



**MASK R-CNN FOR REAL-TIME PARTS CLASSIFICATION ON AN ENAMEL
PAINT COATING LINE.**

by

Tarik Citlak
(21337052)

Submitted in fulfilment of the requirements for the degree of:

MASTER OF ENGINEERING

in the

DEPARTMENT OF ELECTRONIC AND COMPUTER ENGINEERING,

FACULTY OF ENGINEERING AND THE BUILT ENVIRONMENT

at the

DURBAN UNIVERSITY OF TECHNOLOGY

August 2025

Supervisor: Prof. N. Pillay

Co-Supervisor: Dr. N. Singh

PREFACE

I, the undersigned, Mr Tarik Citlak at the Durban University of Technology, declare that the work included in this thesis, “Mask R-CNN For Real-Time Parts Classification on an Enamel Paint Coating Line.”, forms the undertaking which I made concerning the research work carried out under the direction of Nelendran Pillay and Navin Singh at the Durban University of Technology. The motivation behind this research was to implement a deep learning model capable of delivering reliable data in an industrial setting. The current problem faced at the firm is related to manual data collection. The primary objective of this research would be to automate the manual process using the Mask R-CNN model as the tool. The thesis lays out the introduction, methodology, results and analysis of the use of the model. I hope that this thesis offers valuable insight into the application of deep learning models within an industrial environment and sparks further curiosity and innovation in this field. I invite the reader to delve into this work as we uncover the possibilities that these technologies hold for the future of manufacturing and automation.

Tarik Citlak

Durban University of Technology

August 2025

DECLARATION 1: SUPERVISOR

According to the contents of this thesis, as the candidates' Supervisor, I agree to the submission of the thesis.

Prof. N. Pillay

(Main supervisor)

Date: 21-08-2025

Dr. N. Singh

(Co-supervisor)

Date: 21-08-2025

DECLARATION 2: PLAGIARISM

I, **Tarik Citlak (21337052)**, the undersigned, declare that:

- (i) The research reported in this thesis, except where otherwise indicated, is my original work.
- (ii) This thesis has not been submitted for degree or examination at any other university.
- (iii) This thesis does not contain other persons' data, images, graphs, or other information unless specifically acknowledged as being sourced from other persons.
- (iv) This thesis does not contain other persons' writing unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:
 - a. their words have been re-written, but the general information attributed to them has been referenced.
 - b. where their exact words have been used, their writing has been placed inside quotation marks, and referenced.
- (v) Where I have reproduced a publication of which I am an author, co-author or editor, I have indicated in detail which part of the publication was actually written by myself alone and have fully referenced such publications.
- (vi) This thesis does not contain text, graphics, or tables copied and pasted from the Internet unless specifically acknowledged and the source is detailed in the dissertation and the Reference sections.

Tarik Citlak

August 2025

DECLARATION 3: PUBLICATIONS

I, **Tarik Citlak (21337052)** declare that the following publications came of this thesis.

1. T. Citlak and N. Pillay, “Real-Time Object Classification on an Enamel Paint Coating Conveyor Line Using Mask R-CNN,” *Automation*, vol. 5, pp. 213–229, Jun. 2024, doi: 10.3390/automation5030013.
2. T. Citlak, N. Pillay, and N. Singh, “Real-Time Industrial Visual Inspection,” *Sensors*, under review, 2025.

ACKNOWLEDGEMENTS

This thesis would not have been possible without the support, guidance and encouragement of many individuals and I would like to take this opportunity to express my gratitude to each one of them.

First and foremost, I would like to express my deepest gratitude to my supervisor Nelendran Pillay. Your support and insightful feedback have been invaluable in this research journey. Your expertise and patience have guided me through each step of this research.

I would like to express my gratitude to Curtis Louw for providing me with the opportunity and resources to advance this research.

To my Wife, Vilencia, I want to express my gratitude for your patience, support, and encouragement. Thank you for continuously motivating me to push through with the research.

ABSTRACT

An enamel paint coating line has relied heavily on human intervention and observation to determine the total number of parts produced in one shift. This was done by manually counting the parts as they were taken off the conveyor line which has led to inaccurate or lately received production performance data. With a manufacturing process that has a changing number of classes and a variation of the part's physical orientation within the process, the importance of investigating an object detection model was realized.

The rising demand to attain live production data has added more importance to monitoring and reporting within the industrial automation sector. Real-time parts screening requiring human intervention for data input may not be a feasible solution within a fast-moving consumer goods manufacturing facility. This has led to the proposed study using Mask Region-based Convolutional Neural Network (R-CNN) to detect objects within an image.

The state of research surrounding object detection has increased rapidly over the past decade with several models being developed and adapted for industrial automation applications. However, there has been limited research regarding models that can perform instance segmentation to provide production performance data. This has prompted the study within this field.

The objective of the study is to classify and accurately report on manufactured parts identified on an enamel paint-coating conveyor line. At any given instant, the parts may not be in the exact coordinates within the desired area of interest and the classes of objects may vary based on changing production requirements. To mitigate these challenges, this study proposes the use of a trained Mask R-CNN model to detect the objects and their associated class. Images are acquired using a high-definition camera fixed to a position next to the enamel coating line.

This study compared the average precision based on changes made to the learning rate and the intersection over union (IoU) thresholds of the model. The outcome was analysed using the precision-recall curve and the confusion matrix to determine the highest average precision and overall accuracy. The highest achieved average precision obtained from the

model was 98.27% with an overall accuracy of 98.24% using real-time captured images of the manufactured parts. The results satisfied the acceptable standard for the average precision of 97.5% set by the plant production quality engineers. Future research would include the use of pixel-wise segmentation generated from the mask branch in determining the objects' exact location, this would aid in efficient robotic spray positioning.

