics of a cellular neural network is determined by the set of circuit parameters, which are collectively called the cloning template. A genetic algorithm was used to settle the cloning template for the cellular neural network owing to its success in the applications of optimization. This study outperformed the previous study in [212] remarkably as it achieved an accuracy of 99.2%. However, the false alarm rate of this study was relatively high. Furthermore, no explanation was provided for the different accuracy results for using MLP algorithm, despite of using same approaches in both studies. Liu and Chien in [228] used different type of ANNs for different purposes to analyse and classify WBM defects. Cellular neural network was used first to eliminate WBM defect noise and enhance the patterns against random errors. Moment invariant and ART1 were then used for pattern clustering and classification respectively, where moment invariant can make the same shape in one cluster, whether the shape's size or position are changed. ART1 can self-lern by characteristic of each WBM. These algorithms were integrated with a user interface system that allows users to execute the following system operations including: loading the analysed dataset, adding or retracting the decision knowledge, controlling the parameters in WBM patterns clustering system, and monitoring the clustering results. Chien et al. in [227] used ART1 classifier as a final classification step to deal with the wafer being classified as having spatial random defects with higher failure percentage and to find

new defect patterns for further classification. The filtered wafer defects that feed ART1 algorithm came from predefined conditions using SPC charts that cannot detect all types of defects (e.g. sparse defects). In SPC chart a set of thresholds were used upper control limit (UCL) and lower control limit (LCL) such that if the failure percentage of the WBM exceeds the UCL the wafer will be classified as a defect, otherwise if it fall below the LCL will be checked for further defects using various filters and ART1 classifier as mentioned before. Chang et al. in [254] proposed an automatic inspection approach for classifying wafer defects into four classes based on a SOM. The feature vector containing GLCM parameter and colour information were used as an input to the network. The inspection results achieved a sensitivity value of 0.967 and a specificity value of 1.

Unsupervised neural networks involving SOM, ART, and Cellular Neural Networks cannot identify two shift-variant or two rotation-variant defect patterns that in fact belong to the same failure cause (i.e. distinguishing between two parallel scratches or between a top zone and a bottom zone) [211].

2.6 Deep Learning in AOI

Deep learning (DL) concept involves the usage of Deep Neural Networks (DNN), which can handle feature extraction and classification methods in machine vision problems. DNNs learn new useful representations from available features that capture essential statistical regularities present from data itself, then the representation features can be formulated for classification, regression and specific problem in information retrieval. The independence from prior knowledge and human effort in feature design is a major advantage for DL in general and DNNs [339]. For AOI applications reviewed in this paper DNNs are implemented using the following methods: Autoencoders (AEs) and Convolutional Neural Networks (CNNs).

Yu in [234] proposed detection and recognition system for detecting WM defects based on a modified version of SDAE called enhanced stacked denoising autoencoder (ESDAE). They have also integrated manifold regularization in the learning procedure which improve the algorithm's performance effectively due to the preservation of intrinsic information in the data. The overall detection accuracy of the proposed method reached 89.6% which overcame other methods used for comparison such that SDAE, DBN, BPN and LR. However, due to the large number of

parameters used, deep autoencoders in general are easy to suffer from overfitting with small size sample, which limits the generalization ability of the deep autoencoders to learn effective features [340]. A similar approach was also proposed by Lee et al. in [341] to assess wafer's quality. However, sensor data were used instead of WM, which is beyond the scope of this thesis. In a similar study that proposed detection and recognition system for detecting WM defects, Yu et al. in [232] integrated SDAE and CNN to form a new DL model called stacked convolutional sparse denoising auto-encoder SCSDAE. The new model proposed can learn effective features and accumulate the robustness layer by layer, which adopts SDAE as the feature extractor and stacks well- designed fully connected SDAE in a convolutional way to obtain much robust feature representations. The challenges faced this study and others that use same dataset is the imbalance of data, in order to handle this problem under-sampling and over-sampling methods are generally used to against highly imbalanced datasets. Under-sampling randomly eliminates majority class examples, while over-sampling increases the number of instances in the minority class. Both of them aim to obtain approximately the same number of instances of the classes. The detection accuracy rate in this study was 95.13% in the simulation case and 94.75% in the industrial case. The results were compared to other classification algorithms such SDAE, AlexNet, DBN, SVM, BPN, k-NN and C4.5 and outperformed all of them in the simulation and industrial cases. However, the problem of limited training data will lead to overfitting of SCSDAE. Moreover, due to labeled data scarcity for some patterns in WMs, it is a challenging issue for SCSDAE to implement pattern recognition on imbalanced dataset.

Kim et al. in [74] proposed an algorithm based on CNN with stacking ensemble model and MLP, to classify four types of TFT-LCD defects. In their investigation, they have noticed that the size of defects is very small compared to the patterned panel in the background. Therefore, they proposed a feature extraction scheme to eliminate the patterns in the background for better classification results by applying three steps: find the pitch of each panel image, calculate the pitchwise difference score for each pixel and apply thresholding. An algorithm was proposed in this study to check the efficiency of using the pattern elimination technique by using several MLP and CNN models. They have achieved an overall accuracy in their classification of 86.13%. However, they did not mention the time needed for the algorithm as they used too many models which may affect the computational time. Mei et al. in [60] proposed a method that combined

handcrafted features and unsupervised DL-based features to detect Mura defects in TFT-LCD. The method was named joint-feature-representation-based defect recognition framework (JFR-DRF). Various unsupervised algorithms were compatible for the unsupervised DL-based feature extraction part; however, SAE algorithm was used to demonstrate its ability. The handcrafted features were extracted with methods similar to GLCM technique that requires coefficients such as entropy and anisotropy coefficients based on the grayscale histogram. Then, the features extracted by these parts will be merged into a single feature vector, which acts as the representation for the Mura defect image. After subsequent processing, it will be classified using one-against-all SVM classifier. In this study, it was proven from statistical results that combining handcrafted and unsupervised feature extraction outperformed the performance of using only one of them. However, fused feature representations may also have the risk of being affected by poor representations from modalities which are not appropriate for the tasks. Moreover, DL-based methods can require considerable computational resources and time to perform training and inferencing [62].

Lin et al. in [51] used CNN (called LEDNet) for inspecting LED chips defects. They considered two defects in their analysis, line blemishes and scratch marks. The LEDNet they used consisted of six convolutional layers, three max-pooling layers, an average pooling layer and a fully connected layer. Average pooling were used to highlight the importance of each feature and relate them to the correct class (line blemishes or scratch) with the aid of associated weights and class activation mapping in following fully connected layer. Due to the lack of real defected LED chip sample images and since CNN needs large amount of data to be effective, the dataset provided to train the CNN were generated using *geometric transformation technology* that simulates the defects with adjusted images. Rotation, flipping, shift, noising and blurring effects were carried out randomly on the adjusted images to mimic the real situation. The inaccuracy rate of classifying the defects for chip 1 and chip 2 were 5.04% and 5.51% respectively. Their proposed approach were compared to the algorithm used in [47], they got 14.53% and 11.97% inaccuracy rates for chip1 and chip2 respectively which more than twice the inaccuracies used in LEDNet. Even though this study got considerably low inaccuracy results; however, their algorithm were limited to classify only two kind of defects. Furthermore, there was no evidence that their adjusted images for the training set can simulate the real case for defected chips. Yang et al. in

[40] used CNN for the purpose of assessing the diameter of the LED cup aperture. Since no feature extraction needed to feed the CNN, the images were directly used as the input neurons of the network. The network consisted of five convolution layers and two fully connected layers. Training samples for the network were used from the captured images, each sample of the image were rotated 90° four times and horizontal flip were applied to each image to further increase the training samples for the network. The detection rate for unqualified cups was 100% with false alarm rate of 8%. However, the need for a large labelled dataset may limit its application [342]. For semiconductor wafer inspection, Kyeong and Kim in [245] used CNN in their classification, in which they considered the WBM as an input binary matrix of 0s and 1s that have a size of 56×53. In their study three convolutional layers, three pooling layers and a fully connected layer with ReLU and softmax activation functions have been used. The CNN model that they have proposed for classification was designed to detect multiple defects on same wafer. In order to do so, they proposed four individual CNN classifiers for each defect class, in such a way that each classifier determines whether the corresponding pattern exists when several defect patterns are mixed over a wafer. The main drawback of this approach is that to detect a mixed pattern defect, the output of each individual classifier has to be obtained [343]. A similar approach was proposed by Cheon et al. in [275]. The proposed CNN model in this study can extract features from the real wafer images that are acquired using SEM and accurately classify the input data into five different wafer defect classes. Their model can also classify the unknown defect classes by combining the CNN model with k-NN classifier. However, the data used in this study were highly imbalanced and no solution were proposed to mitigate the outcomes of this issue [344]. Nakazawa and Kulkarni in [247] also used CNN with three convolutional layers, three pooling layers and two fully connected layers for wafer inspection. The input wafer image size was 286×400. To evaluate the CNN performance 28,600 WBM images were generated in addition to 1191 real wafers for 22 defect classes which improved the test accuracy up to 98.2%. However, there was high reliance on simulated data for training and validation of the CNN model because real data was highly imbalanced. Furthermore, this study does not provide any information about the defect cluster, size and its location. In a later study conducted by the same authors in [248], Nakazawa and Kulkarni overcame the lack of information problem using different type of CNN models such as Fully Convolutional Network (FCN) [345], SegNet [346] and U-Net [347]. These

models were compared to each other with regards to their classification performance and training time.

Yu et al. in [233] constructed an algorithm based on CNN to recognise and classify WM defects. The proposed method was divided into two parallel approaches: offline modelling and online processing. In the offline modelling stage, a wafer defect pattern recognition model that consisted of two sub-models was constructed. These two sub-models were basically two CNNs, one of them was responsible for wafer defect pattern detection and the other for wafer defect pattern classification. Both CNNs consisted of three convolutional layers and three pooling layers, while the detection CNN consisted of two FC layers and the classification CNN consisted of three FC layers. Dropout method were used in both CNNs to avoid overfitting. Furthermore, PCA method were used in the classification CNN after the second FC layer to reduce the feature dimension and then features were normalised. In the online processing stage, the WM is pre-processed and used as an input to both CNN models. It was noticed in their study that the detection CNN has a simple structure and a fast computing speed compared with the classification CNN due to low probability of process faults in the actual manufacturing process. Their proposed method achieved an average detection accuracy of 93.13%, which outperformed the accuracy of other algorithm that were used for comparison. However, using PCA for feature reduction is redundant since CNN has the ability of considering the important features and discarding the unnecessary ones.

Most of the researchers that investigated WM defects using CNN are using simulated data to solve the problem of noisy and imbalanced data for WM pattern defect identification. In this way, they can improve the classification accuracy, but it becomes costly and time-consuming. Also, denoising the actual data can destroy the actual defect patterns on the wafer images [201].

Cai et al. in [153] inspected the SMT solder joint defects in PCBs by using three types of CNNs. The entire sample images were fed to the first CNN (called CNN-1). Sub-regions which represent the ROI were selected from the entire images manually and fed to the second CNN (called CNN-2). The output for CNN-2 were used as an input for a third CNN (called CNN-3). The outputs for both CNN-1 and CNN-3 were used for classification of the SMT solder joint (Good or bad). The concept of the classification depends of the weighted some of both CNNs and the probability of each. The results of the inspection using this method was compared to

other methods such as Bayes, SVM, G-MLP, GMM and ViBe. It was proved by experimental results that the proposed method overcame the others by remarkable percentage. However, their approach classified the solder joints as good or bad and did not consider classifying the defect type, which does not justify this study to use such a complex algorithm that is optimised for multi-class classification. Yuan et al in [121] proposed a CNN-based approach to classify IR hole defects in mobile screen glass. In order to overcome the lack of training data, they proposed a data generation algorithm that consists of two steps, defect superposition, and data augmentation. The superposition step overlays defects on a chosen non-defected image in randomly selected locations with stochastic sizes, shapes, and severities. The defect images generated from single one template would naturally have the same scale, shape, and background colour, making them not accord with the actual situation. Therefore, the augmentation step is applied followed the superposition step to augment the variety of the generated defect images. Finally, considerable manual labelled real images were collected and used to verify the proposed detection method.

Due to complex and deep structure, CNN requires comparatively large-scale image data, a lot of hyper parameters tuning, high computational power (such as parallel GPUs), and long training time because of the large amount of labelled data used [201]. To overcome these problems, Yang et al. in [62] proposed transfer-learning-based approach called online sequential classifier and transfer learning (OSC-TL) to investigate Mura defects in FPDs. Unlike conventional CNN, which assumes that training data and future data must be in the same feature space and must have the same distribution, transfer-learning algorithm reuses the feature extractor portion of a previously trained network using existing large datasets and retrains only the classification functionality using specific datasets appropriate for different classification tasks. The proposed OSC-TL uses a pretrained CNN using ImageNet LSVRC-2012 dataset for feature extraction step. Then two more modules are used for training and classification of the defects. The proposed algorithm was compared to other DL approaches and showed remarkable overperformance in terms of the computational time, while maintaining similar accuracy levels.

Tables 2.10,2.11,2.12,2.13, and 2.14 summarise the inspection algorithm details for selected research articles.

Table 2.10: Inspection algorithm details for selected articles that investigated wafer defects

Ref.	Feature Extraction & Selection Method	Main classification algorithm	Results		Inspection Time	Other algorithms (or studies) compared with main algorithm
			Detection Overall Accuracy	False Alarm Rate		
[205]	ART1	ART1	Up to 98.73%	-	3s/35samples	SOM
[260]	Thresholding & gray-level values	BPNN & MDC & MLC	95.8%(BPNN) 87.5%(MDC) 87.5%(MLC)	-	-	-
[206]	ART1	ART1	Up to 95%	-	3s/20samples	-
[252]	Thresholding	BPNN & RBF & LVQ	100%(BPNN) RBF(90%) 100%(LVQ)	-	<1s/die	-
[261]	Region growing & surface extraction	MLP	96.5%	2.5%	-	-
[200]	ART1	ART1 & Decision Tree	Up to 100%	-	<1min/pattern	-
[214]	Spatial correlogram	DTW clustering	90 - 100%	-	-	-
[219]	Spatial non-homogeneous Poisson process	Model-based clustering	96.6%	-	75s/wafer	[218]
[220]	Spatial non-homogeneous Poisson process	BCSF & BCNHPP	86.5-95.8% (BCSF) 86-95.5% (BCNHPP)	-	28.6-36.6s (BCSF) 276.8-375.3s (BCNHPP)	-
[225]	Angle & distance variations	Multi-class SVM	91.3725%	-	-	RBF
[217]	gHMT & DWT	MLC	Up to 100%	-	-	WT & HMT
[216]	GLCM (moment invariant)	SOM-SVM	>90%	-	-	SOM-BP
[267]	Template matching	Rule-based	98.9%	-	-	-
[268]	Segmentation	Rule-based	98.9% (Pattern recognition) 95.6% (Defect criticality)	-	-	-
[255]	Template matching & 2D DWT	Rule-based	-	-	0.91s/chip	-
[256]	Template matching & 1D FFT	Rule-based	-	-	0.62s/chip	2D FFT
[222]	nearest-neighbor clutter	Clustering	96.4%	-	15.73%	[219, 220]
[202]	SDC	Rule-based	Up to 90%	1 - 3%	<3s/wafer	-
[226]	Zooming strategy & ART1	Multi-step ART1	84 - 100%	-	≈6s (total)	ART1 without zooming
[199]	HT	C4.5 & Logitboost	92%	-	-	NBC & bagging
[257]	-	SVM	95.6%	-	-	PLS, GRN, C4.5 & k-NN
[194]	Polar Fourier transform & RMI	ADTree	Up to 95%	-	10mins (Training time)	Bayesian and other modified decision tree classifiers
[236]	Morphological Operations	SVM	95%	5%	173s (Total)	Anomaly correlation
[258]	Feature point matching	Rule-based	94%	-	0.26s/24feature points	Subtration, NP subtraction, NCC & eigenvalue
[238]	Distance & statistical features	Random forest	>99%	-	-	-
[273]	Edge detection & template matching	Rule-based	97.4%	-	-	-
[247]	CNN	CNN	98.2%	-	0.13s/WM	-
[245]	CNN	CNN	97.4% (Moderate noise) 91%(severe noise)	-	-	SVM & MLP
[231]	Randon transform	Decision tree	90.5%	-	-	C4.5, ANN, Logistic & SVM
[249]	Splitter	RGRN & CNN	86.17%	-	-	RGRN, DSCN, RF, MLP, SVM & RBF
[244]	Voronoi regions	K-means & DDPFinder	>92%	-	≈10s (total)	SSRN, RGRN, GRN, SMO, PNN, MLP, SVN & RBF
[242]	Thresholding	Decision tree	95.6%	-	-	Bayes classifiers
[275]	CNN	k-NN & CNN	96.2%	-	1.813s (testing)	SAE, MLP & SVM
[234]	DNN	DNN (ESDN)	89.6%	0.05%	0.0294/WM	SDAE, DBN, BPN, & LR
[232]	CNN	Hybrid CNN (SCS-DAE)	94.75% (Industrial) 95.13% (simulation)	-	-	SDAE, AlexNet, DBN, SVM, BPN, k-NN, & C4.5
[233]	PCA	CNN	93.25%	-	2.1-5.7s/sample	SVM, C4.5 & [203]
[250]	Rotation invariant	MLP	92.3%	-	-	Boosting, RF, k-NN, & SVM

Table 2.11: Inspection algorithm details for selected articles that investigated PCB defects

Ref.	Feature Extraction & Selection Method	Main classification algorithm	Results		Inspection Time	Other algorithms (or studies) compared with main algorithm
			Detection Overall Accuracy	False Alarm Rate		
[126]	Geometric & wavelet	MLP & LVQ	98.8%(MLP) 95.4%(LVQ)	-	-	k-NN
[142]	Geometric & wavelet	MLP	99.5%	-	1.3s (total)	LVQ
[132]	Projection profile, moment, zoning & contour profile	MLP	100%	0%	1.844 - 2.093s (feature extraction)	-
[145]	Reflection-area-based	Rule-based	95%	-	-	[348]
[135]	Segmentation	Rule-based	97.3%	2.7%	-	-
[184]	Otsu & entropy	Rule-based	96.43%	-	0.7s/image	-
[185]	Edge detection, colour information & MLP	MLP	96%	-	1.05s/image	-
[146]	-	LVQ	Up to 100%	0%	-	-
[175]	Template matching	Rule-based	Up to 100%	-	125s (total)	-
[143]	Geometric & wavelet	Neuro-fuzzy	97.8%	-	-	MLP & LVQ
[162]	Template matching & PSO	Rule-based	Up to 100%	-	≈1min/image	GA
[158]	DWT & Template matching	Rule-based	76.2 - 96.7%	-	8 - 183ms/image	DCT
[136]	Geometrical & colour features / Advanced adaboost	CART	97.2%	≈0.6%	8.6ms/IC chip	Conventional adaboost
[137]	Shape, digital & logical	Rule-based	98.6%	≈1.45%	-	[126, 141, 349]
[144]	Geometric features	Fuzzy rules	97.83%	-	-	[143]
[176]	Template matching & edge detection	SVM	93.5%	-	124ms/image	[175]
[156]	BICF	Nearest neighbor	97.5%	-	-	LLSTA, LPP, NPE, LDA, DLA & PNDA
[163]	Template matching (NCC) & PSO	Rule-based	Up to 100%	-	3.19 - 28.41s/3 resistor-network	-
[166]	PICC	Rule-based	Up to 100%	0%	0.3s/image	NCC
[148]	NCC & colour features	Bayes & SVM	100%	-	0.53s (SVM)	Decision tree, k-NN & BPNN
[139]	Segmentation & template matching	Rule-based	97.96%	2.61%	-	[135, 138, 145, 348, 349]
[149]	NCC & colour features	BPNN	98.46%	0%	-	Decision tree & SVM
[140]	Region, evaluation, & colour grads features	Rule-based (Boolean rules)	97.7%	-	11s/PCB	[137, 138, 144, 145]
[10]	LDIM	Rule-based	100%	0%	-	-
[151]	ViBe	Rule-based	Up to 100%	0.9%	0.52s inspection	SDL, SAM & G-MLP
[152]	RPCA	Rule-based	Up to 100%	0.72%	-	ViBe, SAM & SDL
[189]	Modified HT	Rule-based	97.89%	-	160ms/BGA	Ground truth, Samsung SMT482 & HALCON
[190]	Line-based clustering	Rule-based	100%	-	<200ms/BGA	Samsung SMT482
[164]	Model-based	Rule-based	97.73 - 99.46%	0.54-2.27%	13.7-16.8ms	HALCON & SM482
[154]	Adaptive template	Rule-based	100%	0%	-	ViBe, SDL, SAM& RPCA
[181]	Systematic matrix	Rule-based	Up to 100%	0.05%	1.64s (total)	[46, 180, 350, 351]
[178]	Textural features & PCA	SVM	81.39%	-	-	Bayes, MLP, decision tree, logistic regression & gradient boost
[177]	SURF	Random forest & WKDE	≈91%	-	-	-
[193]	Model-based	K-means	100%	-	-	-
[127]	INCC	Rule-based	>96%	-	-	NCC
[130]	Regional and colour features & GA	SVM, MLP & decision tree	94.9%(SVM) 94.6%(MLP) 43.6%Decision tree	-	3.7ms/component	-
[150]	Same in [352–354]	K-means, FCM, & Mean-shift	98 - 100%	2.2 - 12%	-	-
[134]	Template matching	Rule-based	-	-	31ms/image	[355, 356]
[182]	Template matching & FFT	Rule-based	100%	0%	0.038s/16subimage	-

Table 2.12: Inspection algorithm details for selected articles that investigated FPD defects

Ref.	Feature Extraction & Selection Method	Main classification algorithm	Results		Inspection Time	Other algorithms (or studies) compared with main algorithm
			Detection Overall Accuracy	False Alarm Rate		
[57]	DCT	Rule-based	-	-	<75ms/image	-
[72]	NCC	Rule-based	-	-	18μs/line image	-
[93]	KPCA	ISVM	96%	-	<1s (total)	k-NN
[63]	T^2 Hotelling & ACO	BPNN	Up to 100%	-	-	Otsu & without ACO
[357]	NCC & ICA	Rule-based	-	0.17%	2ms/image	Fourier analysis
[85]	Thresholding	RBF, BPNN, LVQ1 & LVQ2	87%(RBF) 93%(BPNN) 93%(LVQ1) 97%(LVQ2)	-	<0.5s/defect image	-
[92]	GLCM	Fuzzy-SVDD	Up to 98.9%	2.23-5.29%	11s (total)	SVDD
[65]	Regression diagnostics	Rule-based	-	0%	0.8s/image	[54]
[81]	Adaptive thresholding	Rule-based	78% (precision)	-	>0.203s/image	-
[64]	ICA	Rule-based	-	-	8.1ms/image	-
[67]	Optical flow	Rule-based	-	-	0.054s/image	-
[66]	Modified HT	Rule-based	-	0%	0.696s/image	HT
[96]	Segmentation	F-SVDD	95%	-	7.8s/panel	SVDD
[96]	Segmentation	QK-SVDD	96%	7.54%	60ms/panel	SVDD
[89]	PCA	Rule-based	-	-	0.032s/image	FFT
[68]	LBF	Rule-based	>99% (precision & recall)	0.63%	6.61s (total CPU time)	PF, ICA & MCV
[295]	Thresholding	MLP	-	-	<1s/panel image	-
[97]	Segmentation	Tagushi-BPNN	94%	-	-	-
[118]	Segmentation & PCA	Rule-based	90%	6%	<0.53s/image	-
[99]	GLCM	BPNN	83.3%	-	-	-
[84]	DWT, GLCM & PCA	SVM	89.5%	-	-	MLP & CART
[98]	Template matching & segmentation	BPNN & MDC	93.7%(BPNN) 96.8%(MDC)	-	-	-
[123]	Gray-value & OMP	Rule-based	91.5%	12%	1.15s/image	-
[124]	Segmentation	Rule-based	94 - 99%	-	5.1s/defect	-
[78]	Haar WT & Thresholding	OCSVM	91.7%	-	-	GA & PCO
[28]	colour matching	Rule-based	71.10 - 96.91%	-	2.07s/image	[65, 68]
[115]	NDF & GLCM	Random forest	87 - 98%	-	0.07-0.11ms/defect	DCT, RPCA & Otsu
[120]	CR & IFCM	Rule-based	-	-	1.6601s (total)	Kittler, Niblack & Otsu
[119]	Multifractal features	SVM & ECS	96.61%	1 - 2%	4.4268s (testing)	SVM, B-SVM, US, ROS, SMOTE, GMM, k-NN & Adaboost
[116]	local inlier–outlier ratios augmented with modified LBP	A-SVDD	>96%	≈0.5%	11.3-20.2s training/dataset	[58, 76, 358–360]
[60]	JFR-DRF	SVM	Up to 95%	-	-	Handcraft & K-means
[69]	MSCDAE	MSCDAE	⩽92.04%(Precision)	-	-	-
[361]	FFT & GLCM	BPNN & RBFNN	98.9%(BPNN) 97.3%(RBFNN)	-	2.57s/image	-
[61]	OPBC & RGLS	Rule-based	Up to 100%	-	8.12s (total)	[59]
[62]	OSC-TL	OSC-TL	≈92%	-	1.5ms/image	DNN+ELM, DNN+SVM & DNN+MLP
[121]	CNN	CNN	98.26%	0.588%	-	-
[122]	Regularised Autoencoder	Regularised Autoencoder	-	-	313ms/image	Segmentation & SVD
[117]	Thresholding	Rule-based	94.2%	-	0.068 - 11.163s /image	DWT, DCT & linear regression
[75]	MS-FCAE	MS-FCAE	98.1%(precision) 92.1%(recall)	-	19321ms/dataset	LCA, PHOT, TEXEMS, ACAE, RCAE & MSCDAE
[110]	Segmentation	SVR	≈80%	-	<10ms/defect	-
[74]	Thresholding	MLP & CNN	86.13%	-	-	-

Table 2.13: Inspection algorithm details for selected articles that investigated LED defects

Ref.	Feature Extraction & Selection Method	Main classification algorithm	Results		Inspection Time	Other algorithms (or studies) compared with main algorithm
			Detection Overall Accuracy	False Alarm Rate		
[38]	Segmentation	Rule-based	90%	-	200ms	-
[37]	Wavelet transform & T^2 Hotelling	WNN & WMS	95%(WNN) 92%(WMS)	7.5%(WNN) 5.8%(WMS)		Otsu-thresolding
[43]	CCL	HNN	95%	-	22ms/die	BPNN, LVQ, RBF, K-means, FCM
[39]	BDCT	Rule-based	95.8%	4.2%	0.82s/image	WNN and Otsu method
[36]	Otsu-thresholding	Rule-based	95%	7.1%(Type1) 14.7%(Type2)	0.3s/image	-
[44]	Segmentation	RBFNN	92.5%	-	8.87ms/die	LVQ and K-means
[46]	Optical flow	Rule-based	100%	0%	12ms/image	NCC and Lucas-Kanade
[31]	PCC	One-class SVM	100%	0.13%	1 - 10ms	-
[34]	BDCT	Rule-based	99.67%	0.13%	0.32s/image	Otsu method
[47]	NCC and Thresholding	K-means & MLP	97.83%	-	-	-
[41]	Otsu-thresholding	Rule-based	94.34%(LED_{1S}) 95.17%(LED_{1B}) 96.02%(LED_2)	-	-	-
[49]	NCC and Segmentation	Rule-based	Up to 99%	Up to 2%	0.9ms/chip	-
[40]	CNN	CNN	100%	8%	<1s/cup	-
[51]	CNN	CNN	94.96%	-	-	[47]
[50]	Correlation coefficient & PCA	SVM & DTSVM	96%	-	1s(SVM) 1s(DTSVM)	-
[42]	Thresholding	Rule-based	98.25%	-	-	-

Table 2.14: Inspection algorithm details for selected articles that investigated miscellaneous defects

Ref.	Feature Extraction & Selection Method	Main classification algorithm	Results		Inspection Time	Other algorithms (or studies) compared with main algorithm
			Detection Overall Accuracy	False Alarm Rate		
[284]	T^2 Hotelling & DWT	Rule-based	93.75%	6.4%	≈3s(total)	-
[283]	T^2 Hotelling & DWT	Rule-based	95%	3.2%	0.8s/image	-
[285]	DCT	Rule-based	94.74%	15%	4s/DCT	-
[277]	Segmentation	MLP & Rule-based	99.6%	-	5s/defect	-
[278]	Segmentation	CART	94%	5%	-	-
[280]	Template matching & DWT	Rule-based	96.44%	-	1.05s/sample	-
[279]	Segmentation & similarity matching	SVM	Up to 100%	4.2%	-	-
[286]	NSCT & adaptive thresholding	Rule-based	98.7%	-	0.1s/capacitor	-
[287]	Segmentation	MLP & LVQ	98.5%(MLP) 91.27%(LVQ)	-	0.318s(MLP) 0.297s(LVQ)	-
[281]	Segmentation	SVM	95.83%	-	15s/image	-
[282]	HT	SVM	93.44%	7.69%	0.051s (testing)/image	k-NN & ANN

2.7 Conclusion

Quality monitoring is essential step for minimizing product's defects in various industries. AOI is considered one of the simplest and commonly used quality monitoring approaches used for automatic industrial inspection. The field of AOI is vast as it involves a variety of subjects ranging from hardware setups for image acquisition to inspection and decision-making algorithms. For this reason, the research opportunities in this field are wide open and have the potential to improve in the near future.

Many electronics manufacturers are integrating AOI systems in their production steps to avoid human inspectors' extra cost and possible errors. The majority of these manufacturers are located in Asia. In this study, more than 300 articles that used AOI systems and algorithms to investigate various defects in electronics industry were reviewed and critiqued. During the searching process for relevant papers to be reviewed, it has been found that AOI systems are widely used in four major applications: semiconductor wafers, FPDs, PCBs, and LEDs defect detection. Various types of defects were considered for each inspected component. Semiconductor wafer defects are usually inspected according to their distinguished defect patterns. Solder joints and IC component defects are commonly inspected defects in PCB industry. TFT-LCD defects are widely investigated using AOI among other types in FPD industry. Unlike previously mentioned electronic components, it was observed that there are more variety of defects investigated using AOI in LED industry such as apertures, LED chips, SMD-LED and cosmetic defects. Other miscellaneous electronic components were also considered in this review such as camera modules, thermal fuses, and passive electronic components.

Hardware and software setups play vital role in the quality of the acquired image for inspection. It was found in this chapter that these setups can also affect the accuracy and false alarm rates of the final classification results. Environmental and variable illumination settings are avoided in AOI systems since they increase noise and false alarms. Customised illumination can be used in some applications for better defect description such as using tiered illumination for PCB solder joint inspection to capture the 3D nature of the inspected component. In this chapter, the advantages and limitations for different illumination settings used have been highlighted. Selecting the right image sensor and lens for inspection is also important. Many factors are considered in the selection process such as size of the inspected component, filed of view,

frame rate and resolution. While most of image samples are acquired using image sensors alone, auxiliary systems such as OCT and SEM have been also used to aid the process. Thermography and X-ray can be also used to acquire sample images for certain applications such as in PCB and BGA inspection. Circuit probe inspection is widely used to investigate semiconductor wafer defects such that WM images can be generated for defect classification.

Inspection algorithms that process the acquired images were also introduced. Three main stages are used in applying the inspection algorithm: preprocessing, feature extraction & selection and classification. In preprocessing, several filtering, denoising and geometrical operations are applied to the acquired image. This is very useful in removing the unwanted noise that affects the classification decision. Denoising is essential step in removing random defects from WM patterns such that inspected samples can be classified according to the systematic defect. In the feature extraction & selection, the algorithm tries to isolate and locate the defective features in the image. Several statistical and geometrical measurements are also used to assess the features of defect. Segmentation and template matching are two well-used feature extraction algorithms according to the reviewed papers. Feature selection process can reduce the number of redundant extracted features such that the algorithm performs much faster. PCA is one of the mostly used algorithms for feature selection. Finally, classification is the decision-making stage of inspection algorithm. Rule-based and learning-based are two classification approaches used in the reviewed papers. In rule-based, a set of pre-programmed if-then rules are used to assess the features acquired from the previous step. In learning-based classification, supervised and unsupervised machine learning algorithms such as k-NN, Bayes, decision trees , SVM, clustering and ANNs are used for the classification process. Unsupervised machine learning algorithms are widely used in finding WM patters to investigate semiconductor wafer defects, while supervised learning techniques are used in the rest of studies. In general, machine learning algorithms overcome rule-based learning by their ability to find anomalies in the input data (features) based on the training process and without any need of customised programming. However, overfitting and imbalanced data are two major drawbacks in using learning-based algorithm. These two problems can be solved by having sufficient data for each defect class considered or by applying several techniques such as regularization and bootstrapping.

Many recent articles have also used DL algorithms in investigating defects. The recent

development of CNNs such as AlexNet opened the door for these models to be used in AOI tasks. DL models can do preprocessing, feature extraction & selection and classification all at once within their hidden layers. In DL layers, many complex features can be captured that traditional feature extraction approaches may fail to recognise. The detection accuracy can be also improved using DL due to the large number of parameters used in training the models. Despite previously mentioned advantages, DL models are considered data hungry. Furthermore, it requires large processing and GPU capabilities. The large data required for training are essential in providing high accuracy and to avoid overfitting. Several data augmentation approaches are used to address this issue such as using GAN.

Chapter 3

Deep Learning for Computer Vision - An Overview & Evaluation

"*A learning machine is any device whose actions are influenced by past experience.*"

Nils John Nilsson

3.1 Overview

As mentioned in the previous chapter, there are various ways in conducting AOI using computer vision in electronics industry. The conventional methods uses manual feature extraction that needs expert's knowledge in customising the algorithm to detect the required features. These methods are time consuming and are overly tailored for detecting specific features as they are coded using rule-based programming. We also showed in Section 2.6 that using deep learning (DL) for AOI purposes in electronics industry can reduce the burden of extracting the features manually by letting the algorithm to learn how to differentiate between different features. It also saves time and expert knowledge in customising a rule-based model for classification when complex feature are to be detected. The main purpose of this chapter is to explore the DL algorithms used in computer vision applications and evaluate them using common dataset i.e. MNIST. This will give some sense when applying them in the main focus of this research which is inspecting semiconductor and optoelectronic wafer defects as will be discussed in Chapters 4 and

5 respectively. The discussion in this chapter will begin by differentiating between three concepts, namely, artificial intelligence, machine learning and DL as in Section 3.2. Section 3.3 will present the basic building block of any neural network which is the perceptron. The remaining sections will present the commonly used deep networks for computer vision applications.

3.2 Artificial Intelligence, Machine Learning & Deep Learning

According to John McCarthy in [362] "artificial intelligence (AI) is defined as the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable." Based on the previous definition, AI does not necessarily allow the computers to learn from experience. For instance, a rule-based classifier (discussed in Section 2.5.3.1) can be considered an AI model; however, these models are pre-programmed by an expert to follow certain rules (e.g. conditions).

Machine learning is considered a subset of AI (Figure 3.1) that provides the ability of the algorithm to learn the correlation between inputs and outputs. It can be used to perform certain classification and regression tasks. Classification tasks involve categorised outputs (e.g. defected, not defected), while regression involves continuous value outputs (e.g. 95.2%, 84.6%). Classification is considered the commonly used method in industrial and inspection applications. In general, machine learning algorithms can be divided into 3 categories: supervised, unsupervised, and reinforced learning. In supervised learning the number of outputs is predefined, and the algorithm is trained to predict the value of these outputs based on certain data called training data. The training data are usually provided by human expert that also perform the training process. In manufacturing and inspection applications, supervised machine learning techniques are mostly applied among the other categories [320]. Multi-layer perceptron (MLP), convolutional neural network (CNN), support vector machines (SVM), decision trees (DT) and k-nearest neighbour (k-NN) are all examples of supervised learning algorithms. Unsupervised learning involves the process of developing a model or function without predefining the outputs. This method is typically used for finding meaningful patterns (e.g. WBM defect patterns) or classifications within a large data set [321]. Clustering, adaptive resonance theory network (ART), Hopfield neural network (HNN), Cellular Neural Network, and self-organizing map (SOM) are all examples of

unsupervised learning algorithms. Reinforcement learning involves learning through a predefined reward signal that enables the machine to be able to quantify its performance. Unlike supervised and unsupervised learning, reinforcement learning is used in robotics, gaming and navigation areas and is not common in industrial applications [322]. Machine learning is considered very powerful tool in classifying the defects detected by the image data. However, there are two major problems that must be considered before applying machine learning algorithms to avoid classification errors. These two problems are overfitting and data imbalance. Unfortunately, these problems are very common in defect inspection using machine learning techniques. Overfitting occurs when the classification performance is highly dependent on the training data. Overfitting problems appear when few samples are used for the training process, and it could be a challenge for industries that do not have enough image samples for the defects so they can train their machine learning algorithm. The lack of training data problem can be solved by generating more datasets using several techniques such as bootstrapping and Generative Adversarial Network GAN which are explained in the following sections. On the other hand, data imbalance occurs when the training data for certain classes are more than others. This is very common in visual inspection problems when certain defects occur more than others in the data provided. Data imbalance problem will make the classifier more biased by the data of the dominant class, which in return affect the quality of the classification. One way to solve the data imbalance problem is to create or modify the algorithm, which includes a cost-sensitive method. The other way is using data pre-processing techniques such as sampling, in which either new samples are added, or existing samples are removed from the original data. The process of removing samples is known as under-sampling and the process of adding new samples is known as over-sampling [84]. Hence, GAN and Bootstrapping can be used to solve the data imbalance problem as well.

As mentioned in Section 3.2, DL, which is considered a subset of machine learning (Figure 3.1), involves the usage of DNNs. Generally speaking, any MLP network that has three hidden layers or more is considered a DNN. Unlike, other machine learning tools, DNNs are capable of extracting the features as well as performing classification tasks. The independence from prior knowledge and human effort in feature design is a major advantage for DL in general and DNNs. CNNs, autoencoders and capsule networks are all examples of DNNs. For the purpose of this thesis, we will limit our discussion with DL algorithms that are used for computer vision such as

the ones mentioned previously and exclude the algorithms that are used for other applications such as LSTM and RNNs.

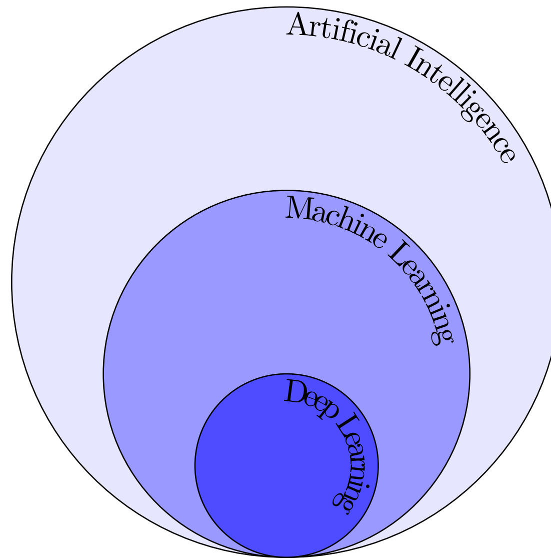


Figure 3.1: Deep learning is a subset of machine learning and machine learning is a subset of artificial intelligence.

3.3 Perceptron

Perceptron is considered the building block for any ANN (including DNNs). It was developed in the 1950s by the scientist Frank Rosenblatt [363]. The perceptron takes one or more inputs that go through specific activation function in a neuron to produce an output, for example the perceptron in Figure 3.2 has n inputs and n corresponding weights that goes inside a neuron and produces one output. A bias b can be also included to increase the number of controllable parameters. The resulting formula from the previous operation is represented in equation 3.1.

$$z = \sum_{i=0}^{n} w_i \cdot x_i + b \tag{3.1}$$

Rosenblatt proposed simple yet effective rules to compute the output by introducing weights; weights are real numbers the describe the importance of each input to the system, in which the input with less importance will be penalised by multiplying it with low or in some cases zero weight, on the other hand the inputs that have high influence on the system will be rewarded by multiplying them with relatively high weights. Activation function is used to add non-linearity to

the process and help in classification, applying the activation function takes place after multiplying the weights with the inputs and summing the results. The resulting output y after applying the activation function is given by equation 3.2. Figure 3.3 demonstrates some of the commonly used activation functions. Hence, the process of applying equations 3.1 and 3.2 is called forward propagation.

$$\hat{\mathbf{y}} = a(z) = a\left(\sum_{i=0}^{n} w_i \cdot x_i + b\right) \tag{3.2}$$

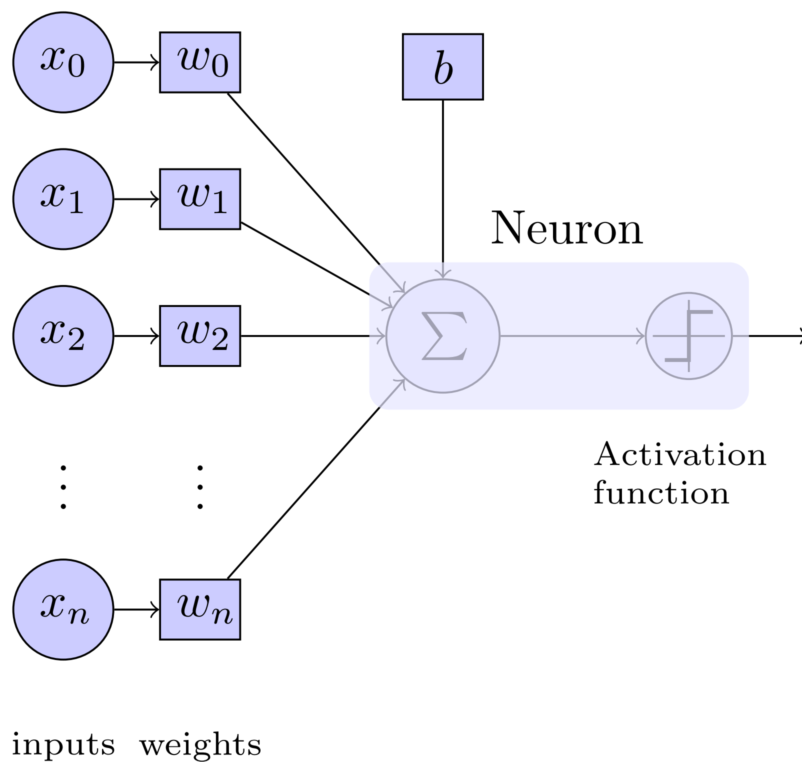
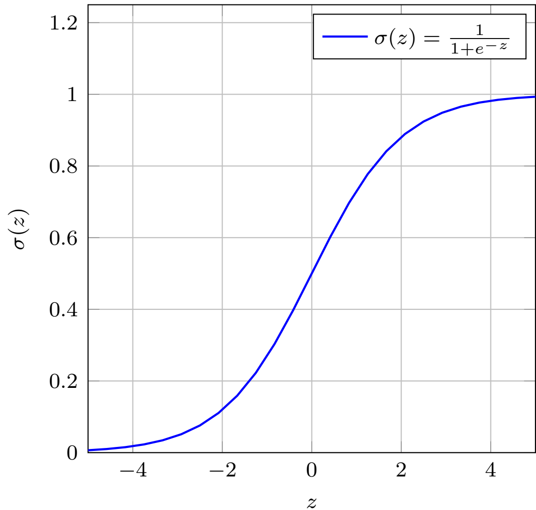
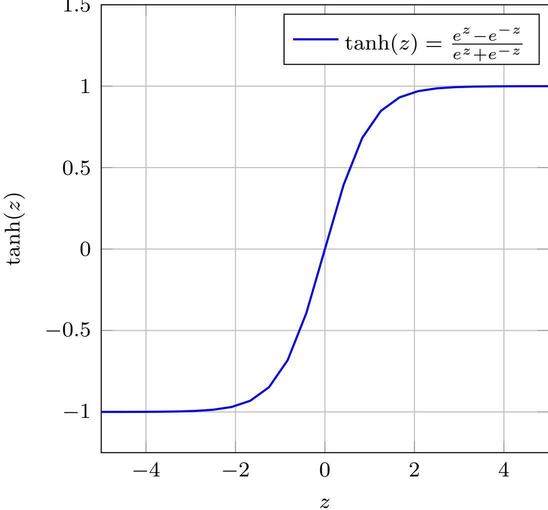


Figure 3.2: Perceptron

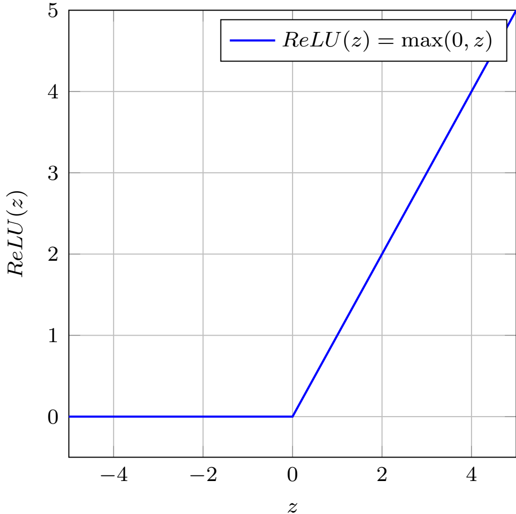
The research conducted at that time did not propose an efficient way to update weights and bias for the algorithm to learn by examples. In 1986, Rumelhart et al. [364] proposed the back-propagation algorithm that can estimate the error in perceptron with the aid of a cost function. The proposed algorithm can be used to update the tunable parameters automatically by applying the gradient by descent principle which will be discussed in the following section.



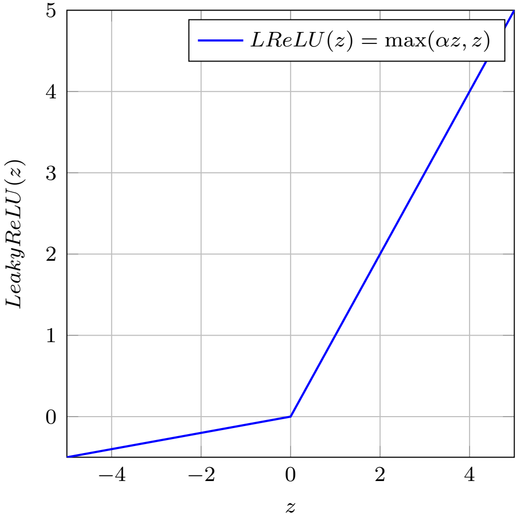
(a) Sigmoid activation function.



(b) Hyperbolic tangent activation function.



(c) ReLU activation function.



(d) LeakyReLU activation function.

Figure 3.3: Commonly used activation functions.

3.3.1 Learning in Perceptron

As stated previously in Section 3.2 There are three types of learning; supervised, unsupervised and reinforcement. In supervised learning, the perceptron will learn by example which means that the perceptron will be presented with a set of inputs $\mathbf{x} = [x_0, x_1, ..., x_n]$ and targets (also known as *labels*) $\mathbf{y} = [y_0, y_1, ..., y_n]$. A cost function J is introduced to compare the estimated output of the perceptron $\hat{\mathbf{y}}$ and real labels $\mathbf{y}$. Ideally, if the inputs $\mathbf{x}$ from the historical data were fed to the perceptron, the corresponding labels $\mathbf{y}$ should match $\hat{\mathbf{y}}$. However, in real cases there is a margin of error that exists. Hence, learning occurs only when the perceptron makes an error.

Many forms of cost functions to measure the error are available. The most commonly used is the mean-squared error (MSE), which is the squared sum of all targets minus outputs, divided by the number of outputs m as shown in equation 3.3.

$$J = \frac{1}{m} \sum_{k=1}^{m} (\hat{y}_k - y_k)^2 \tag{3.3}$$

The objective of this process is to get the optimal set of tunable parameters (weights and biases) that will minimise the cost J. Back-propagation and gradient descent are used to apply these optimisation steps in systematic manner such that the weights and biases are updated using the derivative and chain-rule concepts as described in equations 3.4 and 3.5. The parameter μ is called the learning rate which is selected before the training takes place. If the value of μ is too high, the algorithm will learn faster; however, it may diverge in reaching the minimal value resulting in oscillating behaviour. On the other hand, if the value of μ is too small, the learning process will be slow but more stable. Parameter t refers to the number of iterations (also known as *epochs*) needed to update the weights and biases until the perceptron reaches the desired output. Hence, in the beginning of the training the weights and biases are usually given random values. The learning by gradient descent process is illustrated in Figure 3.4 and the backpropagation with gradient descent is demonstrated in Algorithm 1.

$$\begin{aligned} \Delta w_i &= -\mu \frac{\partial J}{\partial w_i} \\ w_i(t+1) &= w_i(t) + \Delta w_i \end{aligned} \tag{3.4}$$

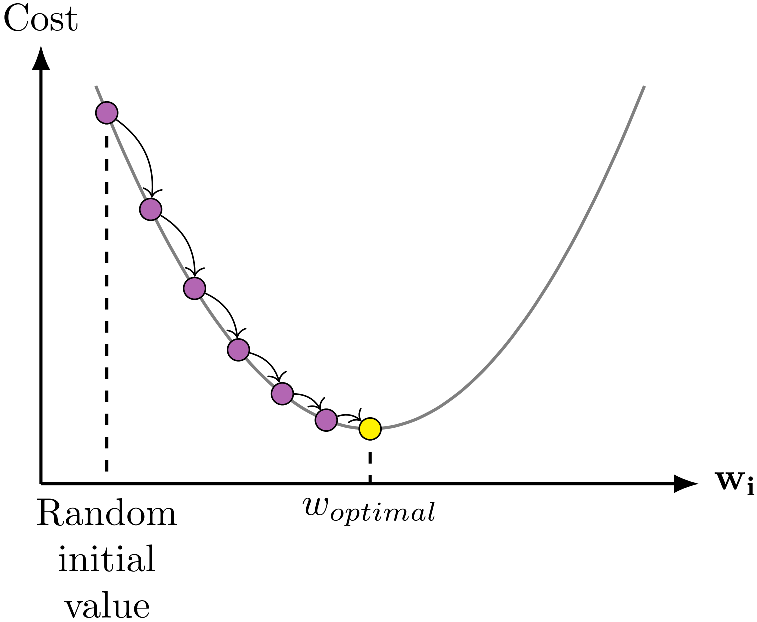
$$\Delta b = -\mu \frac{\partial J}{\partial b}$$
$$b(t+1) = b(t) + \Delta b \tag{3.5}$$

Algorithm 1 Back-propagation with gradient descent algorithm.

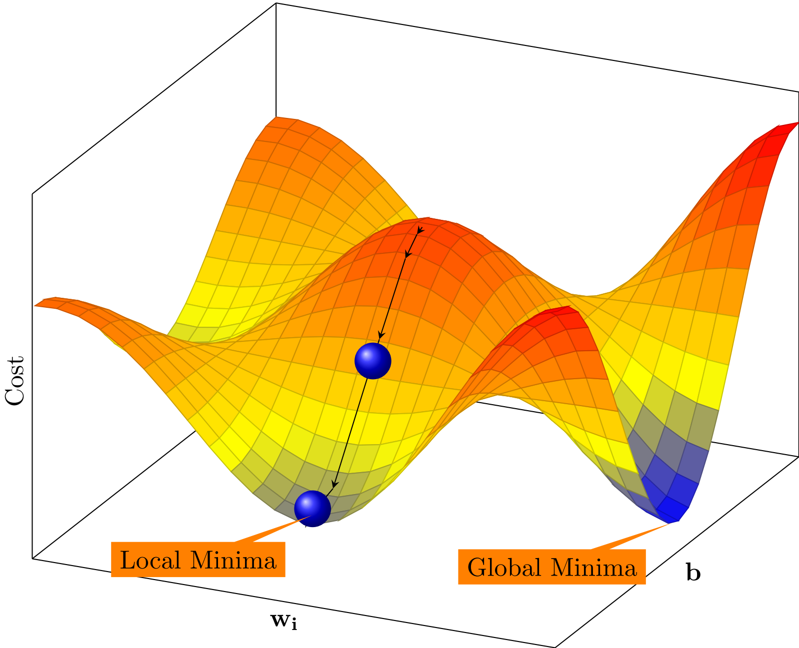
1: **procedure** TRAIN PERCEPTRON
2: **Input**: learning rate μ, number of epochs t, and stopping condition
3: $\mathbf{X} \leftarrow$ Training data set of size m.
4: $\mathbf{y} \leftarrow$ Corresponding labels set of size m.
5: $\mathbf{w}, \mathbf{b} \leftarrow$ Randomly initialise weights and biases.
6: **while** ($N \leq t$ **OR** (stopping condition **NOT** *true*)) **do**
7: $z \leftarrow \sum w_i \cdot x_i + b$
8: $\hat{\mathbf{y}} \leftarrow activation(z)$
9: $J \leftarrow$ Calculate the cost using cost function.
10: $\Delta w_i \leftarrow -\mu \frac{\partial J}{\partial w_i}$
11: $\Delta b \leftarrow -\mu \frac{\partial J}{\partial b}$
12: $w_i \leftarrow w_i + \Delta w_i$
13: $b \leftarrow b + \Delta b$
14: $N \leftarrow N + 1$
15: **end while**
16: **end procedure**

Despite the effectiveness of back-propagation and gradient descent algorithm in learning, there are some shortcomings. Most of these shortcomings can be resolved with proper parameters tuning. First, gradient by descent approach can make the cost function to approach local minima point (as shown in Figure 3.4(b)) which will make the training to stop before reaching the best result[1]. Choosing the right learning rate μ can help in mitigating this problem, such that a relatively low μ will make the training much stable. Some problems may also occur when calculating the derivative of certain activation functions (e.g. Sigmoid), specially when the activation function is approaching a constant value (e.g. 1). Therefore, understanding the problem before attempting to solve it is essential in choosing the optimal parameters for perceptron learning. More problems will also arise when the number of neurons and layers are increased as will see in Section 3.4.

[1]By best result we mean reaching global minima which the best result that perceptron can achieve



(a) Gradient by descent illustration using one parameter.



(b) Gradient by descent illustration using multiple parameters.

Figure 3.4: Gradient descent visualisation.

3.3.2 MNIST Hand Written Digits Dataset

For evaluating the networks' performance discussed in this chapter, we will use the Modified National Institute of Standards and Technology (MNIST) dataset, which contains 70,000 scanned images of handwritten digits, together with their correct labels. These images are scanned handwriting samples from 250 people, half of whom were US Census Bureau employees, and the other half are high school students. The images are in graysacle format and of size 28×28 pixels each. The labels are in integer format such that each image has the correct corresponding label that describes it as a number from 0 to 9. Figure 3.5 shows sample images from MNIST dataset.

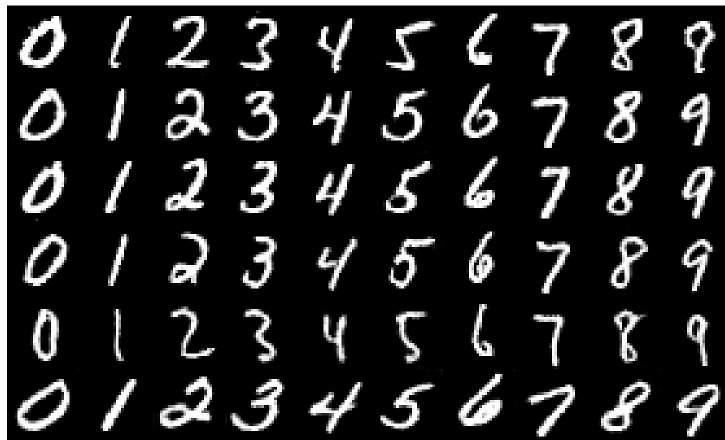


Figure 3.5: MNIST data sample.

The MNIST data are divided into two parts. The first part is the training data that contains 60,000 images. The second part is the validation data that contains 10,000 images. As the name states the training data are used to train the network to classify the images correctly while the validation data are used to evaluate how good was the training of the network [365].

3.3.3 Evaluation Metrics

Evaluation metrics are used to measure the performance of machine learning models. It is very important to evaluate the model after training takes place to decide whether the model can be implemented in the project or not. There are a variety of techniques used for evaluation such as classification accuracy, loss and confusion matrix. Confusion matrix is very useful tool for evaluation. It visualises the predictions for a classification problem against the real labels in table format. Each row of the matrix represents the instances of real labels while each column

represents the instances of predicted labels. The values of the confusion matrix are denoted by four categories, namely, true positive (TP), true negative (TN), false negative (FN) and false positive (FP). The outcomes of a confusion matrix can be visualised in a 2×2 matrix as shown in Figure 3.6.

		Predicted		
		Positive	Negative	Total
Actual	Positive	TP	FN	$TP + FN$
	Negative	FP	TN	$FP + TN$
	Total	$TP + FP$	$FN + TN$	$N = TP + TN + FP + FN$

Figure 3.6: Confusion Matrix.

Many metrics can be obtained from the confusion matrix, such as accuracy, precision, recall and F1-score. The accuracy of the algorithm predicts the number of classes that are classified correctly. The precision measures the exact efficiency of the algorithm for predicting the positive samples. The recall is the measure to calculate the true positive, and the mean harmonic of the recall and precision is determined with F1 score. The formulas of theses metrics are given by equations 3.6-3.9.

$$Accuracy = \frac{TN + TP}{TN + TP + FN + FP} \tag{3.6}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.7}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.8}$$

$$F1 - score = \frac{2(Precision \times Recall)}{Precision + Recall} \tag{3.9}$$

3.3.4 Evaluating Perceptron using MNIST Dataset

We will use the MNIST data described in the previous section to evaluate the perceptron's classification performance. Before the images are fed to the perceptron, they should go through simple pre-processing steps. First, the pixel values are scaled (normalised) to have values between

0 and 1. In order to do that, all the pixel values are divided by 255[2]. Secondly, the image should be transformed to vector format since perceptron can only receive the input data in vector from as shown in Figure 3.7.

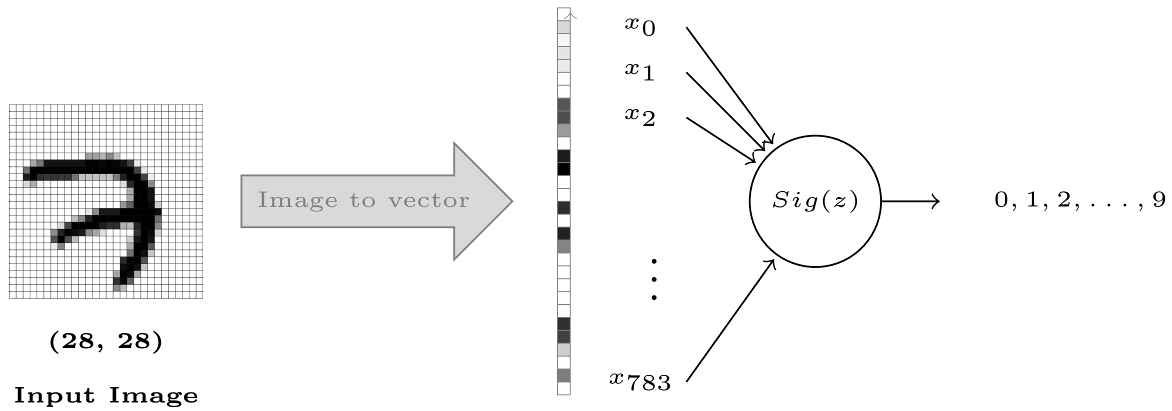


Figure 3.7: Perceptron architecture for classifying MNIST dataset.

Figure 3.7 demonstrates a special case of using perceptron for classification that is called *Logistic Regression*. In this case Sigmoid function is used as an activation function. The forward propagation formulas for this model are given by equations 3.10 and 3.11.

$$z = \sum_{i=0}^{783} w_i \cdot x_i + b \tag{3.10}$$

$$a(z) = Sig(z) \tag{3.11}$$

The training of the perceptron model is done over 100 epochs without applying a stopping condition. During these epochs training and validation accuracies were calculated on each epoch as shown in Figure 3.8. According to Figure 3.8, the training accuracy achieved at epoch 100 is 94% with a validation accuracy of 92.75%.

The confusion matrix for each class of the the validation data has been also calculated along with precision, recall and F1-score as shown in Figure 3.9 and Table 3.1. As we can see from the results the logistic regression classifier performed the best in classifying digit 1 with classification accuracy of 98% and performed the worst in classifying digit 5 with classification accuracy of 87%. This means that the features of digit 1 was easy and distinguishable for the algorithm. On

[2]MNIST grayscale images have 256 different gray values, numbered 0 - 255 (8 bits).

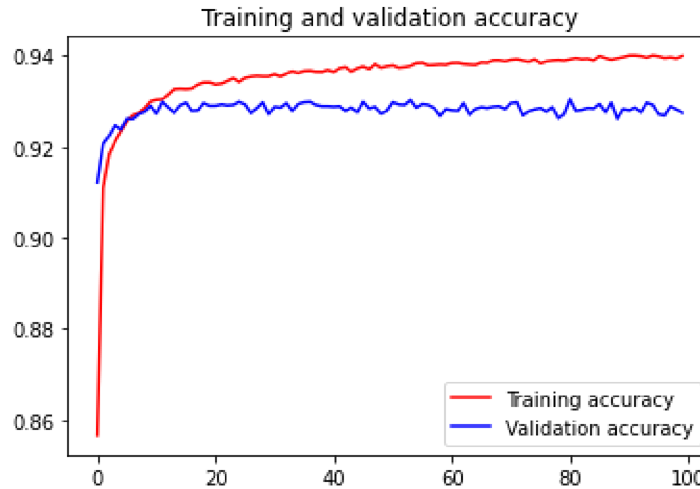


Figure 3.8: Perceptron's training and validation accuracy for MNIST dataset.

the other hand, the features of digit 5, are being mistaken with the features of other digits and therefore there is relatively high misclassification error.

Table 3.1: Metrics for evaluating the validation dataset in percpetron.

Class	Recall	Precision	F1-score
Zero	0.974	0.959	0.967
One	0.98	0.96	0.97
Two	0.904	0.928	0.916
Three	0.92	0.905	0.912
Four	0.94	0.936	0.938
Five	0.868	0.899	0.883
Six	0.949	0.947	0.948
Seven	0.927	0.937	0.932
Eight	0.881	0.885	0.883
Nine	0.927	0.912	0.917

3.4 Multi-layer Perceptron

As the name suggests, Multi-layer Perceptron (MLP) algorithm is the an algorithm with multiple perceptrons and an increased numbers of layers as shown in Figure 3.10. MLP is the most popular architecture in ANNs and feed-forward networks in general. The architecture of MLP consists of an input layer (l_0), a hidden layer or multiple hidden layers (l_n), and an output layer (l_{n+1}).

Actual Class \ Predicted Class	Zero	One	Two	Three	Four	Five	Six	Seven	Eight	Nine
Zero	955 97%	0 0%	1 0%	3 0%	1 0%	9 1%	7 1%	3 0%	1 0%	0 0%
One	0 0%	1112 98%	5 0%	2 0%	0 0%	2 0%	3 0%	1 0%	10 1%	0 0%
Two	5 0%	9 1%	933 90%	14 1%	11 1%	5 0%	9 1%	9 1%	34 3%	3 0%
Three	3 0%	0 0%	17 2%	929 92%	1 0%	20 2%	3 0%	10 1%	18 2%	9 1%
Four	1 0%	4 0%	7 1%	1 0%	923 94%	0 0%	8 1%	6 1%	7 1%	25 3%
Five	9 1%	4 0%	3 0%	36 4%	9 1%	774 87%	14 2%	7 1%	30 3%	6 1%
Six	8 1%	3 0%	9 1%	3 0%	7 1%	16 2%	909 95%	1 0%	2 0%	0 0%
Seven	0 0%	8 1%	24 2%	5 0%	4 0%	1 0%	0 0%	953 93%	2 0%	31 3%
Eight	8 1%	10 1%	5 1%	23 2%	9 1%	29 3%	7 1%	9 1%	858 88%	16 2%
Nine	7 1%	8 1%	1 0%	11 1%	21 2%	5 0%	0 0%	18 2%	7 1%	931 92%

Figure 3.9: Confusion matrix for MNIST dataset classification using perceptron.

The outputs of each layer are fed to the next layer as inputs and so on until final output layer is reached. The input layer denoted by l_0 is not considered as a layer because it does not contain any neurons and therefore it does not perform any form of computations. MLPs are usually described according to the number of neurons and layers used. For example, a 3-5-4 MLP means that this is a two-layer network that consists of: input layer with three inputs, hidden layer with five neurons and output layer with four outputs. The same computations that are governing the perceptron are also applied in MLP, bearing in mind that feed forward and back-propagation equations are calculated after each layer and the weights will be in matrix form instead of the vector form as in the perceptron. When three or more hidden layers are used in MLP the network is then considered as deep neural network (DNN).

Adding large number of hidden layers may improve the network's capability in extracting the important information for classification; however, it will increase the computational time and it could lead to overfitting. Therefore, there should be some starting point in selecting the right amount of hidden layers for a certain problem. Heaton in [366] has established three rules in

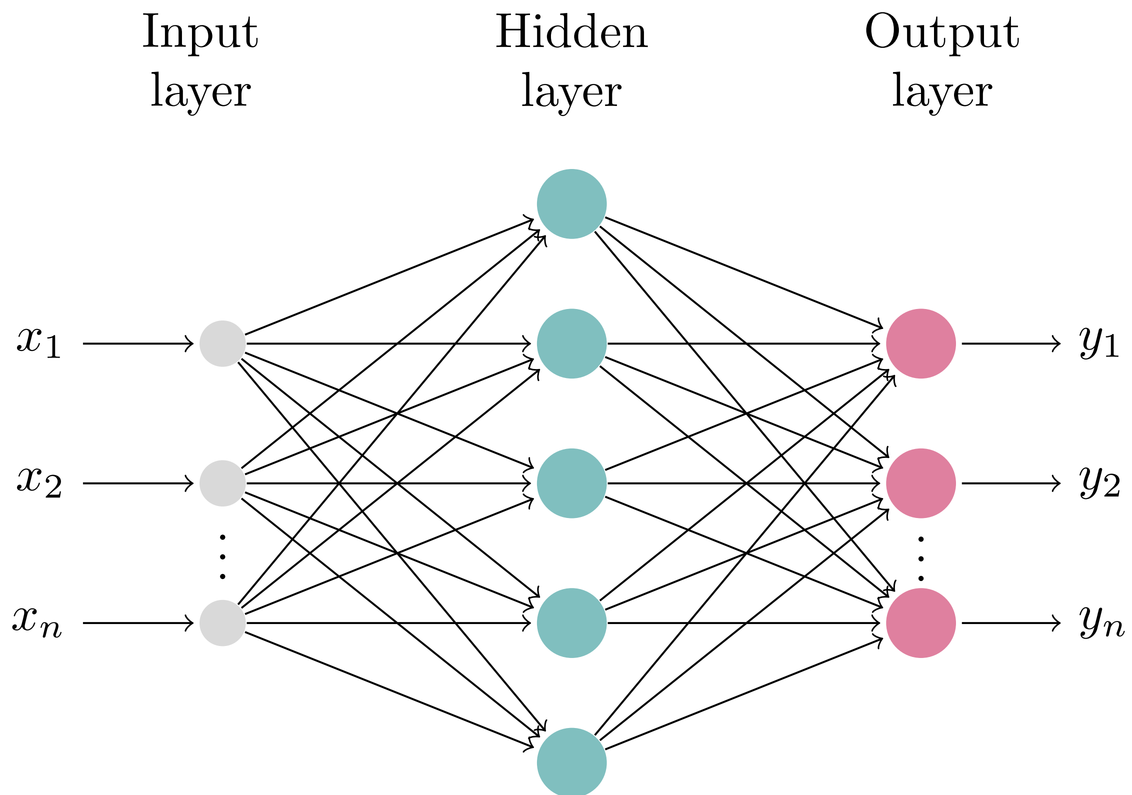


Figure 3.10: MLP.

selecting an appropriate number of hidden layers as follows:

- The number of hidden neurons should be between the size of the input layer and the size of the output layer.
- The number of hidden neurons should be 2/3 the size of the input layer, plus the size of the output layer.
- The number of hidden neurons should be less than twice the size of the input layer.

However, in complex problems that require DNN usage, it is hard to predict the behaviour of the network due to the large amount of trainable parameters. Therefore, trial-and-error is still used for this purpose. Nevertheless, the previous rules can be still valid as a starting point in some cases.

3.4.1 Batch Training

A popular way to accelerate the training of ANNs is to use batch training. In batch training, the training dataset is divided to several subsets which are called minibatches. The batch size should be defined before the training takes place by the user. For instance, assuming that a dataset of 10,000 elements was chosen for training, if the user chose a batch size of 10, it means that the training at each epoch will be over 1000 elements. During training by batches the loss function will follow equation 3.12, where ζ stands for the batch size.

$$J = \frac{1}{\zeta} \sum_{k=1}^{\zeta} Loss \tag{3.12}$$

The gradient computations for each minibatch B can be parallelised, allowing these methods to perform faster in a distributed framework [367]. Several advantages can be obtained by training on mini-batches. First, the gradient of the loss over a mini-batch is an estimate of the gradient over the training set, whose quality improves as the batch size increases. Secondly, computation over a mini-batch can be more efficient than computations for individual examples on modern computing platforms [368].

Another interesting concept in ANNs in general is batch normalisation. It is a technique that helps in coordinating the update of multiple layers in the model resulting in settling the learning process and decreasing the number of epochs required in training [369]. This is particularly useful when the number of layers is large i.e. DNNs as the layers changes over each layer which makes learning process chasing a moving target. For instance, weights and biases of certain layer are updated based on the expectation that the previous layer output values with a given distribution. This distribution is likely changed after the weights of the prior layer are updated [370]. Batch normalization can be applied to any input or hidden layer in a network according to equation 3.13.

$$B' = \frac{B - \mu}{\sigma} \tag{3.13}$$

where μ and σ are vectors containing the mean and standard deviation for each neuron in the layer respectively.

3.4.2 Evaluating MLP using MNIST Dataset

MNIST data will be used again in this section to evaluate the performance of MLP. In MLP the image data should be also vectorised in the same way mentioned in Section 3.3.1. Therefore, the input layer will consists of 784 elements which corresponds to the number of pixels in an MNIST image. In this evaluation, we will use a $784 - 128 - 128 - 128 - 128 - 10$ MLP that also follows the rules of Heaton in [366]. Hence, this MLP is also considered a DNN since the number of hidden layers is more than three. The network architecture is shown in Figure 3.11.

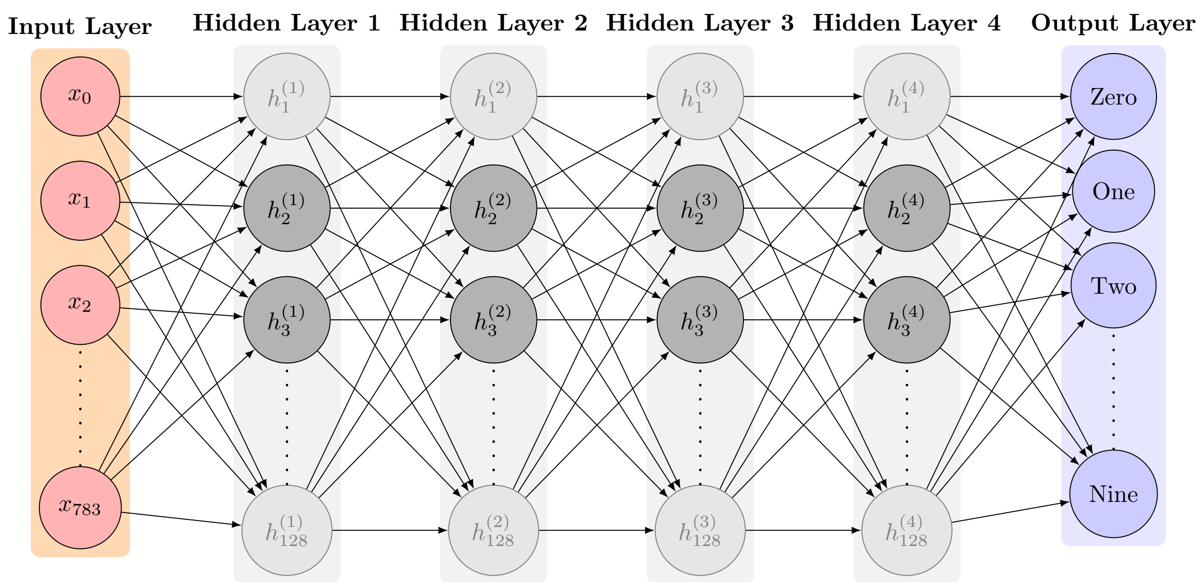


Figure 3.11: MLP architecture for classifying MNIST dataset.

Softmax activation function was used for the neurons of the output layer. In hidden layers, ReLU activation function is used for all the neurons. The training is also done over 100 epochs without a stopping condition. Where the training and validation accuracies are calculated at each epoch as shown in Figure 3.12.

The confusion matrix and other evaluation metrics of the validation data are also obtained in Figure 3.13 and Table 3.2. Compared with the performance of perceptron in classifying MNIST digits we notice that MLP achieved an overall accuracy of 98% which 5.25% more than the accuracy achieved in perceptron. Also, it is observed in Figure 3.13 that some digits zero, one, three and six were classified with an accuracy of 99%.

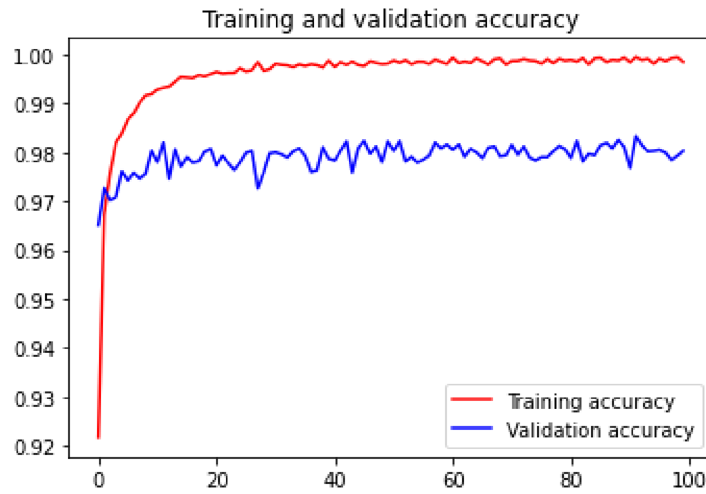


Figure 3.12: MLP's training and validation accuracy for MNIST dataset.

Actual Class \ Predicted Class	Zero	One	Two	Three	Four	Five	Six	Seven	Eight	Nine
Zero	972 99%	0 0%	1 0%	1 0%	0 0%	1 0%	4 0%	0 0%	1 0%	0 0%
One	0 0%	1127 99%	1 0%	1 0%	1 0%	0 0%	2 0%	0 0%	3 0%	0 0%
Two	0 0%	3 0%	1001 97%	3 0%	8 1%	0 0%	3 0%	7 1%	6 1%	1 0%
Three	0 0%	0 0%	2 0%	966 99%	0 0%	2 0%	0 0%	2 0%	7 1%	1 0%
Four	0 0%	0 0%	3 0%	0 0%	961 98%	0 0%	7 1%	1 0%	1 0%	9 1%
Five	1 0%	0 0%	0 0%	9 1%	0 0%	867 97%	5 1%	1 0%	6 1%	3 0%
Six	0 0%	3 0%	0 0%	0 0%	1 0%	3 0%	947 99%	0 0%	4 0%	0 0%
Seven	3 0%	2 0%	4 0%	6 1%	4 0%	1 0%	0 0%	993 97%	3 0%	12 1%
Eight	2 0%	1 0%	1 0%	1 0%	4 0%	2 0%	2 0%	2 0%	956 98%	3 0%
Nine	0 0%	2 0%	0 0%	5 0%	11 1%	3 0%	1 0%	1 0%	3 0%	983 97%

Figure 3.13: Confusion matrix for MNIST dataset classification using MLP.

Table 3.2: Metrics for evaluating the validation dataset in MLP.

Class	Recall	Precision	F1-score
Zero	0.992	0.994	0.993
One	0.993	0.99	0.992
Two	0.97	0.988	0.979
Three	0.986	0.975	0.98
Four	0.979	0.971	0.975
Five	0.972	0.986	0.979
Six	0.989	0.975	0.982
Seven	0.966	0.986	0.976
Eight	0.982	0.966	0.974
Nine	0.974	0.971	0.973

3.4.3 Overfitting

Overfitting is a major problem in DL networks, it occurs when the model is being too reliant on the training data which prevents its ability to generalise. The greater the hidden layers in DNNs, the greater the ability of the DNN to identify existing patterns. However, if the number of hidden neurons is too large, the DNN might simply memorise all training examples. This might prevent it from generalising, or producing correct outputs when presented with data that was not part of the training dataset [371]. For instance, it can be observed from Figure 3.12 that there is slight overfitting in the training process, such that the training accuracy is almost 100% while the validation accuracy is 98%. That means the model is over trained to recognise the training samples more than recognising the validation ones. This problems becomes more obvious when images are getting much complicated and the features are hard to learn. Another reason that can produce overfitting is the lack of enough data for training, which may make the model so dependent on the few data provided and cannot classify data with same class that may have slightly different or complex features.

Many remedies can be used to avoid the overftting problem such as hyperparameter tuning, regularisation, dropout and data augmentation. Hyperparameter tuning involves choosing the optimum hyperparameter such as number of hidden layers, hidden neurons, batch size and learning rate that will lead to the best performance of the model. The tuning can be either done manually (by trial-and-error) or automatically by iteration process. Regularisation is a

from of regression that adds some constraints or penalty term to the cost function which can reduce the variance of the model, without a substantial increase in the bias. This technique can help the model to generalise more and therefore the overfitting is reduced. L1 and L2 are two regularisation techniques used in literature [32, 372]. Dropout can be also considered one of the regularisation techniques that are widely used. Dropout can be used at each training iteration by deactivating random set of neurons, according to certain percentage specified by user, and then training is conducted on the resultant subnet. Dropout has the effect of making the training process noisier by forcing nodes within a layer to probabilistically take on more or less responsibility for the inputs [370, 373]. Hence, batch normalisation can also offer regularisation effect and therefore can be used instead of dropout in some cases to avoid overfitting [369, 370].

Data augmentation is also very efficient in reducing overfitting in the network. It is defined as the process of increasing the dataset for training either by modifying the existing ones or by generating synthetic data that looks similar to the existing. The data can be modified according to their type. For image data, scaling, rotation, and shifting operations can be applied to increase the samples. Creating synthetic data can be done by image processing techniques, such as isolating the region of interest and insert it in multiple locations in different images, or by applying GAN network to generate synthetic images that look very similar to the images of interest as explained in Section 3.7.

3.4.4 Data Imbalance

The data imbalance (also known as class imbalance) problem in DL describes classification tasks where the classes of data are not equally presented and involves heavy skew in class distribution [374]. Deciding whether a dataset has a class imbalance problem depends on the size and nature of the dataset. Generally speaking, a dataset is considered severely class imbalanced when the majority to minority class scale is 100:1 or more [375]. In some applications such as medical imaging analysis, fraud detection in banking or industrial defect detection, data imbalance can lead to serious problems in the learning process by making the model biased towards the majority class. A classifier can still achieve high accuracy depending on the majority class detection only. For instance, suppose a trivial binary classifier that classifies all images samples of an industrial product as “non-defect”. This classifier can achieve 99.9% accuracy assuming 0.1% of the image

samples belong to "defect" class; however, in this case, all defect cases remain undetected [374]. Many techniques are used to mitigate the class imbalance effect on the training, these techniques can be divided into three categories: cost-sensitive, algorithm level methods and data level methods. Cost-sensitive methods aims to provide different misclassification cost for each class in the algorithm. This is not possible in all datasets as in most cases the misclassification cost is unknown. Algorithm level methods involves training multiple algorithms for the relevant classes, this also includes performing ensemble learning or decision fusion. Data level methods consider increasing the dataset of each class aiming to change the class distribution. But since this solution is not always feasible in most of the cases, data augmentation approaches can be used to increase or upsample the data in each class as explained in the previous section. Fortunately, this problem is not serious in the MNIST data that is currently used for evaluating the different DL architectures in this chapter as shown in Figure 3.14. However, in real case-scenarios this problem is very common as will be investigated in Chapters 4 and 5.

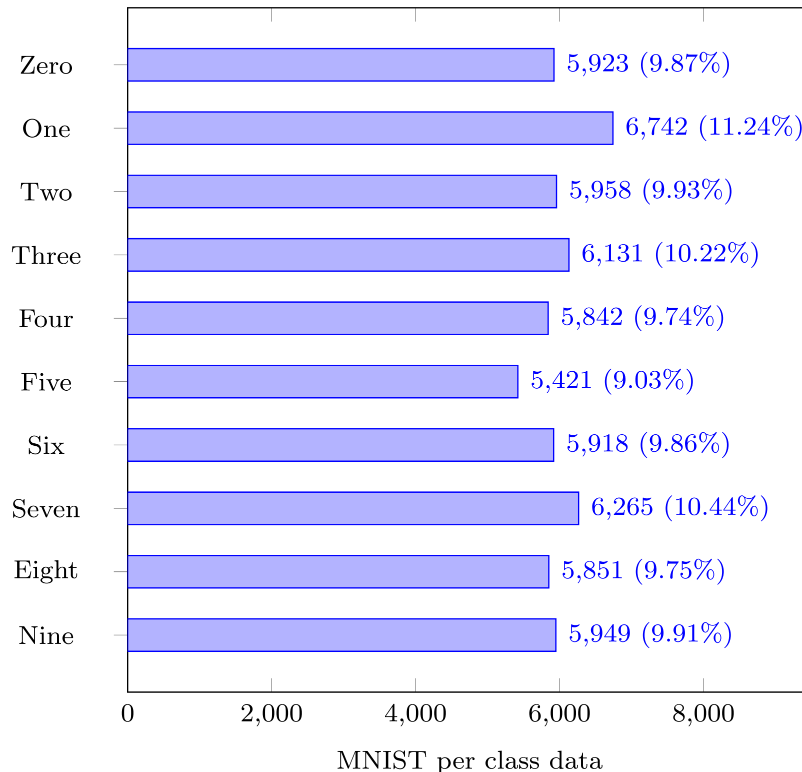


Figure 3.14: MNIST training data count and distribution per class.

3.5 Convolutional Neural Networks

A special case of DNNs is when the matrix multiplications include convolutional filter operations, which is common in DNNs that are designed for image and video analysis. Such models are known as convolutional neural networks (CNNs) [376]. CNNs have been widely known for their high image recognition capabilities in recent years [377, 378]. Usually, the methods based on CNN are preferable in imaging applications and can obtain more precise results compared to traditional methods due to many factors such as; capturing grid-like topology of images effectively, decrease the computation time due to pooling and convolutional layers usage, capability of differentiating large number of classes, and learning image features automatically without using image processing tools for feature extraction [379]. Remarkable achievements in feature extraction and image classification have been produced by CNNs such as AlexNet [380], VGG [381], ResNet [382], and DenseNet [383], which have outperformed conventional classification models such as MLP. A typical CNN consists of input layer, convolutional layers, pooling layers, fully connected (FC) layers, and output layer as shown in Figure 3.15.

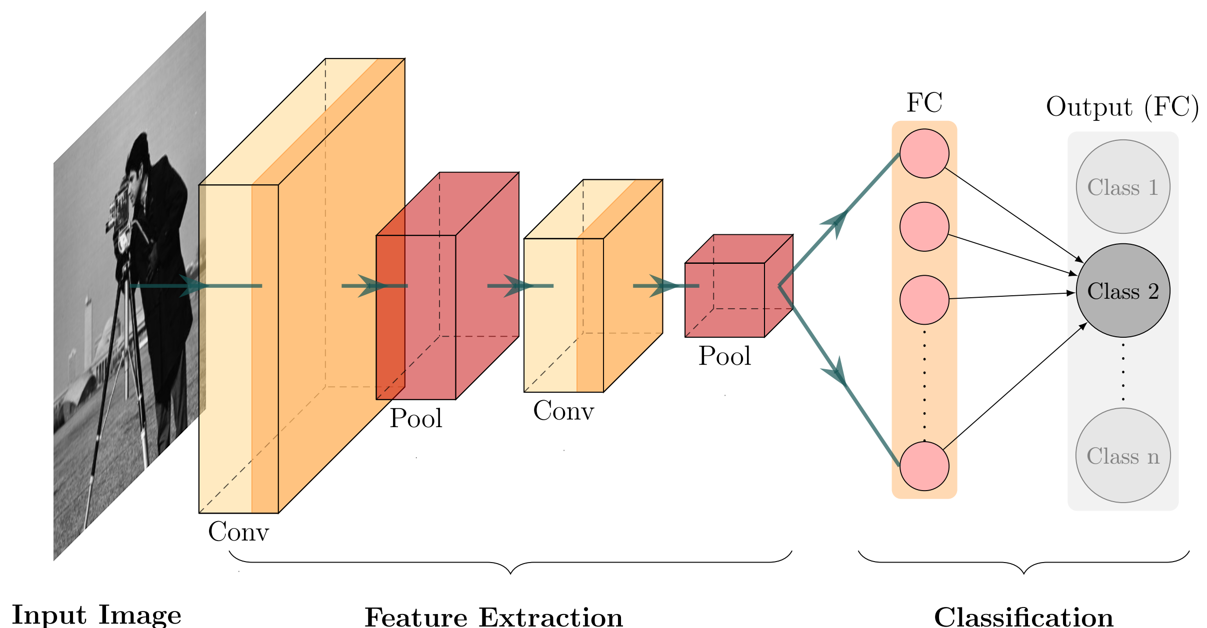


Figure 3.15: CNN architecture.

Convolutional and pooling layers (which performs convolutional and pooling operations respectively) are the main elements that distinguish CNN from other types of ANNs. In CNN, pooling layers lay between two successive convolutional layer and output layer follows the last

fully connected layer which connects all hidden units in previous layers. The convolution layers play an essential role in extracting features from the input images by performing two operations throughout an input array. First, it performs element by element-by-element multiplication (also known as dot product) between a sub-image array (that is equal in size of a kernel) of an input image and kernel with random weights. Secondly, the output of this operation will be added to bias and result in what so called feature map. The feature map dimensions are given by equation 3.14.

$$n_{out} = \left(\frac{n_{in} + 2p - k}{s} \right) + 1 \tag{3.14}$$

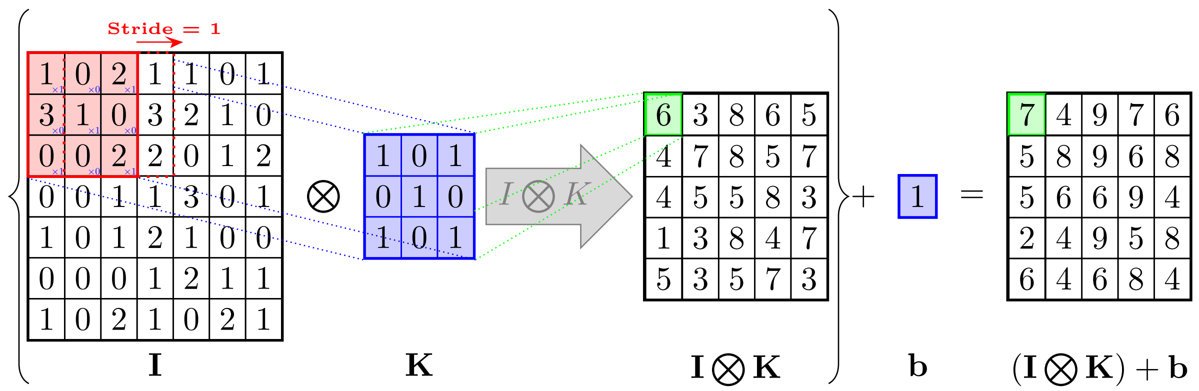
Where n_{out} is the resulting size of feature map, n_{in} is the input feature map to the layer, p is pooling size, k is the kernel size and s is the stride.

The initial weights of the kernels are randomly generated, while the bias can be set based on the networks' configuration. A suitable stride can be defined that identify how many of pixels the kernel can slide across the input image. A careful consideration must be made in selecting the stride size, as a larger slide can reduce computation time, but it may also lose important features from the input image. The convolution operation is demonstrated in Figure 3.16(a). On the other hand, pooling layers are important in improving image classification. It minimizes computation time by reducing the spatial size of the input image array [384]. This process is often called down sampling. There are two widely used pooling functions used for this purpose which are max-pooling and average-pooling (also known as mean-pooling). The pooling operation is demonstrated in Figure 3.16(b). FC layers are responsible in encoding the features from previous layers in order to come up with relevant class and hence perform classification. FC layers can be considered as an MLP network inside the CNN.

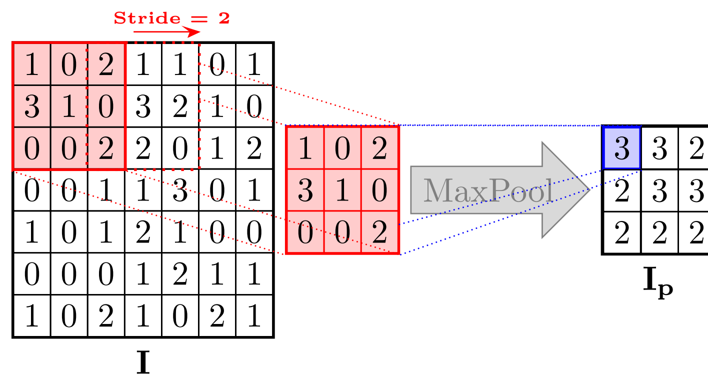
CNNs are usually described in terms of the number of their convolutional and FC layers, which contain the trainable parameters.

3.5.1 Evaluating CNN using MNIST Dataset

Unlike the preprocessing procedure in MLP, CNN does not require the input image to be vectorised. This is because the convolutional operations can handle an image input in matrix format, whether it is grayscale or RGB image. In our case, MNIST data are in grayscale with a size of 28×28. The CNN network architecture used to evaluate MNIST data classification is shown in



(a) Convolutional operation



(b) Pooling operation

Figure 3.16: Convolution and pooling operations.

Figure 3.17. Table 3.3 also shows the details of the network parameters and activation functions used. In this CNN, three convolutional, three pooling and two FC layers were used. Furthermore, dropout after the first FC layer was applied with 50% to reduce the overfitting.