TABLE OF CONTENTS

PREFACE.....	I
DECLARATION 1: SUPERVISOR	II
DECLARATION 2: PLAGIARISM	III
DECLARATION 3: PUBLICATIONS.....	IV
ACKNOWLEDGEMENTS	V
ABSTRACT	VI
LIST OF FIGURES	XII
LIST OF TABLES	XV
LIST OF ABBREVIATIONS	XVI
 CHAPTER 1 INTRODUCTION	 1
1.1 INTRODUCTION.....	1
1.2 RESEARCH PROBLEM STATEMENT	6
1.3 RESEARCH QUESTION	7
1.4 RESEARCH AIMS AND OBJECTIVES.....	7
1.5 STUDY LIMITATIONS.....	9
1.6 RESEARCH CONTRIBUTIONS.....	9
1.7 STRUCTURE OF THE THESIS	10
 CHAPTER 2 LITERATURE REVIEW	 11
2.1 INTRODUCTION.....	11
2.2 TRADITIONAL METHODS FOR PRODUCTION DATA RETRIEVAL.....	11
2.3 CONVOLUTIONAL NEURAL NETWORKS FOR COMPUTER VISION	12
2.3.1 Convolutional Layer	12
2.3.2 Pooling Layer.....	13
2.3.3 Fully Connected Layer.....	14
2.4 ARCHITECTURE OF CONVOLUTIONAL NEURAL NETWORKS	15
2.4.1 LeNet-5	15
2.4.2 AlexNet	16
2.4.3 VGG Net	17

2.4.4	ResNet.....	17
2.5	HISTORICAL CONTEXT OF OBJECT DETECTION	19
2.5.1	R-CNN	19
2.5.2	Fast R-CNN	19
2.5.3	Faster R-CNN	20
2.5.4	SSD	20
2.5.5	YOLO	21
2.5.6	Object Detection Impact and Application.....	21
2.6	INTRODUCTION TO MASK R-CNN	26
2.6.1	Mask R-CNN Architecture	26
2.6.2	Region Proposal Network (RPN).....	27
2.6.3	Feature Pyramid Network (FPN).....	28
2.6.4	ROIAlign.....	29
2.6.5	Mask Branch	30
2.7	APPLICATION OF MASK R-CNN.....	30
2.7.1	Case Studies	30
2.8	PERFORMANCE EVALUATION	37
2.8.1	Performance Criteria	38
2.8.2	Benchmarking and Datasets.....	39
2.8.3	Model Comparisons	40
2.9	CHALLENGES AND LIMITATIONS	45
2.10	SUMMARY	46
CHAPTER 3	METHODOLOGY	47
3.1	INTRODUCTION.....	47
3.2	RESEARCH DESIGN	48
3.3	RESEARCH METHOD	50
3.3.1	Hardware and Software.....	51
3.3.2	Data Collection Method.....	54
3.3.3	VIA Annotation Tool.....	55
3.3.4	Field Setup	56
3.4	DATA PREPROCESSING	58
3.5	FUNCTION SELECTION	61

3.5.1	Learning Rate.....	61
3.5.2	Learning Momentum	62
3.5.3	Activation Function	63
3.5.4	Weight Decay Regularization.....	63
3.5.5	Images per GPUs	64
3.5.6	Epochs.....	64
3.5.7	Steps per Epochs	65
3.5.8	Validation Steps	66
3.5.9	Backbone.....	66
3.5.10	Backbone Strides	67
3.5.11	ROI Positive Ratio	67
3.5.12	Detection Minimum Confidence.....	68
3.5.13	RPN NMS Thresholds	69
3.5.14	Loss Function.....	69
3.6	TRAINING METHODOLOGY	70
3.6.1	Experimental Training Setup	70
3.6.2	Types of Datasets.....	71
3.6.3	Practical Methodology	73
3.7	EVALUATION METRICS.....	75
3.7.1	Losses.....	76
3.7.2	Average Recall.....	77
3.7.3	Average Precision	78
3.7.4	Precision Recall Curve.....	79
3.7.5	Mean Average Precision	79
3.7.6	F1 Score	80
3.7.7	Overall Accuracy	80
3.7.8	Confusion Matrix	81
3.8	EXPERIMENTAL ENVIRONMENT	82
3.9	LIMITATIONS	83
3.10	SUMMARY	85
CHAPTER 4	RESULTS.....	87
4.1	INTRODUCTION.....	87

4.2 DATASET ANOMALIES	87
4.3 PERFORMANCE EVALUATION	91
4.4 OVERALL ACCURACY	94
4.5 COMPARATIVE ANALYSIS	105
CHAPTER 5 ANALYSIS AND CONCLUSION	109
5.1 INTRODUCTION.....	109
5.2 SUMMARY OF RESEARCH OBJECTIVES AND METHODS.....	109
5.3 KEY FINDINGS AND ANALYSIS.....	110
5.4 CONTRIBUTIONS OF FINDINGS	111
5.5 CONCLUSIONS AND FUTURE RESEARCH RECOMMENDATIONS	112
CHAPTER 6 REFERENCES	115