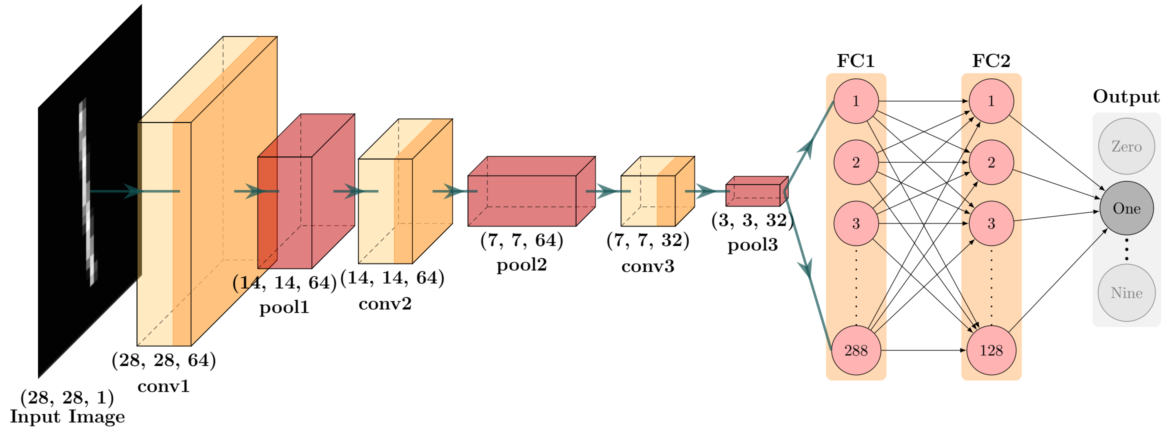


Figure 3.17: CNN architecture for classifying MNIST dataset.

Table 3.3: Suggested CNN network for MNIST classification.

Layer	Type	Input size	Kernel Size/ Stride	Activation	Dropout	Output size
1	Input Image	-	-	-	-	[28,28,1]
2	conv1	[28,28,1]	5/1	ReLU	No	[28,28,64]
3	pool1	[28,28,64]	2/1	-	-	[14,14,64]
4	conv2	[14,14,64]	5/1	ReLU	No	[14,14,64]
5	pool2	[14,14,64]	2/1	-	-	[7,7,64]
6	conv3	[7,7,64]	3/1	ReLU	No	[7,7,32]
7	pool3	[7,7,32]	2/1	-	-	[3,3,32]
8	FC1	[3,3,32]	-	-	-	[288,]
9	FC2	[288,]	-	ReLU	Yes (0.5)	[128,]
10	Output	[128,]	-	Softmax	-	[10,]

The training is also done over 100 epochs without a stopping condition. Where the training and validation accuracies are calculated at each epoch as shown in Figure 3.18 . Compared with the performance of MLP in classifying MNIST digits it was noticed that CNN achieved an overall validation accuracy of 99.2% which is 1.2% more than the accuracy achieved in MLP. Also, it is

observed in Figure 3.18 that the overfitting is reduced as the training accuracy is very close to the validation accuracy with a value of 99.3%. That means the CNN model has the ability to generalise more when compared with MLP's performance in Figure 3.12. The confusion matrix and other evaluation metrics of the validation data are also obtained in Figure 3.19 and Table 3.4.

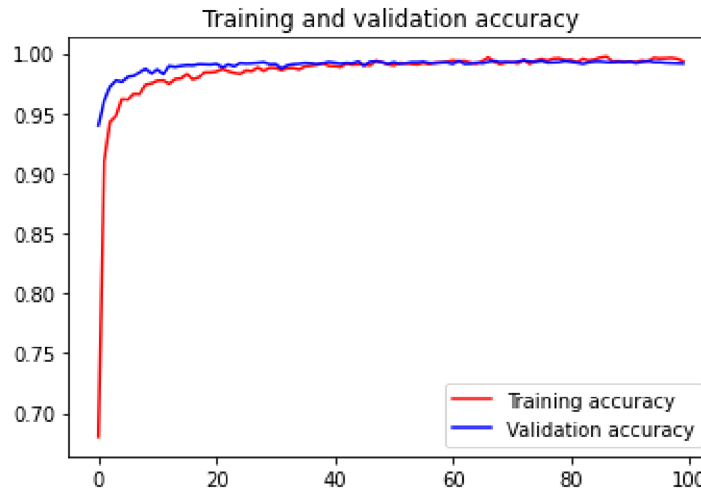


Figure 3.18: CNN's training and validation accuracy for MNIST dataset.

Table 3.4: Metrics for evaluating the validation dataset in CNN.

Class	Recall	Precision	F1-score
Zero	0.999	0.988	0.993
One	0.984	0.996	0.99
Two	0.996	0.986	0.991
Three	0.997	0.989	0.993
Four	0.996	0.995	0.995
Five	0.994	0.988	0.991
Six	0.987	0.989	0.988
Seven	0.988	0.995	0.992
Eight	0.99	0.994	0.992
Nine	0.984	0.996	0.99

3.5.2 Common CNN Architectures

From the late 1990s up to present time, various improvements on CNN architecture have been implemented. Improvements in CNNs included different key aspects such as depth and connectivity

Actual Class \ Predicted Class	Zero	One	Two	Three	Four	Five	Six	Seven	Eight	Nine
Zero	979 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	1 0%	0 0%	0 0%
One	3 0%	1117 98%	2 0%	3 0%	0 0%	0 0%	10 1%	0 0%	0 0%	0 0%
Two	0 0%	1 0%	1028 100%	2 0%	0 0%	0 0%	0 0%	1 0%	0 0%	0 0%
Three	0 0%	0 0%	0 0%	1007 100%	0 0%	2 0%	0 0%	0 0%	0 0%	1 0%
Four	0 0%	0 0%	1 0%	0 0%	978 100%	0 0%	1 0%	0 0%	0 0%	2 0%
Five	1 0%	0 0%	0 0%	4 0%	0 0%	887 99%	0 0%	0 0%	0 0%	0 0%
Six	5 1%	0 0%	0 0%	0 0%	1 0%	6 1%	946 99%	0 0%	0 0%	0 0%
Seven	0 0%	3 0%	8 1%	0 0%	0 0%	0 0%	0 0%	1016 99%	0 0%	1 0%
Eight	3 0%	1 0%	4 0%	2 0%	0 0%	0 0%	0 0%	0 0%	964 99%	0 0%
Nine	0 0%	0 0%	0 0%	0 0%	4 0%	3 0%	0 0%	3 0%	6 1%	993 98%

Figure 3.19: Confusion matrix for MNIST dataset classification using CNN.

of layers, hyperparameters, optimisation strategies and regularisation techniques [385]. Noteworthy, these improvements were possible due to the computing and processing power breakthroughs lately. This has led to common CNN architecture that can be used for multiple applications, begging from LeNet-5 that classified MNIST digits and ending at ResNext model in 2017 [386]. Figure 3.20 summarises these architectures. Hence, there is a huge gap between LeNet-5 and AlexNet, which is due to the lack of computing units that can process DNNs during this time.

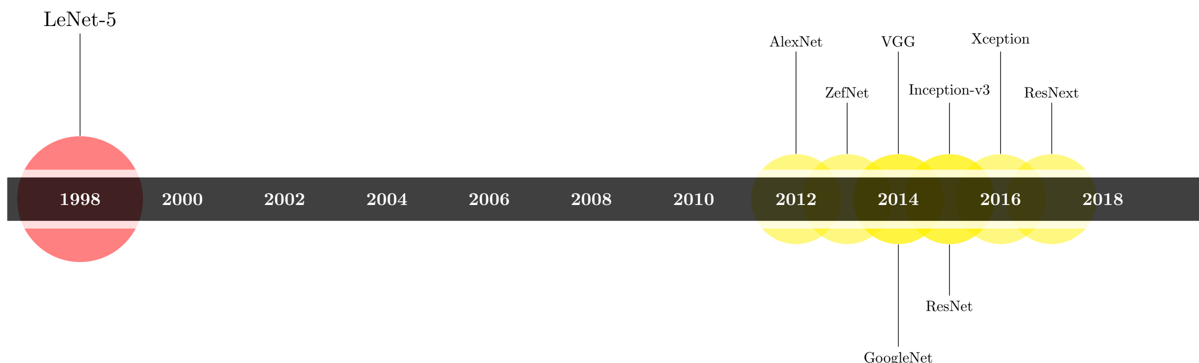


Figure 3.20: Evolution of common CNN architectures.

3.5.2.1 LeNet-5

LeNet-5 is considered the first deep CNN, number five refers to the total number of convolution and FC layers used in the network. It was proposed by LeCun et al. [379] in 1998 for MNIST hand written digits classification. The size the MNIST images at that time was 32×32 as shown in Figure 3.21 that demonstrated LeNet-5 architecture. As shown in Figure 3.21, the network consists of three convolutional layers, two pooling layers and two FC layers.

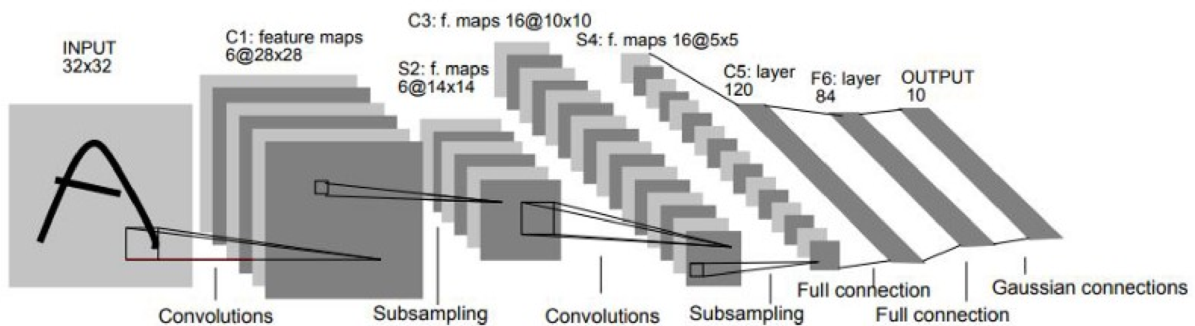


Figure 3.21: LeNet-5 architecture.

3.5.2.2 AlexNet

AlexNet was proposed by Krizhevesky et al. in 2012 [380]. At the time of publication of their article, the author pointed that their architecture was one the deepest CNNs. The architecture peoposed consisted of five convolutional layers, three pooling layers and three FC layers. ImageNet dataset was used to evaluate the performance of this algorithm with an image size of $227 \times 227 \times 3$. To ensure the robustness of feature extraction process and avoid overfitting, the authors introduced the dropout concept in their network. Furthermore, to accelerate the early stage of learning, ReLU activation. function was utilised. Other modifications were made such as using considerably large-size filters (5×5 and 11×11) in the earlier layers.

3.5.2.3 ZefNet

Generally, CNN training was conducted by performing trial-and-error in modifying different hyperparameters. In 2013, ZefNet was proposed by Zeiler and Fergus in [387]. They introduced the concept of visualising the inner layers of CNNs which helped in finding methods to optimise the network's performance. They have also introduced in utilising the deconvolution concept

which helped in the formation of convoutional autoencoder and deep convolutional generative adversarial networks (discussed in Sections 3.6.2 and 3.7.2). The network was evaluated using ImageNet dataset.

3.5.2.4 VGG-16

VGG network was proposed by Visual Geometry Group in 2014 [381]. The network consisted of 13 convolutional, five pooling and three FC layers, carrying with them the ReLU tradition from AlexNet. VGG utilised a layer of the heap of 3×3 filters rather than 5×5 and 11×11 filters as in AlexNet and ZefNet. This has proven that the parallel assignment of these filters could produce the same influence as the large-size filters in some datasets such as ImageNet. However, VGG's computational cost was large due the utilisation of 138 million parameters, which represented its main shortcoming [386]. A deeper version called VGG-19 was also designed by the same group.

3.5.2.5 GoogleNet

GoogleNet (also called Inception-V1) was proposed in 2014 [388]. It consists of 22-layer architecture with 5 millions parameters. Three main contributions were observed in the architecture [389]:

1. It has parallel paths of convolutions with multiple filter sizes, followed by concatenation, that captures different features and cluster them.
2. Computational bottlenecks were removed by using 1×1 convolutions.
3. Authors have introduced two auxiliary classifiers to encourage discrimination in the lower stages of classification. Auxiliary classifier can also increase the gradient signal that gets propagated back, and it provides additional regularisation effect.

3.5.2.6 Inception-V3

Inception-V3 is the successor of GoogleNet. It was proposed by the same authors in 2015 [390]. Batch normalisation was the key aspect that distinguished this architecture form the previous one. Other aspects were also considered such as convolution layers number and size.

3.5.2.7 ResNet

Residual Network (ResNet) was proposed by He et al. [382] in 2015. In addition to the previous architectures which proposed layer and hyperparameters modifications, ResNet has proposed also the idea of shortcut connections (also know as residuals or residual layers). Unlike traditional CNNs that each layer in it feeds into the next layer, in a network with residual layers each layer feeds into the next layer and also to the next layers by skipping the layers in between as shown in Figure 3.22. Due to this concept, the output of the layer is not the same now as the input 'x'

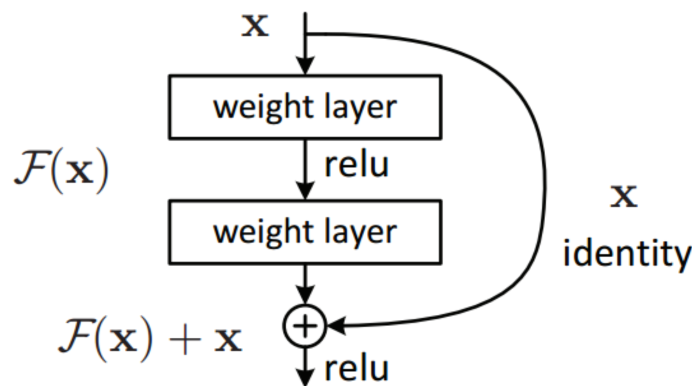


Figure 3.22: Residual layers concept [382].

gets multiplied by the weights of the layer followed by adding a bias term. In return, this speeds up the learning procedure by using fewer layers and reduces the overfitting. It also offers lower computational complexity and memory usage. This structure contained 49 convolutional layers plus a single FC layer, and it is usually called ResNet-50. A deeper version of this network was also proposed called ResNet-100.

3.5.2.8 Xception

Xception was proposed by Chollet in 2016 [391]. It is an adaptation from Inception-v1, where the Inception-v1 modules have been replaced with depthwise separable convolutions. It has also almost the same number of parameters as Inception-v1. The Xception is considered computationally inexpensive through the use of the decoupling channel and spatial correspondence. Furthermore, it performs mapping by applying 1×1 convolutions to every channel then 3×3 to each output. This is identical to replacing the Inception module with depthwise separable convolutions [386, 389].

3.5.2.9 ResNext

ResNext as an enhanced version of ResNet, VGG and GoogleNet networks. It was proposed by Xie et al. in 2017 [392]. It employed the VGG deep homogeneous topology with the basic architecture of GoogleNet by setting 3×3 filters as spatial resolution inside the blocks of transform, split, and merge. It also added parallel towers/branches/paths within each module to the original ResNet [386, 389].

3.6 Autoencoder

The previously discussed networks are working in supervised manner, in which labelled examples are necessary in order for the algorithms to carry on their task of learning and generating the desired output. Autoencoders (AEs) are branch of DL networks that conduct unsupervised learning strategy which does not require the training samples to be labelled. AE consists of three main parts encoder, decoder and bottleneck as shown in Figure 3.23. The encoder maps the input data from a high-dimensional space into codes in a low-dimensional space by extracting the meaningful features and discarding the redundant ones. The learned representations are encoded at the bottleneck layer, known as the code or latent space, which typically has a much lower dimensionality than the input data [393]. The decoder reconstructs the inputs from the corresponding latent space. AE can learn features in unsupervised manner by minimising reconstructed errors between input data and output. It can be used in various applications such as image denoising [232] and anomaly detection [69, 394].

3.6.1 Evaluating AE using MNIST Dataset

In this section, the reconstruction performance of AE using MNIST dataset will be evaluated. The AE architecture used is simply a $784 - 64 - 784$ MLP as shown in Figure 3.24.

The input image of 28×28 should be vectorised to $[784, 1]$ as has been done with MLP and perceptron in the previous sections. The encoding happens when the parameters are feedforwarded from input layer to the bottleneck and the decoding happens when the parameters are feedforwarded from bottleneck to output layer resulting on the reconstructed image. In this example, since there is not classification accuracy, we are more interested in the loss function as

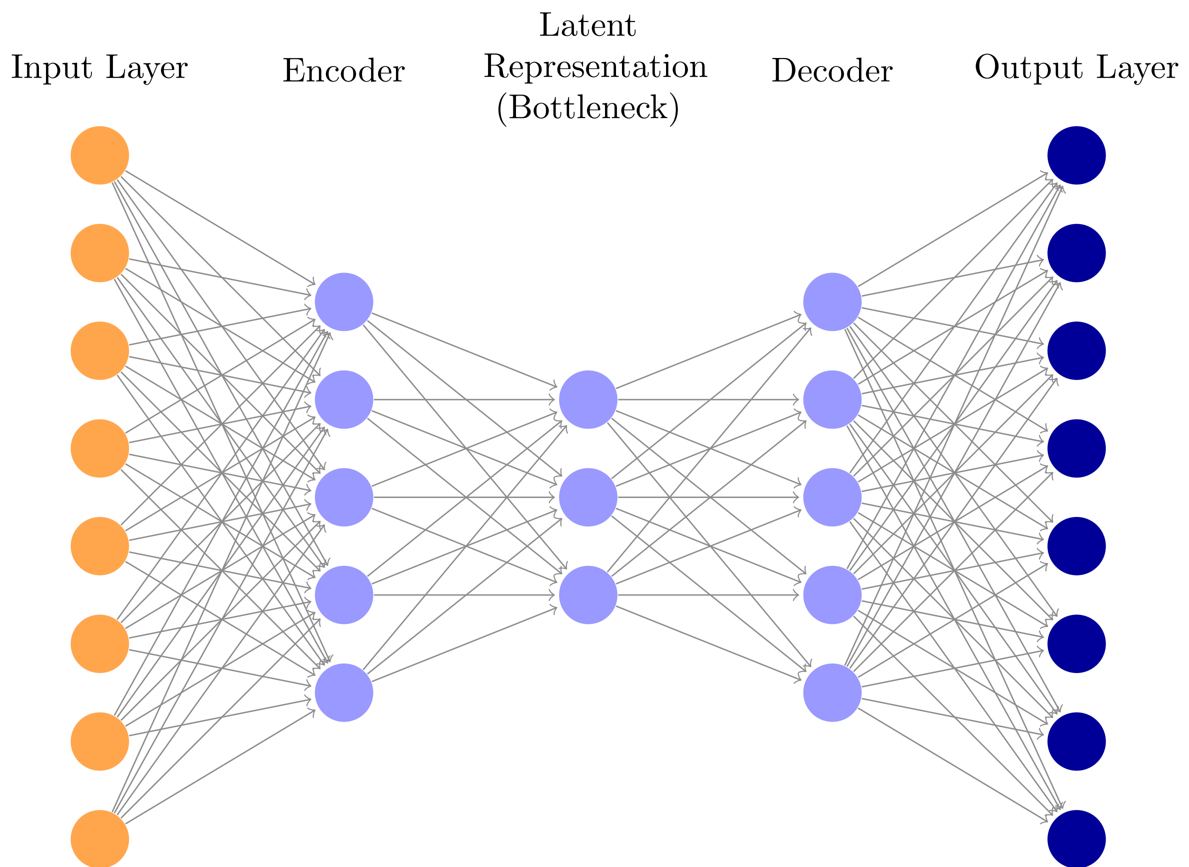


Figure 3.23: AE architecture.

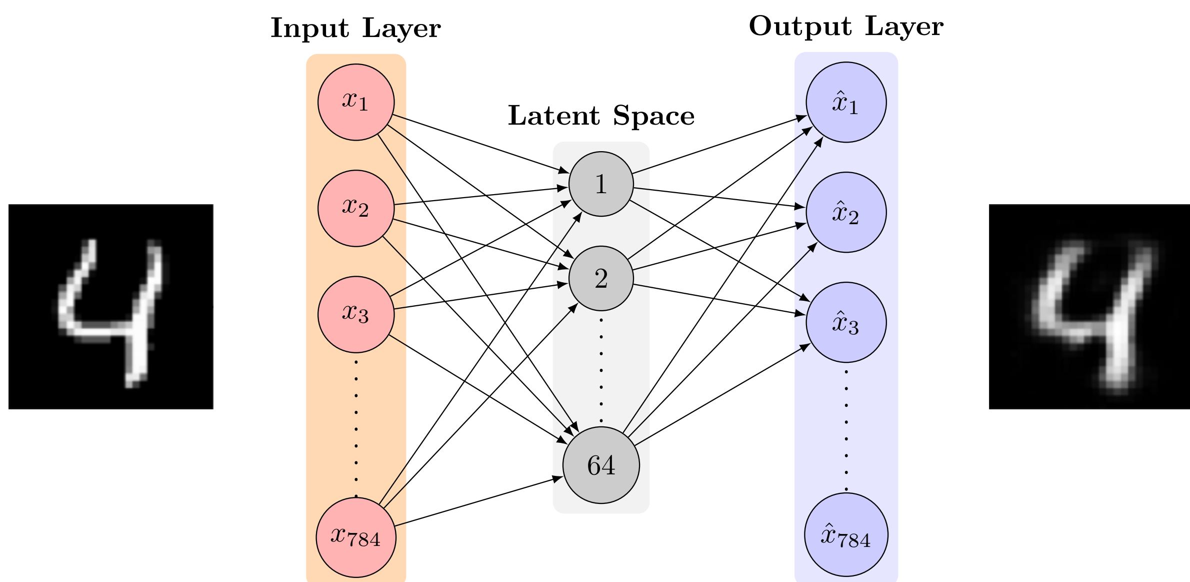


Figure 3.24: AE architecture for reconstructing MNIST dataset.

shown in Figure 3.25. The training of this AE has gone through 100 epochs without a stopping

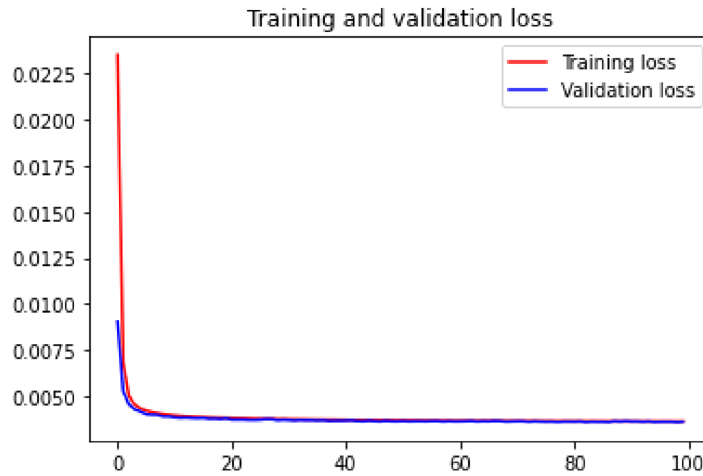


Figure 3.25: AE's training and validation loss for MNIST dataset.

condition. The performance of the AE in reconstruction image sample is demonstrated in Figure 3.26.

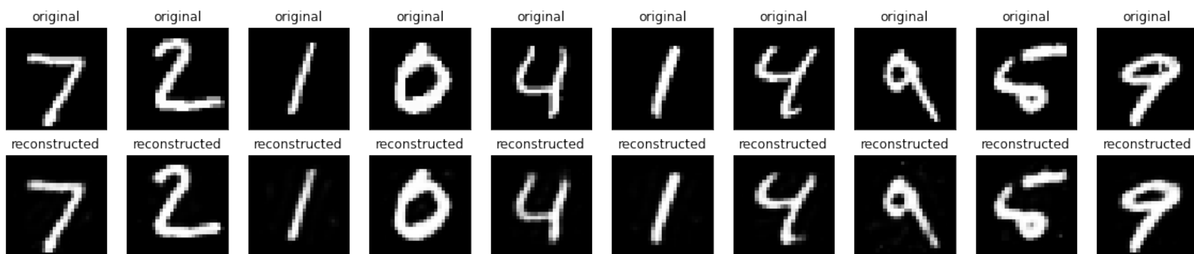


Figure 3.26: Original vs reconstructed results of AE.

3.6.2 Convolutional Autoencoder

Instead of using FC layers to construct AE as shown in Figure 3.23, convolutional layers can be also used to form what so called convolutional autoencoder (CAE). CAE take the advantage of convolutional layers that leverage the ideas of sparse connectivity, parameter sharing and equivariant representation. Therefore, they provide more effective reconstruction of image data when compared to AE with FC layers. The architecture of CAE is demonstrated in Figure 3.27.

Upon first glance, the process of reconstructing an image may sound trivial and it is hard to imagine how this can be related to the process of anomaly detection. However, when the AE is well-trained on reconstructing a set of data (e.g. a sample of normal images), it means

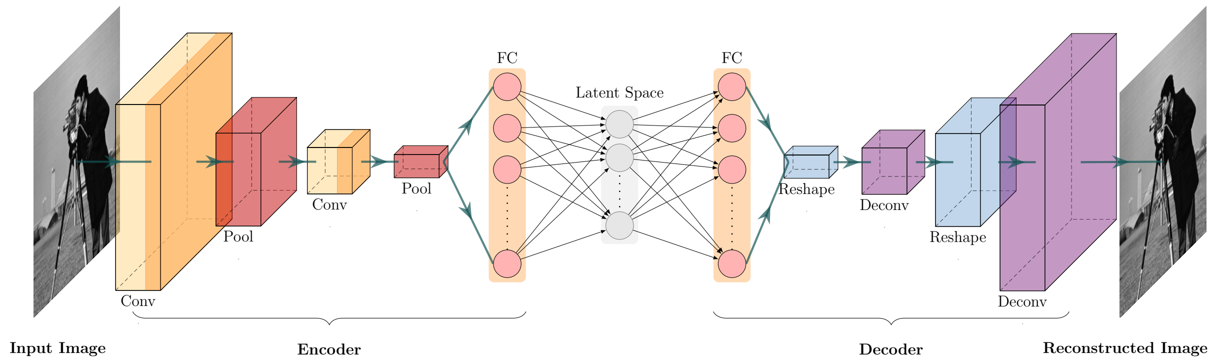


Figure 3.27: CAE architecture.

that it can recognise these data very well and therefore, the latent space of these data can be analysed and compared with the latent space of data that does not look similar (e.g. anomaly images) [395] as will be explored in Section 5.8.1. This process can help in binary classification for anomaly detection application.

3.6.3 Evaluating CAE using MNIST Dataset

Same as had been done in Section 3.6.1, the reconstructing performance of MNIST dataset will be evaluated using CAE. As in CNN, CAE can deal with image data as they are without the need of converting them into vector form. The CAE architecture used for evaluation is shown in Figure 3.28.

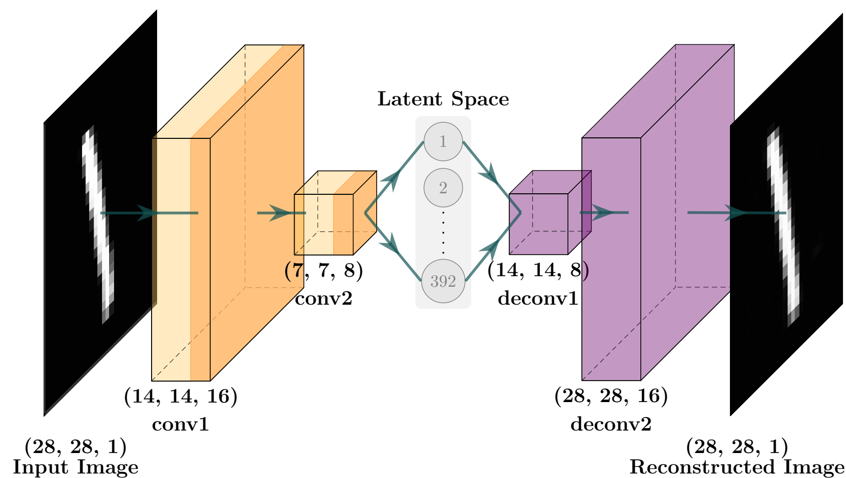


Figure 3.28: CAE architecture for reconstructing MNIST dataset.

The encoding happens when the parameters are feedforwarded from input layer through convolutional layers to the bottleneck and the decoding happens when the parameters are feed-

forwarded from bottleneck through deconvolutional layers to output layer resulting on the reconstructed image. The loss function of CAE is shown in Figure 3.29. It can be observed that the loss is much lower when compared to the performance of AE in Figure 3.29.

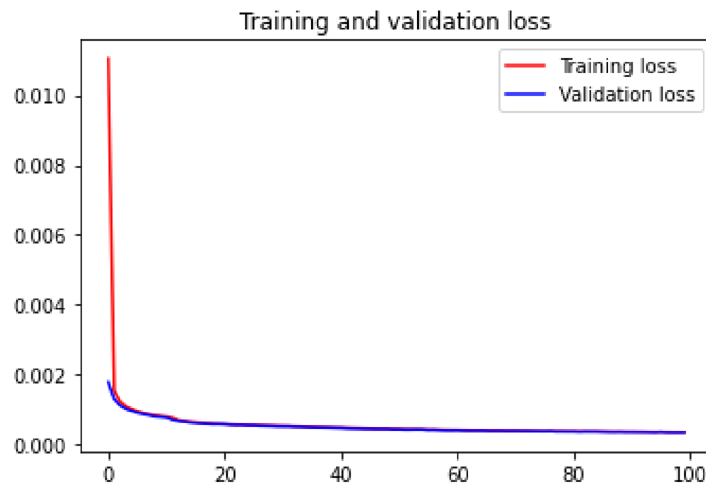


Figure 3.29: CAE's training and validation loss for MNIST dataset.

The training was also done over 100 epochs the and reconstructed images from the original are shown in Figure 3.30.

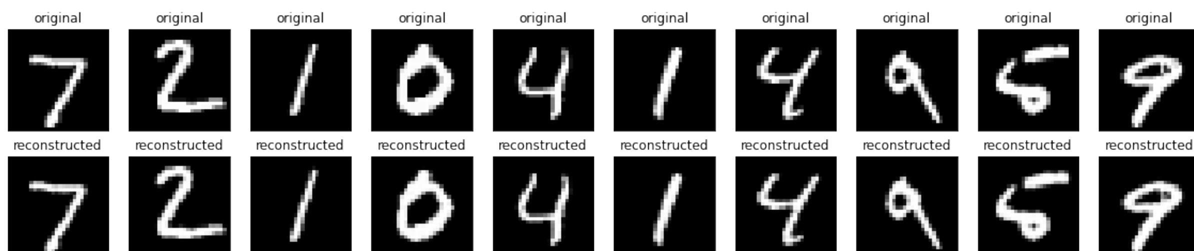


Figure 3.30: Original vs reconstructed results of CAE.

3.7 Generative Adversarial Networks

Generative Adversarial Networks (GAN) was first introduced in 2014 by Goodfellow et al. [396]. It consists of two ANNs, namely, generator and discriminator. The original paper suggested that both generator and discriminator are MLPs. The generator is responsible for producing synthetic images that look like the training dataset and supply these produced images for the discriminator. The discriminator is responsible for making the decision whether the produced images from the

generator look similar to the real data or not using binary classification scheme. This framework can be used to generate realistic new images that are almost identical to pre-existing training dataset by training the generator and discriminator simultaneously using adversarial process. The GAN architecture is demonstrated in Figure 3.31.

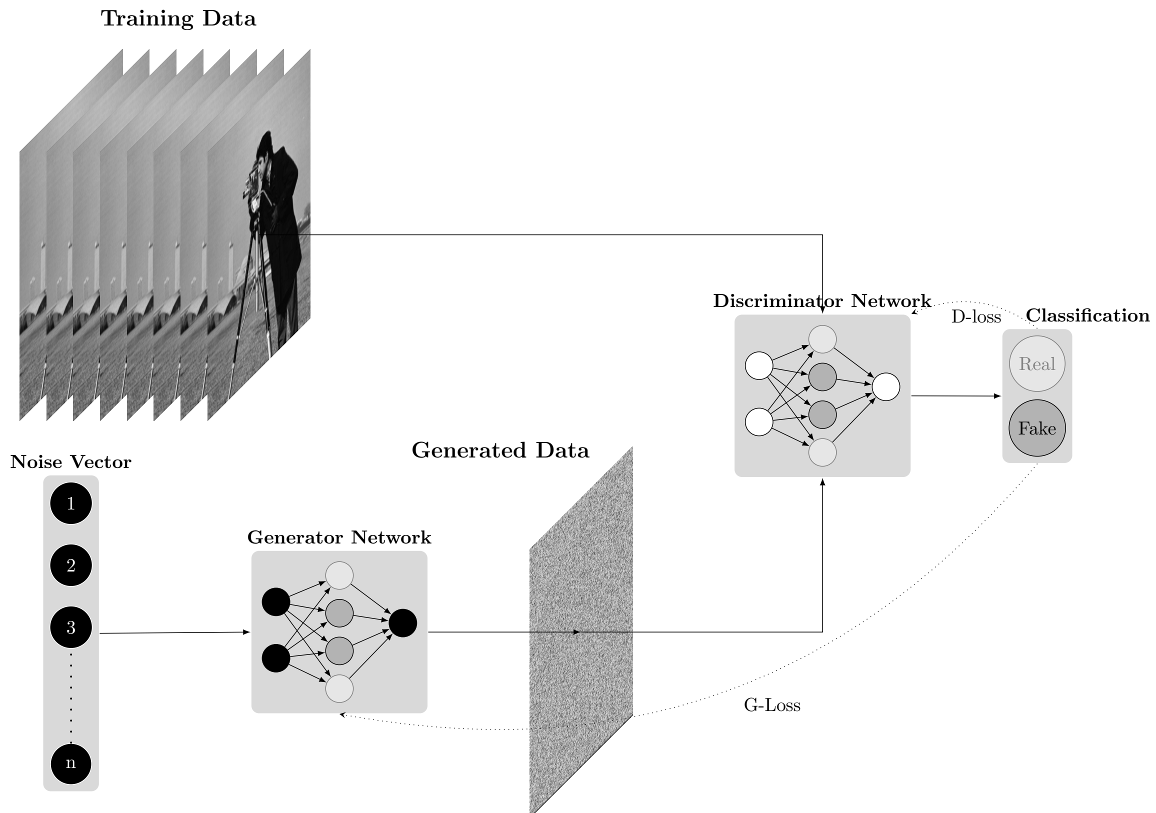


Figure 3.31: GAN architecture.

During training, the generator progressively becomes better at creating images that look real, while the discriminator becomes better at distinguishing real from synthetic images. The process reaches equilibrium when the discriminator can no longer distinguish between real and synthetic images. The minimax concept suggested by Goodfellow et al. [396] to fulfill this objective, which is given by Equation 3.15. Equation 3.15 indicates that the discriminator is trained to maximize the probability $\log(D(x))$ of assigning the correct label to both the generated synthetic data probability and real training data using while the generator is trained to minimize

$\log(1 - D(G(z)))$.

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)} [\log (D(x))] + \mathbb{E}_{z \sim p_z(z)} [\log (1 - D(G(z)))] \tag{3.15}$$

where $D(x)$ is the probability that x belong to the original data distribution, $G(z)$ is the generator function that maps to the data space, $\mathbb{E}_{x \sim p_{data}(x)}$ is the expected value over all real samples, $\mathbb{E}_{z \sim p_z(z)}$ is the expected value over all fake samples [397].

Since discriminator is performing binary classification (real or fake), binary cross entropy (BCE) loss function (given by equation 3.16) is used for the discriminator to penalize itself for misclassifying real and fake images.

$$J_q(w) = -\frac{1}{N} \sum_{n=1}^{N} y_n \cdot \log (q(y_n)) + (1 - y_n) \cdot \log (1 - q(y_n))) \tag{3.16}$$

where y_n is the label for training example n (1 for real samples and 0 for fake), $q(y)$ is the predicted probability of the point being real for all N points.

3.7.1 Evaluating GAN Using MNIST Dataset

In this section the discriminator will be trained to distinguish between real and fake MNIST samples by supplying MNIST training examples to it. The discriminator is an MLP of $784 - 128 - 256 - 512 - 1$. LeakyReLU with $\alpha = 0.2$ activation function was used at each layer except on the output layer where sigmoid activation function was used. The generator was trained to optimise the random noise vector of $[100,]$ to an image that look similar to MNIST dataset of size 28×28. The generator is also an MLP of $100 - 128256 - 512 - 784$. As in the discriminator LeakyReLU activation function was used in all layers expect the output layer. In the output layer tanh activation function was used. The training took place over 100 epochs and loss of the training and validation are demonstrated in Figure 3.32.

Figure 3.33 also shows the development stages of the constructed synthetic image of the generator at each epoch.

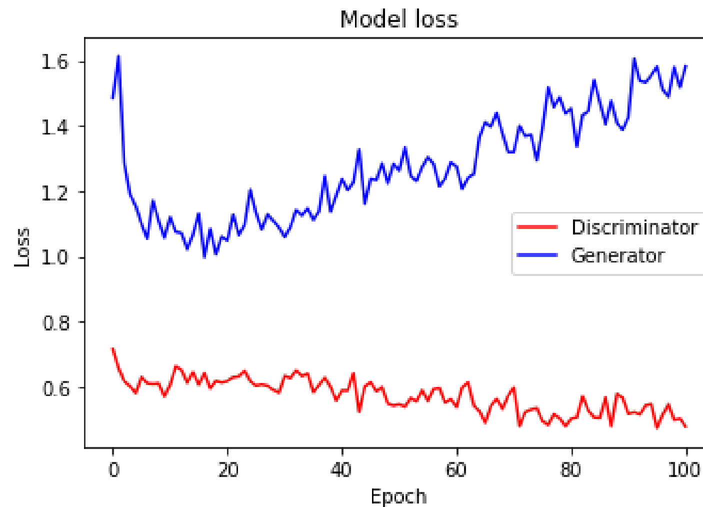


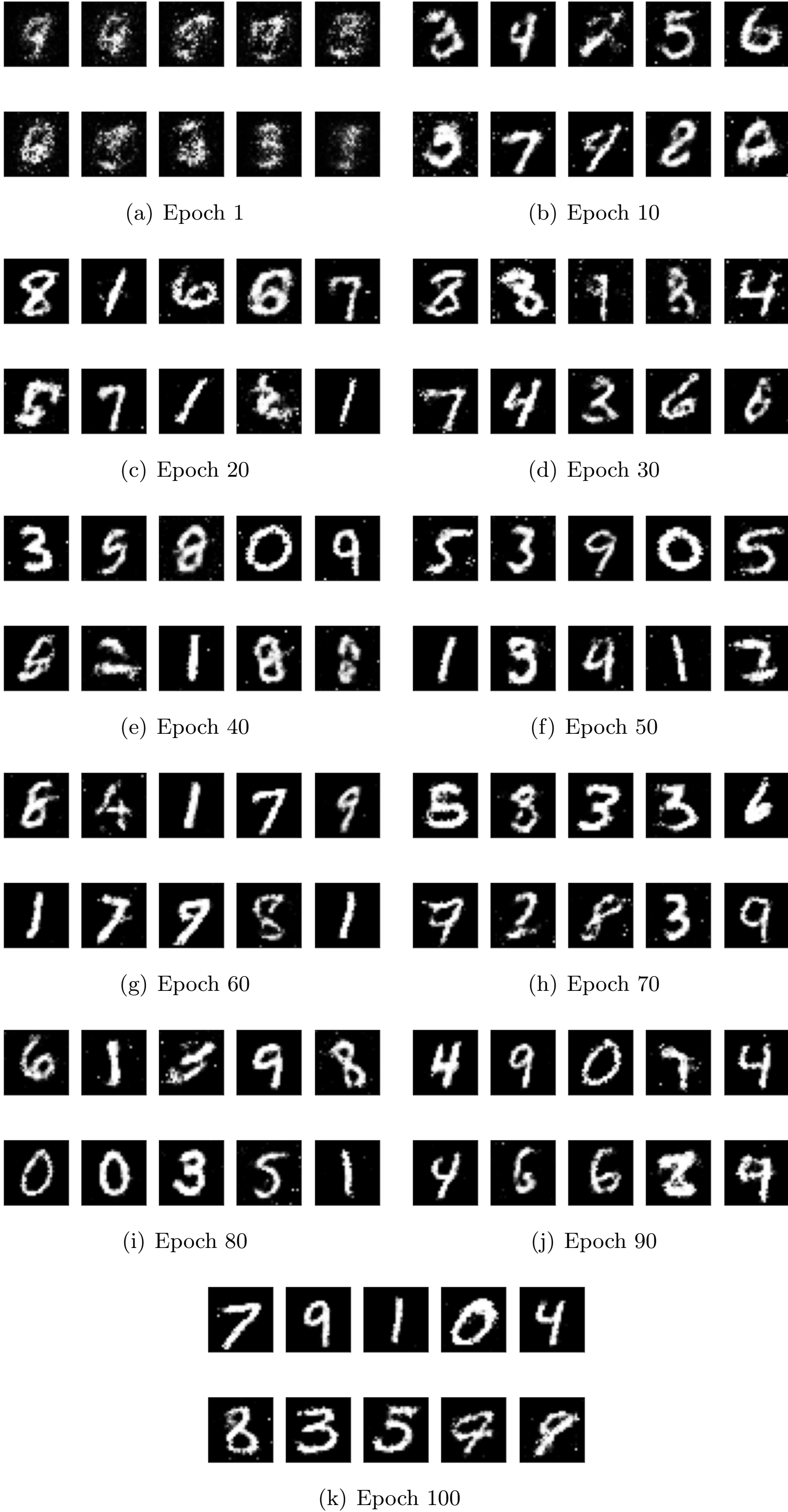
Figure 3.32: GAN loss in training MNIST dataset.

3.7.2 Deep Convolutional Generative Adversarial Networks

Deep Convolutional Generative Adversarial Networks (DCGAN) was proposed in 2016 by Radford et al. [398] and is considered an extension of the original GAN proposed by Goodfellow et al. in [396]. The architecture of DCGAN is almost the same in the original GAN except that convolutional and convolutional transpose (deconvolutional) layers are used in discriminator and generator networks respectively instead of the MLP structure. In DCGAN, the generator network receives a one-dimensional random Gaussian vector (usually of size $[100,]$) as an input. Multiple transpose convolutional layers are then applied to upscale the vector into random noise image. All the deconvolutional layers are followed by ReLU activation function.

3.7.3 Evaluating DCGAN Using MNIST Dataset

As has been done in Section 3.7.1, the performance of DCGAN using MNIST dataset will be evaluated. The discriminator network proposed is a CNN with three convolutional layers. LeakyRelU activation with $\alpha = 0.2$ with batch normalisation were used in the convolutional layers. The output layer is dense layer with sigmoid activation function. The generator is also a CNN with three deconvolutional layers. ReLU activation function and batch normalisation were used at the first three deconvolutioal layers, while tanh activation was used at the output layer. The training took place over 100 epochs and loss of the training and validation are demonstrated in Figure 3.34.



(a) Epoch 1
(b) Epoch 10
(c) Epoch 20
(d) Epoch 30
(e) Epoch 40
(f) Epoch 50
(g) Epoch 60
(h) Epoch 70
(i) Epoch 80
(j) Epoch 90
(k) Epoch 100

Figure 3.33: GAN epochs.

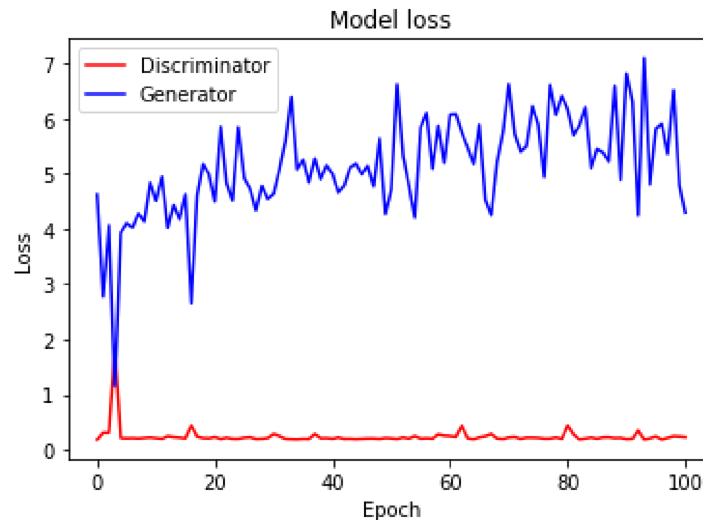


Figure 3.34: DCGAN loss in training MNIST dataset.