LIST OF FIGURES

Figure 1.1. Model of enamel coating spray booth.....	5
Figure 1.2. Model of enamel coating spray booth with proposed camera setup.	8
Figure 2.1. CNN layers.....	12
Figure 2.2. 2-D convolution.	13
Figure 2.3. Average pooling operation on a 5 x 5 input.....	14
Figure 2.4. Fully connected layer.	15
Figure 2.5. LeNet-5 architecture [56].....	16
Figure 2.6. AlexNet architecture [15].....	16
Figure 2.7. VGGNet architecture [57].....	17
Figure 2.8. Training error and test error on CIFAR-10 with a plain network [58].	18
Figure 2.9. Residual learning building block.	18
Figure 2.10. R-CNN algorithm framework [16].....	19
Figure 2.11. Fast R-CNN algorithm framework [17].....	20
Figure 2.12. Faster R-CNN algorithm framework [18].....	20
Figure 2.13. SSD algorithm framework.[62].....	21
Figure 2.14. YOLO algorithm framework.[19].....	21
Figure 2.15. Mask R-CNN algorithm framework for parts detection [20].....	27
Figure 2.16. RPN structure [84].	28
Figure 2.17. Feature Pyramid Network [90].....	29
Figure 2.18. ROIAlign [20].	29
Figure 3.1. Object detection framework.....	51
Figure 3.2. Global shutter type.	52
Figure 3.3. Hardware configuration of field devices.....	53
Figure 3.4. VIA annotation tool.	56
Figure 3.5. Field Setup.	57
Figure 3.6. Classes used in the dataset with the alphabets representing the classes.	57
Figure 3.7. Random set of images from the dataset used in the model.....	58
Figure 3.8. (a,b) Raw images with shadows circled; (c,d) augmented images with improved object details.....	60
Figure 3.9. Losses over epochs.....	65
Figure 3.10. Training process flow.....	71

Figure 3.11. Sample of an additional dataset.	72
Figure 3.12. a) The second panel contains class B and the third panel contains class A; (b) the second panel contains class B and the third panel contains class C; (c) the second panel contains class E and the third panel contains no class; (d) the second panel contain class D and the third panel contains class A.	73
Figure 3.13. Confusion matrix.....	76
Figure 3.14. Precision–Recall curve.....	80
Figure 3.15. Confusion matrix layout.....	81
Figure 3.16. (a) Positive detection of class B and C; (b) positive detection of class A and B; (c) positive detection of class A and D; (d) positive detection of class B and E.....	83
Figure 4.1. Challenges faced during the testing process.	90
Figure 4.2. Mask R-CNN + ResNet50: (a) learning rate 0.00002 with an IoU of 50%, 75% and 90%; (b) learning rate 0.0002 with an IoU of 50%, 75% and 90%; (c) learning rate 0.002 with an IoU of 50%, 75% and 90%.	91
Figure 4.3. Mask R-CNN + ResNet101: (a) learning rate 0.00002 with an IoU of 50%, 75% and 90%; (b) learning rate 0.0002 with an IoU of 50%, 75% and 90%; (c) learning rate 0.002 with an IoU of 50%, 75% and 90%.	92
Figure 4.4. Mask R-CNN + ResNet50: (a) losses of the model with a learning rate 0.00002 and an IoU of 50%, 75% and 90%; (b) losses of the model with a learning rate 0.0002 and an IoU of 50%, 75% and 90%; (c) losses of the model with learning rate 0.002 and an IoU of 50%, 75% and 90%.	93
Figure 4.5. Mask R-CNN + ResNet101: (a) losses of the model with a learning rate 0.00002 and an IoU of 50%, 75% and 90%; (b) losses of the model with a learning rate 0.0002 and an IoU of 50%, 75% and 90%; (c) losses of the model with learning rate 0.002 and an IoU of 50%, 75% and 90%.	94
Figure 4.6. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.00002 and an IoU of 50%.	95
Figure 4.7. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.00002 and an IoU of 75%.	96
Figure 4.8. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.00002 and an IoU of 90%.	96
Figure 4.9. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.0002 and an IoU of 50%.	97

Figure 4.10. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.0002 and an IoU of 75%.	97
Figure 4.11. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.0002 and an IoU of 90%.	98
Figure 4.12. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.002 and an IoU of 50%.	98
Figure 4.13. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.002 and an IoU of 75%.	99
Figure 4.14. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.002 and an IoU of 90%.	99
Figure 4.15. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.00002 and an IoU of 50%.	100
Figure 4.16. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.00002 and an IoU of 75%.	101
Figure 4.17. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.00002 and an IoU of 90%.	101
Figure 4.18. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.0002 and an IoU of 50%.	102
Figure 4.19. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.0002 and an IoU of 75%.	102
Figure 4.20. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.0002 and an IoU of 90%.	103
Figure 4.21. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.002 and an IoU of 50%.	103
Figure 4.22. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.002 and an IoU of 75%.	104
Figure 4.23. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.002 and an IoU of 90%.	104
Figure 4.24. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.002 and an IoU of 90% trained on additional data.	105

LIST OF TABLES

Table 2.1. COCO dataset two-stage based models.....	44
Table 3.1. Dataset split.	54
Table 3.2. Image Preprocessing.....	60
Table 3.3. ResNet101 hyperparameter values.	75
Table 3.4. Precision- Recall table.....	79
Table 4.1. Average precision over all classes.....	106
Table 4.2. Overall accuracy using Mask R-CNN + ResNet50.....	107
Table 4.3. Overall accuracy using Mask R-CNN + ResNet101.....	108

LIST OF ABBREVIATIONS

ANN	Artificial Neural Network
AP	Average Precision
AP50	Average Precision at IoU = 50%
AP75	Average Precision at IoU = 75%
AP90	Average Precision at IoU = 90%
API	Application Programming Interface
BG	Background
CMOS	Complementary Metal-Oxide-Semiconductor
CNN	Convolutional Neural Network
COCO	Common Objects in Context
DL	Deep Learning
FAIR	Facebook™ Artificial Intelligence Research
FC	Fully Connected
FCN	Fully Convolutional Network
FN	False Negative
FP	False Positive
FPN	Feature Pyramid Network
FPS	Frames Per Second
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HOG	Histogram of Oriented Gradients
IoU	Intersection over Union
LSVRC	Large Scale Visual Recognition Challenge
mAP	Mean Average Precision
NMS	Non-Maximum Suppression
OA	Overall Accuracy
PCB	Printed Circuit Board
POESTA	Powder Electrostatic Application
R-CNN	Region-based Convolutional Neural Network
RFID	Radio Frequency Identification
RGB	Red, Green, Blue

ROI	Region of Interest
RPN	Region Proposal Network
SIFT	Scale-Invariant Feature Transform
SPPNet	Spatial Pyramid Pooling Networks
SQL	Structured Query Language
SSD	Single Shot Detector
SVM	Support Vector Learning Machines
TP	True Positive
UAV	Unmanned Aerial Vehicles
VGG	Visual Geometry Group
VIA	VGG Image Annotator
VOC	Visual Object Classes
YOLO	You Only Look Once

CHAPTER 1 Introduction

1.1 Introduction

An enamel paint coating line has relied heavily on human intervention and observation to determine the total number of parts produced in one shift through manual counting. Current techniques for object detection, other than human intervention, include placing RFID tags on the parts themselves or using light curtains to determine the class from its size. Manual counting has introduced human error into production data, while light curtains aren't able to accurately determine classes when parts are too close to one another.

Digital transformation within the manufacturing industry has attracted significant attention due to the integration of Industry 4.0 technologies. This has posed a lot of challenges within the manufacturing industry. The organisational and technological structural changes required within industrial firms have not shown their full potential from digital transformation as these industries have been slow to implement. The challenges are generally related to realizing the impact or cost of the process [1]. In the transition towards digital transformation within the manufacturing industry, data analytics has become imperative within an industrial firm to realize production process improvements. However, data analytics requires readily available data from the processes within the industrial firm. The challenge of missing or non-existent data within an industrial firm may be due to missing technological infrastructure or cost related to models used to extract that data [2]. The missing data required may be collected traditionally using paper-based instruments. However, using these traditional techniques may lead to inaccuracies as data may be lost between the field and the management office [3]. Furthermore, the data would need to be checked for accuracy, adding more time and resources to the process, which may add additional inaccuracies due to human error [3]. The challenges posed by digital transformation and data analytics within an industrial firm have resulted in a lack of real-time analysis and decision-making within the process [4,5]. Studies of human manual data capturing have indicated a 6.5% error rate, and for spreadsheet data entry it is expected to be in the range of 5% [6], [7]. While this is relative to the amount of data and the complexity

of the entered data. It has been shown that the human error rate may even vary between 5% and 26.9% [8].

The use of object detection applications in the manufacturing sector has seen a rise. This is evident from the state of research surrounding object detection which has increased rapidly over the last decade with several deep learning (DL) models being developed and adapted for industrial and robotic applications used in achieving improved production rates [9-12]. The use of DL models with the focus being placed on object detection applications within a production environment was realised to provide a solution to the challenges of digital transformation.

Before DL models became prevalent, object detection applications used features such as History of Orientated Gradients (HOG) and machine learning models called Support Vector Learning Machines (SVM) as the classifiers. The earliest milestone contribution to object detection applications using HOG and SVM was documented in 2005 [13,14]. N.Dalal and B.Triggs focused on human detection [13], having achieved results that greatly contributed to the object detection field with a processing speed of 1 frame per second (FPS) with 320 x 240 images [14]. Early contributions in DL-based models for image classification were documented in 2012, with the release of AlexNet, which discussed using convolution neural networks (CNN) based on the ImageNet Large Scale Visual Recognition Challenge (LSVRC) – 2010 dataset. Having achieved a top-1 and top-5 error rate of 37.5% and 17% respectively, performing better than previous state-of-the-art models in that year [15]. This development spurred the interest in using DL-based models for object detection applications. Thus Region-based Convolutional Neural Networks (R – CNN) were proposed in 2014, having achieved a mean average precision (mAP) of 53.3%, a greater than 30% improvement relative to the best-performing previous model based on the PASCAL Visual Object Classes (VOC) dataset [16].

As progress based on Scale – Invariant Feature Transform (SIFT) and HOG were noted as “small gains” in the researched field between 2010 – 2012 [16], the interest in DL models for object detection applications increased. The proposed Fast R – CNN model was introduced in 2015, being 9 times faster to train on the Visual Geometry Group (VGG)-16

architecture as compared to R – CNN and with mAP of 4% greater than R – CNN based on the PASCAL VOC dataset [17]. With the advancement in Spatial Pyramid Pooling Networks (SPPNet) and Fast R – CNN in reducing computational resources and increasing network performance, a focus area was established to further improve previous models. Faster R-CNN was introduced in 2016, with the introduction of the Region Proposal Network (RPN), focusing on decreasing the computational time required for region proposals [18]. Faster R-CNN achieved a mAP of 70.4%, 4.7% higher than Fast R–CNN and the overall best-performing model against SPPNet and R – CNN based on the PASCAL VOC dataset [18]. You Only Look Once (YOLO) was introduced in 2016 with the main objective of being fast and accurate [19]. While YOLO was able to achieve an FPS of 155 which outperformed Faster R-CNN at 7 FPS with the selected hardware, it did not surpass the mAP of 73.2% for Faster R–CNN, achieving 52.7% for YOLO based on the PASCAL VOC dataset [19].