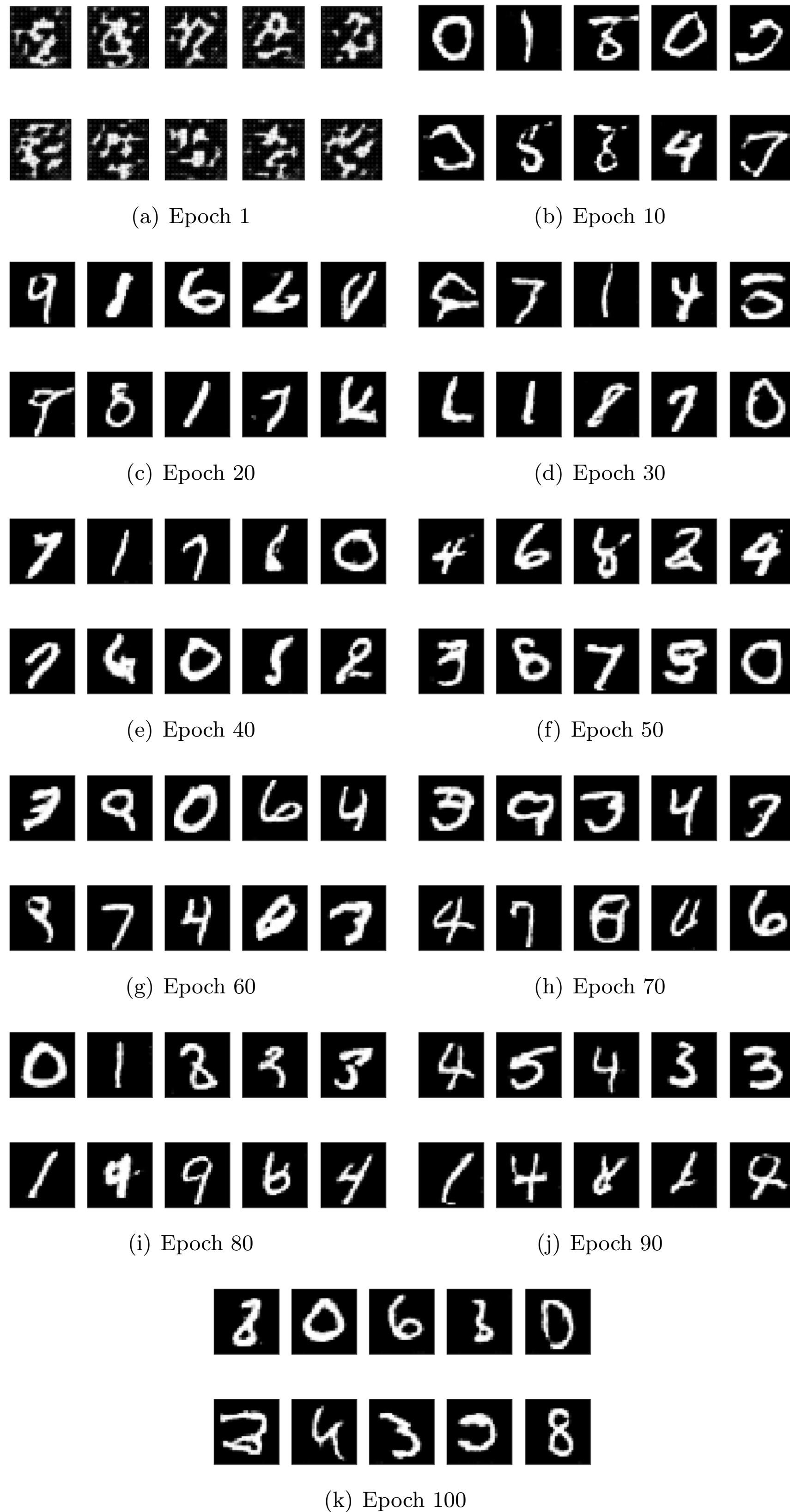
Figure 3.35 also shows the development stages of the constructed synthetic image of the generator at each epoch.

3.8 Capsule Networks

CNNs have been for a long time a popular DL tool in dealing with machine vision problems [399]. Despite their remarkable performance in image classification tasks, CNNs have several drawbacks. For instance, the pooling layers in the CNN can decrease the number of features extracted by the network and therefore valuable information in the image will be lost. Furthermore, CNNs are not very good at detecting the spatial location of the features in the image [400]. CapsNet is a newly proposed neural network that can overcome the previous problems. It was originally proposed by Sabour et al. in 2017 [401] to classify MNIST handwritten digits.

Two key aspects distinguish CapsNets from CNNs, which are layer-based squashing and dynamic routing [402]. CapsNet replaces the scalar-output feature detectors of CNNs with vector-output capsules and replaces pooling with routing-by-agreement. Each neuron in the capsule represents various features in particular parts of an image. In this way, the whole entity of the image can be recognized by considering each part [403]. The architecture of the original CapsNet proposed in Sabour et al. paper [401] is shown in Figure 3.36.

As demonstrated in Figure 3.36, the original network is made up of four major layers [404]:



(a) Epoch 1 (b) Epoch 10 (c) Epoch 20 (d) Epoch 30 (e) Epoch 40 (f) Epoch 50 (g) Epoch 60 (h) Epoch 70 (i) Epoch 80 (j) Epoch 90 (k) Epoch 100

Figure 3.35: DCGAN epochs.

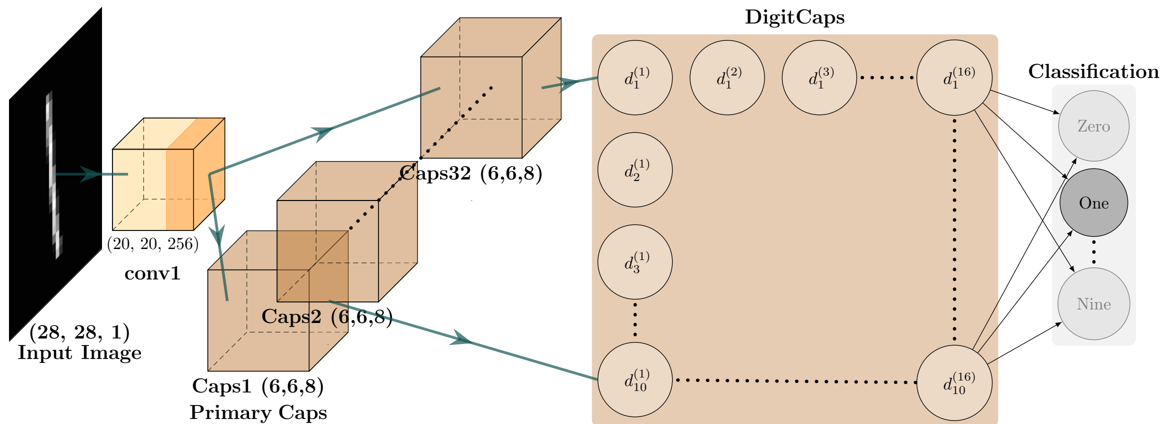


Figure 3.36: Originally proposed CapsNet for MNIST handwritten digit classification.

- Convolutional layer, which is a standard convolutional layer as the one found in CNNs. The input image size for this layer is 28×28. 265 kernels were used along with stride of 1 and ReLU activation function and to generate the feature maps and each kernel has a size of 9×9. The resulting feature map output after applying the layer has a size of $20 \times 20 \times 265$.

- PrimaryCaps. In this layer, the feature maps resulted from the convolutional layer were split into 32 capsules where convolutional operations are performed. The kernel is size in this layer is again 9×9 and a stride of 2. This will produce an output size of $6 \times 6 \times 8$ for each capsule.

- DigitCaps. This layer has 16 capsules per digit class and each of these capsules receives input from all the capsules in the previous layer.

- Fully connected layer for classification. The input image size was 28×28 to match the size of MNIST data used for evaluating the network.

Else than the first layer that shares some similarities with CNN in the feature extraction process, the other layers behave in a different manner. In the second layer (PrimaryCaps), each capsule i from the 32 has an activity vector $\mathbf{u}_i$ to encode the spatial information in the form of instantiation parameters. Then the output of $\mathbf{u}_i$ is fed to the next layer (DigitCaps), such that each capsule j from the 16 per digit class will receive $\mathbf{u}_i$ and multiply with the weight matrix $\mathbf{W}_{ij}$. This will result in the prediction vector $\hat{\mathbf{u}}_{j|i}$, which indicates the amount of contribution for capsule i in

the PrimaryCaps on capsule j in the DigitCaps as given by equation 3.17.

$$\hat{\mathbf{u}}_{j|i} = \mathbf{W}_{ij}\mathbf{u}_i \tag{3.17}$$

The predictions are then multiplied by a coefficient called coupling coefficient c that represents the agreement between capsules as in equation. Hence, coefficient c is updated based on iterative process to form what so called "Dynamic Routing". This process can be determined by routing softmax function whose initial logits b_{ij} are the log prior probabilities that capsule i in the PrimaryCaps should be coupled to capsule j in the DigitCaps. This operation can be demonstrated by equations 3.18-3.21.

$$a_{ij} = \mathbf{s}_j \cdot \hat{\mathbf{u}}_{j|i} \tag{3.18}$$

$$b_{ij} = b_{ij} + a_{ij} \tag{3.19}$$

$$c_{ij} = \frac{\exp\left(b_{ij}\right)}{\sum_k \exp\left(b_{ij}\right)} \tag{3.20}$$

$$\mathbf{s}_j = \sum_i c_{ij}\hat{\mathbf{u}}_{j|i} \tag{3.21}$$

where $\mathbf{s}_j$ is weighted sum that is calculated to obtain the candidates for a squashing function v_j. The squashing operation is responsible for creating a normalized vector from the multiple neurons contained in the capsule. The activation function used in this step is given by equation 3.22.

$$v_j = \frac{\|\mathbf{s}_j\|^2}{1+\|\mathbf{s}_j\|^2} \cdot \frac{\mathbf{s}_j}{\|\mathbf{s}_j\|} \tag{3.22}$$

A margin loss function was defined to assist the classification process. The function evaluates the loss term coming from the output vector of DigitCaps. This will help in deciding whether the selected digit capsule matches the real target value of class k. The formula of the margin loss function is given by equation 3.23.

$$L_k = T_k \max\left(0, m^+ - \|v_k\|\right)^2 + \lambda\left(1 - T_k\right) \max\left(0, \|v_k\| - m^-\right)^2 \tag{3.23}$$

Where T_k is a label (0 or 1) indicating whether a class k is present "1" or not "0". Terms m^+, m^-, and λ are the hyperparameters of the model such that $m^+ = 0.9$, $m^- = 0.1$ and $\lambda = 0.5$. The routing procedure can be further described in Algorithm 2.

Algorithm 2 Routing algorithm.

1: **procedure** ROUTING($\hat{\mathbf{u}}_{j|i}, r, l$)
2: for all capsules i in layer l and capsule j in layer $(l+1)$: $b_{ij} \leftarrow 0$.
3: **for** r iterations **do**
4: for all capsule i in layer l: $\mathbf{c}_i \leftarrow \texttt{softmax}(\mathbf{b}_i)$
5: for all capsule j in layer $(l+1)$: $\mathbf{s}_j \leftarrow \sum_i c_{ij}\hat{\mathbf{u}}_{j|i}$
6: for all capsule j in layer $(l+1)$: $\mathbf{v}_j \leftarrow \texttt{squash}(\mathbf{s}_j)$
7: for all capsule i in layer l and capsule j in layer $(l+1)$:
8: $b_{ij} \leftarrow b_{ij} + \hat{\mathbf{u}}_{j|i} \cdot \mathbf{v}_j$
9: **return** $\mathbf{v}_j$
10: **end for**
11: **end procedure**

3.8.1 Evaluating CapsNet using MNIST Dataset

The original authors of Sabour et al. [401] used MNIST to evaluate the performance of their proposed CapsNet. The original architecture of the network were used to obtain more detailed metrics of each class. The training took place over 100 epochs and the validation and training accuracies were estimated at each epoch as shown in Figure 3.37.

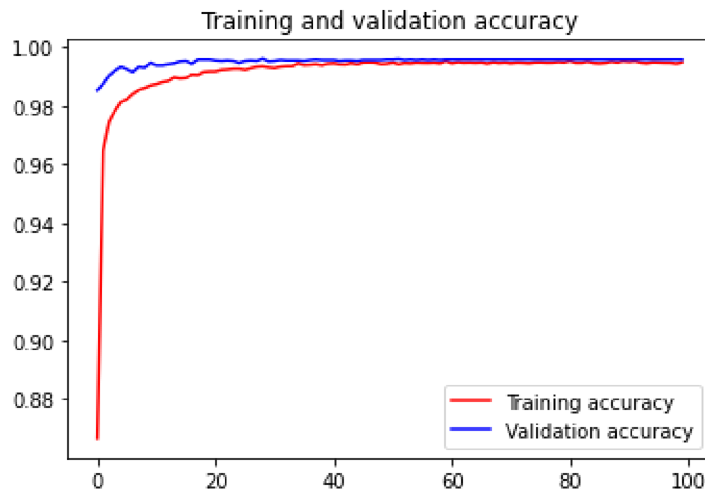


Figure 3.37: CapsNet's training and validation accuracy for MNIST dataset.

The estimated training accuracy 99.5% and the validation accuracy was 99.6%, we note form these results that there was not any overfitting and the validation accuracy was 0.3% higher of

that in the CNN. Confusion matrix in Figure 3.38 were also obtained that shows the detection accuracy of each class. Table 3.5 shows more evaluation metrics.

Actual Class \ Predicted Class	Zero	One	Two	Three	Four	Five	Six	Seven	Eight	Nine
Zero	980 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
One	0 0%	1131 100%	0 0%	0 0%	0 0%	0 0%	1 0%	3 0%	0 0%	0 0%
Two	2 0%	0 0%	1028 100%	0 0%	0 0%	0 0%	0 0%	2 0%	0 0%	0 0%
Three	0 0%	0 0%	0 0%	1009 100%	0 0%	1 0%	0 0%	0 0%	0 0%	0 0%
Four	0 0%	0 0%	0 0%	0 0%	979 100%	0 0%	0 0%	1 0%	0 0%	2 0%
Five	0 0%	0 0%	0 0%	4 0%	0 0%	886 99%	2 0%	0 0%	0 0%	0 0%
Six	4 0%	1 0%	0 0%	0 0%	0 0%	0 0%	952 99%	0 0%	1 0%	0 0%
Seven	0 0%	1 0%	5 0%	0 0%	0 0%	0 0%	0 0%	1022 99%	0 0%	0 0%
Eight	0 0%	0 0%	2 0%	1 0%	1 0%	1 0%	1 0%	0 0%	967 99%	1 0%
Nine	0 0%	0 0%	0 0%	0 0%	2 0%	1 0%	0 0%	2 0%	0 0%	1004 100%

Figure 3.38: Confusion matrix for MNIST dataset classification using CapsNet.

Noteworthy, capsule network can produce reconstruction images as well. Figure 3.39 shows a reconstructed image sample at 100th epoch.

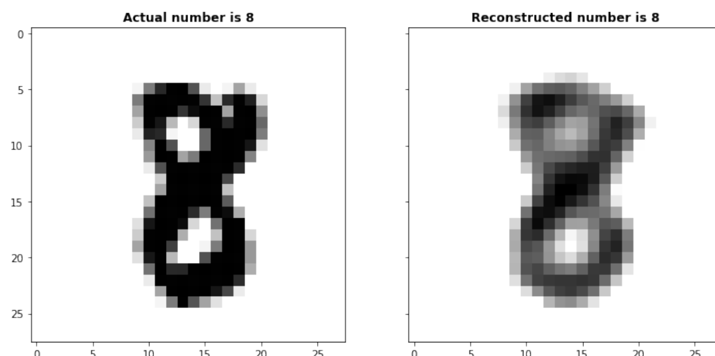


Figure 3.39: Original vs reconstructed results of CapsNet.

Table 3.5: Metrics for evaluating the test data of mixed (DCGAN & Original) dataset for different deep learning models

Class	Recall	Precision	F1-score
Zero	1.0	0.994	0.997
One	0.996	0.998	0.997
Two	0.996	0.993	0.994
Three	0.999	0.995	0.997
Four	0.997	0.997	0.997
Five	0.993	0.997	0.995
Six	0.994	0.996	0.995
Seven	0.994	0.992	0.993
Eight	0.993	0.999	0.996
Nine	0.995	0.997	0.996

3.9 Conclusion

This chapter presented a comprehensive background theory of common DL algorithms that are used in computer vision applications. The basic building block of any DNN is the perceptron, therefore we started our discussion from there until we got to more complex DL structures such as CNN and CapsNet. In this chapter, we also presented how data augmentation can be performed using DL methods such as GAN and DCGAN. Furthermore, we discussed about unsupervised learning methods such as AE and CAE that can preform image reconstruction. For evaluating these networks and architectures we used the well-known MNIST dataset and presented several metrics for evaluations such as confusion matrix, accuracy, recall, precision and F1-score. This chapter has also discussed some challenges in performing DL approaches such as lack of data for training and overfitting.

Chapter 4

Implementation of Deep Learning for Semiconductor Wafer Inspection [1]

> "*I would say that hardware is the bone of the head, the skull. The semiconductor is the brain within the head. The software is the wisdom and data is the knowledge.*"
>
> Masayoshi Son

4.1 Overview

Advances in semiconductor technology and design have been the driving forces behind the successful progress of microelectronic and optoelectronic devices. The majority of these devices are manufactured using semiconductor wafers that consist of several hundreds of integrated circuits (ICs) (also called dies) [32, 234]. However, the fabrication process for the semiconductor wafers is complex and consist of many stages that should take place in a clean room environment, such as oxidation, photolithography, etching, ion implementation, and metallization, which requires monitoring many key process parameters. The complexity of these steps makes the wafer prone to many kinds of defects and failures; therefore, wafer testing is an essential step in order to pro-

[1] The contents of this chapter were published in the following article: **A. M. Abu Ebayyeh** , S. Danishvar and A. Mousavi, "An Improved Capsule Network (WaferCaps) for Wafer Bin Map Classification Based on DCGAN Data Upsampling", *IEEE Transactions on Semiconductor Manufacturing*, vol. 35, no. 1, 2022. doi: 10.1109/TSM.2021.3134625

vide necessary information on specific manufacturing problems, which can then reduce products' flaws and lead to early prevention [203]. One of the testing methods used in inspecting the wafer is called circuit probe in which each die in the wafer is tested using an electrical probe to form a binary map image called 'Wafer Bin Map' (WBM). In WBM, the defective dies are represented with logic '1' and the normal dies with logic '0'.

Defective dies in the wafer tend to form a spatial pattern [197]. Figure 2.8 shows common inspected defect patterns in WBMs. An experienced inspector can identify the cause of defect depending on the WBM's pattern. The process of manually inspecting these defects is time consuming and may be affected by the fatigue's level of the inspector, especially because modern semiconductors manufacturers produce several thousands of wafers every week [230, 405]. Therefore, many semiconductor manufacturing are investigating this problem using machine learning and computer vision techniques to perform automatic defect detection [203].

In this chapter, WM-811k dataset that contains eight defect patterns will be investigated. Two DL approaches will be applied; DCGAN and an improved capsule network (WaferCaps). DCGAN will be used to increase the dataset sample and upsample them in the same time so each class of the eight will have similar number of samples. WaferCaps will be used to classify the original and augmented data according to the corresponding class.

4.2 Motivation

Many studies investigated WBM defect patterns using WM-811K dataset as summarised in Table 2.5. The majority of these studies implemented conventional computer vision techniques by extracting the defect pattern feature and use them as an input for rule-based classifier to obtain the relevant class. More recent research articles such as in applied DL algorithms (e.g. CNN) to classify the WBM defects automatically without the need of performing feature extraction manually. However, none of these articles gave a valid answer to the class imbalance problem within WM-811K data. Although, the total accuracy obtained in most of these articles are acceptable, but when a class imbalance exists in the data measuring the total accuracy will not be a good indicator to evaluate the classification performance as explained in Section 3.4.4. Furthermore, none of these articles paid attention to the spatial location of the pattern defect feature and how it can affect the performance of the classification specially when defect patterns

have similarities with each other (e.g. loc, and edge-loc) as will be seen in the next section.

To solve the previous problems, this chapter proposes a DL framework to augment the data so that each class has similar number of samples for training and an algorithm to classify the samples taking in consideration the spatial locations of the features. The following are the contributions of this chapter:

- A DCGAN algorithm, which is trained on each class of the dataset separately in order to upsample the data such as each class has similar number of samples. The proposed algorithm will significantly increase the diversity of data available for training without the need for obtaining real data. This technique also goes far beyond the conventional data augmentation by generating completely new data rather than performing simple rotation or scaling to an existing image.
- Proposing a new algorithm "WaferCaps" which is an improvement of CapsNet for considering the spatial location of the features in the data and enhancing the classification accuracy.

4.3 WM-811K Dataset

WM-811K is an open source dataset that include semiconductor WBM collected from 46,393 lots in real-world semiconductor industries. The same logic for WBM applies here such that defective dies have logic '1' and normal dies are represented by logic '0'. The properties included in the WM-811K dataset are `waferMap`, `dieSize`, `lotName`, `waferIndex`, `trainTestLabel`, and `failureType`. In total the data consists of 811,475 WBMs, 21.3% of the WBMs in the dataset have labels. Among labeled WBMs, 3.1% have failure patterns while 18.2% do not have patterns. The failure patters are eight in total namely Centre (Figure 4.1), Donut (Figure 4.2), Edge-Loc (Figure 4.3), Edge-Ring (Figure 4.4), Loc (Figure 4.5), Near-Full (Figure 4.6), Random (Figure 4.7), and Scratch (Figure 4.8). Figure 4.9(a) summarises the distribution of data for WM-811k dataset. In this chapter, labelled and patterned data that account for 3.1% of the total will be considered.

It is clear that the labeled and patterned WBMs data considered are highly imbalanced as shown in Figure 4.9(b). For instance, the Edge-Ring class has 8,268 samples while the Donut

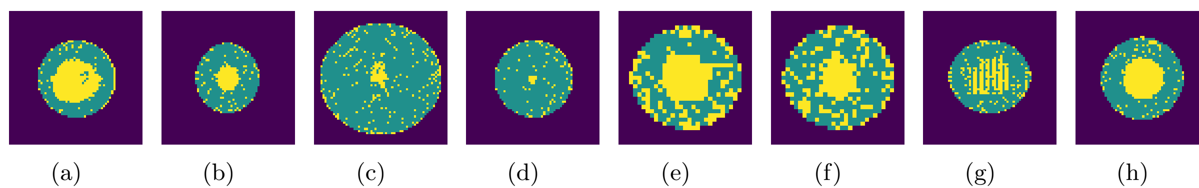


Figure 4.1: Centre WBM patterns (yellow dots indicate defective dies).

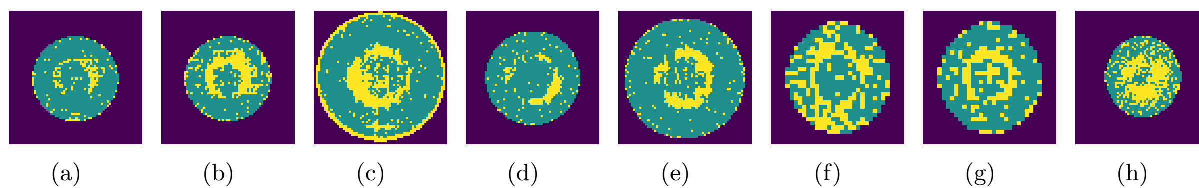


Figure 4.2: Donut WBM patterns (yellow dots indicate defective dies).

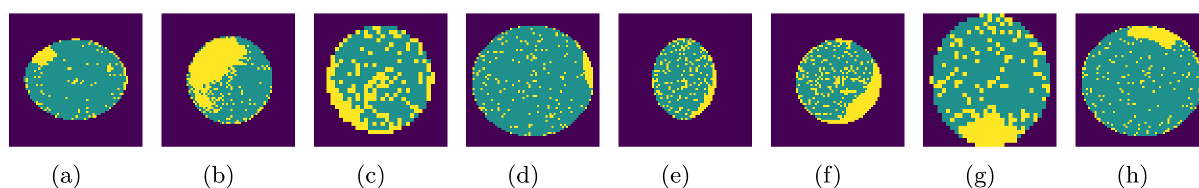


Figure 4.3: Edge-Loc WBM patterns (yellow dots indicate defective dies).

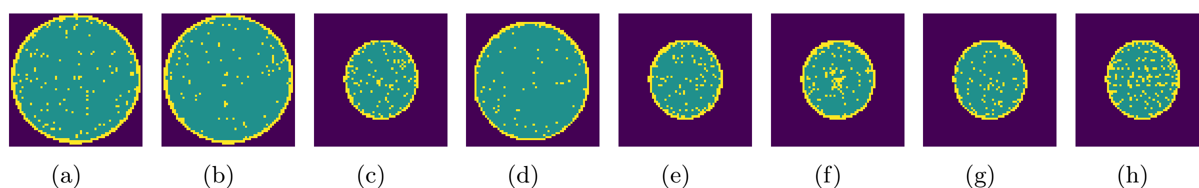


Figure 4.4: Edge-Ring WBM patterns (yellow dots indicate defective dies).

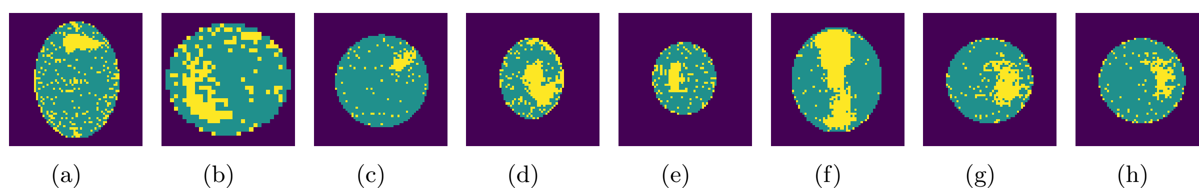


Figure 4.5: Loc WBM patterns (yellow dots indicate defective dies).

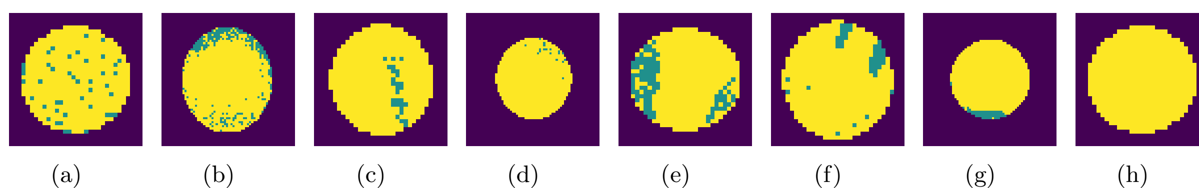


Figure 4.6: Near-Full WBM patterns (yellow dots indicate defective dies).

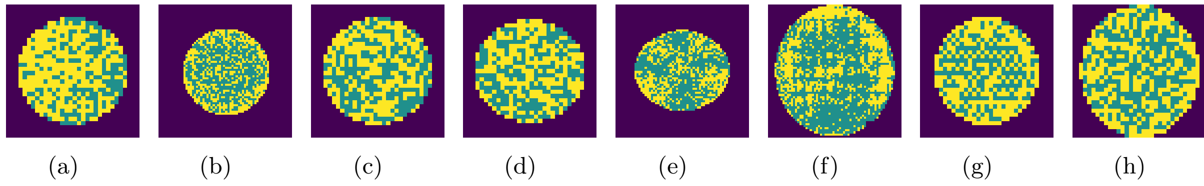


Figure 4.7: Random WBM patterns (yellow dots indicate defective dies).

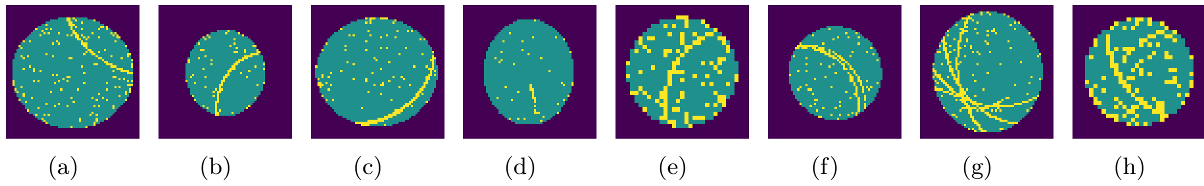
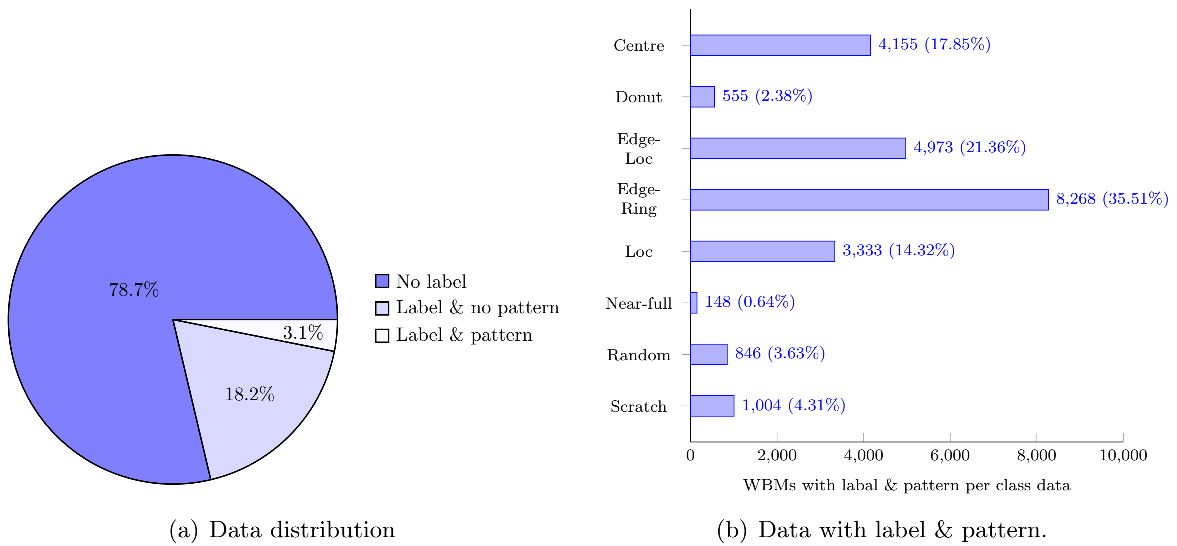


Figure 4.8: Scratch WBM patterns (yellow dots indicate defective dies).



(a) Data distribution

(b) Data with label & pattern.

Figure 4.9: WM-811k data availability

class has 555 samples. Furthermore, the number of images of near-full is 148 samples, which is not enough number for training a DL network. Data imbalance problem plays negative role in the overall classification performance as it will be biased towards the majority class of the data as previously explained in Section 3.4.4. Therefore, DCGAN was used to upsample the data such that each class will contain 10,000 WBM images as will be explained in details in Section 4.5.

4.4 Data Preprocessing

The original data are of different image sizes. Therefore, the size of each image pattern for the purpose of our research was unified to 64×64. Since the WBMs that are dealt with have only three intensity values; background, logic 1 (represent defective dies) and logic 0 (represent normal dies), there is no need to deal with RGB images and therefore it will be sufficient to process the images in grayscale format. These preprocessed images are first fed for DCGAN to apply data augmentation and upsampling on them. The generated augmented images and the original ones are then fed to a classifier in order to detect the defect pattern.

4.5 Data Augmentation & Upsampling

As can be seen from Figure 4.9(b), the WM-811K dataset with pattern and label is considered highly imbalanced, for instance, the Edge-Ring class has 8,268 sample images, whereas the number of samples in the Near-full class is 148 images only. Accordingly, some of the classes do not have sufficient dataset for the training process and will make the accuracy biased towards the dominant class. Therefore, upsampling is necessary to optimise the classification results and to ensure that the total accuracy is not biased to any class more than the other. For this purpose, DCGAN were used for upsampling the dataset and for increasing the training data for better accuracy.

In the proposed DCGAN, the generator network receives a one-dimensional random Gaussian vector of size 100 as an input. Multiple deconvolutional layers are then applied to upscale the vector into 64×64 random noise image. All the deconvolutional layers are followed by ReLU activation function and batch normalization was used in these layers in order to stabilise the learning process. The final layer will have the same size as the target image (64×64), which

will be a noise image in the first epoch of training that will evolve in each epoch to produce the wanted synthetic image. The layers of the generator network are explained in Table 4.1.

Table 4.1: Generator network

Layer	Type	Input size	Kernel Size/ Stride	Activation	Dropout	Output size
1	FC	[100,]	3/1	-	No	[16,16,128]
2	Deconv	[16,16,128]	3/2	ReLU	Yes	[32,32,64]
3	Deconv	[32,32,64]	3/1	ReLU	Yes	[32,32,32]
4	Deconv	[32,32,32]	3/2	ReLU	No	[64,64,1]

The disciminator in the proposed DCGAN is a CNN that does binary classification; it receives the training WBMs and label them real (class 1), while the output generated from the generator is labeled as fake (class 0). Each convolutional block in the discriminator is followed by LeakyReLU activation function with $\alpha = 0.2$. Dropout and batch normalization has been used in the first three convolutional layers to stabilise the learning process. The output layer in the discriminator uses a Sigmoid function for the classification process (Real or Fake). The layers of the discriminator network are explained in Table 4.2. BCE loss function was used in the proposed architecture for the discriminator to penalise itself for misclassifying real and fake images. The proposed DCGAN architecture can be visualised in Figure 4.10.

Table 4.2: Discriminator network

Layer	Type	Input size	Kernel Size/ Stride	Activation	Dropout	Output size
1	Conv	[64,64,1]	3/2	L.ReLU	Dropout	[32,32,32]
2	Conv	[32,32,32]	3/2	L.ReLU	Batch N.	[16,16,64]
3	Conv	[16,16,64]	3/2	L.ReLU	Batch N.	[8,8,128]
4	FC	[8,8,128]	-	-	-	[8192,]
5	FC	[8192,]	-	Sigmoid	-	[1,]

A post-processing step using Pearson correlation coefficients (PCC) is employed to determine whether to accept the DCGAN generated images or dismiss them. Pearson's method is widely used in statistical analysis, pattern recognition and applications that require matching

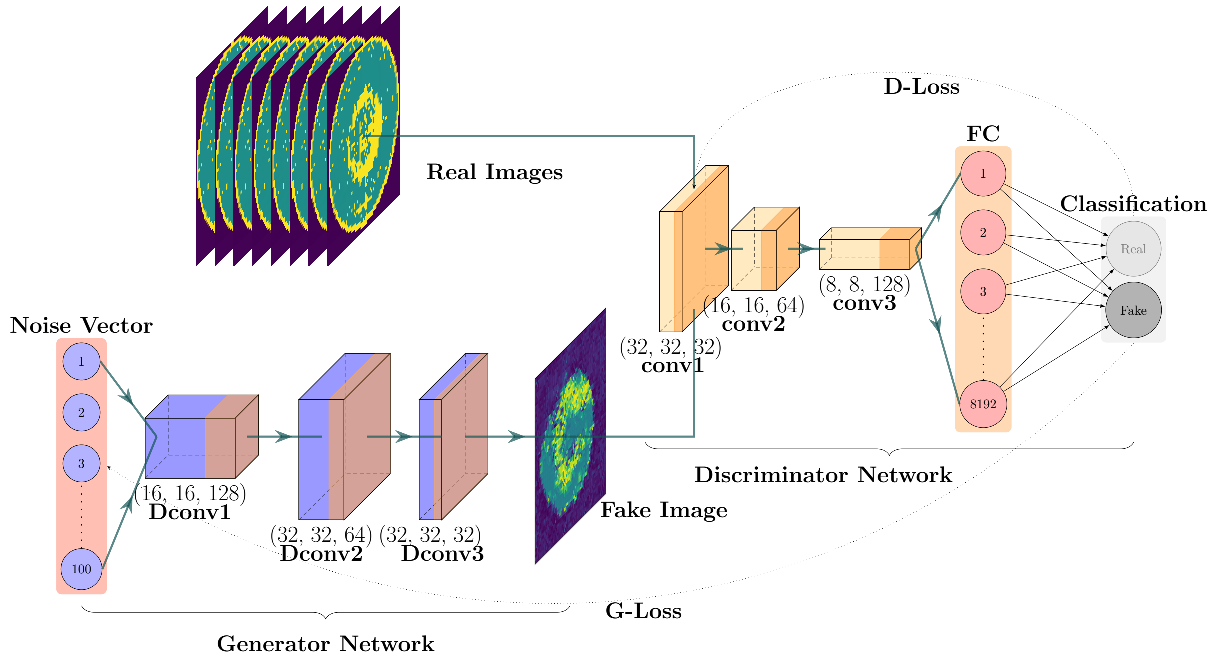


Figure 4.10: Architecture of the proposed DCGAN for generating synthetic WBMs.

and comparing two images as described in equation 4.1 [406].

$$r_1 = \frac{\sum_i (x_i - x_m) \cdot (y_i - y_m)}{\sqrt{\sum_i (x_i - x_m)^2} \cdot \sqrt{\sum_i (y_i - y_m)^2}} \tag{4.1}$$

where x_i is the intensity of the ith pixel in image A, y_i is the intensity of the ith pixel in image B. x_m is the mean intensity of image A, and y_m is the mean intensity of image B. The absolute values of PCC are between 0 and 1. They show how much two images are similar. The closer the coefficient to 1, the more the two images are similar to each other. Based on piratical experiment, a threshold of 0.92 were used for comparison, where the output of the DCGAN networks is compared with all the database images of the class using Pearson's method. If the similarity of the DCGAN generated image is higher than this threshold, the image will be considered; otherwise, it will be ignored. Experiments show that considering this threshold, almost 50% of DCGANs output are accepted and added to the database. The accepted dataset are then used as training set for the proposed WaferCaps to perform classification. Using this method, the required synthetic images could be generated for training the proposed WaferCaps.

4.6 Classification using WaferCaps

CapsNet provides its ability in classifying MNIST images more accurately when compared to CNN and MLP as explained Section 3.8. Despite the efficiency of the original CapsNet in dealing with MNIST handwritten digits classification [401], the architecture has some drawbacks that must be tackled to fit the WM-811K dataset. The original architecture employs two convolutional layers to extract image features, which is not proper for complex images. Furthermore, the size of the convolutional kernels in the original CapsNet is 9×9 that compatible with the MNIST dataset (28×28). For datasets with big images size, this kernel produces a large number of training parameters whic makes the network to overfit on the training data. In this research, a new Wafer Capsule Network (WaferCaps) was proposed to overcome the limitations of the original CapsNet. To intensify the capability of convolutional layers to extract image features, two more convolutional layers were added and established a dropout layer to avoid overfitting after each layer. The input size of the network has also been modified to 64×64 to match our WBMs size. The architecture of the proposed WaferCaps is shown in Figure 4.11. Also, Table 4.3 provides the layers details used in the architecture. As Figure 4.11 shows, conv1, conv2 and conv3 have 256, 512 and 1024 depth layers, respectively. All convolutional layers have 15×15 convolution kernels with a stride of 1 and ReLU activation. These layers transform pixel intensities to local feature detectors' activities fed as inputs to the primary capsules. The PrimaryCaps layer is a convolutional capsule layer with 128 channels of convolutional capsules. Each primary capsule comprises eight convolutional units with a 9×9 kernel and a stride of 2. A justification of using these parameters for the proposed WaferCaps is further explained in the ablation study of Section 4.7.3.

4.7 Results

In this section, different sets of experiments were implemented to study and evaluate the performance of the proposed WaferCaps & DCGAN method. This study adopts accuracy, recall, precision and F1-score metrics to evaluate the performances of these methods as has been done in Chapter 3.

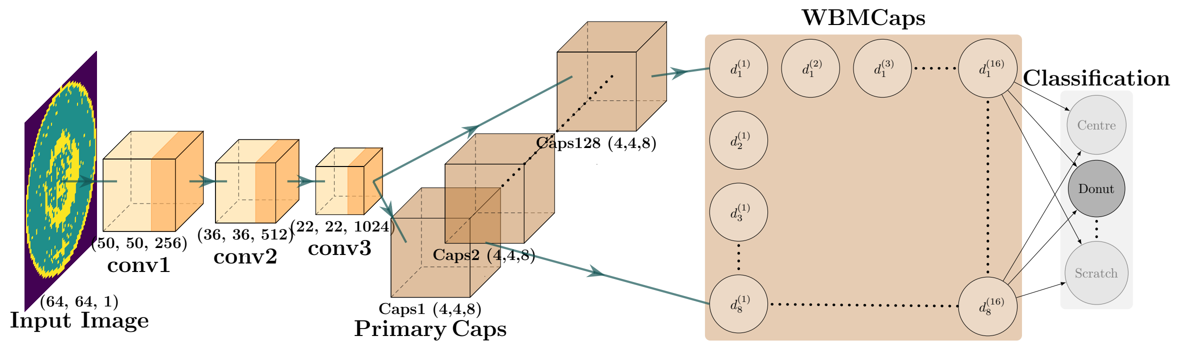


Figure 4.11: Our proposed WaferCaps architecture.

Table 4.3: Layers of the prposed WaferCaps

Layer	**Type**	**Input size**	**Kernel Size/ Stride**	**Activation**	**Dropout**	**Output size**
1	conv1	[64,64,1]	15/1	ReLU	Yes	[50,50,256]
2	conv2	[50,50,256]	15/1	ReLU	Yes	[36,36,512]
3	conv3	[36,36,512]	15/1	ReLU	Yes	[22,22,1024]
4	PrimCaps	[22,22,1024]	9/2	ReLU	No	[4,4,8,128]
5	WBMCaps	[4,4,8,128]	-	Squash	No	[16,8]
6	FC	[16,8]	-	Softmax	No	[8,]

4.7.1 DCGAN

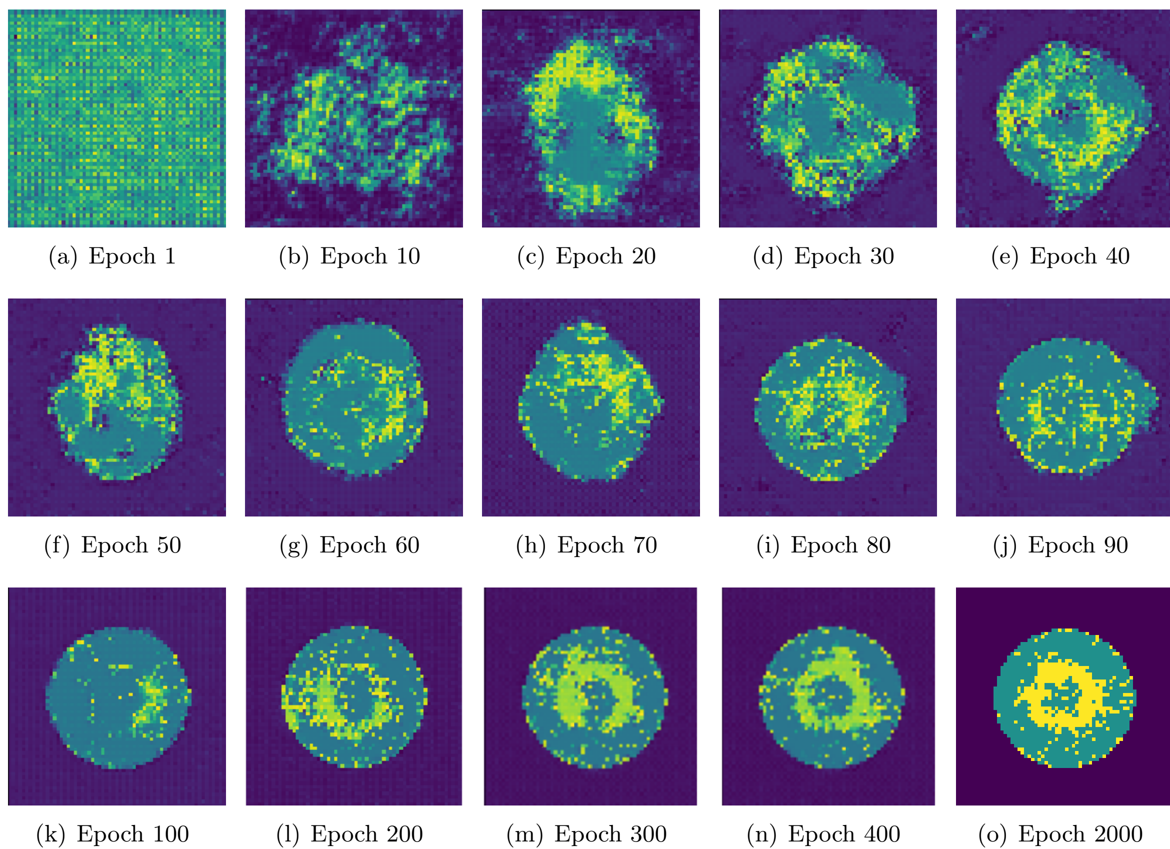
The training of DCGAN is performed over two steps. First step is training the discriminator alone on fake and real data, such that the discriminator can classify them efficiently. In this step, the generator is standby and is not being trained yet. The discriminator loss penalises the discriminator for misclassifying a real sample as fake or a fake sample as real while updating the weights via backpropagation. The second step involves training the generator to create synthetic images such that the generator's loss penalises the generator for producing a sample that the discriminator network classifies as fake. In this study, the training data of each class of the eight was fed to DCGAN to generate synthetic data.

All the classes were upsampled such that each class will contain 10,000 samples (including original samples). The training for each class procedure involved 2,000 epochs. An acceptable results started to appear after epoch 100 approximately and an almost identical results after epoch 1,000. Figure 4.12 hows how the results are improved as the number of epochs is increasing for the donut class.

Using this method, we were able to generate the required synthetic images for training the proposed WaferCaps. However, our DCGAN could not generate synthetic WBMs for the scratch defect pattern. Therefore, different approaches were used for upsampling the WBMs of this class. Our approach proposed isolating the scratch pattern from each WBM and apply rotation operations of the pattern on the WBM without the pattern. Figure 4.14 demonstrates applying this approach on one of the WBM samples that contain scratch defect pattern.

4.7.2 Experimental Data

The dimension of each WBM in the sample data is 64×64, and consists of eight labels: center, donut, edge-loc, edge-ring, loc, near-full, random, and scratch (shown in Figures 4.1 - 4.8). The WBM data were divided into two sets in order to verify and compare our methods; the first set is the original labelled and patterned data in WM-811K dataset, this dataset is called *original dataset*. The total WBMs used for training are 17,804, for validation 4,333 and for testing 2,165. The second set is a combination of the original data and the DCGAN generated WBMs, this dataset is called *mixed dataset*. In both groups it was assured that the WBMs used for testing are the same and are all of the original dataset. The total WBMs used for training are 63,200 and for



(a) Epoch 1 (b) Epoch 10 (c) Epoch 20 (d) Epoch 30 (e) Epoch 40

(f) Epoch 50 (g) Epoch 60 (h) Epoch 70 (i) Epoch 80 (j) Epoch 90

(k) Epoch 100 (l) Epoch 200 (m) Epoch 300 (n) Epoch 400 (o) Epoch 2000

Figure 4.12: Generated synthetic WBMs for DCGAN over multiple epochs.

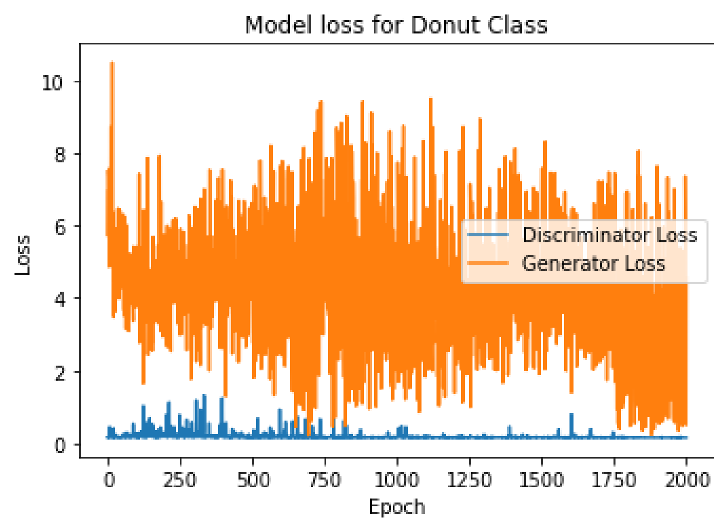


Figure 4.13: Model loss for donut class.

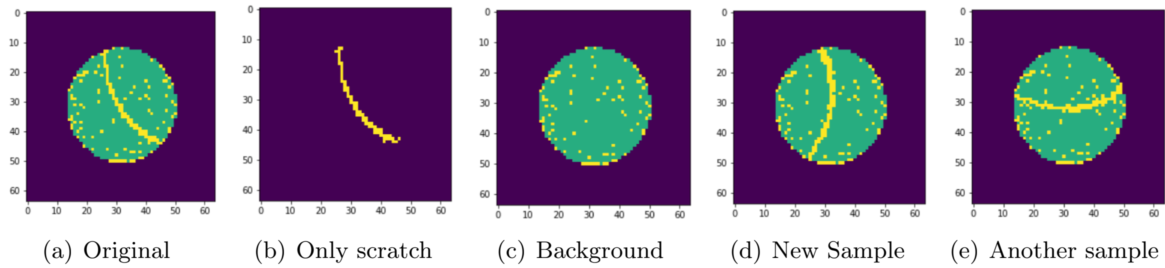

(a) Original (b) Only scratch (c) Background (d) New Sample (e) Another sample

Figure 4.14: Image upsampling procedure for Scratch class.

validation are 15,600. Using these groups will allow us to explore the influence of using DCGAN generated data on the testing accuracy. Figure 4.15 shows the data used for the classifiers in this study to validate the performance of DCGAN.

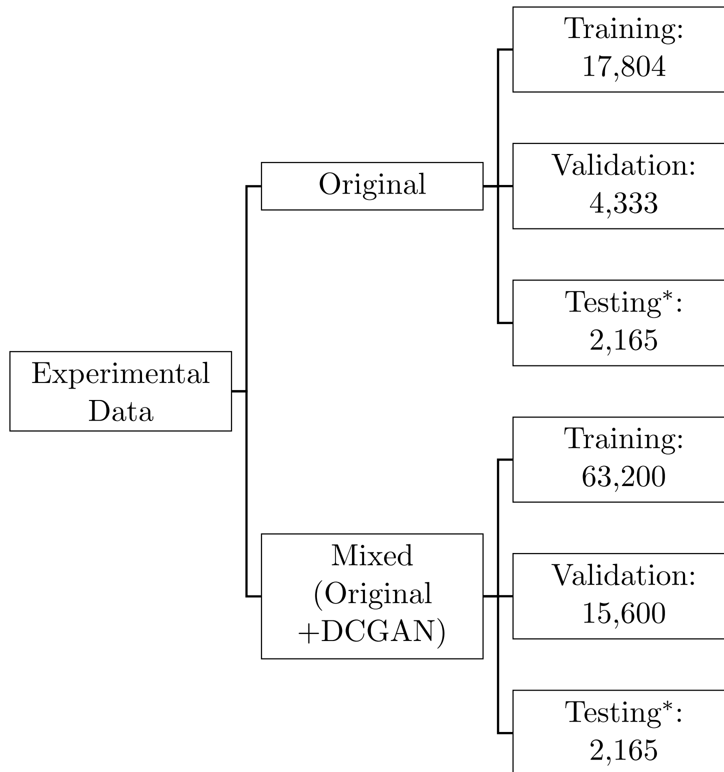


Figure 4.15: Experimental data used.

4.7.3 Ablation Study: Parameter Influence on CapsNet

Multiple parameters in DL models can present a significance difference in the model's performance. All the possible parameters of the original CapsNet were varied in order to observe the effect of that and come up with the proposed WaferCaps. Some of these parameters had

minor noticeable effect on our model while varying others such as image size, dropout, number of convolutional layers, and kernel size demonstrated significant improvement on the model's overall accuracy. In order to justify the selection of the optimised WaferCaps for this specific application, an ablation study was made by varying these parameters while fixing the others. Ablation study on ANNs in general is widely used technique when a new algorithm is proposed to enhance an original reference algorithm as in our case [407]. Therefore, in this study, a step by step ablation is performed to transform an optimised version of CapsNet until WaferCaps is reached as will be seen in the next sections using the test data for evaluation.

4.7.3.1 The Influence of Image Size

The ablation study will start with the input image size of CapsNet. Similar to CNNs, capsule neural networks receive one fixed size of an image for all the samples supplied for training. The larger the image input size, the less shrinking required and therefore the less deformation of features and patterns inside the image [408, 409]. In this section, the originally proposed CapsNet by Sabour et al. [401] for WBM of size 28×28 (similar to the size of MNIST handwritten digits data they used) will be investigated and compare the performance of this size with WBM of size 64×64.

The comparison shows that by considering a size of 28×28, a test accuracy of 80.6% was achieved, while using an image size of 64×64 improved the test accuracy to be 82.9%. Figures 4.16(a) and 4.17(a) represent the confusion matrices for test data upon using image sizes of 28×28 and 64×64 respectively on original proposed CapsNet [401].

4.7.3.2 The Influence of Dropout

In this section, the best CapsNet from Section 4.7.3.1 (image size 64×64) will be used and investigate the effect of applying dropout on the test accuracy. As discussed previously in Section 3.4.3, dropout has a regularisation effect when applied on neural networks which in return can reduce the overfitting. Hence, the dropout will be applied on the convolutional layer before the PrimaryCaps with a percentage of 50%. Figure 4.17(b) show the confusion matrix for the test data with applying dropout. It can be observed from Figure 4.17(b) that the test accuracy achieved was 84.5% when dropout was used, which is better by 1.6% compared with no dropout.

Actual Class \ Predicted Class	Centre	Donut	E-Loc	E-Ring	Loc	Near-full	Random	Scratch
Centre	118 79%	0 0%	11 7%	0 0%	17 11%	1 1%	3 2%	0 0%
Donut	0 0%	149 99%	0 0%	0 0%	0 0%	0 0%	1 1%	0 0%
E-Loc	0 0%	0 0%	121 81%	0 0%	28 19%	0 0%	1 1%	0 0%
E-Ring	0 0%	0 0%	0 0%	150 100%	0 0%	0 0%	0 0%	0 0%
Loc	13 9%	0 0%	44 29%	1 1%	91 61%	0 0%	1 1%	0 0%
Near-full	3 2%	0 0%	23 15%	1 1%	2 1%	104 69%	15 10%	2 1%
Random	4 3%	0 0%	40 27%	0 0%	18 12%	0 0%	84 56%	4 3%
Scratch	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	150 100%

(a) Mixed Data CapsNet

Actual Class \ Predicted Class	Centre	Donut	E-Loc	E-Ring	Loc	Near-full	Random	Scratch
Centre	131 87%	0 0%	3 2%	0 0%	9 6%	1 1%	3 2%	3 2%
Donut	0 0%	141 94%	1 1%	0 0%	7 5%	0 0%	1 1%	0 0%
E-Loc	0 0%	0 0%	132 88%	1 1%	15 10%	0 0%	0 0%	2 1%
E-Ring	0 0%	0 0%	1 1%	149 99%	0 0%	0 0%	0 0%	0 0%
Loc	10 7%	0 0%	29 19%	0 0%	103 69%	0 0%	0 0%	8 5%
Near-full	0 0%	1 2%	1 2%	0 0%	0 0%	29 45%	34 52%	0 0%
Random	3 2%	2 1%	10 7%	1 1%	5 3%	3 2%	125 83%	1 1%
Scratch	5 3%	0 0%	56 37%	1 1%	83 55%	0 0%	0 0%	5 3%

(b) Original Data CapsNet

Figure 4.16: Confusion matrix for test data according to original CapsNet (image size of 28×28).

4.7.3.3 The Influence of Number of Layers

Having successive convolutional networks in DL model will contribute to the learning process of the model in identifying complex and important features. Therefore, the number of layers will highly affect the outcomes of classification [410]. The original CapNet has only one convolutional layer. In this section, the best CapsNet from Section 4.7.3.2 (with dropout) will be modified and alternate the number of convolutional layers to be two and three layers. It was noticed from the experiments that the test accuracy for two and three convolutional layers were 85.8% and 88.4% respectively, which means that increasing contributed to better feature extraction and generalisation of the network. Figures 4.17(c) and 4.17(d) demonstrate the confusion matrices after applying these updates. Therefore, it was concluded that using three convolutional layer resulted in better test accuracy.

4.7.3.4 The Influence of Kernel Size

Due to the working principle of convolutional layers, all the kernels are sliding on the image. Depending on kernel size, each successive layer will have different feature map size and depth according to equation 3.14. Large kernel size will result in deeper convolutional layers and vice-versa [411]. The original kernel size proposed by Sabour et al. [401] is 9×9. In this section, the best CapsNet from Section 4.7.3.3 will be used and alternate the kernel size into other suitable

Actual Class \ Predicted Class	Centre	Donut	E-Loc	E-Ring	Loc	Near-full	Random	Scratch
Centre	124 83%	1 1%	7 5%	0 0%	14 9%	1 1%	3 2%	0 0%
Donut	0 0%	145 97%	1 1%	0 0%	3 2%	0 0%	1 1%	0 0%
E-Loc	2 1%	0 0%	107 71%	2 1%	39 26%	0 0%	0 0%	0 0%
E-Ring	0 0%	0 0%	0 0%	150 100%	0 0%	0 0%	0 0%	0 0%
Loc	14 9%	0 0%	41 27%	0 0%	95 62%	0 0%	4 3%	0 0%
Near-full	3 2%	0 0%	20 13%	1 1%	1 1%	113 75%	8 5%	4 3%
Random	5 3%	1 1%	25 17%	0 0%	19 13%	0 0%	89 59%	11 7%
Scratch	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	150 100%

(a) Original CapsNet with image size 64 × 64

Actual Class \ Predicted Class	Centre	Donut	E-Loc	E-Ring	Loc	Near-full	Random	Scratch
Centre	119 79%	2 1%	7 5%	0 0%	19 13%	0 0%	3 2%	0 0%
Donut	0 0%	148 99%	0 0%	0 0%	1 1%	0 0%	1 1%	0 0%
E-Loc	0 0%	0 0%	123 82%	0 0%	26 17%	0 0%	1 1%	0 0%
E-Ring	0 0%	0 0%	0 0%	150 100%	0 0%	0 0%	0 0%	0 0%
Loc	11 7%	0 0%	28 19%	0 0%	107 71%	0 0%	4 3%	0 0%
Near-full	3 2%	0 0%	20 13%	1 1%	0 0%	107 71%	13 9%	6 4%
Random	4 3%	0 0%	13 9%	0 0%	19 13%	0 0%	110 73%	4 3%
Scratch	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	150 100%

(b) Effect of using dropout

Actual Class \ Predicted Class	Centre	Donut	E-Loc	E-Ring	Loc	Near-full	Random	Scratch
Centre	131 87%	0 0%	6 4%	0 0%	9 6%	1 1%	3 2%	0 0%
Donut	0 0%	145 97%	0 0%	0 0%	4 3%	0 0%	1 1%	0 0%
E-Loc	2 1%	0 0%	128 85%	0 0%	20 13%	0 0%	0 0%	0 0%
E-Ring	0 0%	0 0%	0 0%	150 100%	0 0%	0 0%	0 0%	0 0%
Loc	19 13%	0 0%	42 28%	0 0%	88 59%	0 0%	1 1%	0 0%
Near-full	2 1%	0 0%	20 13%	0 0%	0 0%	103 69%	25 17%	0 0%
Random	3 2%	0 0%	3 2%	0 0%	10 7%	0 0%	134 89%	0 0%
Scratch	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	150 100%

(c) Effect of using two convolutional layers

Actual Class \ Predicted Class	Centre	Donut	E-Loc	E-Ring	Loc	Near-full	Random	Scratch
Centre	129 86%	0 0%	3 2%	1 1%	12 8%	1 1%	4 3%	0 0%
Donut	0 0%	146 97%	0 0%	0 0%	3 2%	0 0%	1 1%	0 0%
E-Loc	0 0%	0 0%	125 83%	0 0%	23 15%	0 0%	2 1%	0 0%
E-Ring	0 0%	0 0%	0 0%	150 100%	0 0%	0 0%	0 0%	0 0%
Loc	14 9%	0 0%	24 16%	1 1%	109 73%	0 0%	2 1%	0 0%
Near-full	5 3%	2 1%	10 7%	0 0%	0 0%	112 75%	21 14%	0 0%
Random	3 2%	0 0%	1 1%	0 0%	6 4%	0 0%	140 93%	0 0%
Scratch	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	150 100%

(d) Effect of using three convolutional layers

Actual Class \ Predicted Class	Centre	Donut	E-Loc	E-Ring	Loc	Near-full	Random	Scratch
Centre	133 89%	0 0%	5 3%	0 0%	8 5%	1 1%	3 2%	0 0%
Donut	0 0%	148 99%	0 0%	0 0%	1 1%	0 0%	1 1%	0 0%
E-Loc	1 1%	0 0%	115 77%	0 0%	33 22%	0 0%	1 1%	0 0%
E-Ring	0 0%	0 0%	1 1%	149 99%	0 0%	0 0%	0 0%	0 0%
Loc	19 13%	0 0%	32 21%	0 0%	99 66%	0 0%	0 0%	0 0%
Near-full	2 1%	0 0%	13 9%	0 0%	0 0%	127 85%	8 5%	0 0%
Random	3 2%	0 0%	14 9%	0 0%	10 7%	0 0%	123 82%	0 0%
Scratch	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	150 100%

(e) Effect of using a kernel size of 7 × 7

Figure 4.17: Confusion matrices for the test data with different parameter scenarios.

sizes such as 7×7 and 15×15. Our experiments showed that using a kernel size of 7×7 resulted in a test accuracy of 87% which is less than using a kernel size of 9×9 as shown in previous section. However, using a kernel size of 15×15 improved the test accuracy to 91.4%. Figures 4.17(e) and 4.18(a) demonstrate confusion matrices of the test data after changing the kernel size to 7×7 and 15×15 respectively.

Kernel size is the last parameter was varied in this study, and by applying it the proposed WaferCaps that is described in Figure 4.11 and Table 4.3 is reached.

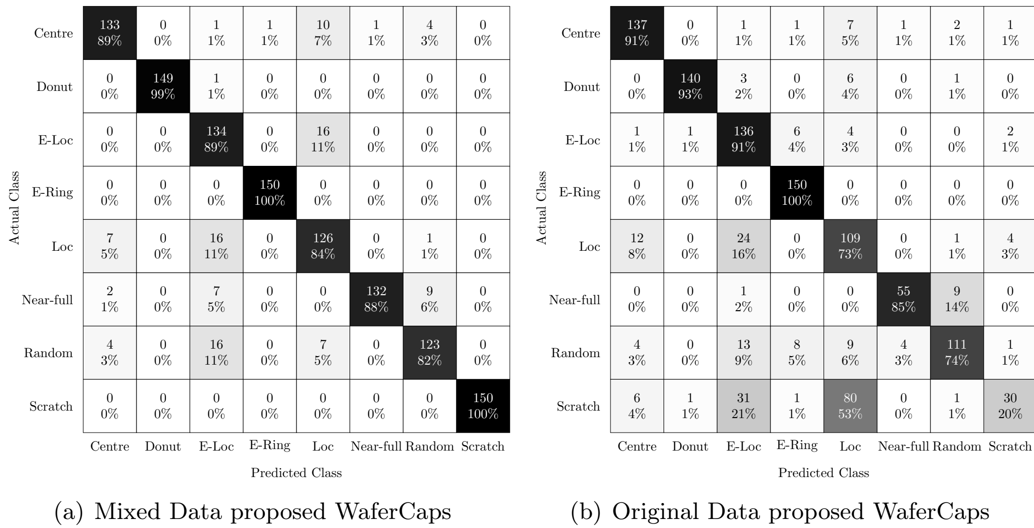
4.7.3.5 The Influence of Data Size

In this section, the influence of the training dataset size on our proposed WaferCaps will be discussed. The original and mixed datasets for this comparison were considered to investigate the effect of using the synthetic WBMs generated by DCGAN on the test accuracy of our proposed WaferCaps. By using original dataset only for training, the test accuracy drops dramatically to 78.2% from 91.4% with using mixed dataset. Figure 4.18 show the confusion matrices of test data for using mixed and original dataset. This result proves the necessity of using DCGAN for data upsampling to get better test results. Tables 4.4 and 4.5 also effect of changing the training dataset on the training and validation accuracies.

4.7.4 Comparison with Other Deep Learning Models

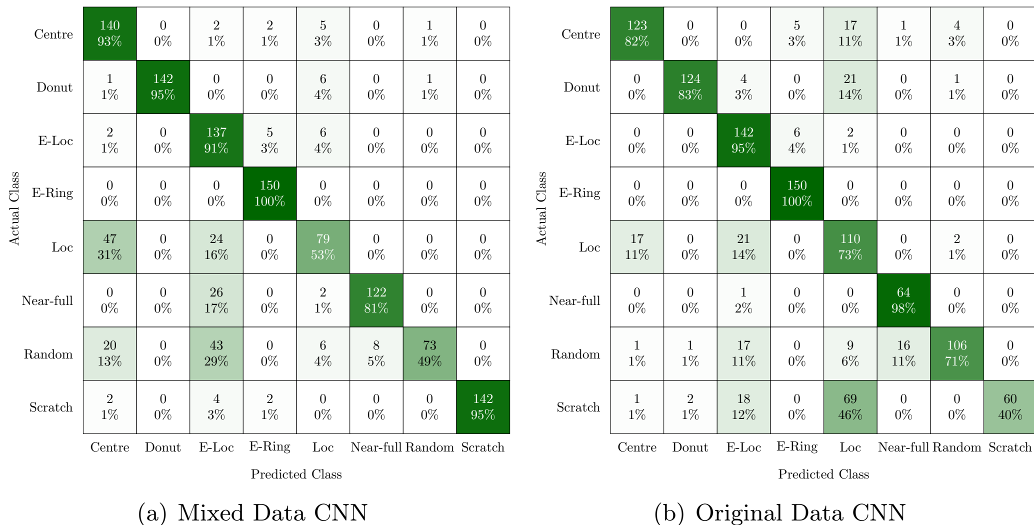
A series of experiments are conducted on both original and mixed dataset to compare our proposed WaferCaps method with several DL models, such as the original CapsNet [401], which we had to resize the WBM to a size of 28×28 to match the architecture of the CapsNet used. The CNN described in Figure 4.21, and the MLP described in Figure 4.22 were also used for comparison. In CNN, the WBM images used for training are fed to the network in the same way as in CapNet and WaferCaps. However, in MLP, the training images were flattened, such that the input image of size 64×64 is transformed into a vector of size $4,096 \times 1$ to be used in the input layer as shown in Figure 4.22. In this section, it was also demonstrated that all DL models perform better when they are trained with the mixed dataset rather than training them using original dataset only (see Tables 4.4 and 4.5). Figures 4.16, 4.18, 4.19 and 4.20 represent the confusion matrices of the test data for applying CapsNet [401], proposed WaferCaps, CNN

and MLP respectively.



(a) Mixed Data proposed WaferCaps (b) Original Data proposed WaferCaps

Figure 4.18: Confusion matrices for the test data with proposed WaferCaps.



(a) Mixed Data CNN (b) Original Data CNN

Figure 4.19: Confusion matrices for the test data with CNN.

4.7.5 Results Discussion

The efficiency of using DCGAN for all the defect pattern was discussed in Section 4.7.1. However, the DCGAN did not perform well when attempting to generate scratch defect patterns. In order to mitigate this issue the method described in Figure 4.14 were used. Despite of the efficiency of this method, the number of samples achieved using this method was 10,000 approximately

Actual Class \ Predicted Class	Centre	Donut	E-Loc	E-Ring	Loc	Near-full	Random	Scratch
Centre	111 74%	0 0%	20 13%	1 1%	14 9%	0 0%	3 2%	1 1%
Donut	0 0%	112 75%	1 1%	0 0%	36 24%	0 0%	0 0%	1 1%
E-Loc	1 1%	0 0%	127 85%	3 2%	12 8%	0 0%	2 1%	5 3%
E-Ring	0 0%	0 0%	1 1%	149 99%	0 0%	0 0%	0 0%	0 0%
Loc	7 5%	0 0%	50 33%	0 0%	55 37%	0 0%	5 3%	33 22%
Near-full	0 0%	0 0%	2 1%	0 0%	1 1%	126 84%	21 14%	0 0%
Random	3 2%	0 0%	4 3%	0 0%	5 3%	0 0%	86 57%	52 35%
Scratch	0 0%	0 0%	2 1%	0 0%	0 0%	0 0%	2 1%	146 97%

(a) Mixed Data MLP

Actual Class \ Predicted Class	Centre	Donut	E-Loc	E-Ring	Loc	Near-full	Random	Scratch
Centre	115 77%	0 0%	9 6%	0 0%	22 15%	1 1%	3 2%	0 0%
Donut	0 0%	113 75%	2 1%	0 0%	34 23%	0 0%	1 1%	0 0%
E-Loc	2 1%	0 0%	115 77%	4 3%	29 19%	0 0%	0 0%	0 0%
E-Ring	0 0%	0 0%	1 1%	149 99%	0 0%	0 0%	0 0%	0 0%
Loc	16 11%	0 0%	22 15%	0 0%	108 72%	0 0%	0 0%	4 3%
Near-full	0 0%	0 0%	14 22%	0 0%	2 3%	23 35%	26 40%	0 0%
Random	4 3%	1 1%	70 47%	0 0%	15 10%	4 3%	56 37%	0 0%
Scratch	5 3%	0 0%	66 44%	0 0%	79 53%	0 0%	0 0%	0 0%

(b) Original Data MLP

Figure 4.20: Confusion matrices for the test data with MLP.

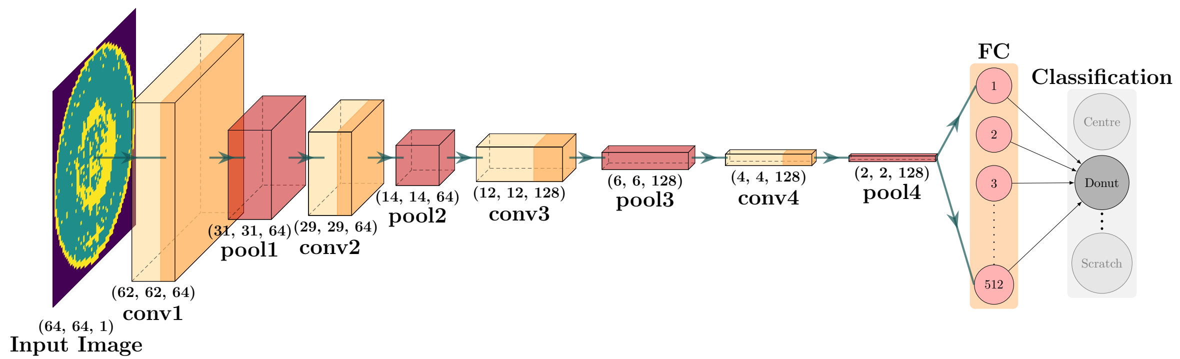


Figure 4.21: CNN architecture used for comparison.

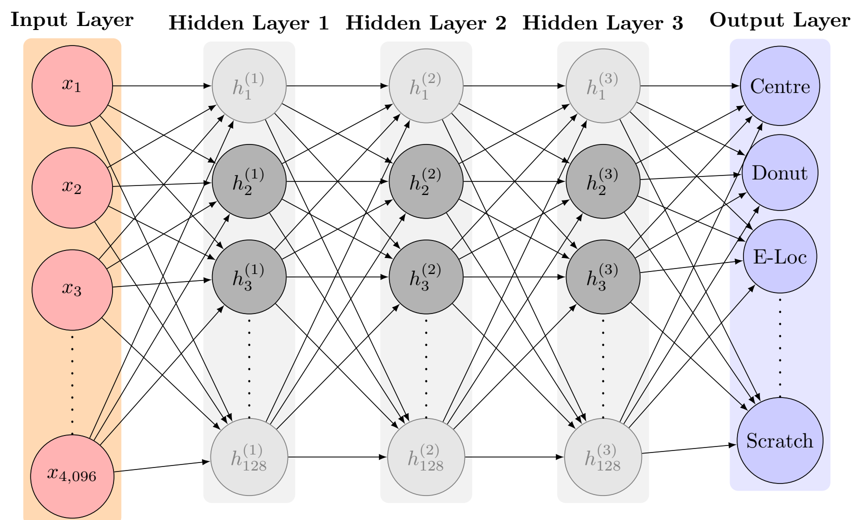


Figure 4.22: MLP architecture used for comparison.

Table 4.4: Overall training, validation and test accuracies using mixed dataset

Model	**Training Accuracy**	**Validation Accuracy**	**Test Accuracy**
WaferCaps (proposed)	**99.59%**	**97.53%**	**91.4%**
CapsNet [401]	99.9%	95.48%	80.6%
CNN	93%	92.7%	82.1%
MLP	96.8%	92.2%	76%

Table 4.5: Overall training, validation and test accuracies using original dataset

Model	**Training Accuracy**	**Validation Accuracy**	**Test Accuracy**
WaferCaps (proposed)	**99.89%**	**88.92%**	**78.2%**
CapsNet [401]	97.84%	81.12%	73.1%
CNN	90%	89.7%	78.8%
MLP	91.72%	75.17%	60.9%

Table 4.6: Metrics for evaluating the test data of mixed (DCGAN & Original) dataset for different deep learning models

Model	**Metric**	**Center**	**Donut**	**E-Loc**	**E-Ring**	**Loc**	**N-full**	**Random**	**Scratch**
WaferCaps (proposed)	**Recall**	**0.887**	**0.993**	**0.893**	**1.0**	**0.84**	**0.88**	**0.82**	**1.0**
	Precision	**0.911**	**1.0**	**0.766**	**0.993**	**0.792**	**0.992**	**0.898**	**1.0**
	F1-score	**0.899**	**0.997**	**0.825**	**0.997**	**0.816**	**0.933**	**0.857**	**1.0**
CapsNet [401]	Recall	0.787	0.993	0.807	1.0	0.607	0.693	0.56	1.0
	Precision	0.855	1.0	0.506	0.987	0.583	0.99	0.8	0.962
	F1-score	0.819	0.997	0.622	0.993	0.595	0.816	0.659	0.98
CNN	Recall	0.933	0.947	0.913	1.0	0.527	0.813	0.487	0.947
	Precision	0.660	1.0	0.581	0.943	0.76	0.94	0.973	1.0
	F1-score	0.773	0.973	0.71	0.971	0.622	0.871	0.65	0.973
MLP	Recall	0.74	0.747	0.847	0.993	0.367	0.84	0.573	0.973
	Precision	0.91	1.0	0.614	0.974	0.447	1.0	0.723	0.613
	F1-score	0.816	0.855	0.711	0.983	0.403	0.913	0.639	0.753

Table 4.7: Metrices for evaluating the test data of original only dataset for different deep learning models

Model	Metric	Center	Donut	E-Loc	E-Ring	Loc	N-full	Random	Scratch
WaferCaps (proposed)	**Recall**	**0.887**	**0.907**	**0.927**	**1.0**	**0.773**	**0.646**	**0.773**	**0.267**
	Precision	**0.875**	**1.0**	**0.698**	**0.993**	**0.483**	**0.875**	**0.841**	**0.784**
	F1-score	**0.881**	**0.951**	**0.797**	**0.997**	**0.595**	**0.743**	**0.806**	**0.398**
CapsNet [401]	Recall	0.873	0.94	0.88	0.993	0.687	0.446	0.833	0.033
	Precision	0.879	0.979	0.567	0.98	0.464	0.879	0.767	0.363
	F1-score	0.876	0.959	0.689	0.987	0.554	0.592	0.799	0.059
CNN	Recall	0.82	0.827	0.947	1.0	0.733	0.985	0.707	0.4
	Precision	0.866	0.976	0.7	0.932	0.482	0.79	0.938	1.0
	F1-score	0.842	0.895	0.805	0.965	0.582	0.877	0.806	0.571
MLP	Recall	0.767	0.753	0.767	0.993	0.72	0.354	0.373	0
	Precision	0.81	0.991	0.385	0.974	0.374	0.821	0.651	0
	F1-score	0.788	0.856	0.512	0.983	0.492	0.495	0.475	0

due to the limitations of the number of scenarios that can be generated. Therefore, this number of samples had to be matched to the other patterns in order to have balanced data for all the eight classes. In an upcoming research, an attempt to optimise DCGAN or using more powerful GAN (e.g. CapsuleGAN [412]) can be investigated in order to generate scratch defect patterns so that unlimited number of data can be produced for all the classes and observe the accuracy of classification according to that.

The results obtained showed that our proposed WaferCaps performed the best in terms of test accuracy when mixed dataset was used for training instead of the original dataset only. The same was also observed when other DL models were used such as CapNet [401], CNN, and MLP.

However, from Table 4.5, it can be observed that CNN has slightly outperformed our proposed WaferCaps when original dataset used only for training. Table 4.4 also shows that CapNet [401] performed better than our proposed WaferCaps in terms of the training accuracy; however, the better result in training accuracy is due to overfitting as both validation and test accuracies scored better when WaferCaps was used. Hence, the Dropout method was used to prevent overfitting.

Noteworthy, in Section 4.7.3 that changing certain parameters such as image size, number of convolutional layers, and kernel size can affect the test accuracy when both datasets used (mixed and original).

From the confusion matrices in Figures 4.16, 4.17,4.19 and 4.20, it can be observed that most

of the defect patterns such as Donut, Edge-Ring and Scratch were very easy to detect when mixed dataset are used for training. While it was noticed that it becomes much harder to detect some classes such as Scratch when original dataset was used in training. This conclusion could lead to future work, where one could optimise a classification algorithm for each class in a way that binary classification be performed separately as one-against-all approach.

4.8 Conclusion

In this chapter, a DCGAN and Capsule Network-based framework (referred to as WaferCaps) were proposed to generate synthetic WBM images and classify them according to eight different defect patterns, namely, Center, Donut, Edge-Loc, Edge-Ring, Loc, Near-full, Scratch and Random. For this purpose, labelled and patterned dataset of WM-811k data were used. DCGAN was utilised first in order to upsample the data such that each class will be increased into 10,000 samples. A different method was used to upsample the Scratch class, in which the defect pattern was isolated and rotated into different angles to increase the number of defect scenarios.

Two main datasets were then created for our analysis namely, original and mixed. The original dataset contained the WBMs available in the WM-811k dataset only, while the mixed dataset contained synthetic and original WBMs together with maintaining the test set the same for both of them which contained original samples. This process was essential in order to observe the effect of using synthetic WBMs generated by DCGAN on the test accuracy.

The mixed dataset were then used on the CapsNet with different parameters and the proposed WaferCaps were obtained according to the best scenario of a specified series of high impact parameters that lead to the best test accuracy. The WaferCaps performance was then compared with the occasion when the original dataset was used for training. Our experiments showed that the proposed WaferCaps achieved a training, validation and test accuracies of 99.59%, 97.53% and 91.41% respectively when mixed dataset was used. While it achieved a training, validation and test accuracies of 99.89%, 88.92% and 78.2% respectively when only original dataset was used.

Also, the performance of the proposed WaferCaps was compared with different DL models such as the original CapsNet proposed by Sabour et al. [401] to classify MNIST handwritten digits, CNN, and MLP. The experiments show that the proposed WaferCaps outperformed all

the other DL models that were compared to when mixed dataset was used.

Chapter 5

Implementation of Deep Learning for Optoelectronic Wafer Inspection [1]

"*Focus like a laser, not a flashlight.*"

Michael Jordan

5.1 Overview

High-performance indium phosphide (InP) quantum cascade lasers (QCLs) play a key role in many optoelectronic applications in the mid to far infrared (IR) regions such as IR imaging and spectroscopy [413]. QCLs are unipolar semiconductor lasers that emit in the range between 4 and 12 μm of wavelength. The first QCL was invented and experimented in 1994 at Bell labs by [414]. Since then and due to the extensive knowledge acquired in the field of epi-material growth, wafer processing, and die packaging, QCLs were developed rapidly [415, 416]. Beside their unique combination of compactness, room temperature operation, high power output and narrow linewidth, QCLs are well-known in their tuning abilities of IR light [417]. A typical QCL wafer (Figure 5.1) consists of multiple laser devices and is made of a stack of InGaAs/AlInAs layers on InP substrate using a series of chemical processes such as molecular beam epitaxy (MBE) or metal-organic chemical vapour deposition (MOCVD) [418]. Like other semiconductor

[1] The contents of this chapter is considered for publication according to the following details: **A. M. Abu Ebayyeh** , A. Mousavi, S. Danishvar, T. Gresch, O. Landry and A. Müller, "A Waveguide Quality Inspection in Quantum Cascade Lasers: A Capsule Neural Network Approach", *Expert Systems with Applications* Status: **Under Review (2nd Round)**

lasers, each QCL in the wafer consists of a laser core, a waveguide (Figure 5.2), and a feedback mechanism [413].

Figure 5.1: QCL wafer.

Due to the complexity of the fabrication process, QCL wafers are prone to multiple flaws. Where some of these malfunctions can only be detected using special tests, a discontinuity in the waveguide of the QCL can be detected visually with the aid of microscope and digital camera. This defect occurs when the hard-mask defining the waveguide is locally damaged and detaches during the etching process of the waveguide. The detachment causes the waveguide to be damaged or interrupted. Another common problem is the settlement of dirt over the waveguide of the QCL. Although dirt does not necessarily cause damage to the wavegiude, but the inspector should be alerted to this case since a damage can happen beneath the dirt on the waveguide. Dirt inspection on the waveguide is visually conducted using microscope and digital camera.

High-power QCL have cavity lengths between 6 and 9 mm and the waveguide needs to be in impeccable condition to ensure reliable operation. Therefore, quality monitoring of the waveguide is essential.

Defects and dirt detection on the waveguide in majority of cases are conducted manually in small and make to order (low volume high variety) plants. In order to increase productivity and production rate to meet demand, AOI techniques can be utilised to automate and increase the

Figure 5.2: Waveguide in QCL.

accuracy of the inspection and quality control task [26, 32]. Furthermore, AOI would reduce the rate of false alarms caused by colour perception induced human error [27]. It is therefore safe to claim that a bespoke robust high quality AOI system could be a desirable tool for detection of anomalies for QCL wafers [32, 419].

5.2 Motivation

To the time of this chapter was written, no research has ever dealt with the inspection of optoelectronic wafer automatically. The inspection process is usually done manually using microscope to inspect each device in the wafer. It also requires expert knowledge in identifying the spots needed for inspection and adjusting the wafer manually for this purpose. According to the experts in optoelectronic wafer industry, an interruption in the waveguide (defect) and the dirt accumulated above it are currently the most occurring anomalies that requires the attention in inspection. This chapter reports on the efforts to find an industrial solution to automatically and accurately detect dirt and defects that occurs in waveguides of QCL wafer. A bespoke dedicated vision system equipped with a novel DL solution that combines CNN and the proposed WaferCaps from Chapter 4 is proposed for this purpose. The solution is tested and validated in an actual and real QCL wafers collected from a laser manufacturing plant in Europe.

This chapter's primary contributions are as follows:

- Providing DL framework to assess the quality of the waveguide in QCLs.
- Generating waveguide samples library by applying manual and automatic augmentation approaches of the minority classes.

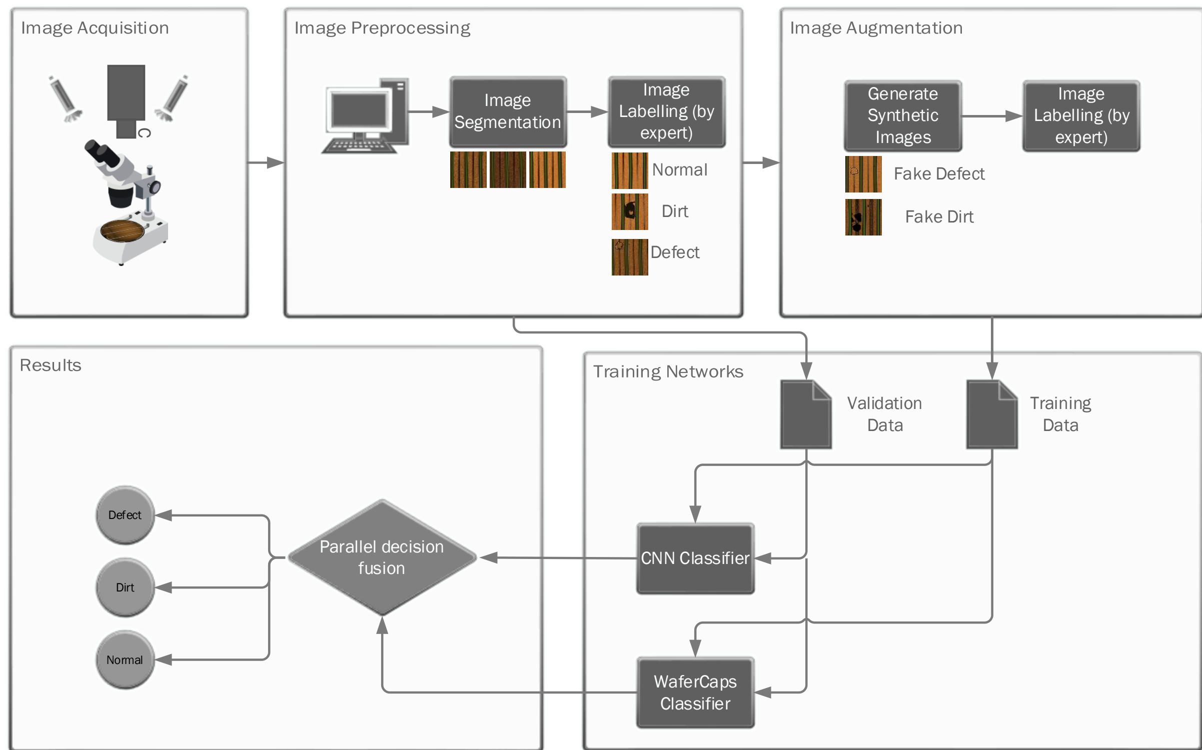


Figure 5.3: Framework

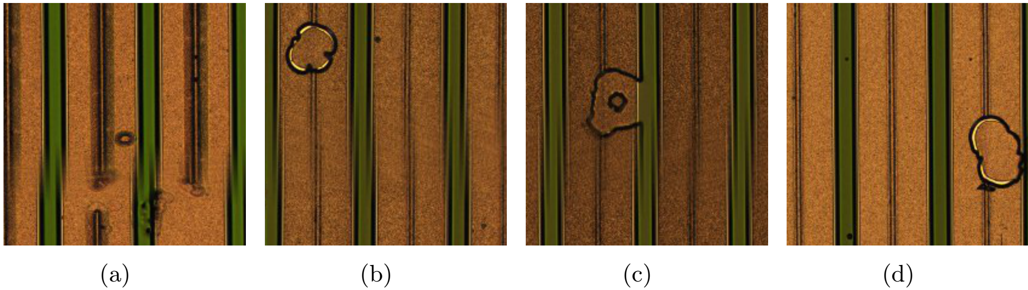
- Integrate the performance of CNN and WaferCaps using parallel decision fusion to deploy the classification of the anomaly and normal samples.
- Compare the performance of the proposed approach with standalone models (without combination), several state-of-the-art DL algorithms such as CapsNet [401], ResNet50 [382], Inception-V3 [390], Xception [391] and MLP.

This work could reduce the burden on human inspectors that usually use microscope to detect waveguide anomalies of QCLs. The framework of the proposed approach can be illustrated in Figure 5.3.

5.3 Investigated Anomalies

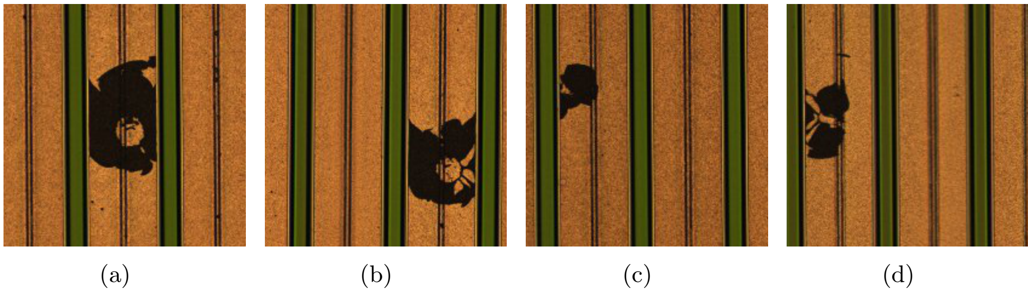
Based on expert recommendation in the field of optoelectronics, two commonly occurring types of anomalies in QCL wafer manufacturing are considered, defected waveguide and dirt. Defected waveguides can be recognised visually by the discontinuity of the black lines that represent the waveguide as in Figure 5.4. Dirt anomaly occurs when the waveguide is covered with external

material that are represented in the form of black clusters (Figure 5.5).



(a) (b) (c) (d)

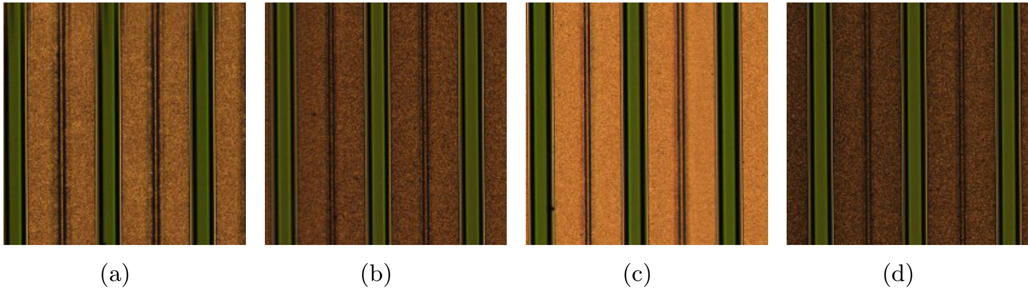
Figure 5.4: Samples for waveguide defects in QCL wafer.



(a) (b) (c) (d)

Figure 5.5: Samples for dirt on waveguide in QCL wafer.

Figure 5.6 shows normal (non-defective) samples of waveguide. The DL algorithm used for detection will classify the sample images according to three classes; Defect, Normal and Dirt.



(a) (b) (c) (d)

Figure 5.6: Normal waveguide samples in QCL wafer.