The Mask R-CNN model was trained and evaluated in this study to perform the task of predicting the classes and their related masks. The Mask R–CNN model was introduced in 2017, and was developed by the Facebook™ Artificial Intelligence Research (FAIR) group [20], is an extension of Faster R-CNN by the addition of the mask branch. The addition of this branch performs the task of predicting the segmentation mask of each Region of Interest (RoI). This task is executed in parallel with the existing branch from Faster R-CNN that performs the task of classification and bounding box regression [20]. Mask R-CNN is capable of instance segmentation due to the addition of the mask branch making it unique in comparison to previous state-of-the-art models. Mask R-CNN outperformed Faster R-CNN with the highest AP across all categories of 62.3% compared to 57.5% achieved by Faster R-CNN, based on the Common Objects in Content (COCO) dataset [20]. The model detected parts on a conveyor line within a fast-moving consumer goods firm that manufactured household appliances.

The manufacturing of household appliances follows a process starting from the raw material. Sheet metal forming takes place within a mould under a precalculated pressure setpoint, which results in the deformation of a thin metal sheet to produce a part that resembles predesigned drawings [21]. These formed parts are to be used in the assembly of products in the automotive, space, aviation and household appliance industries [22-24]. Before the

newly formed parts are sent to the assembly line, the metal surfaces of those parts would need to be enamel coated for protection against corrosion, high temperatures and high pressures [25]. The enamelling process follows the Powder Electrostatic Application (POESTA), which is the application of the coating material in an electric field [25]. The basic principle of POESTA is based on the electric charge transferred to the particles of the dry powder. In corona-charging systems, an electric field is created between the spray gun and the part through the generation of a high-potential electrode [26]. The dry powder particles passing through this electric field would become charged. The process of the dry powder particles being charged in the electric field of corona discharge is administered through Pauthenier's Equation (1.1) [26]. Where Q is the charge accumulated, R is the radius of the sphere, E is the electric field, ϵ_0 is the permittivity of free space and ρ is the material-dependent constant.

$$Q_{max} = 4\pi R^2 \epsilon_0 \rho E \quad (1.1)$$

As the charged dry powder particle travels from the spray gun through the electric field, the force applied to the particle is the electric force equal to the charge of the particle which is multiplied by the strength of the electric field that pushes the particle towards the parts surface [25, 26]. Once the part has been fully coated in powder, it would need to be cured in an enamelling furnace. Figure 1.1 illustrates a model of the enamel coating spray booth where the parts would be passing to be coated.