5.4 Image Acquisition

Images of the opto-electronic wafer are acquired at premises using a S-Neox optical profilometer, by Sensofar, using a microscope and digital camera objective with 5x magnification and white

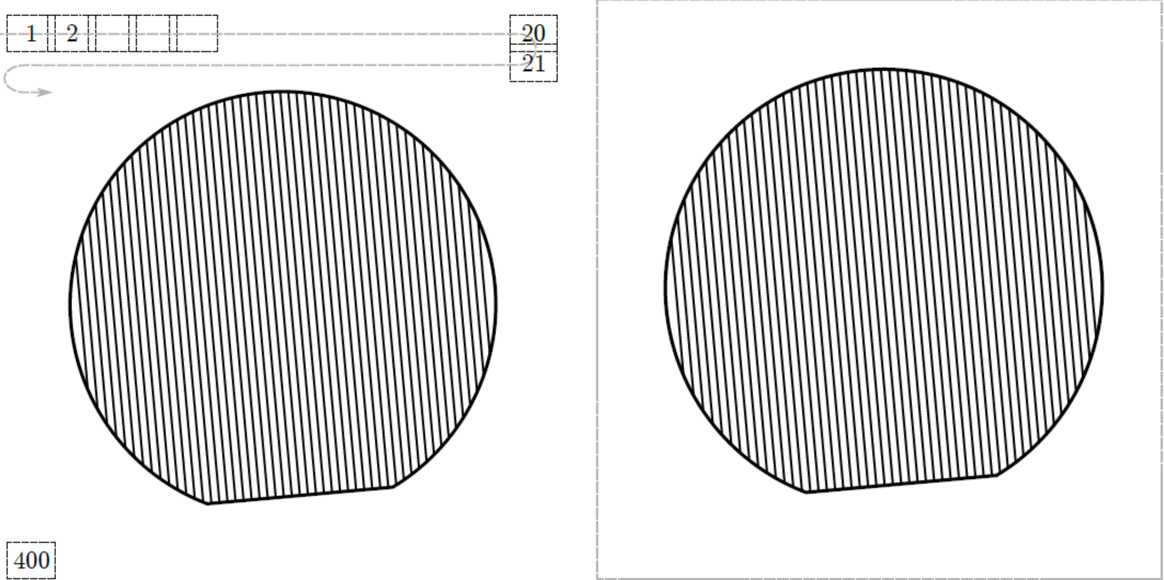
light illumination. The sensofar tool has a x-y table that is controlled by the software, this table is adjusted so that the waveguide looks vertical in the picture corrected for rotation. For coverage of a full 2″ wafer, about 400 individual images are acquired in a raster-scan with an overlap of 10%, while adjusting for the image focus to compensate for wafer bending. The individual images are subsequently stitched together into a single image, normalised to $20,000 \times 20,000$ pixels. Figure 5.7 demonstrate the image acquisition process and Figure 5.1 shows a complete wafer image acquired.

Next step is automatic segmentation process. The images are broken down into multiple segments such that each wafer will result into up to 430 images. In this study three wafer images will be segmented. These segmented images are then labelled according to expert view as defect, dirt or normal. The observed samples during the experiment contained 1,000 of normal, 36 defective and 240 dirt classes.

5.5 Data Augmentation & Upsampling

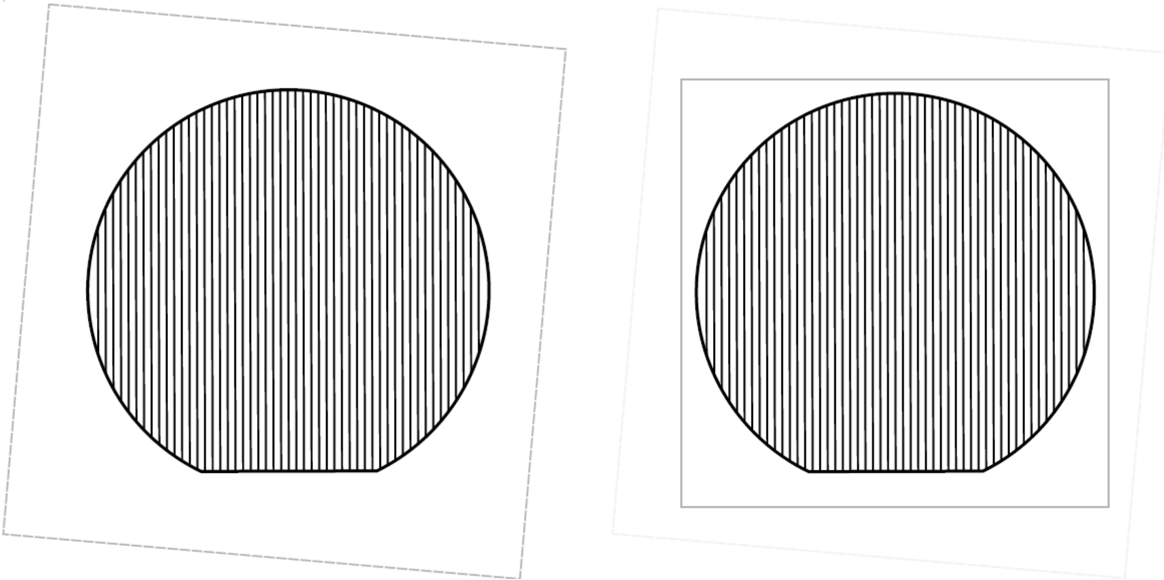
One of the main challenges of the deploying learning models is limitation of available data (analysis span) and lack of sufficient historical image library, therefore risk of overfitting and low validation accuracy. In our case, this challenge is visible in the defect and dirt samples as observed from the previous section. One way to mitigate the situation would be image augmentation. Augmentation involves creating new dataset based on existing data for the purpose of increasing the number of samples. A combination of automatic and manual augmenting was implemented to increase the number of dirt and defect samples to match the normal samples. In the manual augmentation synthetic defect and dirt features were transposed on the images (Figure 5.8). The large library of produced synthetic images were validated and verified by experts on whether the could be realistic occurrences.

The automatic augmentation approach applies different image transformation operations on the original and manually augmented images such as cropping, flipping, scaling, mirroring and scaling to further increase the dataset for training.



(a) Raster scan with sensofar tool (20×20 images)

(b) Stitching with stitching-tool provided by sensofar



(c) Rotation of the image such that waveguides are oriented vertically

(d) Cropping to $20,000 \times 20,000$ pixels

Figure 5.7: Image acquisition approach.

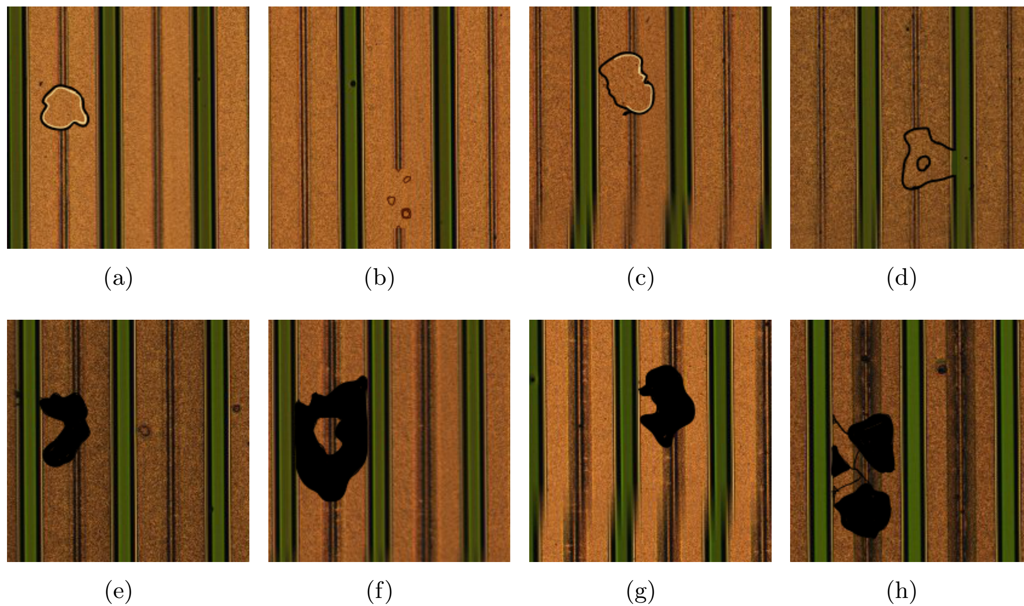


Figure 5.8: Manually augmented samples for defect and dirt anomalies.

5.6 Classification with Parallel Decision Fusion: WaferCaps+CNN

A lot of research indicates that a decision fusion approach improves classification performance significantly [420–424]. Different classification techniques may have varying classification performances for the same problem. Fusion of decisions is a way of integrating the results of multiple classifiers into a common conclusion about an event and has the potential to integrate various decision rules in a fully tunable way. As a result of the decision fusion, classification accuracy can be increased.

There are several types of decision fusion techniques, based on the fusion architecture used [424, 425]:

- Serial decision fusion: One way to implement serial decision fusion is by arranging the classifiers one after another in a series; each classifier's output is used as an input to the next.
- Parallel decision fusion: Two or more classifiers work together in parallel to perform classification simultaneously, and then the classifiers combine the results.
- Hybrid decision fusion: This refers to a hierarchical classification process.

In this section, a parallel decision fusion based approach is employed by combining the per-

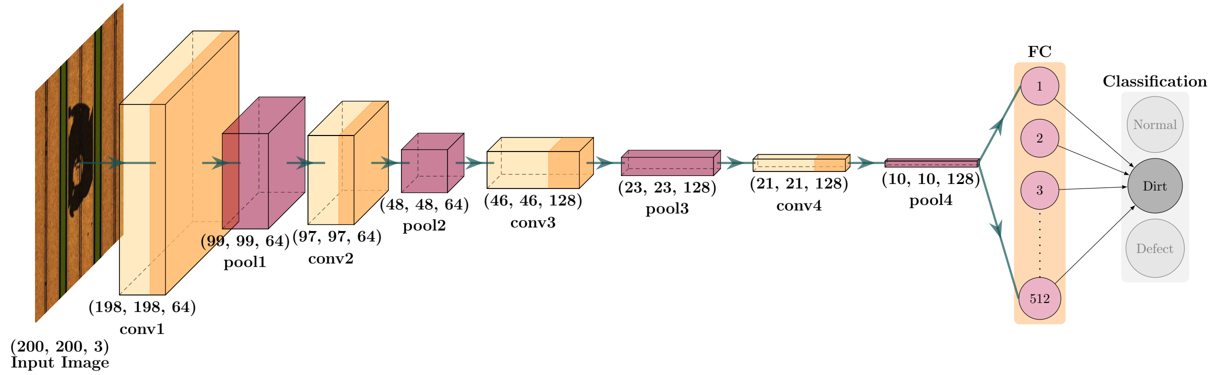


Figure 5.9: CNN architecture.

formance of CNN in Figure 5.9 and WaferCaps in Figure 5.10. As will be observed in Section 5.7 the reason of combining these two classifiers is that WaferCaps performed the best in classifying normal and defect samples, while CNN detected dirt samples more accurately. As a result, the decision fusion approach will increase the final classification accuracy for all the classes. The combined classifier consists of two main layers that are individual classifiers and selection rules. In the first layer, two individual classifiers are settled, which are built using the same training dataset. The second layer represents the selection methods that receive all individual classifiers' outputs and produces the final result. For every input image, each classifier generates the output that represents the probability of each class. The decimal numbers between 0 and 1 can be interpreted as a percentage of confidence.

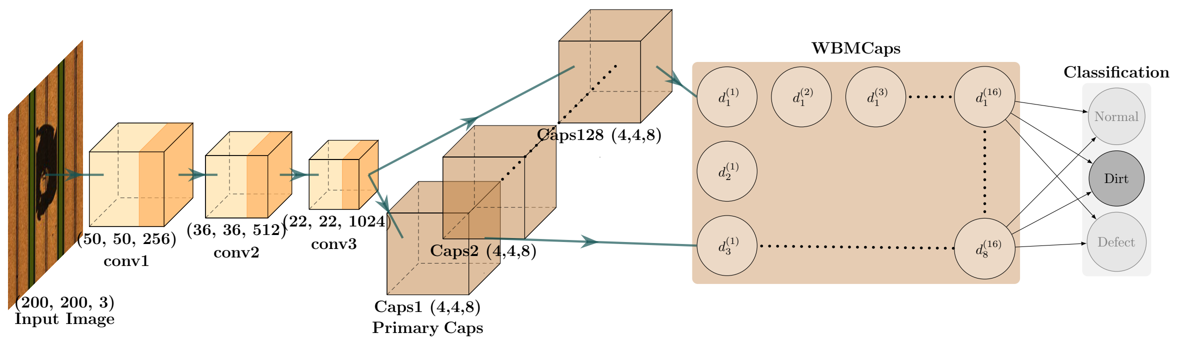


Figure 5.10: WaferCaps proposed

In our classifiers, p_1, p_2 and p_3 are probabilities of Defect, Dirt and Normal classes, respectively and (p_1^{Caps},p_2^{Caps} ,p_3^{Caps}) and (p_1^{CNN},p_2^{CNN} ,p_3^{CNN}) are the outputs of classifiers in the first layers. On one side, the performances of individual classifiers show that WaferCaps classifier

has higher sensitivity for Defect and Normal classes than CNN classifier. Therefore, in the next layer, more decision weight is given for WaferCaps when a decision is made for these two classes. On the other side, CNN provides high sensitivity for the Dirt class, thus CNN's output is more reliable.

In the second layer, the selection rules are described as below and Algorithm 3:

- **Rule 1**: If $p_1^{Caps} > 0.8$ OR $p_1^{CNN} > 0.95$, then the class of the input image is Defect, otherwise $\max(\max(p_1^{Caps}, p_2^{Caps}, p_3^{Caps})\ ,\max(p_1^{CNN}, p_2^{CNN}, p_3^{CNN}))$ represent the predicted class.
- **Rule 2**: If $p_2^{CNN} > 0.95$, then the class of the input image is Dirt otherwise $\max(\max(p_1^{Caps}, p_2^{Caps}, p_3^{Caps})\ ,\max(p_1^{CNN}, p_2^{CNN}, p_3^{CNN}))$ represent the predicted class.
- **Rule 3**: If $p_3^{CNN} > 0.95$, then the class of the input image is Normal otherwise $\max(\max(p_1^{Caps}, p_2^{Caps}, p_3^{Caps}), \max(p_1^{CNN}, p_2^{CNN}, p_3^{CNN}))$ represent the predicted class.

Algorithm 3 Selection rules.

```
1:  procedure SELECTION(p1^Caps, p2^Caps, p3^Caps, p1^CNN, p2^CNN, p3^CNN)
2:      (p1^Caps, p2^Caps, p3^Caps) ← probabilities of three classes in WaferCaps
3:      (p1^CNN, p2^CNN, p3^CNN) ← probabilities of three classes in CNN
4:      if p1^Caps > 0.8 OR p1^CNN > 0.95 then
5:          predicted class ← Defect
6:      else if p2^CNN > 0.95 then
7:          predicted class ← Dirt
8:      else if p3^CNN > 0.95 then
9:          predicted class ← Normal
10:     else
11:         predicted class ← max(max(p1^Caps, p2^Caps, p3^Caps), max(p1^CNN, p2^CNN, p3^CNN))
12:     end if
13:     return predicted class
14: end procedure
```

Applying these rules to the output of the two classifiers merge the advantages of both classifiers to achieve high accuracy.

5.7 Results

In this section, the performance of the proposed combined "WaferCaps+CNN" is evaluated against using non-combined networks and other DL models. In the evaluation, different metrices

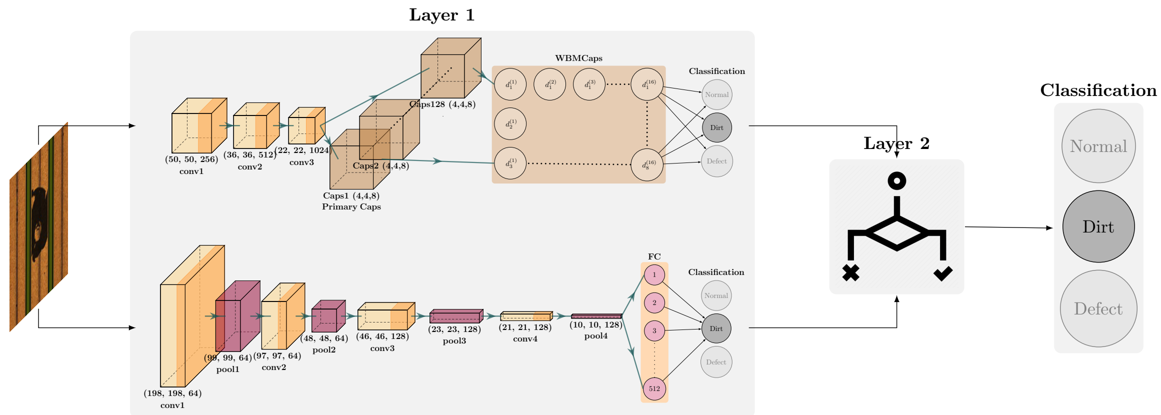


Figure 5.11: Architecture of the combined WaferCaps+CNN proposed approach.

were used such as confusion matrices, accuracy, recall, precision, and F1-score.

5.7.1 Experimental Data

All data were processed in RGB format with a dimension of 200×200 according to three classes: normal, dirt and defect. The data were divided into two categories; namely, training and validation. In order to investigate the performance of the tested networks, as well as the quality of the augmented data, all of the real samples were used for the dirt and defect classes in the validation dataset, while all the augmented data were used in the training of the networks. The total data samples used were 1,000 images per class. Such that the training dataset consisted of 800 samples, while the validation dataset consisted of 200 samples.

5.7.2 Proposed method vs. standalone models

A series of experiments were conducted to investigate the performance of using CNN and WaferCaps models separately and compare it with the performance of using the proposed combined model "WaferCaps+CNN". The results of classification for the CNN can be found in the confusion matrix of Figure 5.12(b) and metrices in Table 5.1. It can be observed that CNN achieved higher overall accuracy when compared to WaferCaps, and the performance of the dirt classification was also remarkable in CNN; however, despite of the better overall performance in CNN, WaferCaps achieved better classification results in both defect and normal classes. When both models' performance is combined the overall accuracy is further increased to 98.5%, and it also

took the best classification results for the normal and dirt class. However, the classification accuracy for the defect class was slightly dropped when compared to the WaferCaps model.

Actual Class \ Predicted Class	Defect	Dirt	Normal
Defect	194 97%	1 0%	5 2%
Dirt	14 7%	164 82%	22 11%
Normal	0 0%	0 0%	200 100%

(a) WaferCaps

Actual Class \ Predicted Class	Defect	Dirt	Normal
Defect	186 93%	11 5%	3 1%
Dirt	0 0%	199 99%	1 0%
Normal	0 0%	2 1%	198 99%

(b) CNN

Actual Class \ Predicted Class	Defect	Dirt	Normal
Defect	192 96%	4 2%	4 2%
Dirt	0 0%	199 99%	1 0%
Normal	0 0%	0 0%	200 100%

(c) WaferCaps+CNN

Figure 5.12: Confusion matrix for the validation dataset for different network setups.

5.7.3 Proposed method vs. other DL models

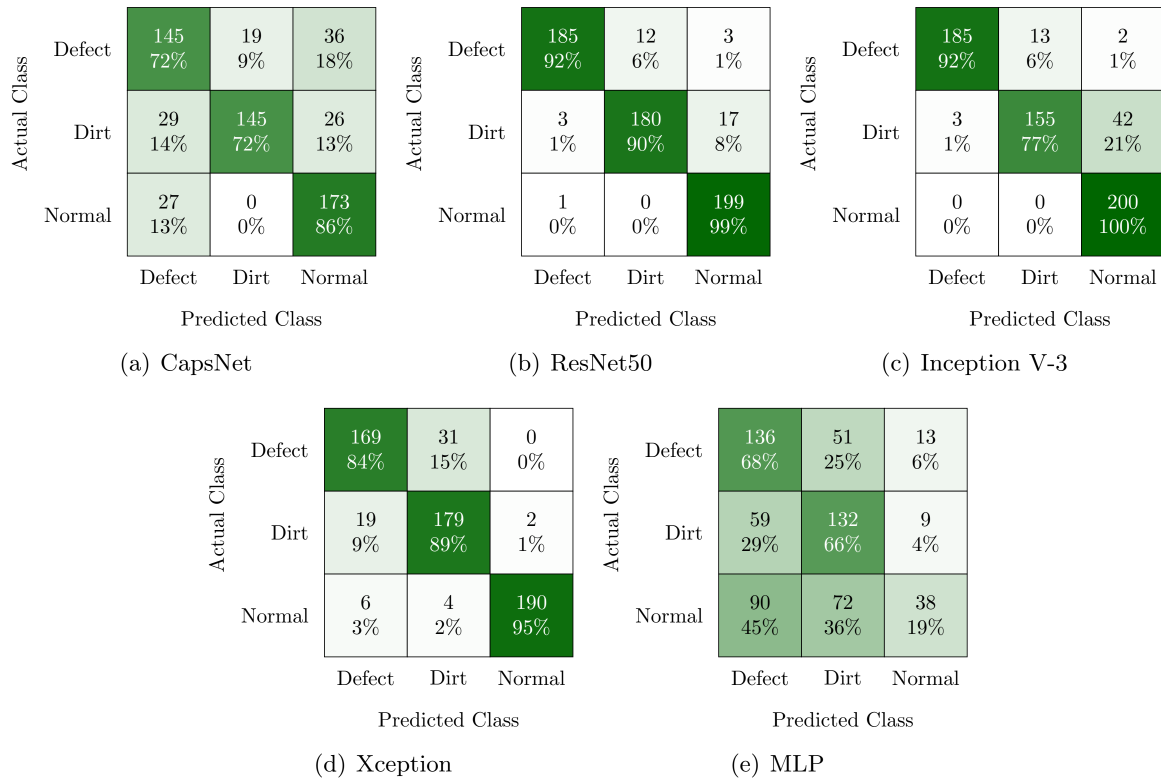
The proposed approach was further compared with relevant popular DL algorithms to validate its performance such as CapsNet, ResNet50 and Inception-V3. The performance of these networks is demonstrated in Figure 5.13 and Table 5.1. Comparing the five networks, ResNet50 achieved the best overall accuracy with 93.75%, followed by Inception-V3 with 90%, then Xception with 89.7%, then CapsNet with 77.2%, and finally MLP with 51%. The proposed approach of "Wafer-Caps+CNN" outperformed these networks. Figure 5.14 compares the proposed approach with all other models used in terms of the misclassified samples for each of them.

5.8 Attempted Results

Before applying the main methods in augmenting and classifying the image data that are presented in this chapter, several solutions were implemented that could prevent us in performing the manual augmentation such as using unsupervised learning with CAE for classifying or applying DCGAN to generate more synthetic images as has been done in Chapter 4. Unfortunately, none of them gave adequate results.

Table 5.1: Metrics for evaluating the validation data against different DL networks.

Model	Overall Accuracy	Metric	Defect	Dirt	Normal
WaferCaps + CNN (proposed)	**0.985**	**Recall**	**0.96**	**0.995**	**1.0**
		Precision	**1.0**	**0.9803**	**0.9756**
		F1-score	**0.9796**	**0.9876**	**0.9876**
WaferCaps	0.93	Recall	0.97	0.82	1.0
		Precision	0.9327	0.9939	0.8811
		F1-score	0.951	0.8986	0.9368
CNN	0.972	Recall	0.93	0.995	0.99
		Precision	1.0	0.9387	0.9802
		F1-score	0.9637	0.966	0.9851
CapsNet	0.772	Recall	0.725	0.725	0.865
		Precision	0.7214	0.8841	0.7362
		F1-score	0.7232	0.7967	0.7954
ResNet50	0.9375	Recall	0.925	0.9	0.995
		Precision	0.9788	0.9375	0.9087
		F1-score	0.9512	0.9184	0.9499
Inception-V3	0.9	Recall	0.925	0.775	1.0
		Precision	0.984	0.9226	0.8197
		F1-score	0.9536	0.8424	0.901
Xception	0.897	Recall	0.845	0.895	0.95
		Precision	0.8711	0.8364	0.9896
		F1-score	0.8579	0.8647	0.9693
MLP	0.51	Recall	0.68	0.66	0.19
		Precision	0.4772	0.5176	0.6333
		F1-score	0.5608	0.5802	0.2923



(a) CapsNet (b) ResNet50 (c) Inception V-3

(d) Xception (e) MLP

Figure 5.13: Confusion matrix for the validation dataset for different network setups.

5.8.1 CAE

CAE was used as a mean of unsupervised learning. The plan was to train the CAE using normal images only so it can reconstruct them and distinguish between the normal and anomaly classes (defect and dirt). The best CAE with minimal loss (Figure 5.15) consisted of an encoder with three convolutional layers and decoder with three deconvolutional layers. The comparison performed using density scores of the reconstructed images (pixels) of each CAE as shown in Figure 5.17. Unfortunately, the density score are overlapped with each other which means that CAE could not decaffeinate between normal class and anomaly classes. It is also clear that the CAE could not reconstruct the images accurately as shown in Figure 5.17. This is due to the low number of samples provided which made model incapable of capturing the difference between the features of normal samples and anomaly ones.

If this approach worked as supposed to, the number of samples for anomaly classes will not be a problem since the unsupervised learning process is trained on normal class images only.

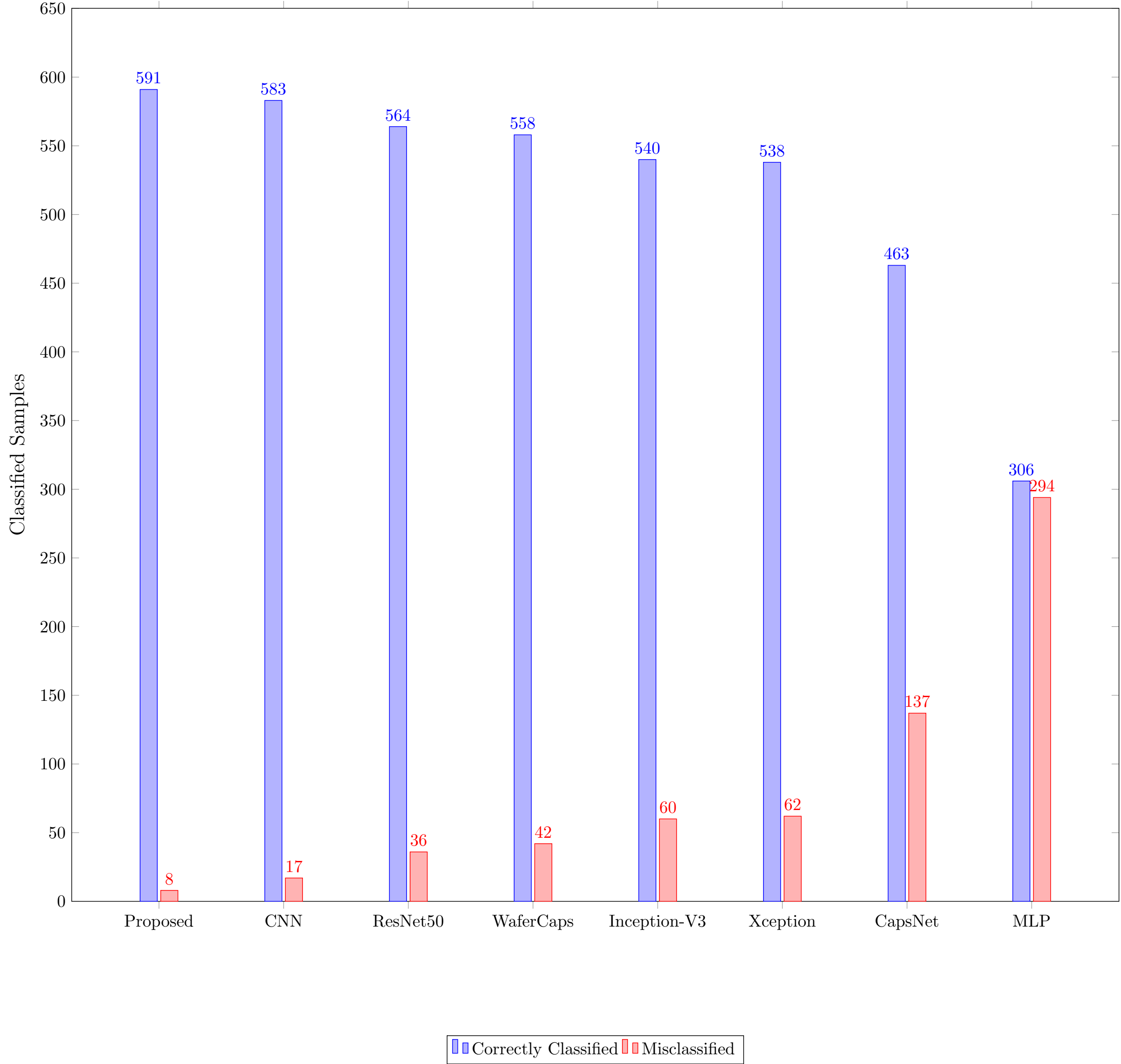


Figure 5.14: Classified vs. misclassified samples for proposed approach against other models.

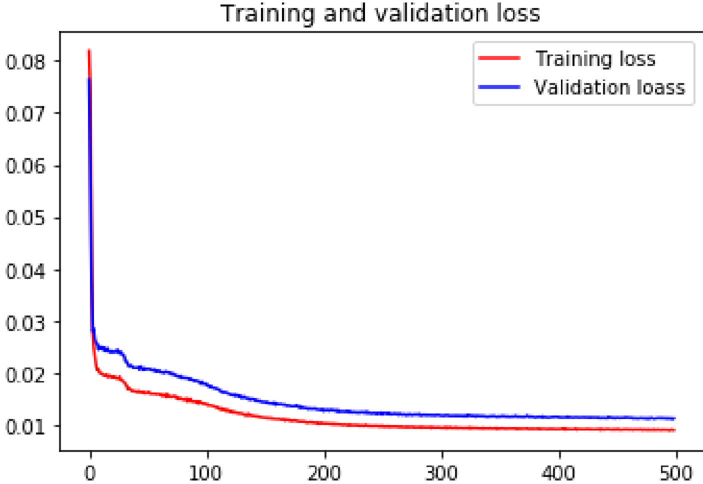
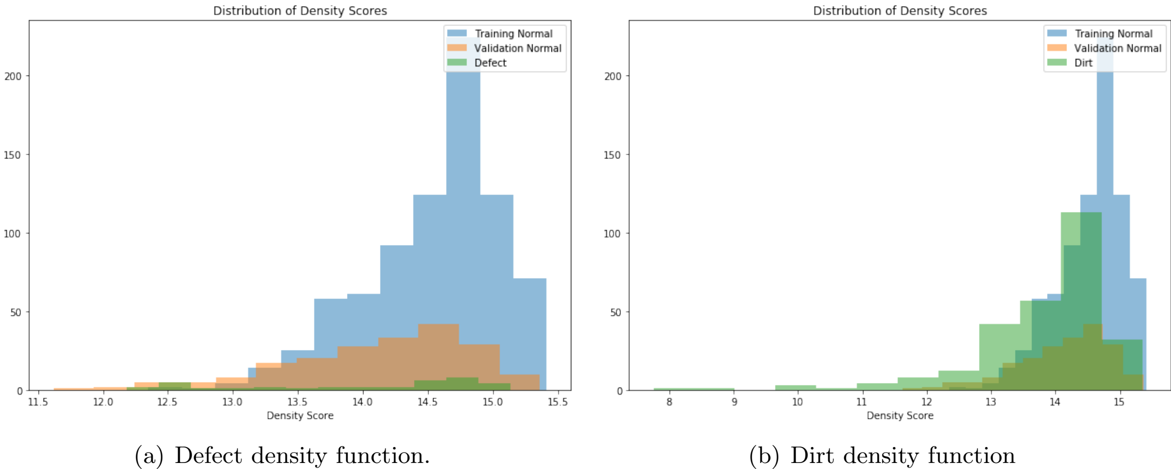


Figure 5.15: MNIST data sample.



(a) Defect density function.

(b) Dirt density function

Figure 5.16: WaferCaps proposed

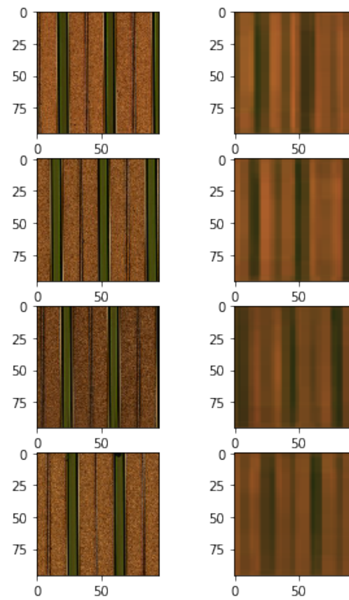


Figure 5.17: Original images (left) and reconstructed images (right).

5.8.2 DCGAN

Similar to what has been done in Chapter 4, an attempt of increasing the data using DCGAN was implemented. Multiple DCGANs were used in the attempts, and the best DCGAN architecture consisted of a discriminator of six convolutional layers and generator of six deoncovolutional layers.

In this DCGAN, it was taken into consideration that they type of images dealt with is in RGB format and therefore the output of the generator should generate and RGB image. Also, the noise vector size was increased to $[4000, 1]$ in order to accommodate the RGB image size requirements and the resulting image is of size 192×192. Therefore, all the images for training the discrimiator were resized from 200×200 to 192×192. The results of the synthetic images generated are shown in Figure 5.18. Unfortunately, DCGAN could not generate synthetic images with high resemblance with the original that can be used in the training due to the lack of samples provided for DCGAN for training. Therefore, manual data augmentation along with performing image correction operations (e.g. scaling, rotation and noise generation) were the best alternative for performing data augmentation and gave excellent results as demonstrated in the previous section.

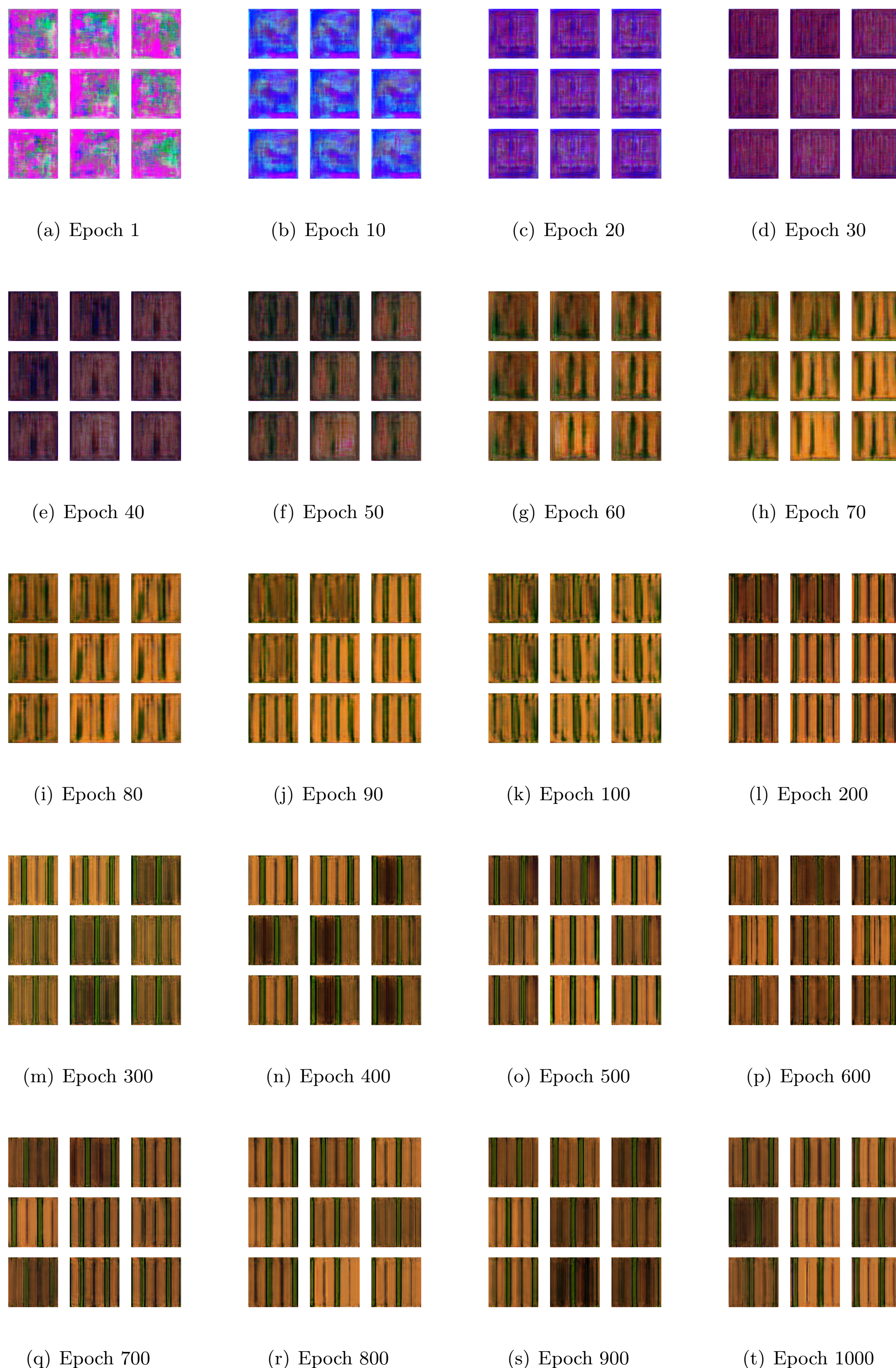


Figure 5.18: Manually augmented samples for defect and dirt anomalies.

5.9 Conclusion

Development of industry ready AOI solution to classify waveguide anomalies in QCL wafers according to three classes (defect, dirt and normal) was the aim of this chapter. In order to achieve this, a mechanism of image acquisition, data augmentation and DL classifier that combines two algorithms (WaferCaps and CNN) were proposed. The customised image acquisition system consists of a microscope and camera. The wafer images were then segmented into multiple samples, such that each sample has 200×200 of size in RGB format. Data augmentation approach was then used to increase the samples of the dirt and defect anomalies in a way that 1,000 images were used in each category. The total dataset was manually labelled by experts. All the generated dataset was used for training, and the real samples were used for validation.

The proposed DL classifier (WaferCaps+CNN) combines the performance of WaferCaps and CNN by rule-based selection of the predictions generated. It showed accurate and robust capability to detect QCL anomalies with high validation accuracy of 98.5%. The performance of the proposed solution was compared with performance of using standalone CNN and WaferCaps as well as with other DL such as ResNet-50, Inception-V3, CapsNet, Xception and MLP. It was proven that the overall performance of WaferCaps+CNN outperformed them all.

Despite the remarkable achievement of identifying normal, dirt and defect samples accurately, this research has some limitations. First, in case of the change of production methods, other possible anomalies and defects feature may rise, which cannot be identified by this approach that is trained to classify three classes. This may open the door on using uncertainty learning. Second, augmenting samples manually is time consuming and may be not the best way in dealing with big data. Therefore, more samples will be acquired in the future to use more effective methods in data augmentation such as DCGAN. Finally, the algorithm cannot specify the location of the anomaly and can only classify the sample. In a future research, more samples will be used and a dedicated algorithm to specify the location of the anomaly accurately will be proposed.

Chapter 6

Conclusion and Future Work

> "*Do not say a little in many words, but a great deal in a few*"
>
> Pythagoras

6.1 Overview

This chapter summarises the entire works carried out in the thesis. In addition, the chapter aims to highlight the main findings and contributions. Based on the findings and contributions, research limitations are discussed and future work to overcome these limitations is presented.

6.2 Review of Main Findings

The purpose of this research was to identify optoelectronic and semiconductor wafer defects as they are considered an early stage in manufacturing complex electronic devices. Detecting these defects and flaws in the early stage can help in achieving the bigger goal of zero-defect manufacturing. DL and computer vision techniques were used to solve this problem by assessing the images of these products that contains the defect features. Several challenges were faced in this research such as lack of image data for training the DL network and class imbalance. Image augmentation approaches were used to mitigate these challenges by increasing the image samples automatically by using DCGAN and image operations such as rotation, scaling etc and manually by mimicking the anomaly feature on normal images.

Chapter 2 achieved the primary goal of reviewing the literature for the similar research and existing techniques for AOI in electronics industry in general. The defects and anomalies of the commonly inspected electronic components, such as semiconductor wafers, flat panel displays, printed circuit boards and light emitting diodes, are first explained. Hardware setups used in acquiring images are then discussed in terms of the camera and lighting source selection and configuration. The inspection algorithms used for detecting the defects in the electronic components are discussed in terms of the preprocessing, feature extraction and classification tools used for this purpose. Recent articles that used DL algorithms were also reviewed.

In Chapter 3 the discussion is limited by focusing on the theory behind DL techniques and assessing DL architectures that are used for image recognition and computer vision using the open source data MNIST. It was noticed in this chapter how using deeper networks can help in better feature extraction process and therefore, the classification process will be more accurate and overfitting can be avoided. It was also demonstrated the use of DCGAN in data augmentation, which can be useful in upsampling and increasing the dataset used in training.

In Chapter 4, the open access dataset WM-811K was used to classify the defect patterns in WBMs. The main challenge faced in this study was the data imbalance and the lack of data in some classes such as the near-full class. In order to upsample the defect patters and increase the dataset amount, DCGAN was proposed to apply the data augmentation. DCGAN were performed on each class separately and the increase the data for minor classes at the same time. An improved CapsNet classifier (WaferCaps) was also proposed to classify the original and DCGAN generated samples into eight patterns. The overall performance of WaferCaps achieved training accuracy of 99.59%, validation accuracy of 97.53% and test accuracy of 91.4%. The performance of WaferCaps was also compared to other DL algorithms such as CapsNet, CNN and MLP.

Finally, in Chapter 5 we investigated optoelectronic wafer anomalies in sample images collected from a manufacturer in Europe. Our main aim was to assess the waveguide quality in QCL devices in wafers according to two anomalies i.e. dirt and discontinuities (defect). Defects in waveguide cause interruption in the laser beam which in return result in QCL to malfunction. Dirt does not necessarily cause damage to the wavegiude; however, the inspector should be alerted to this case since a damage can happen beneath the dirt on the waveguide. The aquired

image samples for the wafers were first segmented, and each segment of these were added in an image library according to three labels provided by experts (dirt, normal and defect). Manual and automatic image augmentation were used to upsample and increase the dataset in the dirt and defect classes as the number of samples in these classes were not enough for training a classifier model. A combinaion of CNN and WaferCaps (used in Chapter 4) classifiers using decision fusion were used to classify the samples. This choice were done since each of the classifier considered gave different results for the classified samples, and therefore the performance for the best classifier were chosen for each class.

6.3 Contributions

The main contributions of this research are as follows:

- Preparing a comprehensive and systematic literature review on applying AOI techniques in electronics industry.
- Use DCGAN to generate synthetic data for the WM-811k dataset successfully. These generated images are used to upsample and increase the number of samples in each class.
- Evidence that using DCGAN to increase the dataset enhanced the classification accuracy and decreased the overfitting in training.
- Proposing a novel improved capsule neural network called WaferCaps.
- Evidence that the proposed WaferCaps for classsifying outperformed other state-of-the-art DL models such as CapsNet, CNN and MLP.
- Investigating opto-electronic wafer defets using AOI and DL approaches.
- Generating image library for waveguide anomalies using manual and automatic augmentation techniques.
- Apply parallel decision fusion approach to combine the performance of CNN and WaferCaps.

- Evidence that combination of CNN and WaferCaps models performed the best in classifying waveguide anomalies when compared with standalone models and state-of-the-art DL models such as CapsNet, ResNet-50, Xception, Inception-V3 and MLP.

6.4 Limitations

This research proposed reliable DL algorithms for classifying and augmenting smeiconductor and optoelectronic wafer samples. However, like any other research, a number of limitations existed in conducting the research:

- In dealing with WM-811K, DCGAN could not generate synthetic images for the scratch class, which forced us to perform the data augmentation process using image processing techniques. Unlike DCGAN, data augmentation using image processing is limited and generating an infinite or large number of samples is not possible option. Therefore in order to keep the number of samples the same for all the classes in WM-811K, the same number of samples for other classes were generated using DCGAN to match the number of samples in the scratch class. This limitation has affected the number of generated samples and made us to be capped with 10,000 images for each class.

- Optoelectronic wafer data were not enough to perform DL data augmentation techniques such as DCGAN. Therefore, manual augmentation were performed by copying the defect and dirt features on normal samples. This technique is time consuming and inconvenient when dealing with big data.

- The algorithms in this thesis are well-designed to recognise the pre-defined classes. But what if a new type of defect appeared due to the change of the production parameters of processes. This scenario will confuse the networks and it will generate false alarm rates in classification.

6.5 Future Work

While the results are encouraging, there is still a room for improvement and hopefully a way to overcome the previously mentioned limitations in this research. Two methods in this section will

be suggested that can be implemented to enhance the classification which are uncertainty in DL and CapsuleGAN.

6.5.1 Uncertainty in Deep Learning

Just like a human being (maybe we can exclude the arrogant ones), a good DL model should tell you "sorry, I do not know" if an unknown new data has been not recognised by the model. Unfortunately, this is not the case in most of the computer vision applications that use DL techniques (including this research). What makes things worse that the model classifies these unknown data according to the pre-existing classes. The answer why most models are behaving like this is simply because they are not trained to do so. Therefore, there should be a way to assess the "uncertainty in DL".