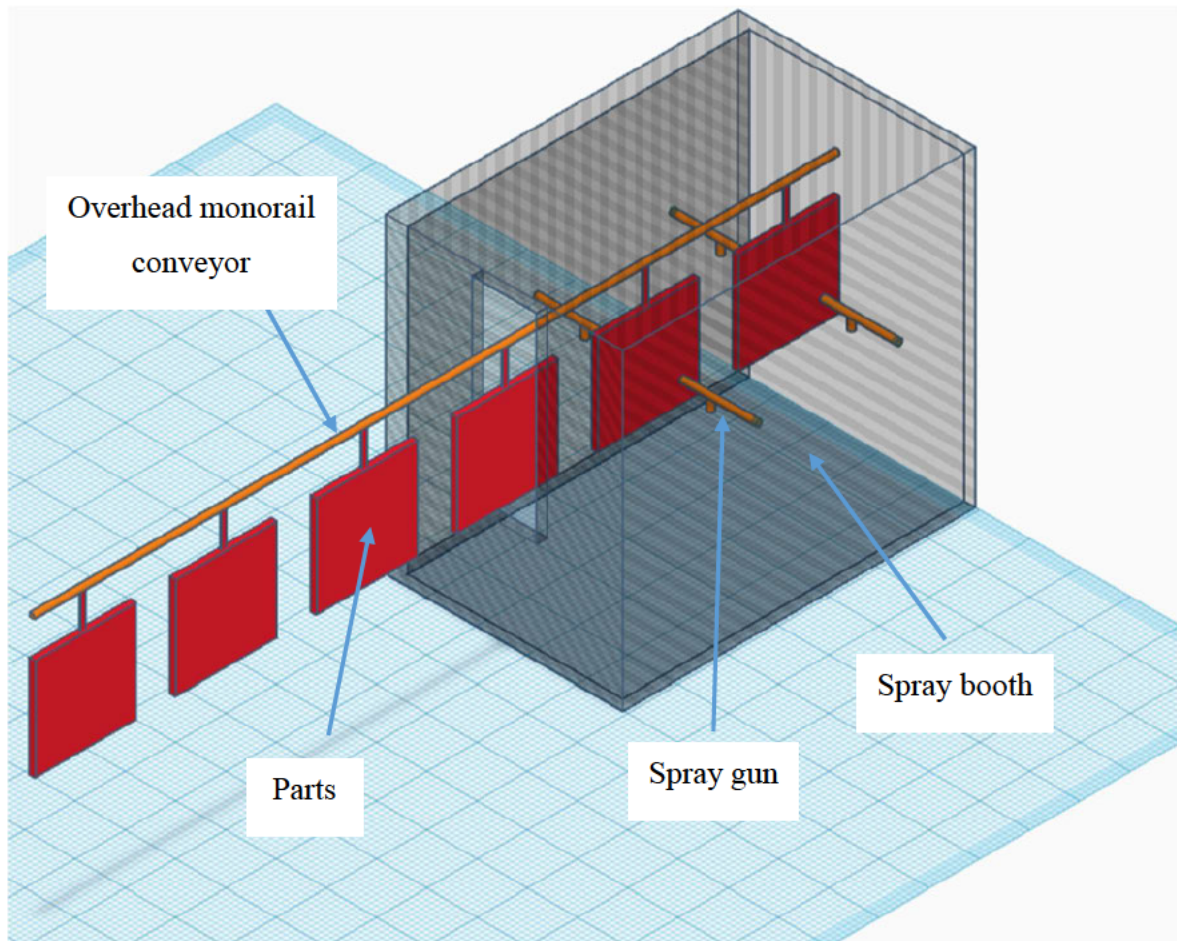


Figure 1.1. Model of enamel coating spray booth.

Batch production involves the manufacturing of a predefined number of parts belonging to a certain class of product, and is usually done within a precalculated time frame [27, 28]. This may be the production of medium-sized allocations of products which may run in single batches within a given shift or multiple repeated runs within a given shift [27, 29]. One of the advantages of batch production is the lower capital investment needed within a firm to manufacture a certain product [27, 28]. Another advantage of batch production may be realized with the effective use of planning to meet the customer's continuous demands as the operation is executed and completed before moving on to the next operation [30]. This may only be beneficial if there are corrective actions put in place when irregularities arise within a process [30]. These irregularities within the process have changed over the years with how humans approach the situation. These changes are affected by the evolution of technology, the ageing workforce and societal changes [31]. These uncontrollable factors have resulted in a poor understanding of production data which has negatively impacted the batch process within this case study. With the irregularities present within the process and the lack of real-

time monitoring, effective planning is not achieved in realizing the full potential of batch production.

1.2 Research Problem Statement

An enamel paint coating line within a household appliance manufacturing facility has relied heavily on human observation to determine the total number of parts produced in one production shift. This was performed by manually counting the parts as they were taken off the conveyor line and stacked into a trolley to be transported to the desired assembly workstation. This technique has introduced unpredictable errors within production data. The requirements needed for the currently available system such as Radio Frequency Identification (RFID) tags [32] required for the hangers or skids to be made specifically for the parts it would be transporting through the line as the tags would be placed on those hangers and not the part itself. The hangers on the coating line, that the research would be based on, are universal in design and can carry more than one model design. Therefore, each hanger cannot be attached to a single model and RFID tags being placed on these hangers would not provide reliable data, hence RFID tags were not a viable option. Alternatively, pattern recognition using light curtains may also be considered to identify certain classes of objects as they pass through the light curtain. This is accomplished through target objects blocking the transmission of light to the receiving sensor. The different classes may be defined by the hardware or software configuration. This approach, however, resulted in very poor accuracy due to the parts being in very close proximity to each other. As a result, the use of human intervention for manual data entry was instated. This archaic approach introduced inaccuracies and significant delays which hindered the process. This study focuses on applying an object detection model to automatically recognize the total parts per group/class to achieve the required production information and increase efficiency. With a manufacturing process that has a changing number of classes and variations of the part's physical orientation within the process line. Therefore, the importance of investigating the use of an object detection model was identified. This study compared the Mask R – CNN DL model with the average precision across all classes based on changes made to the learning rate and the intersection over union (IoU) thresholds. To the author's knowledge, research on enamel-coated parts detection and classification has not been as prevalent in the

literature and this makes this application unique within similar industrial applications and the computer vision research community. Therefore, this research study proposes the Mask R-CNN model for metal parts detection and classification based on the results provided in the subsequent sections. During this case study, the industrial environment was controlled in terms of lighting, a predetermined fixed distance between the objects of interest and the camera to achieve a high level of average precision. One of the main aims of this study was to achieve an average precision that was greater than 97.5% to be deemed as a viable solution to meet the industry standard for usable information within the business which was far superior to manual data entry.

1.3 Research Question

The research aimed to implement a Mask R-CNN object detection model to detect and classify parts hanging on a monorail conveyor line. The question asked, “Can the use of Mask R-CNN object detection models to detect parts on a conveyor line be a viable option for monitoring and reporting production performance data as compared to traditional methods?”

1.4 Research Aims and Objectives

The research aims to solve the issue of incorrect data being received from the process line by incorporating an automated system that would replace a human-performed task that is currently being performed. This research would aim to provide a unique solution compared to what is currently available in the coating industry. The objectives are listed as follows:

1. Conduct a literature review on DL-based object detection models that can perform object detection and classification of each class detected within an image.
2. Install the required hardware to train, validate and test datasets on a DL model.
3. Implement image augmentation on a dataset based on a literature review.
4. Apply and compare fine-tuned hyperparameters of the model.
5. Test and evaluate the performance of the model using the confusion matrix.
6. Provide future recommendations that relate to the use of instance segmentation within the same process of this research.

One of the research's aims would provide a solution as to whether universal hanger designs can be made to carry more than one product at a time while the tracking/data capturing system would be done through object detection.

Figure 1.2 illustrates the proposed setup and the adaptation within the existing process. The camera was installed before the parts entered the spray booth. This area was selected as the finish of the metal surface was uniform before the process of enamel coating. The photoelectric proximity sensors were placed in an area that was triggered as the hangers passed it. The Mask R-CNN model [33] was used in this research to train, validate and test with the dataset.

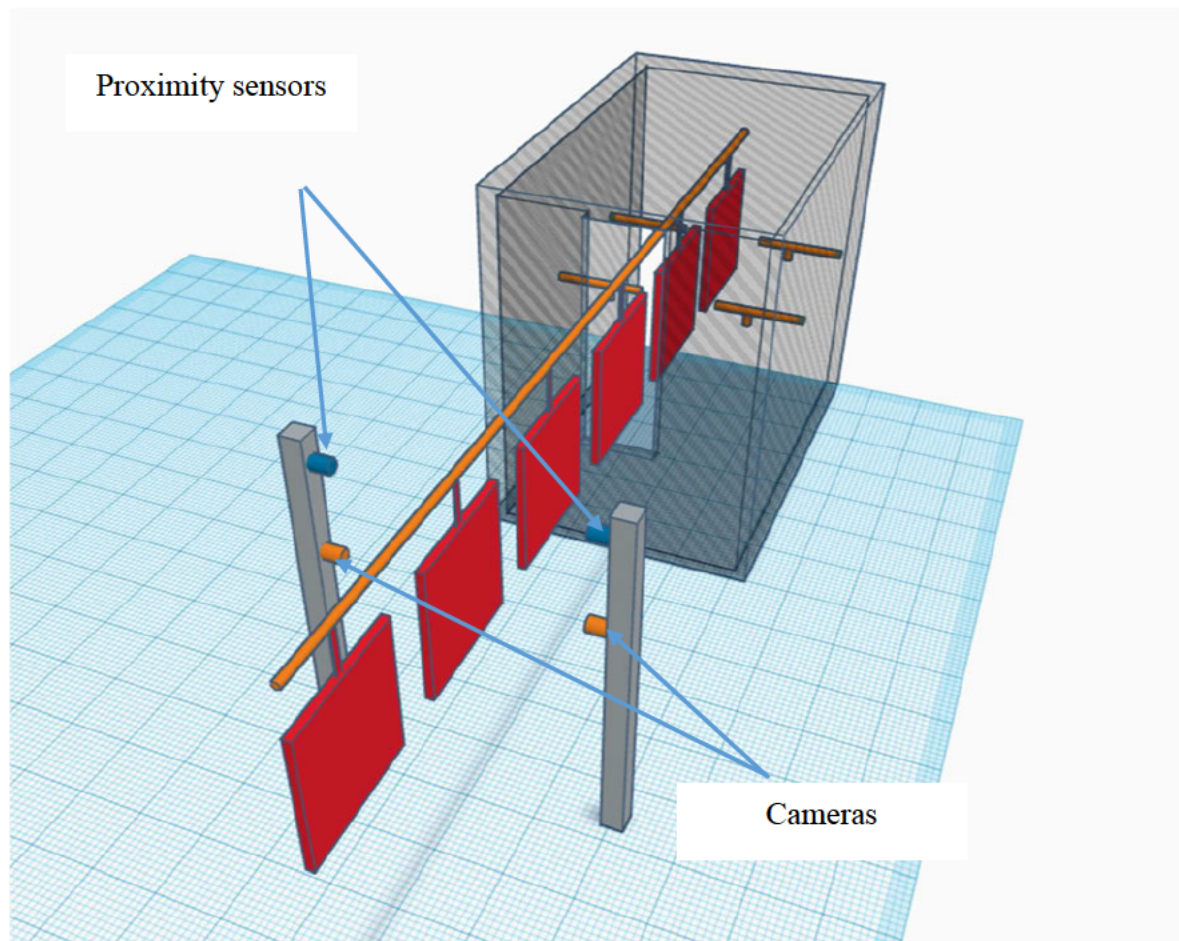


Figure 1.2. Model of enamel coating spray booth with proposed camera setup.

1.5 Study limitations

DL models require labelled data that would be used to train, validate and test. Inadequate data may lead to poor model performance or models that may overfit and not generalize well with new data [34]. This is a study limitation and is usually understood during the training and testing phase of the process. However, annotation of data is time-consuming and may add to poor model performance due to human error [34]. DL models require a powerful Graphics Processing Unit (GPU) to reduce the computational time required during training albeit at a high computational cost [35], and results may differ based on hardware configuration. Based on the requirements within the research, it must be determined whether a fast inference time is desired more than accuracy or vice versa. To achieve the objective in this work a trade off is required [34]. Furthermore, DL model hyperparameter tuning is time-consuming in achieving optimal performance from the model. Variations in lighting conditions within the environment pose a risk as models tend to struggle under these challenging conditions [34].

1.6 Research contributions

This main contribution of this study is summarized as follows:

1. We introduce a technique to automatically classify manufactured metal parts on an enamel paint-coating conveyor line, thus eliminating the need for repetitive human intervention.
2. A trained Mask R-CNN model is proposed for object detection and classification tasks, which is a sophisticated DL technique known in the literature for its accuracy in detecting and segmenting objects in captured images.
3. The study addresses the challenges of objects being outside the area of interest and shadows that distort the features of objects in an image.
4. Based on production requirements, variability in different classes of objects is accounted for, therefore ensuring that the system can accurately adapt to changing production requirements.
5. Finally, the proposed system achieves a high average precision and overall accuracy to meet the stringent demands of production quality requirements.