There are two main types of uncertainty, i.e., epistemic and aleatoric [426]. Epistemic uncertainty occurs due to inadequate knowledge of the model (training data were not enough for generalisation). On the other hand, aleatoric uncertainty occurs from the natural stochasticity of observations. Therefore, is not a property of the model, but rather is an inherent property of the data distribution, and hence, it cannot be resolved even if more data were provided.

Several Bayesian approaches can be used with DL model to mitigate this problem and develop a class to accommodate uncertainty issues [427]. This can be an investigation for future work to consider unknown images or features by the model. This is very useful in the case of identifying semiconductor or optoelectronic wafer anomalies and defect patterns in case of a new anomaly/pattern generated from the production steps. Applying uncertainty in DL will reduce the false alarm rates in inspection.

6.5.2 CapsuleGAN

Considering powerful networks for the discriminator and generator in GAN plays vital role in the quality of the augmented images. DCGAN showed remarkable results in generating synthetic images of the WM-811K dataset. However, it neither could generate scratch samples nor optoelectronic wafer samples efficiently. Considering capusleGAN in generating synthetic images could solve the problem by capturing more features in the training images due to the spatial characteristics of CapsNet in memorising the location of the features [412]. The only concern

would be the computational power need to train two CapsNet networks. This again could be a subject of a future research in generating synthetic images of optoelectronic and semiconductor wafers.

6.6 Conclusion

This research proposed DL methods for data augmentation and classification of electronic wafers (i.e. semiconductor and optoelectronic). The work used DCGAN to upsample the data of WM-811K for more accurate classification results. A novel capsule neural network (WaferCaps) was proposed to classify the data according to the defect pattern. Manual augmentation techniques were also used to increase the library for samples of optoelectronic wafer segments. The waveguide quality was the main focus in this type of wafers. A fusion of CNN and WaferCaps was proposed to classify the data into three classes namely, normal, defect and dirt.

Bibliography

[1] J. A. Carvajal Soto, F. Tavakolizadeh, and D. Gyulai, *An online machine learning framework for early detection of product failures in an Industry 4.0 context*, International Journal of Computer Integrated Manufacturing **32**, 452–465 (2019).

[2] Y. Liu, C. Jiang, and H. Zhao, *Using contextual features and multi-view ensemble learning in product defect identification from online discussion forums*, Decision Support Systems **105**, 1–12 (2018).

[3] T. Czimmermann, G. Ciuti, M. Milazzo, M. Chiurazzi, S. Roccella, C. M. Oddo, and P. Dario, *Visual-based defect detection and classification approaches for industrial applications—A SURVEY*, Sensors (Switzerland) **20**, 1–25 (2020).

[4] S. H. Huang and Y. C. Pan, *Automated visual inspection in the semiconductor industry: A survey*, Computers in Industry **66**, 1–10 (2015).

[5] S. Lv, H. Kim, B. Zheng, and H. Jin, *A review of data mining with Big Data towards its applications in the electronics industry*, Applied Sciences (Switzerland) **8**, 1–34 (2018).

[6] IBISWorld. *Global Consumer Electronics Manufacturing Industry - Market Research Report.* Technical report, (2019).

[7] M. Liukkonen, E. Havia, and Y. Hiltunen, *Computational intelligence in mass soldering of electronics - A survey*, Expert Systems with Applications **39**, 9928–9937 (2012).

[8] D. M. Tsai and Y. C. Hsieh, *Machine Vision-Based Positioning and Inspection Using Expectation-Maximization Technique*, IEEE Transactions on Instrumentation and Measurement **66**, 2858–2868 (2017).

[9] S. Gebus and K. Leiviskä, *Knowledge acquisition for decision support systems on an electronic assembly line*, Expert Systems with Applications **36**, 93–101 (2009).

[10] Y. Xie, Y. Ye, J. Zhang, L. Liu, and L. Liu, *A physics-based defects model and inspection algorithm for automatic visual inspection*, Optics and Lasers in Engineering **52**, 218–223 (2014).

[11] Toshiba. *How to make semiconductors, the brains of everything*, (2021). Accessed online: https://www.toshiba-clip.com/en/detail/p=719.

[12] J. Chai, H. Zeng, A. Li, and E. W. Ngai, *Deep learning in computer vision: A critical review of emerging techniques and application scenarios*, Machine Learning with Applications **6**, 100134 (2021).

[13] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus. *Deconvolutional networks.* In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2528–2535, (2010).

[14] M. D. Zeiler, G. W. Taylor, and R. Fergus. *Adaptive deconvolutional networks for mid and high level feature learning.* In *2011 International Conference on Computer Vision*, pages 2018–2025, (2011).

[15] M. Gupta, M. A. Khan, R. Butola, and R. M. Singari, *Advances in applications of Non-Destructive Testing (NDT): A review*, Advances in Materials and Processing Technologies **0**, 1–22 (2021).

[16] S. K. Dwivedi, M. Vishwakarma, and P. Soni, *Advances and Researches on Non Destructive Testing: A Review*, Materials Today: Proceedings **5**, 3690–3698 (2018).

[17] T. D'Orazio, M. Leo, A. Distante, C. Guaragnella, V. Pianese, and G. Cavaccini, *Automatic ultrasonic inspection for internal defect detection in composite materials*, NDT & E International **41**, 145–154 (2008).

[18] S. Gholizadeh, *A review of non-destructive testing methods of composite materials*, Procedia Structural Integrity **1**, 50–57 (2016).

[19] J. García-Martín, J. Gómez-Gil, and E. Vázquez-Sánchez, *Non-Destructive Techniques Based on Eddy Current Testing*, Sensors (Switzerland) **11**, 2525–2565 (2011).

[20] S. Mukhopadhyay, *Novel Planar Electromagnetic Sensors: Modeling and Performance Evaluation*, Sensors (Switzerland) **5**, 546–579 (2005).

[21] K. Chomsuwan, S. Yamada, and M. Iwahara, *Bare PCB Inspection System With SV-GMR Sensor Eddy-Current Testing Probe*, IEEE Sensors Journal **7**, 890–896 (2007).

[22] T. G. Santos, R. Miranda, and C. C. de Carvalho, *A new NDT technique based on bacterial cells to detect micro surface defects*, NDT & E International **63**, 43–49 (2014).

[23] P. Aryan, S. Sampath, and H. Sohn, *An Overview of Non-Destructive Testing Methods for Integrated Circuit Packaging Inspection*, Sensors (Switzerland) **18**, 1981 (2018).

[24] H. Shih-Cheng, Y. Feng-Yi, and L. Shieh-Shing, *Applying PSO and OCBA to Minimize the Overkills and Re-Probes in Wafer Probe Testing*, IEEE Transactions on Semiconductor Manufacturing **25**, 531–540 (2012).

[25] T. S. Newman and A. K. Jain, *A Survey of Automated Visual Inspection*, Computer Vision and Image Understanding **61**, 231–262 (1995).

[26] M. J. J. Wang and C. L. Huang, *Evaluating the eye fatigue problem in wafer inspection*, IEEE Transactions on Semiconductor Manufacturing **17**, 444–447 (2004).

[27] M. H. Kim, T. Weyrich, and J. Kautz, *Modeling human color perception under extended luminance levels*, ACM Transactions on Graphics **28**, 1–9 (2009).

[28] G. Nam, H. Lee, S. Oh, and M. H. Kim, *Measuring color defects in flat panel displays using HDR imaging and appearance modeling*, IEEE Transactions on Instrumentation and Measurement **65**, 297–304 (2016).

[29] R. T. Chin and C. A. Harlow, *Automated Visual Inspection: A Survey*, IEEE Transactions on Pattern Analysis and Machine Intelligence **4**, 557–573 (1982).

[30] A. R. Rao, *Future directions in industrial machine vision: A case study of semiconductor manufacturing applications*, Image and Vision Computing **14**, 3–19 (1996).

[31] F. Timm and E. Barth, *Novelty detection for the inspection of light-emitting diodes*, Expert Systems with Applications **39**, 3413–3422 (2012).

[32] A. M. Abu Ebayyeh and A. Mousavi, *A Review and Analysis of Automatic Optical Inspection and Quality Monitoring Methods in Electronics Industry*, IEEE Access **8**, 183192–183271 (2020).

[33] R. Carnell, N. Mapa, I. Pang, and P. Sakpal, *Asia and the global tech slump: The chips are down*, ING (2019).

[34] Y. S. Chiu and H. D. Lin, *An innovative blemish detection system for curved LED lenses*, Expert Systems with Applications **40**, 471–479 (2013).

[35] M. Bürmen, F. Pernuš, and B. Likar, *Automated optical quality inspection of light emitting diodes*, Measurement Science and Technology **17**, 1372–1378 (2006).

[36] D. B. Perng, H. W. Liu, and C. C. Chang, *Automated SMD LED inspection using machine vision*, International Journal of Advanced Manufacturing Technology **57**, 1065–1077 (2011).

[37] H. D. Lin, *Automated defect inspection of light-emitting diode chips using neural network and statistical approaches*, Expert Systems with Applications **36**, 219–226 (2009).

[38] M. H. Ahmed Fadzil and C. J. Weng, *LED Cosmetic Flaw Vision Inspection System*, Pattern Analysis and Applications **1**, 62–70 (1998).

[39] H. D. Lin and S. W. Chiu, *Flaw detection of domed surfaces in LED packages by machine vision system*, Expert Systems with Applications **38**, 15208–15216 (2011).

[40] Y. Yang, Y. Lou, M. Gao, and G. Ma, *An automatic aperture detection system for LED cup based on machine vision*, Multimedia Tools and Applications **77**, 23227–23244 (2018).

[41] D. B. Perng, H. W. Liu, and S. H. Chen, *A vision-based LED defect auto-recognition system*, Nondestructive Testing and Evaluation **29**, 315–331 (2014).

[42] C. F. J. Kuo, T. Fang, C. L. Lee, and H. C. Wu, *Automated optical inspection system for surface mount device light emitting diodes*, Journal of Intelligent Manufacturing **30**, 641–655 (2019).

[43] C. Y. Chang, C. H. Li, S. Y. Lin, and M. D. Jeng, *Application of two Hopfield neural networks for automatic four-element LED inspection*, IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews **39**, 352–365 (2009).

[44] C. Y. Chang, C. H. Li, Y. C. Chang, and M. Jeng, *Wafer defect inspection by neural analysis of region features*, Journal of Intelligent Manufacturing **22**, 953–964 (2011).

[45] H. D. Lin, G. C. Lin, C. Y. Chung, and W. T. Lin, *Wavelet-based neural network and statistical approaches applied to automated visual inspection of LED chips*, Journal of Scientific and Industrial Research **67**, 412–420 (2008).

[46] D. M. Tsai, I. Y. Chiang, and Y. H. Tsai, *A shift-tolerant dissimilarity measure for surface defect detection*, IEEE Transactions on Industrial Informatics **8**, 128–137 (2012).

[47] C. F. J. Kuo, C. T. M. Hsu, Z. X. Liu, and H. C. Wu, *Automatic inspection system of LED chip using two-stages back-propagation neural network*, Journal of Intelligent Manufacturing **25**, 1235–1243 (2014).

[48] C. C. Hsu and M. S. Chen, *Intelligent maintenance prediction system for LED wafer testing machine*, Journal of Intelligent Manufacturing **27**, 335–342 (2016).

[49] F. Zhong, S. He, and B. Li, *Blob analyzation-based template matching algorithm for LED chip localization*, International Journal of Advanced Manufacturing Technology **93**, 55–63 (2017).

[50] C. F. J. Kuo, C. P. Tung, and W. H. Weng, *Applying the support vector machine with optimal parameter design into an automatic inspection system for classifying micro-defects on surfaces of light-emitting diode chips*, Journal of Intelligent Manufacturing **30**, 727–741 (2019).

[51] H. Lin, B. Li, X. Wang, Y. Shu, and S. Niu, *Automated defect inspection of LED chip using deep convolutional neural network*, Journal of Intelligent Manufacturing **30**, 2525–2534 (2019).

[52] R. F. Fung, C. Y. Yang, and C. T. Lai, *Graphic supervisory control of an automatic optical*

inspection for LED properties, Measurement: Journal of the International Measurement Confederation **44**, 1349–1360 (2011).

[53] C. Ngo, Y. J. Park, J. Jung, R. U. Hassan, and J. Seok, *A new algorithm on the automatic TFT-LCD mura defects inspection based on an effective background reconstruction*, Journal of the Society for Information Display **25**, 737–752 (2017).

[54] J. Y. Lee and S. I. Yoo, *Automatic detection of region-mura defect in TFT-LCD*, IEICE Transactions on Information and Systems **E87-D**, 2371–2378 (2004).

[55] B. C. Jiang, C. C. Wang, and H. C. Liu, *Liquid crystal display surface uniformity defect inspection using analysis of variance and exponentially weighted moving average techniques*, International Journal of Production Research **43**, 67–80 (2005).

[56] S. L. Chen and J. H. Chang, *TFT-LCD Mura defects automatic inspection system using linear regression diagnostic model*, Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture **222**, 1489–1501 (2008).

[57] L. C. Chen and C. C. Kuo, *Automatic TFT-LCD mura defect inspection using discrete cosine transform-based background filtering and 'just noticeable difference' quantification strategies*, Measurement Science and Technology **19** (2008).

[58] S. L. Chen and S. T. Chou, *TFT-LCD Mura Defect Detection Using Wavelet and Cosine Transforms*, Journal of Advanced Mechanical Design, Systems, and Manufacturing **2**, 441–453 (2008).

[59] X. Bi, C. Zhuang, and H. Ding, *A New Mura Defect Inspection Way for TFT-LCD Using Level Set Method*, IEEE Signal Processing Letters **16**, 311–314 (2009).

[60] S. Mei, H. Yang, and Z. Yin, *Unsupervised-learning-based feature-level fusion method for mura defect recognition*, IEEE Transactions on Semiconductor Manufacturing **30**, 105–113 (2017).

[61] H. Yang, K. Song, S. Mei, and Z. Yin, *An Accurate Mura Defect Vision Inspection Method Using Outlier-Prejudging-Based Image Background Construction and Region-Gradient-*

Based Level Set, IEEE Transactions on Automation Science and Engineering **15**, 1704–1721 (2018).

[62] H. Yang, S. Mei, K. Song, B. Tao, and Z. Yin, *Transfer-Learning-Based Online Mura Defect Classification*, IEEE Transactions on Semiconductor Manufacturing **31**, 116–123 (2018).

[63] Y. S. P. Chiu and H. D. Lin, *A hybrid approach based on Hotelling statistics for automated visual inspection of display blemishes in LCD panels*, Expert Systems with Applications **36**, 12332–12339 (2009).

[64] Y. H. Tsneg and D. M. Tsai, *Defect detection of uneven brightness in low-contrast images using basis image representation*, Pattern Recognition **43**, 1129–1141 (2010).

[65] S. K. S. Fan and Y. C. Chuang, *Automatic detection of Mura defect in TFT-LCD based on regression diagnostics*, Pattern Recognition Letters **31**, 2397–2404 (2010).

[66] W. C. Li and D. M. Tsai, *Defect Inspection in Low-Contrast LCD Images Using Hough Transform-Based Nonstationary Line Detection*, IEEE Transactions on Industrial Informatics **7**, 136–147 (2011).

[67] D. M. Tsai and H. Y. Tsai, *Low-contrast surface inspection of mura defects in liquid crystal displays using optical flow-based motion analysis*, Machine Vision and Applications **22**, 629–649 (2011).

[68] Y. Gan and Q. Zhao, *An effective defect inspection method for LCD using active contour model*, IEEE Transactions on Instrumentation and Measurement **62**, 2438–2445 (2013).

[69] S. Mei, H. Yang, and Z. Yin, *An unsupervised-learning-based approach for automated defect inspection on textured surfaces*, IEEE Transactions on Instrumentation and Measurement **67**, 1266–1277 (2018).

[70] D. M. Tsai and C. Y. Hung, *Automatic defect inspection of patterned thin film transistor-liquid crystal display (TFT-LCD) panels using one-dimensional Fourier reconstruction and wavelet decomposition*, International Journal of Production Research **43**, 4589–4607 (2005).

[71] D. M. Tsai, S. T. Chuang, and Y. H. Tseng, *One-dimensional-based automatic defect inspection of multiple patterned TFT-LCD panels using Fourier image reconstruction*, International Journal of Production Research **45**, 1297–1321 (2007).

[72] D. M. Tsai and S. T. Chuang, *1D-based defect detection in patterned TFT-LCD panels using characteristic fractal dimension and correlations*, Machine Vision and Applications **20**, 423–434 (2009).

[73] Y. G. Cen, R. Z. Zhao, L. H. Cen, L. H. Cui, Z. J. Miao, and Z. Wei, *Defect inspection for TFT-LCD images based on the low-rank matrix reconstruction*, Neurocomputing **149**, 1206–1215 (2015).

[74] M. Kim, M. Lee, M. An, and H. Lee, *Effective automatic defect classification process based on CNN with stacking ensemble model for TFT-LCD panel*, Journal of Intelligent Manufacturing **31** (2019).

[75] H. Yang, Y. Chen, K. Song, and Z. Yin, *Multiscale Feature-Clustering-Based Fully Convolutional Autoencoder for Fast Accurate Visual Inspection of Texture Surface Defects*, IEEE Transactions on Automation Science and Engineering **16**, 1450–1467 (2019).

[76] C. J. Lu and D. M. Tsai, *Defect inspection of patterned thin film transistor-liquid crystal display panels using a fast sub-image-based singular value decomposition*, International Journal of Production Research **42**, 4331–4351 (2004).

[77] C. J. Lu and D. M. Tsai, *Automatic defect inspection for LCDs using singular value decomposition*, International Journal of Advanced Manufacturing Technology **25**, 53–61 (2005).

[78] A. Wu, J. Zhu, Z. Tao, and C. Mao, *Automatic inspection and classification for thin-film transistor liquid crystal display surface defects based on particle swarm optimization and one-class support vector machine*, Advances in Mechanical Engineering **8**, 1–11 (2016).

[79] Y. G. Yoon, S. L. Lee, C. W. Chung, and S. H. Kim, *An effective defect inspection system for polarized film images using image segmentation and template matching techniques*, Computers and Industrial Engineering **55**, 567–583 (2008).

[80] C. F. J. Kuo, C. H. Chiu, and Y. C. Chou, *Research and development of intelligent on-line real-time defect inspection system for polymer polarizer*, Polymer - Plastics Technology and Engineering **48**, 185–192 (2009).

[81] C. H. Noh, S. L. Lee, D. H. Kim, C. W. Chung, and S. H. Kim, *An effective and efficient defect inspection system for TFT-LCD polarised films using adaptive thresholds and shape-based image analyses*, International Journal of Production Research **48**, 5115–5135 (2010).

[82] D. Lim and D. H. Jeong, *Zone-Based Inspection and Defect Classification for LCD Manufacturing: Trivial Defect Free Procedure for TFT Glass Inspection*, International Journal of Optomechatronics **1**, 312–330 (2007).

[83] S. M. Chao and D. M. Tsai, *An anisotropic diffusion-based defect detection for low-contrast glass substrates*, Image and Vision Computing **26**, 187–200 (2008).

[84] A. Yousefian-Jazi, J. H. Ryu, S. Yoon, and J. J. Liu, *Decision support in machine vision system for monitoring of TFT-LCD glass substrates manufacturing*, Journal of Process Control **24**, 1015–1023 (2014).

[85] L. F. Chen, C. T. Su, and M. H. Chen, *A neural-network approach for defect recognition in TFT-LCD photolithography process*, IEEE Transactions on Electronics Packaging Manufacturing **32**, 1–8 (2009).

[86] C. S. Lin, W. Z. Wu, Y. L. Lay, and M. W. Chang, *A digital image-based measurement system for a LCD backlight module*, Optics and Laser Technology **33**, 499–505 (2001).

[87] D. M. Tsai, P. C. Lin, and C. J. Lu, *An independent component analysis-based filter design for defect detection in low-contrast surface images*, Pattern Recognition **39**, 1679–1694 (2006).

[88] S. M. Chao and D. M. Tsai, *Anisotropic diffusion with generalized diffusion coefficient function for defect detection in low-contrast surface images*, Pattern Recognition **43**, 1917–1931 (2010).

[89] D. M. Tsai, M. C. Chen, W. C. Li, and W. Y. Chiu, *A fast regularity measure for surface defect detection*, Machine Vision and Applications **23**, 869–886 (2012).

[90] W. J. Lin and S. S. Jhuo, *A fast luminance inspector for backlight modules based on multiple kernel support vector regression*, IEEE Transactions on Components, Packaging and Manufacturing Technology **4**, 1391–1401 (2014).

[91] Y. H. Liu, Y. K. Huang, and M. J. Lee, *Automatic inline defect detection for a thin film transistor–liquid crystal display array process using locally linear embedding and support vector data description*, Measurement Science and Technology **19**, 095501 (2008).

[92] Y. H. Liu, S. H. Lin, Y. L. Hsueh, and M. J. Lee, *Automatic target defect identification for TFT-LCD array process inspection using kernel FCM-based fuzzy SVDD ensemble*, Expert Systems with Applications **36**, 1978–1998 (2009).

[93] Y. H. Liu, C. K. Wang, Y. Ting, W. Z. Lin, Z. H. Kang, C. S. Chen, and J. S. Hwang, *In-TFT-Array-Process Micro Defect Inspection Using Nonlinear Principal Component Analysis*, International Journal of Molecular Sciences **10**, 4498–4514 (2009).

[94] Y. H. Liu and Y. J. Chen, *Automatic defect detection for TFT-LCD array process using quasiconformal kernel support vector data description*, International Journal of Molecular Sciences **12**, 5762–5781 (2011).

[95] Yi-Hung Liu, Yan-Chen Liu, and Yen-Jen Chen, *Fast Support Vector Data Descriptions for Novelty Detection*, IEEE Transactions on Neural Networks **21**, 1296–1313 (2010).

[96] Y. H. Liu, Y. C. Liu, and Y. Z. Chen, *High-speed inline defect detection for TFT-LCD array process using a novel support vector data description*, Expert Systems with Applications **38**, 6222–6231 (2011).

[97] C. F. Kuo, C. T. M. Hsu, C. H. Fang, S. M. Chao, and Y. D. Lin, *Automatic defect inspection system of colour filters using Taguchi-based neural network*, International Journal of Production Research **51**, 1464–1476 (2013).

[98] C. F. J. Kuo, K. C. Peng, H. C. Wu, and C. C. Wang, *Automated inspection of micro-defect recognition system for color filter*, Optics and Lasers in Engineering **70**, 6–17 (2015).

[99] S. W. Yang, C. S. Lin, S. K. Lin, and H. T. Chiang, *Automatic defect recognition of TFT array process using gray level co-occurrence matrix*, Optik **125**, 2671–2676 (2014).

[100] C. S. Lin, K. H. Huang, Y. L. Lay, K. C. Wu, Y. C. Wu, and J. M. Lin, *An improved pattern match method with flexible mask for automatic inspection in the LCD manufacturing process*, Expert Systems with Applications **36**, 3234–3239 (2009).

[101] C. S. Lin, K. H. Huang, T. C. Lin, H. J. Shei, and C. L. Tien, *An automatic inspection method for the fracture conditions of anisotropic conductive film in the TFT-LCD assembly process*, International Journal of Optomechatronics **5**, 286–298 (2011).

[102] X. Sheng, L. Jia, Z. Xiong, Z. Wang, and H. Ding, *ACF-COG interconnection conductivity inspection system using conductive area*, Microelectronics Reliability **53**, 622–628 (2013).

[103] G. Ni, L. Liu, X. Du, J. Zhang, J. Liu, and Y. Liu, *Accurate AOI inspection of resistance in LCD Anisotropic Conductive Film bonding using differential interference contrast*, Optik **130**, 786–796 (2017).

[104] M. F. Shirazi, K. Park, R. E. Wijesinghe, H. Jeong, S. Han, P. Kim, M. Jeon, and J. Kim, *Fast industrial inspection of optical thin film using optical coherence tomography*, Sensors (Switzerland) **16**, 1–13 (2016).

[105] J. Y. Lee, T. W. Kim, and H. J. Pahk, *Robust defect detection method for a non-periodic TFT-LCD pad area*, International Journal of Precision Engineering and Manufacturing **18**, 1093–1102 (2017).

[106] Q. Lu, Q. Fu, L. Luo, Q. Yuan, W. Hua, and Y. Yunguang, *Measurement method of LCD surface deformation for smartphone based on optical vision sensing system*, Optik **172**, 1079–1088 (2018).

[107] W. W. Lai, X. X. Zeng, J. He, and Y. L. Deng, *Aesthetic defect characterization of a polymeric polarizer via structured light illumination*, Polymer Testing **53**, 51–57 (2016).

[108] Y. Long Deng, S. Peng Xu, and W. Wei Lai, *A novel imaging-enhancement-based inspection method for transparent aesthetic defects in a polymeric polarizer*, Polymer Testing **61**, 333–340 (2017).

[109] Y. Long Deng, S. Peng Xu, H. Quan Chen, Z. Heng Liang, and C. Long Yu, *Inspection*

of extremely slight aesthetic defects in a polymeric polarizer using the edge of light between black and white stripes, Polymer Testing **65**, 169–175 (2018).

[110] Y. Deng, X. Pan, X. Wang, and X. Zhong, *Vison-Based 3D Shape Measurement System for Transparent Microdefect Characterization*, IEEE Access **7**, 105721–105733 (2019).

[111] R. E. Wijesinghe, K. Park, Y. Jung, P. Kim, M. Jeon, and J. Kim, *Industrial resin inspection for display production using automated fluid-inspection based on multimodal optical detection techniques*, Optics and Lasers in Engineering **96**, 75–82 (2017).

[112] F. P. Ferreira, P. M. Forte, P. E. R. Felgueiras, B. P. J. Bret, M. S. Belsley, and E. J. Nunes-Pereira, *Evaluating sub-pixel functional defects of a display using an arbitrary resolution camera*, Displays **49**, 40–50 (2017).

[113] D. B. Perng and S. H. Chen, *Directional textures auto-inspection using discrete cosine transform*, International Journal of Production Research **49**, 7171–7187 (2011).

[114] S. H. Chen and D. B. Perng, *Directional textures auto-inspection using principal component analysis*, International Journal of Advanced Manufacturing Technology **55**, 1099–1110 (2011).

[115] Y. Park and I. S. Kweon, *Ambiguous Surface Defect Image Classification of AMOLED Displays in Smartphones*, IEEE Transactions on Industrial Informatics **12**, 597–607 (2016).

[116] V. A. Sindagi and S. Srivastava, *Domain Adaptation for Automatic OLED Panel Defect Detection Using Adaptive Support Vector Data Description*, International Journal of Computer Vision **122**, 193–211 (2017).

[117] J. Kwak, K. B. Lee, J. Jang, K. S. Chang, and C. O. Kim, *Automatic inspection of salt-and-pepper defects in OLED panels using image processing and control chart techniques*, Journal of Intelligent Manufacturing **30**, 1047–1055 (2019).

[118] D. Li, L. Q. Liang, and W. J. Zhang, *Defect inspection and extraction of the mobile phone cover glass based on the principal components analysis*, International Journal of Advanced Manufacturing Technology **73**, 1605–1614 (2014).

[119] C. Jian, J. Gao, and Y. Ao, *Imbalanced defect classification for mobile phone screen glass using multifractal features and a new sampling method*, Multimedia Tools and Applications **76**, 24413–24434 (2017).

[120] C. Jian, J. Gao, and Y. Ao, *Automatic surface defect detection for mobile phone screen glass based on machine vision*, Applied Soft Computing Journal **52**, 348–358 (2017).

[121] Z. C. Yuan, Z. T. Zhang, H. Su, L. Zhang, F. Shen, and F. Zhang, *Vision-Based Defect Detection for Mobile Phone Cover Glass using Deep Neural Networks*, International Journal of Precision Engineering and Manufacturing **19**, 801–810 (2018).

[122] H. Jo and J. Kim, *Regularized Auto-Encoder-Based Separation of Defects from Backgrounds for Inspecting Display Devices*, Electronics **8**, 533 (2019).

[123] L. Q. Liang, D. Li, X. Fu, and W. J. Zhang, *Touch screen defect inspection based on sparse representation in low resolution images*, Multimedia Tools and Applications **75**, 2655–2666 (2016).

[124] M. Chang, B. C. Chen, J. L. Gabayno, and M. F. Chen, *Development of an optical inspection platform for surface defect detection in touch panel glass*, International Journal of Optomechatronics **10**, 63–72 (2016).

[125] C. Benedek, O. Krammer, M. Janoczki, and L. Jakab, *Solder paste scooping detection by multilevel visual inspection of printed circuit boards*, IEEE Transactions on Industrial Electronics **60**, 2318–2331 (2013).

[126] G. Acciani, G. Brunetti, and G. Fornarelli, *Application of neural networks in optical inspection and classification of solder joints in surface mount technology*, IEEE Transactions on Industrial Informatics **2**, 200–209 (2006).

[127] M. H. Annaby, Y. M. Fouda, and M. A. Rushdi, *Improved Normalized Cross-Correlation for Defect Detection in Printed-Circuit Boards*, IEEE Transactions on Semiconductor Manufacturing **32**, 199–211 (2019).

[128] S. S. Zakaria, A. Amir, N. Yaakob, and S. Nazemi, *Automated Detection of Printed Circuit*

Boards (PCB) Defects by Using Machine Learning in Electronic Manufacturing: Current Approaches, IOP Conference Series: Materials Science and Engineering **767**, 012064 (2020).

[129] W. Dai, A. Mujeeb, M. Erdt, and A. Sourin, *Soldering defect detection in automatic optical inspection*, Advanced Engineering Informatics **43**, 101004 (2020).

[130] J. D. Song, Y. G. Kim, and T. H. Park, *SMT defect classification by feature extraction region optimization and machine learning*, International Journal of Advanced Manufacturing Technology **101**, 1303–1313 (2019).

[131] D. W. Capson and S. K. Eng, *A Tiered-Color Illumination Approach for Machine Inspection of Solder Joints*, IEEE Transactions on Pattern Analysis and Machine Intelligence **10**, 387–393 (1988).

[132] R. Nagarajan, S. Yaacob, P. Pandian, M. Karthigayan, S. H. Amin, and M. Khalid, *A real time marking inspection scheme for semiconductor industries*, International Journal of Advanced Manufacturing Technology **34**, 926–932 (2007).

[133] B. C. Jiang, S. L. Tasi, and C. C. Wang, *Machine vision-based gray relational theory applied to IC marking inspection*, IEEE Transactions on Semiconductor Manufacturing **15**, 531–539 (2002).

[134] F. H. Hsu and C. A. Shen, *The Design and Implementation of an Embedded Real-Time Automated IC Marking Inspection System*, IEEE Transactions on Semiconductor Manufacturing **32**, 112–120 (2019).

[135] B. C. Jiang, C. C. Wang, and Y. N. Hsu, *Machine vision and background remover-based approach for PCB solder joints inspection*, International Journal of Production Research **45**, 451–464 (2007).

[136] H. Xie, X. Zhang, Y. Kuang, and G. Ouyang, *Solder joint inspection method for chip component using improved adaboost and decision tree*, IEEE Transactions on Components, Packaging and Manufacturing Technology **1**, 2018–2027 (2011).

[137] F. Wu and X. Zhang, *Feature-extraction-based inspection algorithm for IC solder joints*,

IEEE Transactions on Components, Packaging and Manufacturing Technology **1**, 689–694 (2011).

[138] N. S. Mar, P. K. Yarlagadda, and C. Fookes, *Design and development of automatic visual inspection system for PCB manufacturing*, Robotics and Computer-Integrated Manufacturing **27**, 949–962 (2011).

[139] W. Y. Wu, C. W. Hung, and W. B. Yu, *The development of automated solder bump inspection using machine vision techniques*, International Journal of Advanced Manufacturing Technology **69**, 509–523 (2013).

[140] F. Wu and X. Zhang, *An inspection and classification method for chip solder joints using color grads and Boolean rules*, Robotics and Computer-Integrated Manufacturing **30**, 517–526 (2014).

[141] W. Ko, K and H. S. Cho, *Solder joints inspection using a neural network and fuzzy rule-based classification method*, IEEE Transactions on Electronics Packaging Manufacturing **23**, 78 (2000).

[142] G. Acciani, G. Brunetti, and G. Fornarelli, *A Multiple Neural Network System to Classify Solder Joints on Integrated Circuits*, International Journal of Computational Intelligence Research **2**, 337–348 (2006).

[143] A. Giaquinto, G. Fornarelli, G. Brunetti, and G. Acciani, *A Neurofuzzy method for the evaluation of soldering global quality index*, IEEE Transactions on Industrial Informatics **5**, 56–66 (2009).

[144] G. Acciani, G. Fornarelli, and A. Giaquinto, *A fuzzy method for global quality index evaluation of solder joints in surface mount technology*, IEEE Transactions on Industrial Informatics **7**, 115–124 (2011).

[145] S. N. Chiu and M. H. Perng, *Reflection-area-based feature descriptor for solder joint inspection*, Machine Vision and Applications **18**, 95–106 (2007).

[146] T. Y. Ong, Z. Samad, and M. M. Ratnam, *Solder joint inspection with multi-angle imag-*

ing and an artificial neural network, International Journal of Advanced Manufacturing Technology **38**, 455–462 (2008).

[147] N. Mar, C. Fookes, and P. Yarlagadda, *Design of automatic vision-based inspection system for solder joint segmentation*, Journal of Achievements in Materials and Mainfacturing Engineering **34**, 145–151 (2009).

[148] H. Wu, X. Zhang, H. Xie, Y. Kuang, and G. Ouyang, *Classification of solder joint using feature selection based on bayes and support vector machine*, IEEE Transactions on Components, Packaging and Manufacturing Technology **3**, 516–522 (2013).

[149] H. Wu, X. Zhang, Y. Kuang, G. Ouyang, and H. Xie, *Solder joint inspection based on neural network combined with genetic algorithm*, Optik **124**, 4110–4116 (2013).

[150] C. L. Fonseka and J. A. Jayasinghe, *Implementation of an Automatic Optical Inspection System for Solder Quality Classification of THT Solder Joints*, IEEE Transactions on Components, Packaging and Manufacturing Technology **9**, 353–366 (2019).

[151] N. Cai, J. Lin, Q. Ye, H. Wang, S. Weng, and B. W. K. Ling, *A New IC Solder Joint Inspection Method for an Automatic Optical Inspection System Based on an Improved Visual Background Extraction Algorithm*, IEEE Transactions on Components, Packaging and Manufacturing Technology **6**, 161–172 (2016).

[152] N. Cai, Y. Zhou, Q. Ye, G. Liu, H. Wang, and X. Chen, *IC Solder Joint Inspection via Robust Principle Component Analysis*, IEEE Transactions on Components, Packaging and Manufacturing Technology **7**, 300–309 (2017).

[153] N. Cai, G. Cen, J. Wu, F. Li, H. Wang, and X. Chen, *SMT Solder Joint Inspection via a Novel Cascaded Convolutional Neural Network*, IEEE Transactions on Components, Packaging and Manufacturing Technology **8**, 670–677 (2018).

[154] Q. Ye, N. Cai, J. Li, F. Li, H. Wang, and X. Chen, *IC Solder Joint Inspection Based on an Adaptive-Template Method*, IEEE Transactions on Components, Packaging and Manufacturing Technology **8**, 1121–1127 (2018).

[155] T. W. Hui and G. K. H. Pang, *Solder paste inspection using region-based defect detection*, International Journal of Advanced Manufacturing Technology **42**, 725–734 (2009).

[156] J. Jiang, J. Cheng, and D. Tao, *Color biological features-based solder paste defects detection and classification on printed circuit boards*, IEEE Transactions on Components, Packaging and Manufacturing Technology **2**, 1536–1544 (2012).

[157] A. J. Crispin and V. Rankov, *Automated inspection of PCB components using a genetic algorithm template-matching approach*, International Journal of Advanced Manufacturing Technology **35**, 293–300 (2007).

[158] H. J. Cho and T. H. Park, *Wavelet transform based image template matching for automatic component inspection*, International Journal of Advanced Manufacturing Technology **50**, 1033–1039 (2010).

[159] A. A. I. Hassanin, F. E. Abd El-Samie, and G. M. El Banby, *A real-time approach for automatic defect detection from PCBs based on SURF features and morphological operations*, Multimedia Tools and Applications **78**, 34437–34457 (2019).

[160] M. E. Zervakis, S. K. Goumas, and G. A. Rovithakis, *A Bayesian Framework for Multilead SMD Post-Placement Quality Inspection*, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics **34**, 440–453 (2004).

[161] S. K. Goumas, I. N. Dimou, and M. E. Zervakis, *Combination of multiple classifiers for post-placement quality inspection of components: A comparative study*, Information Fusion **11**, 149–162 (2010).

[162] C. H. Wu, D. Z. Wang, A. Ip, D. W. Wang, C. Y. Chan, and H. F. Wang, *A particle swarm optimization approach for components placement inspection on printed circuit boards*, Journal of Intelligent Manufacturing **20**, 535–549 (2009).

[163] N. Dong, C. H. Wu, W. H. Ip, Z. Q. Chen, and K. L. Yung, *Chaotic species based particle swarm optimization algorithms and its application in PCB components detection*, Expert Systems with Applications **39**, 12501–12511 (2012).

[164] L. Bai, X. Yang, and H. Gao, *Corner Point-Based Coarse-Fine Method for Surface-Mount Component Positioning*, IEEE Transactions on Industrial Informatics **14**, 877–886 (2018).

[165] S. H. Chen and D. B. Perng, *Automatic optical inspection system for IC molding surface*, Journal of Intelligent Manufacturing **27**, 915–926 (2016).

[166] C. C. Wang, B. C. Jiang, J. Y. Lin, and C. C. Chu, *Machine vision-based defect detection in IC images using the partial information correlation coefficient*, IEEE Transactions on Semiconductor Manufacturing **26**, 378–384 (2013).

[167] W. C. Wang, S. L. Chen, L. B. Chen, and W. J. Chang, *A Machine Vision Based Automatic Optical Inspection System for Measuring Drilling Quality of Printed Circuit Boards*, IEEE Access **5**, 10817–10833 (2017).

[168] G. Duan, H. Wang, Z. Liu, and Y. W. Chen, *A machine learning-based framework for automatic visual inspection of microdrill bits in PCB production*, IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews **42**, 1679–1689 (2012).

[169] G. Duan, H. Wang, Z. Liu, J. Tan, and Y. W. Chen, *Automatic optical phase identification of micro-drill bits based on improved ASM and bag of shape segment in PCB production*, Machine Vision and Applications **25**, 1411–1422 (2014).

[170] B. C. Jiang, Y. M. Wang, and C. C. Wang, *Bootstrap sampling technique applied to the PCB golden fingers defect classification study*, International Journal of Production Research **39**, 2215–2230 (2001).

[171] D. M. Tsai and B. T. Lin, *Defect detection of gold-plated surfaces on PCBs using entropy measures*, International Journal of Advanced Manufacturing Technology **20**, 420–428 (2002).

[172] B. C. Jiang, C. C. Wang, and P. L. Chen, *Logistic regression tree applied to classify PCB golden finger defects*, International Journal of Advanced Manufacturing Technology **24**, 496–502 (2004).

[173] H. Rau and C. H. Wu, *Automatic optical inspection for detecting defects on printed circuit*

board inner layers, International Journal of Advanced Manufacturing Technology **25**, 940–946 (2005).

[174] Z. Ibrahim and S. A. R. Al-Attas, *Wavelet-based printed circuit board inspection algorithm*, Integrated Computer-Aided Engineering **12**, 201–213 (2005).

[175] P. C. Chang, L. Y. Chen, and C. Y. Fan, *A case-based evolutionary model for defect classification of printed circuit board images*, Journal of Intelligent Manufacturing **19**, 203–214 (2008).

[176] C. T. Liao, W. H. Lee, and S. H. Lai, *A flexible PCB inspection system based on statistical learning*, Journal of Signal Processing Systems **67**, 279–290 (2012).

[177] E. H. Yuk, S. H. Park, C. S. Park, and J. G. Baek, *Feature-learning-based printed circuit board inspection via speeded-up robust features and random forest*, Applied Sciences (Switzerland) **8** (2018).

[178] T. Vafeiadis, N. Dimitriou, D. Ioannidis, T. Wotherspoon, G. Tinker, and D. Tzovaras, *A framework for inspection of dies attachment on PCB utilizing machine learning techniques*, Journal of Management Analytics **5**, 81–94 (2018).

[179] S. F. Chuang, W. T. Chang, C. C. Lin, and Y. S. Tarng, *Misalignment inspection of multilayer PCBs with an automated X-ray machine vision system*, International Journal of Advanced Manufacturing Technology **51**, 995–1008 (2010).

[180] D. M. Tsai and R. H. Yang, *An eigenvalue-based similarity measure and its application in defect detection*, Image and Vision Computing **23**, 1094–1101 (2005).

[181] V. H. Gaidhane, Y. V. Hote, and V. Singh, *An efficient similarity measure approach for PCB surface defect detection*, Pattern Analysis and Applications **21**, 277–289 (2018).

[182] D. M. Tsai and C. K. Huang, *Defect Detection in Electronic Surfaces Using Template-Based Fourier Image Reconstruction*, IEEE Transactions on Components, Packaging and Manufacturing Technology **9**, 163–172 (2019).

[183] C. S. Lin and L. W. Lue, *Image system for fast positioning and accuracy inspection of ball grid array boards*, Microelectronics Reliability **41**, 119–128 (2001).

[184] C. C. Chu, B. C. Jiang, and C. C. Wang, *Modified gamma correction method to enhance ball grid array image for surface defect inspection*, International Journal of Production Research **46**, 2165–2178 (2008).

[185] Y. C. Chiou, C. S. Lin, and B. C. Chiou, *The feature extraction and analysis of flaw detection and classification in BGA gold-plating areas*, Expert Systems with Applications **35**, 1771–1779 (2008).

[186] A. F. Said, B. L. Bennett, L. J. Karam, and J. S. Pettinato, *Automated detection and classification of non-wet solder joints*, IEEE Transactions on Automation Science and Engineering **8**, 67–80 (2011).

[187] A. F. Said, B. L. Bennett, L. J. Karam, A. Siah, K. Goodman, and J. S. Pettinato, *Automated void detection in solder balls in the presence of vias and other artifacts*, IEEE Transactions on Components, Packaging and Manufacturing Technology **2**, 1890–1901 (2012).

[188] J. Li, B. L. Bennett, L. J. Karam, and J. S. Pettinato, *Stereo Vision Based Automated Solder Ball Height and Substrate Coplanarity Inspection*, IEEE Transactions on Automation Science and Engineering **13**, 757–771 (2016).

[189] W. Jin, W. Lin, X. Yang, and H. Gao, *Reference-free path-walking method for ball grid array inspection in surface mounting machines*, IEEE Transactions on Industrial Electronics **64**, 6310–6318 (2017).

[190] H. Gao, W. Jin, X. Yang, and O. Kaynak, *A Line-Based-Clustering Approach for Ball Grid Array Component Inspection in Surface-Mount Technology*, IEEE Transactions on Industrial Electronics **64**, 3030–3038 (2017).

[191] M. van Veenhuizen, *Void detection in solder bumps with deep learning*, Microelectronics Reliability **88-90**, 315–320 (2018).

[192] L. Bai, X. Yang, and H. Gao, *A Novel Coarse-Fine Method for Ball Grid Array Component Positioning and Defect Inspection*, IEEE Transactions on Industrial Electronics **65**, 5023–5031 (2018).

[193] X. Lu, Z. He, L. Su, M. Fan, F. Liu, G. Liao, and T. Shi, *Detection of micro solder balls using active thermography technology and K-means algorithm*, IEEE Transactions on Industrial Informatics **14**, 5620–5628 (2018).

[194] M. P. L. Ooi, H. K. Sok, Y. C. Kuang, S. Demidenko, and C. Chan, *Defect cluster recognition system for fabricated semiconductor wafers*, Engineering Applications of Artificial Intelligence **26**, 1029–1043 (2013).

[195] V. Perminov, V. Putrolaynen, M. Belyaev, E. Pasko, and K. Balashkov, *Automated image analysis for evaluation of wafer backside chipping*, International Journal of Advanced Manufacturing Technology **99**, 2015–2023 (2018).

[196] J. Kim, Y. Lee, and H. Kim, *Detection and clustering of mixed-type defect patterns in wafer bin maps*, IISE Transactions **50**, 99–111 (2018).

[197] C. H. Jin, H. J. Na, M. Piao, G. Pok, and K. H. Ryu, *A Novel DBSCAN-Based Defect Pattern Detection and Classification Framework for Wafer Bin Map*, IEEE Transactions on Semiconductor Manufacturing **32**, 286–292 (2019).

[198] F. Adly, O. Alhussein, P. D. Yoo, Y. Al-Hammadi, K. Taha, S. Muhaidat, Y. S. Jeong, U. Lee, and M. Ismail, *Simplified subspaced regression network for identification of defect patterns in semiconductor wafer maps*, IEEE Transactions on Industrial Informatics **11**, 1267–1276 (2015).

[199] C. W. Chang, T. M. Chao, J. T. Horng, C. F. Lu, and R. H. Yeh, *Development pattern recognition model for the classification of circuit probe wafer maps on semiconductors*, IEEE Transactions on Components, Packaging and Manufacturing Technology **2**, 2089–2097 (2012).

[200] S. C. Hsu and C. F. Chien, *Hybrid data mining approach for pattern extraction from wafer bin map to improve yield in semiconductor manufacturing*, International Journal of Production Economics **107**, 88–103 (2007).

[201] M. Saqlain, B. Jargalsaikhan, and J. Y. Lee, *A voting ensemble classifier for wafer map defect patterns identification in semiconductor manufacturing*, IEEE Transactions on Semiconductor Manufacturing **32**, 171–182 (2019).

[202] M. P. L. Ooi, E. K. J. Joo, Y. C. Kuang, S. Demidenko, L. Kleeman, and C. W. K. Chan, *Getting more from the semiconductor test: Data mining with defect-cluster extraction*, IEEE Transactions on Instrumentation and Measurement **60**, 3300–3317 (2011).

[203] J. Yu and X. Lu, *Wafer Map Defect Detection and Recognition Using Joint Local and Non-local Linear Discriminant Analysis*, IEEE Transactions on Semiconductor Manufacturing **29**, 33–43 (2016).

[204] H. W. Hsieh and F. L. Chen, *Recognition of defect spatial patterns in semiconductor fabrication*, International Journal of Production Research **42**, 4153–4172 (2004).

[205] F. L. Chen and S. F. Liu, *A neural-network approach to recognize defect spatial pattern in semiconductor fabrication*, IEEE Transactions on Semiconductor Manufacturing **13**, 366–373 (2000).

[206] S. F. Liu, F. L. Chen, and W. B. Lu, *Wafer bin map recognition using a neural network approach*, International Journal of Production Research **40**, 2207–2223 (2002).

[207] F. Di Palma, G. De Nicolao, G. Miraglia, E. Pasquinetti, and F. Piccinini, *Unsupervised spatial pattern classification of electrical-wafer-sorting maps in semiconductor manufacturing*, Pattern Recognition Letters **26**, 1857–1865 (2005).

[208] C. H. Wang, S. J. Wang, and W. D. Lee, *Automatic identification of spatial defect patterns for semiconductor manufacturing*, International Journal of Production Research **44**, 5169–5185 (2006).

[209] C. H. Wang, *Recognition of semiconductor defect patterns using spatial filtering and spectral clustering*, Expert Systems with Applications **34**, 1914–1923 (2008).

[210] C. H. Wang, W. Kuo, and H. Bensmail, *Detection and classification of defect patterns on semiconductor wafers*, IIE Transactions (Institute of Industrial Engineers) **38**, 1059–1068 (2006).

[211] C. H. Wang, *Separation of composite defect patterns on wafer bin map using support vector clustering*, Expert Systems with Applications **36**, 2554–2561 (2009).

[212] C. J. Huang, *Clustered defect detection of high quality chips using self-supervised multilayer perceptron*, Expert Systems with Applications **33**, 996–1003 (2007).

[213] C. J. Huang, Y. J. Chen, C. F. Wu, and Y. A. Huang, *Application of neural networks and genetic algorithms to the screening for high quality chips*, Applied Soft Computing Journal **9**, 824–832 (2009).

[214] Y. S. Jeong, S. J. Kim, and M. K. Jeong, *Automatic identification of defect patterns in semiconductor wafer maps using spatial correlogram and dynamic time warping*, IEEE Transactions on Semiconductor Manufacturing **21**, 625–637 (2008).

[215] K. P. White, B. Kundu, and C. M. Mastrangelo, *Classification of defect clusters on semiconductor wafers via the hough transformation*, IEEE Transactions on Semiconductor Manufacturing **21**, 272–277 (2008).

[216] T. S. Li and C. L. Huang, *Defect spatial pattern recognition using a hybrid SOM-SVM approach in semiconductor manufacturing*, Expert Systems with Applications **36**, 374–385 (2009).

[217] J. Chen, C. J. Hsu, and C. C. Chen, *A self-growing hidden Markov tree for wafer map inspection*, Journal of Process Control **19**, 261–271 (2009).

[218] J. Y. Hwang and W. Kuo, *Model-based clustering for integrated circuit yield enhancement*, European Journal of Operational Research **178**, 143–153 (2007).

[219] T. Yuan and W. Kuo, *A model-based clustering approach to the recognition of the spatial defect patterns produced during semiconductor fabrication*, IIE Transactions (Institute of Industrial Engineers) **40**, 93–101 (2008).

[220] T. Yuan and W. Kuo, *Spatial defect pattern recognition on semiconductor wafers using model-based clustering and Bayesian inference*, European Journal of Operational Research **190**, 228–240 (2008).

[221] T. Yuan, S. J. Bae, and J. I. Park, *Bayesian spatial defect pattern recognition in semiconductor fabrication using support vector clustering*, International Journal of Advanced Manufacturing Technology **51**, 671–683 (2010).

[222] T. Yuan, W. Kuo, and S. J. Bae, *Detection of Spatial Defect Patterns Generated in Semiconductor Fabrication Processes*, IEEE Transactions on Semiconductor Manufacturing **24**, 392–403 (2011).

[223] Q. Zhou, L. Zeng, and S. Zhou, *Statistical detection of defect patterns using hough transform*, IEEE Transactions on Semiconductor Manufacturing **23**, 370–380 (2010).

[224] J. S. Lin, *Constructing a yield model for integrated circuits based on a novel fuzzy variable of clustered defect pattern*, Expert Systems with Applications **39**, 2856–2864 (2012).

[225] L. C. Chao and L. I. Tong, *Wafer defect pattern recognition by multi-class support vector machines by using a novel defect cluster index*, Expert Systems with Applications **36**, 10158–10167 (2009).

[226] G. Choi, S. H. Kim, C. Ha, and S. J. Bae, *Multi-step ART1 algorithm for recognition of defect patterns on semiconductor wafers*, International Journal of Production Research **50**, 3274–3287 (2012).

[227] C. F. Chien, S. C. Hsu, and Y. J. Chen, *A system for online detection and classification of wafer bin map defect patterns for manufacturing intelligence*, International Journal of Production Research **51**, 2324–2338 (2013).

[228] C. W. Liu and C. F. Chien, *An intelligent system for wafer bin map defect diagnosis: An empirical study for semiconductor manufacturing*, Engineering Applications of Artificial Intelligence **26**, 1479–1486 (2013).

[229] F. Adly, P. D. Yoo, S. Muhaidat, Y. Al-Hammadi, U. Lee, and M. Ismail, *Randomized general regression network for identification of defect patterns in semiconductor wafer maps*, IEEE Transactions on Semiconductor Manufacturing **28**, 145–152 (2015).

[230] M. J. Wu, J. S. R. Jang, and J. L. Chen, *Wafer map failure pattern recognition and similarity ranking for large-scale data sets*, IEEE Transactions on Semiconductor Manufacturing **28**, 1–12 (2015).

[231] M. Piao, C. H. Jin, J. Y. Lee, and J. Y. Byun, *Decision tree ensemble-based wafer map*

failure pattern recognition based on radon transform-based features, IEEE Transactions on Semiconductor Manufacturing **31**, 250–257 (2018).

[232] J. Yu, X. Zheng, and J. Liu, *Stacked convolutional sparse denoising auto-encoder for identification of defect patterns in semiconductor wafer map*, Computers in Industry **109**, 121–133 (2019).

[233] N. Yu, Q. Xu, and H. Wang, *Wafer defect pattern recognition and analysis based on convolutional neural network*, IEEE Transactions on Semiconductor Manufacturing **32**, 566–573 (2019).

[234] J. Yu, *Enhanced stacked denoising autoencoder-based feature learning for recognition of wafer map defects*, IEEE Transactions on Semiconductor Manufacturing **32**, 613–624 (2019).

[235] J. Wang, Z. Yang, J. Zhang, Q. Zhang, and W. T. K. Chien, *AdaBalGAN: An Improved Generative Adversarial Network with Imbalanced Learning for Wafer Defective Pattern Recognition*, IEEE Transactions on Semiconductor Manufacturing **32**, 310–319 (2019).

[236] C. S. Liao, T. J. Hsieh, Y. S. Huang, and C. F. Chien, *Similarity searching for defective wafer bin maps in semiconductor manufacturing*, IEEE Transactions on Automation Science and Engineering **11**, 953–960 (2014).

[237] L. Xie, R. Huang, N. Gu, and Z. Cao, *A novel defect detection and identification method in optical inspection*, Neural Computing and Applications **24**, 1953–1962 (2014).

[238] S. Kang, S. Cho, D. An, and J. Rim, *Using Wafer Map Features to Better Predict Die-Level Failures in Final Test*, IEEE Transactions on Semiconductor Manufacturing **28**, 431–437 (2015).

[239] S. C. Tan, J. Watada, Z. Ibrahim, and M. Khalid, *Evolutionary fuzzy ARTMAP neural networks for classification of semiconductor defects*, IEEE Transactions on Neural Networks and Learning Systems **26**, 933–950 (2015).

[240] B. Kim, Y. S. Jeong, S. H. Tong, I. K. Chang, and M. K. Jeong, *A regularized singular*

value decomposition-based approach for failure pattern classification on fail bit map in a DRAM wafer, IEEE Transactions on Semiconductor Manufacturing **28**, 41–49 (2015).

[241] B. Kim, Y. S. Jeong, S. H. Tong, I. K. Chang, and M. K. Jeong, *Step-Down Spatial Randomness Test for Detecting Abnormalities in DRAM Wafers with Multiple Spatial Maps*, IEEE Transactions on Semiconductor Manufacturing **29**, 57–65 (2016).

[242] B. Kim, Y. S. Jeong, S. H. Tong, and M. K. Jeong, *A generalised uncertain decision tree for defect classification of multiple wafer maps*, International Journal of Production Research **0**, 1 (2019).

[243] K. Nakata, R. Orihara, Y. Mizuoka, and K. Takagi, *A Comprehensive Big-Data-Based Monitoring System for Yield Enhancement in Semiconductor Manufacturing*, IEEE Transactions on Semiconductor Manufacturing **30**, 339–344 (2017).

[244] K. Taha, K. Salah, and P. D. Yoo, *Clustering the Dominant Defective Patterns in Semiconductor Wafer Maps*, IEEE Transactions on Semiconductor Manufacturing **31**, 156–165 (2018).

[245] K. Kyeong and H. Kim, *Classification of Mixed-Type Defect Patterns in Wafer Bin Maps Using Convolutional Neural Networks*, IEEE Transactions on Semiconductor Manufacturing **31**, 395–402 (2018).

[246] M. B. Alawieh, F. Wang, and X. Li, *Identifying wafer-level systematic failure patterns via unsupervised learning*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **37**, 832–844 (2018).

[247] T. Nakazawa and D. V. Kulkarni, *Wafer map defect pattern classification and image retrieval using convolutional neural network*, IEEE Transactions on Semiconductor Manufacturing **31**, 309–314 (2018).

[248] T. Nakazawa and D. V. Kulkarni, *Anomaly detection and segmentation for wafer defect patterns using deep Convolutional Encoder-Decoder Neural Network Architectures in Semiconductor Manufacturing*, IEEE Transactions on Semiconductor Manufacturing **32**, 250–256 (2019).

[249] G. Tello, O. Y. Al-Jarrah, P. D. Yoo, Y. Al-Hammadi, S. Muhaidat, and U. Lee, *Deep-structured machine learning model for the recognition of mixed-defect patterns in semiconductor fabrication Processes*, IEEE Transactions on Semiconductor Manufacturing **31**, 315–322 (2018).

[250] R. Wang and N. Chen, *Wafer Map Defect Pattern Recognition Using Rotation-Invariant Features*, IEEE Transactions on Semiconductor Manufacturing **32**, 596–604 (2019).

[251] P. Xie and S. U. Guan, *Golden-template self-generating method for patterned wafer inspection*, Machine Vision and Applications **12**, 149–156 (2000).

[252] C. T. Su, T. Yang, and C. M. Ke, *A neural-network approach for semiconductor wafer post-sawing inspection*, IEEE Transactions on Semiconductor Manufacturing **15**, 260–266 (2002).

[253] S. U. Guan, P. Xie, and H. Li, *A golden-block-based self-refining scheme for repetitive patterned wafer inspections*, Machine Vision and Applications **13**, 314–321 (2003).

[254] C. Y. Chang, C. H. Li, J. W. Chang, and M. D. Jeng, *An unsupervised neural network approach for automatic semiconductor wafer defect inspection*, Expert Systems with Applications **36**, 950–958 (2009).

[255] H. Liu, W. Zhou, Q. Kuang, L. Cao, and B. Gao, *Defect detection of IC wafer based on two-dimension wavelet transform*, Microelectronics Journal **41**, 171–177 (2010).

[256] H. Liu, W. Zhou, Q. Kuang, C. Lei, and G. Bo, *Defect Detection of IC Wafer Based on Spectral Subtraction*, IEEE Transactions on Semiconductor Manufacturing **23**, 141–147 (2010).

[257] R. Baly and H. Hajj, *Wafer classification using support vector machines*, IEEE Transactions on Semiconductor Manufacturing **25**, 373–383 (2012).

[258] H. W. Kim and S. I. Yoo, *Defect detection using feature point matching for non-repetitive patterned images*, Pattern Analysis and Applications **17**, 415–429 (2014).

[259] P. Bourgeat, F. Meriaudeau, K. W. Tobin, and P. Gorria, *Gabor filtering for feature extraction on complex images: Application to defect detection on semiconductors*, Imaging Science Journal **54**, 200–210 (2006).

[260] R. Aghaeizadeh Zoroofi, H. Taketani, S. Tamura, Y. Sato, and K. Sekiya, *Automated inspection of IC wafer contamination*, Pattern Recognition **34**, 1307–1317 (2001).

[261] T. Kubota, P. Talekar, X. Ma, and T. S. Sudarshan, *A nondestructive automated defect detection system for silicon carbide wafers*, Machine Vision and Applications **16**, 170–176 (2005).

[262] N. G. Shankar and Z. W. Zhong, *Defect detection on semiconductor wafer surfaces*, Microelectronic Engineering **77**, 337–346 (2005).

[263] N. G. Shankar and Z. W. Zhong, *A rule-based computing approach for the segmentation of semiconductor defects*, Microelectronics Journal **37**, 500–509 (2006).

[264] N. G. Shankar and Z. W. Zhong, *Improved segmentation of semiconductor defects using area sieves*, Machine Vision and Applications **17**, 1–7 (2006).

[265] C. H. Yeh, C. H. Chen, F. C. Wu, and K. Y. Chen, *Validation and evaluation for defect-kill-rate and yield estimation models in semiconductor manufacturing*, International Journal of Production Research **45**, 829–844 (2007).

[266] H. D. Lin and D. C. Ho, *Detection of pinhole defects on chips and wafers using DCT enhancement in computer vision systems*, International Journal of Advanced Manufacturing Technology **34**, 567–583 (2007).

[267] R. Nakagaki, T. Honda, and K. Nakamae, *Automatic recognition of defect areas on a semiconductor wafer using multiple scanning electron microscope images*, Measurement Science and Technology **20** (2009).

[268] R. Nakagaki, Y. Takagi, and K. Nakamae, *Automatic recognition of circuit patterns on semiconductor wafers from multiple scanning electron microscope images*, Measurement Science and Technology **21** (2010).

[269] C. H. Yeh, F. C. Wu, W. L. Ji, and C. Y. Huang, *A wavelet-based approach in detecting visual defects on semiconductor wafer dies*, IEEE Transactions on Semiconductor Manufacturing **23**, 284–292 (2010).

[270] S. Usuki, H. Nishioka, S. Takahashi, and K. Takamasu, *Experimental verification of super-resolution optical inspection for semiconductor defect by using standing wave illumination shift*, International Journal of Advanced Manufacturing Technology **46**, 863–875 (2010).

[271] Z. Pan, L. Chen, W. Li, G. Zhang, and P. Wu, *A novel defect inspection method for semiconductor wafer based on magneto-optic imaging*, Journal of Low Temperature Physics **170**, 436–441 (2013).

[272] C. S. Chen, C. L. Huang, and C. W. Yeh, *A hybrid defect detection for in-tray semiconductor chip*, International Journal of Advanced Manufacturing Technology **65**, 43–56 (2013).

[273] C. L. Tien, Q. H. Lai, and C. S. Lin, *Development of optical automatic positioning and wafer defect detection system*, Measurement Science and Technology **27** (2016).

[274] S. N. David Chua, S. Mohamaddan, S. J. Tanjong, A. Yassin, and S. F. Lim, *Detection of Bond Pad Discolorations at Outgoing Wafer Inspections*, IEEE Transactions on Semiconductor Manufacturing **31**, 144–148 (2018).

[275] S. Cheon, H. Lee, C. O. Kim, and S. H. Lee, *Convolutional Neural Network for Wafer Surface Defect Classification and the Detection of Unknown Defect Class*, IEEE Transactions on Semiconductor Manufacturing **32**, 163–170 (2019).

[276] M. Harada, Y. Minekawa, and K. Nakamae, *Defect detection techniques robust to process variation in semiconductor inspection*, Measurement Science and Technology **30** (2019).

[277] K. W. Ko, D. H. Kim, M. Y. Kim, and J. H. Kim, *An automatic optical inspection system for inspection of cmos compact camera module assembly*, International Journal of Precision Engineering and Manufacturing **10**, 67–72 (2009).

[278] Y. J. Chen, T. H. Lin, K. H. Chang, and C. F. Chien, *Feature extraction for defect classi-*

fication and yield enhancement in color filter and micro-lens manufacturing: An empirical study, Journal of Industrial and Production Engineering **30**, 510–517 (2013).

[279] Y. J. Chen, C. Y. Fan, and K. H. Chang, *Manufacturing intelligence for reducing false alarm of defect classification by integrating similarity matching approach in CMOS image sensor manufacturing*, Computers and Industrial Engineering **99**, 465–473 (2016).

[280] Y. Liu and F. Yu, *Automatic inspection system of surface defects on optical IR-CUT filter based on machine vision*, Optics and Lasers in Engineering **55**, 243–257 (2014).

[281] C. F. Jeffrey Kuo, W. C. Lo, Y. R. Huang, H. Y. Tsai, C. L. Lee, and H. C. Wu, *Automated defect inspection system for CMOS image sensor with micro multi-layer non-spherical lens module*, Journal of Manufacturing Systems **45**, 248–259 (2017).

[282] C. F. Chang, J. L. Wu, K. J. Chen, and M. C. Hsu, *A hybrid defect detection method for compact camera lens*, Advances in Mechanical Engineering **9**, 1–19 (2017).

[283] H. D. Lin, *Computer-aided visual inspection of surface defects in ceramic capacitor chips*, Journal of Materials Processing Technology **189**, 19–25 (2007).

[284] H. D. Lin, *Automated visual inspection of ripple defects using wavelet characteristic based multivariate statistical approach*, Image and Vision Computing **25**, 1785–1801 (2007).

[285] H. D. Lin, *Tiny surface defect inspection of electronic passive components using discrete cosine transform decomposition and cumulative sum techniques*, Image and Vision Computing **26**, 603–621 (2008).

[286] Y. Yang, Z. J. Zha, M. Gao, and Z. He, *A robust vision inspection system for detecting surface defects of film capacitors*, Signal Processing **124**, 54–62 (2016).

[287] T. H. Sun, F. C. Tien, F. C. Tien, and R. J. Kuo, *Automated thermal fuse inspection using machine vision and artificial neural networks*, Journal of Intelligent Manufacturing **27**, 639–651 (2016).

[288] T. Brosnan and D. W. Sun, *Improving quality inspection of food products by computer vision - A review*, Journal of Food Engineering **61**, 3–16 (2004).

[289] M. Teena, A. Manickavasagan, A. Mothershaw, S. El Hadi, and D. S. Jayas, *Potential of Machine Vision Techniques for Detecting Fecal and Microbial Contamination of Food Products: A Review*, Food and Bioprocess Technology **6**, 1621–1634 (2013).

[290] V. Chauhan and B. Surgenor, *Fault detection and classification in automated assembly machines using machine vision*, International Journal of Advanced Manufacturing Technology **90**, 2491–2512 (2017).

[291] C. Vedang, F. Heshan, and S. Brian. *Effect of Illumination Techniques on Machine Vision Inspection for Automated Assembly Machines.* In *Proceedings of The Canadian Society for Mechanical Engineering International Congress*, number June, pages 1–6, Toronto, (2014).

[292] Z. Zeng, L. Z. Ma, and Z. Y. Zheng, *Automated extraction of PCB components based on specularity using layered illumination*, Journal of Intelligent Manufacturing **22**, 919–932 (2011).

[293] Z. Zeng, L. Ma, and M. Suwa, *Algorithm of locating PCB components based on colour distribution of solder joints*, International Journal of Advanced Manufacturing Technology **53**, 601–614 (2011).

[294] F. Wu, S. Li, X. Zhang, and W. Ye, *A design method for LEDs arrays structure illumination*, Journal of Display Technology **12**, 1177–1184 (2016).

[295] T. Y. Li, J. Z. Tsai, R. S. Chang, L. W. Ho, and C. F. Yang, *Pretest Gap mura on TFT LCDs using the optical interference pattern sensing method and neural network classification*, IEEE Transactions on Industrial Electronics **60**, 3976–3982 (2013).

[296] J. Z. Tsai, R. S. Chang, and T. Y. Li, *Detection of gap mura in TFT LCDs by the interference pattern and image sensing method*, IEEE Transactions on Instrumentation and Measurement **62**, 3087–3092 (2013).

[297] C. Yu, P. Chen, X. Zhong, X. Pan, and Y. Deng, *Saturated imaging for inspecting transparent aesthetic defects in a polymeric polarizer with black and white stripes*, Materials **11** (2018).

[298] National Instruments, *IMAQ Vision Concepts Manual* (2000).

[299] S. Shirmohammadi and A. Ferrero, *Camera as the instrument: the rising trend of vision based measurement*, IEEE Instrumentation & Measurement Magazine **17**, 41–47 (2014).

[300] E. N. Malamas, E. G. Petrakis, M. Zervakis, L. Petit, and J. D. Legat, *A survey on industrial vision systems, applications and tools*, Image and Vision Computing **21**, 171–188 (2003).

[301] S. H. Kim, J. H. Kim, and S. W. Kang, *Nondestructive defect inspection for LCDs using optical coherence tomography*, Displays **32**, 325–329 (2011).

[302] N. H. Cho, K. Park, J. Y. Kim, Y. Jung, and J. Kim, *Quantitative assessment of touch-screen panel by nondestructive inspection with three-dimensional real-time display optical coherence tomography*, Optics and Lasers in Engineering **68**, 50–57 (2015).

[303] T. K. Lin, *Automatic inspection of an LCD light-guide plate based on weighted central moments combined with neural networks*, Journal of the Chinese Institute of Engineers **36**, 890–901 (2013).

[304] J. Zhang, *China needs 'five to 10 years' to catch up in semiconductors, Peking University professor Zhou Zhiping says*, South China Morning Post (2019).

[305] U. C. Ltd, *Semiconductor wafer inspection* (2018). Accessed: Jul 2020 [Online]. Available: https://www.utechzone.com.tw/en/html/product/index.php?cid=67.

[306] S. Purandare, J. Zhu, R. Zhou, G. Popescu, A. Schwing, and L. L. Goddard, *Optical inspection of nanoscale structures using a novel machine learning based synthetic image generation algorithm*, Optics Express **27**, 17743 (2019).

[307] X. Lu, G. Liao, Z. Zha, Q. Xia, and T. Shi, *A novel approach for flip chip solder joint inspection based on pulsed phase thermography*, NDT and E International **44**, 484–489 (2011).

[308] Y. J. Roh, W. S. Park, and H. Cho, *Correcting image distortion in the X-ray digital tomosynthesis system for PCB solder joint inspection*, Image and Vision Computing **21**, 1063–1075 (2003).

[309] Y. Wang, M. Wang, and Z. Zhang, *Microfocus X-ray printed circuit board inspection system*, Optik **125**, 4929–4931 (2014).

[310] H. Uğuz, *A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm*, Knowledge-Based Systems **24**, 1024–1032 (2011).

[311] S. U. Lee, S. Yoon Chung, and R. H. Park, *A comparative performance study of several global thresholding techniques for segmentation*, Computer Vision, Graphics and Image Processing **52**, 171–190 (1990).

[312] K. S. Fu and J. K. Mui, *A survey on image segmentation*, Pattern Recognition **13**, 3–16 (1981).

[313] H. F. Ng, *Automatic thresholding for defect detection*, Pattern Recognition Letters **27**, 1644–1649 (2006).

[314] F. Yan, H. Zhang, and C. R. Kube, *A multistage adaptive thresholding method*, Pattern Recognition Letters **26**, 1183–1191 (2005).

[315] N. Otsu, *A Threshold Selection Method from Gray-Level Histograms*, IEEE Transactions on Systems, Man, and Cybernetics **9**, 62–66 (1979).

[316] A. K. Jain, R. P. Duin, and J. Mao, *Statistical pattern recognition: A review*, IEEE Transactions on Pattern Analysis and Machine Intelligence **22**, 4–37 (2000).

[317] N. Dong, C. H. Wu, W. H. Ip, Z. Q. Chen, C. Y. Chan, and K. L. Yung, *An improved species based genetic algorithm and its application in multiple template matching for embroidered pattern inspection*, Expert Systems with Applications **38**, 15172–15182 (2011).

[318] M. Arebey, M. A. Hannan, R. A. Begum, and H. Basri, *Solid waste bin level detection using gray level co-occurrence matrix feature extraction approach*, Journal of Environmental Management **104**, 9–18 (2012).

[319] C. E. Honeycutt and R. Plotnick, *Image analysis techniques and gray-level co-occurrence matrices (GLCM) for calculating bioturbation indices and characterizing biogenic sedimentary structures*, Computers and Geosciences **34**, 1461–1472 (2008).

[320] T. Wuest, D. Weimer, C. Irgens, and K. D. Thoben, *Machine learning in manufacturing: Advantages, challenges, and applications*, Production and Manufacturing Research **4**, 23–45 (2016).

[321] D. H. Kim, T. J. Kim, X. Wang, M. Kim, Y. J. Quan, J. W. Oh, S. H. Min, H. Kim, B. Bhandari, I. Yang, and S. H. Ahn, *Smart Machining Process Using Machine Learning: A Review and Perspective on Machining Industry*, International Journal of Precision Engineering and Manufacturing - Green Technology **5**, 555–568 (2018).

[322] Z. Ge, Z. Song, S. X. Ding, and B. Huang, *Data Mining and Analytics in the Process Industry: The Role of Machine Learning*, IEEE Access **5**, 20590–20616 (2017).

[323] Y. Freund and M. Llew. *The Alternating Decision Tree Learning Algorithm.* In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 124–133. Morgan Kaufmann Publishers Inc., (1999).

[324] E. Osuna, R. Freund, and F. Girosit. *Training support vector machines: an application to face detection.* In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 130–136. IEEE Comput. Soc, (1997).

[325] Chih-Wei Hsu and Chih-Jen Lin, *A comparison of methods for multiclass support vector machines*, IEEE Transactions on Neural Networks **13**, 415–425 (2002).

[326] Y. J. Cha, K. You, and W. Choi, *Vision-based detection of loosened bolts using the Hough transform and support vector machines*, Automation in Construction **71**, 181–188 (2016).

[327] G. Cavallaro, M. Riedel, M. Richerzhagen, J. A. Benediktsson, and A. Plaza, *On Understanding Big Data Impacts in Remotely Sensed Image Classification Using Support Vector Machine Methods*, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing **8**, 4634–4646 (2015).

[328] M. Alper Selver, O. Akay, F. Alim, S. Bardakçı, and M. Ölmez, *An automated industrial conveyor belt system using image processing and hierarchical clustering for classifying marble slabs*, Robotics and Computer-Integrated Manufacturing **27**, 164–176 (2011).

[329] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of Massive Datasets*, Cambridge University Press (2020).

[330] G. Gan, C. Ma, and J. Wu, *Data Clustering: Theory, Algorithms, and Applications*, Society for Industrial and Applied Mathematics, Philadelphiaa, Pennsylvania (2007).

[331] M.-S. Chen, P. S. Yu, and B. Liu, editors, *Advances in Knowledge Discovery and Data Mining*, Springer Berlin Heidelberg, Berlin, H. (2002).

[332] F. Petitjean, A. Ketterlin, and P. Gançarski, *A global averaging method for dynamic time warping, with applications to clustering*, Pattern Recognition **44**, 678–693 (2011).

[333] Y. Liu and S. Zhou, *Detecting point pattern of multiple line segments using hough transformation*, IEEE Transactions on Semiconductor Manufacturing **28**, 13–24 (2015).

[334] D. Birant and A. Kut, *ST-DBSCAN: An algorithm for clustering spatial-temporal data*, Data and Knowledge Engineering **60**, 208–221 (2007).

[335] M. R. Meireles, P. E. Almeida, and M. G. Simões, *A comprehensive review for industrial applicability of artificial neural networks*, IEEE Transactions on Industrial Electronics **50**, 585–601 (2003).

[336] T. Kohonen. *Improved versions of learning vector quantization.* In *1990 IJCNN International Joint Conference on Neural Networks*, pages 545–550 vol.1. IEEE, (1990).

[337] A. J. Al-Mahasneh, S. G. Anavatti, and M. A. Garratt, *Review of Applications of Generalized Regression Neural Networks in Identification and Control of Dynamic Systems*, (2018).

[338] A. Jain, Jianchang Mao, and K. Mohiuddin, *Artificial neural networks: a tutorial*, Computer **29**, 31–44 (1996).

[339] A. Shrestha and A. Mahmood, *Review of deep learning algorithms and architectures*, IEEE Access **7**, 53040–53065 (2019).

[340] X. Liu, X. Zhang, W. Zhang, W. Wei, Y. Zhao, Y. Xi, and S. Cao. *Data Mining from Haier Air-Conditioner Equipment Running Data for Fault Prediction.* In *2018 IEEE SmartWorld,*

Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), pages 1829–1836. IEEE, (2018).

[341] H. Lee, Y. Kim, and C. O. Kim, *A Deep Learning Model for Robust Wafer Fault Monitoring With Sensor Measurement Noise*, IEEE Transactions on Semiconductor Manufacturing **30**, 23–31 (2017).

[342] J. Wang, P. Fu, and R. X. Gao, *Machine vision intelligence for product defect inspection based on deep learning and Hough transform*, Journal of Manufacturing Systems **51**, 52–60 (2019).

[343] B. Devika and N. George. *Convolutional Neural Network for Semiconductor Wafer Defect Detection.* In *2019 10th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2019*, pages 1–6. IEEE, (2019).

[344] M. Saqlain, Q. Abbas, and J. Y. Lee, *A Deep Convolutional Neural Network for Wafer Defect Identification on an Imbalanced Dataset in Semiconductor Manufacturing Processes*, IEEE Transactions on Semiconductor Manufacturing , 1–1 (2020).

[345] J. Long, E. Shelhamer, and T. Darrell. *Fully Convolutional Networks for Semantic Segmentation.* In *arXiv:1411.4038. [Online]. Available: http://arxiv.org/abs/1411.4038*, (2015).

[346] V. Badrinarayanan, A. Kendall, and R. Cipolla, *SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence **39**, 2481–2495 (2017).

[347] O. Ronneberger, P. Fischer, and T. Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation.* In *arXiv:1505.04597. [Online]. Available: http://arxiv.org/abs/1505.04597*, (2015).

[348] T. H. Kim, T. H. Cho, Y. S. Moon, and S. H. Park, *Visual inspection system for the classification of solder joints*, Pattern Recognition **32**, 565–575 (1999).

[349] Horng-Hai Loh and Ming-Sing Lu, *Printed circuit board inspection using image analysis*, IEEE Transactions on Industry Applications **35**, 426–432 (1999).

[350] M. Li and B. Yuan, *2D-LDA: A statistical linear discriminant analysis for image matrix*, Pattern Recognition Letters **26**, 527–532 (2005).

[351] X. Chen, N. Liu, B. You, and B. Xiao, *A novel method for surface defect inspection of optic cable with short-wave infrared illuminance*, Infrared Physics & Technology **77**, 456–463 (2016).

[352] C. L. S. C. Fonseka and J. A. K. S. Jayasinghe, *Color Model Analysis for Solder Pad Segmentationon Printed Circuit Boards*, International Journal of Scientific and Research Publications (IJSRP) **6**, 212–225 (2016).

[353] C. L. Fonseka and J. A. Jayasinghe, *Localization of component lead inside a THT solder joint for solder defects classification*, Journal of Achievements in Materials and Manufacturing Engineering **83**, 57–66 (2017).

[354] C. L. Fonseka and J. A. Jayasinghe, *Feature extraction and template matching algorithm classification for PCB fiducial verification*, Journal of Achievements in Materials and Manufacturing Engineering **86**, 14–32 (2018).

[355] W. C. Lee and C. H. Chen, *A fast template matching method with rotation invariance by combining the circular projection transform process and bounded partial correlation*, IEEE Signal Processing Letters **19**, 737–740 (2012).

[356] B. Zhang, H. Yang, and Z. Yin, *A Region-Based Normalized Cross Correlation Algorithm for the Vision-Based Positioning of Elongated IC Chips*, IEEE Transactions on Semiconductor Manufacturing **28**, 345–352 (2015).

[357] D. M. Tsai and S. C. Lai, *Defect detection in periodically patterned surfaces using independent component analysis*, Pattern Recognition **41**, 2812–2832 (2008).

[358] J. Choi and C. Kim. *Unsupervised detection of surface defects: A two-step approach.* In *2012 19th IEEE International Conference on Image Processing*, pages 1037–1040, (2012).

[359] V. A. Sindagi and S. Srivastava. *OLED panel defect detection using local inlier-outlier ratios and modified LBP*. In *2015 14th IAPR International Conference on Machine Vision Applications (MVA)*, pages 214–217, (2015).

[360] F. Tajeripour, E. Kabir, and A. Sheikhi, *Fabric Defect Detection Using Modified Local Binary Patterns*, EURASIP Journal on Advances in Signal Processing **2008** (2007).

[361] C. F. J. Kuo, C. Y. Lai, C. H. Kao, and C. H. Chiu, *Integrating image processing and classification technology into automated polarizing film defect inspection*, Optics and Lasers in Engineering **104**, 204–219 (2018).

[362] J. McCarthy. *What is Artificial Intelligence?*, (2007).

[363] F. Rosenblatt, *The perceptron: A probabilistic model for information storage and organization in the brain.*, Psychological Review **65**, 386–408 (1958).

[364] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning representations by back-propagating errors*, Nature **323**, 533–536 (1986).

[365] M. Nielsen, *Neural Networks and Deep Learning*, Determination Press, 1 edition (2015).

[366] J. Heaton. *The Number of Hidden Layers*, (2017).

[367] A. Cotter, O. Shamir, N. Srebro, and K. Sridharan. *Better Mini-Batch Algorithms via Accelerated Gradient Methods*. In *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., (2011).

[368] S. Ioffe and C. Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 448–456. JMLR.org, (2015).

[369] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press (2016). http://www.deeplearningbook.org.

[370] J. Brownlee, *Better Deep Learning: Train Faster, Reduce Overfitting, and Make Better Predictions*, Machine Learning Mastery, 1 edition (2019).

[371] M. Negnevitsky, *Artificial Intelligence: A Guide to Intelligent Systems*, Pearson Education UK, 2 edition (2005).

[372] Y. Tian and Y. Zhang, *A comprehensive survey on regularization strategies in machine learning*, Information Fusion **80**, 146–166 (2021).

[373] S. Cai, Y. Shu, G. Chen, B. C. Ooi, W. Wang, and M. Zhang. *Effective and Efficient Dropout for Deep Convolutional Neural Networks*, (2020).

[374] G. Douzas, F. Bacao, and F. Last, *Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE*, Information Sciences **465**, 1–20 (2018).

[375] H. He and E. Garcia, *Learning from Imbalanced Data*, IEEE Transactions on Knowledge and Data Engineering **21**, 1263–1284 (2009).

[376] J. Chen and X. Ran, *Deep Learning With Edge Computing: A Review*, Proceedings of the IEEE **107**, 1655–1674 (2019).

[377] Y. J. Cha, W. Choi, and O. Büyüköztürk, *Deep Learning-Based Crack Damage Detection Using Convolutional Neural Networks*, Computer-Aided Civil and Infrastructure Engineering **32**, 361–378 (2017).

[378] J. Ker, L. Wang, J. Rao, and T. Lim, *Deep Learning Applications in Medical Image Analysis*, IEEE Access **6**, 9375–9379 (2017).

[379] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, Proceedings of the IEEE **86**, 2278–2324 (1998).

[380] A. Krizhevsky, I. Sutskever, and G. E. Hinton. *ImageNet Classification with Deep Convolutional Neural Networks.* In *Advances in Neural Information Processing Systems 25*, pages 1097–1105, (2012).

[381] K. Simonyan and A. Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition.* In *International Conference on Learning Representations (ICLR)*, (2015).

[382] K. He, X. Zhang, S. Ren, and J. Sun. *Deep Residual Learning for Image Recognition.* In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, (2016).

[383] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. *Densely connected convolutional networks.* In *arXiv:1608.06993. [Online]. Available: http://arxiv.org/abs/1608.06993*, (2016).

[384] J. K. Park, B. K. Kwon, J. H. Park, and D. J. Kang, *Machine learning-based imaging system for surface defect inspection*, International Journal of Precision Engineering and Manufacturing - Green Technology **3**, 303–310 (2016).

[385] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, *A survey of the recent architectures of deep convolutional neural networks*, Artificial Intelligence Review **53**, 5455–5516 (2020).

[386] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, *Review of deep learning: concepts, CNN architectures, challenges, applications, future directions*, Journal of Big Data **8** (2021).

[387] M. D. Zeiler and R. Fergus. *Visualizing and Understanding Convolutional Networks.* In *European conference on computer vision*, pages 818–833. Springer International Publishing, (2014).

[388] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. *Going deeper with convolutions.* In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, (2015).

[389] R. Karim. *Illustrated: 10 CNN Architectures*, (2019). Accessed online: `https://towardsdatascience.com/illustrated-10-cnn-architectures-95d78ace614d`.

[390] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. *Rethinking the Inception Architecture for Computer Vision.* In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, (2016).

[391] F. Chollet. *Xception: Deep Learning with Depthwise Separable Convolutions.* In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1800–1807, (2017).

[392] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. *Aggregated Residual Transformations*

for Deep Neural Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, (2017).

[393] J. Chow, Z. Su, J. Wu, P. Tan, X. Mao, and Y. Wang, *Anomaly detection of defects on concrete structures with the convolutional autoencoder*, Advanced Engineering Informatics **45**, 101105 (2020).

[394] L. Wang, D. Zhang, J. Guo, and Y. Han, *Image Anomaly Detection Using Normal Data Only by Latent Space Resampling*, Applied Sciences **10**, 8660 (2020).

[395] C. Zhou and R. C. Paffenroth. *Anomaly Detection with Robust Deep Autoencoders.* In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 665–674, New York, NY, USA, (2017). Association for Computing Machinery.

[396] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. *Generative Adversarial Nets.* In *Advances in Neural Information Processing Systems 27*, pages 2672–2680, (2014).

[397] S. Venu and S. Ravula, *Evaluation of Deep Convolutional Generative Adversarial Networks for Data Augmentation of Chest X-ray Images*, Future Internet **13**, 8 (2020).

[398] A. Radford, L. Metz, and S. Chintala, *Unsupervised representation learning with deep convolutional generative adversarial networks*, arXiv **arXiv:1511.06434** (2015).

[399] A. Punjabi, J. Schmid, and A. Katsaggelos, *Examining the Benefits of Capsule Neural Networks*, CoRR **abs/2001.10964** (2020).

[400] F. Deng, S. Pu, X. Chen, Y. Shi, T. Yuan, and S. Pu, *Hyperspectral Image Classification with Capsule Network Using Limited Training Samples*, Sensors **18** (2018).

[401] S. Sabour, N. Frosst, and G. E. Hinton. *Dynamic Routing between Capsules.* In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 3859–3869, Red Hook, NY, USA, (2017). Curran Associates Inc.

[402] H. Wang, K. Shao, and X. Huo, *An improved CapsNet applied to recognition of 3D vertebral images*, Applied Intelligence **50**, 3276–3290 (2020).

[403] M. Patrick, A. Adekoya, A. Mighty, and B. Edward, *Capsule Networks - A survey*, Journal of King Saud University - Computer and Information Sciences (2019).

[404] Z. Zhu, G. Peng, Y. Chen, and H. Gao, *A convolutional neural network based on a capsule network with strong generalization for bearing fault diagnosis*, Neurocomputing **323**, 62–75 (2019).

[405] H. Kahng and S. B. Kim, *Self-Supervised Representation Learning for Wafer Bin Map Defect Pattern Classification*, IEEE Trans. Semicond. Manuf. **34**, 74–86 (2020).

[406] Y. Zhou, Q. Zhang, and V. P. Singh, *An adaptive multilevel correlation analysis: a new algorithm and case study*, Hydrological Sciences Journal **61**, 2718–2728 (2016).

[407] R. Meyes, M. Lu, C. W. de Puiseau, and T. Meisen. *Ablation Studies in Artificial Neural Networks*, (2019).

[408] M. Hashemi, *Enlarging smaller images before inputting into convolutional neural network: zero-padding vs. interpolation*, Journal of Big Data **6** (2019).

[409] J. J. Luke, R. Joseph, and M. Balaji, *Impact of Image Size on Accuracy and Generalization of Convolutional Neural Networks*, International Journal of Research and Analytical Reviews (IJAR) **6**, 70–80 (2019).

[410] S. S. Basha, S. R. Dubey, V. Pulabaigari, and S. Mukherjee, *Impact of fully connected layers on performance of convolutional neural networks for image classification*, Neurocomputing **378**, 112–119 (2020).

[411] O. M. Khanday and S. Dadvandipour, *Convolutional Neural Networks and Impact of Filter Sizes on Image Classification*, Multidiszciplináris Tudományok **10**, 55–60 (2020).

[412] A. Jaiswal, W. AbdAlmageed, and P. Natarajan. *CapsuleGAN: Generative Adversarial Capsule Network.* In *Proceedings of the European Conference on Computer Vision (ECCV)*, (2018).

[413] M. Razeghi, *High-Performance InP-Based Mid-IR Quantum Cascade Lasers*, IEEE Journal of Selected Topics in Quantum Electronics **15**, 941–951 (2009).

[414] J. Faist, F. Capasso, D. L. Sivco, C. Sirtori, A. L. Hutchinson, and A. Y. Cho, *Quantum Cascade Laser*, Science **264**, 553–556 (1994).

[415] P. Figueiredo, M. Suttinger, R. Go, E. Tsvid, C. K. N. Patel, and A. Lyakh, *Progress in high-power continuous-wave quantum cascade lasers*, Applied Optics **56**, H15–H23 (2017).

[416] R. F. Curl, F. Capasso, C. Gmachl, A. A. Kosterev, B. McManus, R. Lewicki, M. Pusharsky, G. Wysocki, and F. K. Tittel, *Quantum cascade lasers in chemical physics*, Chemical Physics Letters **487**, 1–18 (2010).

[417] N. Villa, G. Strübi, T. Gresch, J. Butet, S. Blaser, and A. Müller, *Quantum cascade lasers with discrete and non equidistant extended tuning tailored by simulated annealing*, Optics Express **27**, 26701 (2019).

[418] R. Pecharroman-Gallego, *An Overview on Quantum Cascade Lasers: Origins and Development*, Quantum Cascade Lasers (2017).

[419] I. Edinbarough, R. Balderas, and S. Bose, *A vision and robot based on-line inspection monitoring system for electronic manufacturing*, Computers in Industry **56**, 986–996 (2005).

[420] S. A. Singh and S. Majumder. *Chapter one - Short and noisy electrocardiogram classification based on deep learning.* In *Deep Learning for Data Analytics*, pages 1–19. Academic Press, (2020).

[421] Y. Li, H. Wang, J. Yang, K. Wang, and G. Qi, *A non-intrusive load monitoring algorithm based on multiple features and decision fusion*, Energy Reports **7**, 1555–1562 (2021).

[422] J. E. Arco, J. Ramírez, J. M. Górriz, and M. Ruz, *Data fusion based on Searchlight analysis for the prediction of Alzheimer's disease*, Expert Systems with Applications **185**, 115549 (2021).

[423] T. T. Nguyen, M. T. Dang, V. A. Baghel, A. V. Luong, J. McCall, and A. W.-C. Liew, *Evolving interval-based representation for multiple classifier fusion*, Knowledge-Based Systems **201-202**, 106034 (2020).

[424] B. Khaleghi, A. Khamis, F. O. Karray, and S. N. Razavi, *Multisensor data fusion: A review of the state-of-the-art*, Information Fusion **14**, 28–44 (2013).

[425] Y. Chandola, J. Virmani, H. Bhadauria, and P. Kumar. *Chapter 4 - End-to-end pre-trained CNN-based computer-aided classification system design for chest radiographs.* In *Deep Learning for Chest Radiographs*, Primers in Biomedical Imaging Devices and Systems, pages 117–140. Academic Press, (2021).

[426] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, V. Makarenkov, and S. Nahavandi, *A review of uncertainty quantification in deep learning: Techniques, applications and challenges*, Information Fusion **76**, 243–297 (2021).

[427] K. Wickstrøm, M. Kampffmeyer, and R. Jenssen, *Uncertainty and interpretability in convolutional neural networks for semantic segmentation of colorectal polyps*, Medical Image Analysis **60**, 101619 (2020).