1.7 Structure of the thesis

Chapter 1 is an introduction to the research.

Chapter 2 provides a comprehensive literature review.

Chapter 3 presents a detailed methodology.

Chapter 4 gives comprehensive results using the Mask R-CNN model.

Chapter 5 presents a detailed analysis of results obtained from the model with a conclusion.

CHAPTER 2 Literature Review

2.1 Introduction

This chapter provides a comprehensive review of traditional methods used within the enamel coating line with a focus on production monitoring. Further in-depth literature review will be given on previous state-of-the-art object detection models and the architectures used within those models. The convolutional neural network layers would be expanded on together with illustrations. A detailed review of the Mask R-CNN architecture will be elaborated on. The training and implementation process and challenges would be explained relating to object detection applications. The literature review will provide evidence on applications that have used the Mask R-CNN model in different settings, thereby showing its efficacy.

2.2 Traditional Methods for Production Data Retrieval

For specific parts detection on a conveyor line, a common approach is to use Radio Frequency Identification (RFID) tags that are either attached to the product itself or the hanger on which the product is being conveyed [32,36-41]. These tags are thermally armoured to withstand the high temperature within the furnace after passing through the enamel coating process [38]. This method does not apply to the industrial application in this study, as an RFID tag would be required for each part which is not economically sustainable. In addition, due to workflow, quality, and company design standards, a single hanger was used to carry more than one product at any given instant. Therefore, RFID tags were not a viable option. Alternatively, pattern recognition using light curtains may also be used to identify certain classes of objects as they pass through the light curtain. This is accomplished through target objects blocking the transmission of light to the receiving sensor [42-45]. The different classes may be defined by the hardware or software configuration. This approach, however, resulted in very poor accuracy due to the parts being in very close proximity to each other. As a result, the use of human intervention for manual data was instated. This archaic approach introduced inaccuracies and significant delays which hindered the process.

2.3 Convolutional Neural Networks for Computer Vision

Convolutional neural networks (CNN) have revolutionized the field of computer vision. While it has been known to be used since the late 1980s for a few computer vision tasks [46], progress was slow until the mid-2000s when computing power and availability of data increased [46]. Deep CNNs have allowed models with multiple processing layers to be able to learn details of data with the abstraction of these details occurring over multiple layers [47]. The use of CNNs has eliminated the use of manual feature extraction as the model can extract features directly from pixel data [48]. CNNs consist of key layer types, which are the convolutional layers, pooling layers and the fully connected layers [47, 49]. This is illustrated in Figure 2.1 with further explanation given in the subsections.

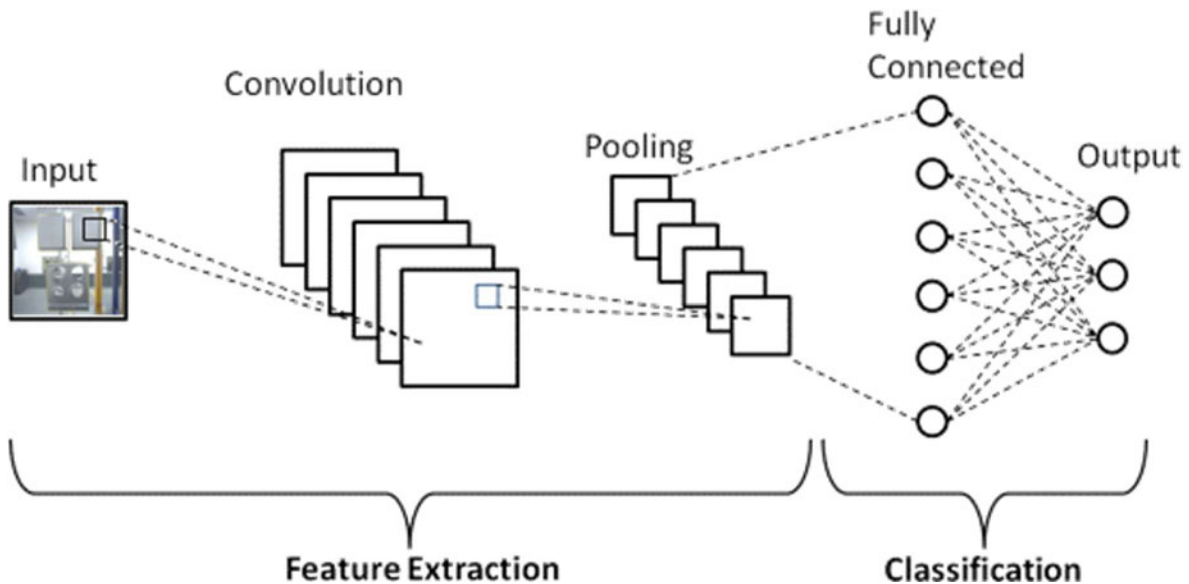


Figure 2.1. CNN layers.

2.3.1 Convolutional Layer

The convolutional layer is the first layer that is used to extract local features from the input image. This is done through the application of multiple filters (or kernels) across the image. This process would create feature maps that would add to the creation of a new image with the selected filter [50]. An image filter is a matrix with predefined image pixel values that would be used in an element-wise multiplication with the receptive field. This is used to detect a pattern in the data or image that is used in the input [50]. This convolution operation is described by Equation 2.1. Where the output represents the feature map, x represents the

receptive field from the input image, w represents the kernel matrix and t represents the time index. This is further illustrated in Figure 2.2 with an example of a 2-D convolution.

$$S(t) = (x \star w)(t) \quad (2.1)$$

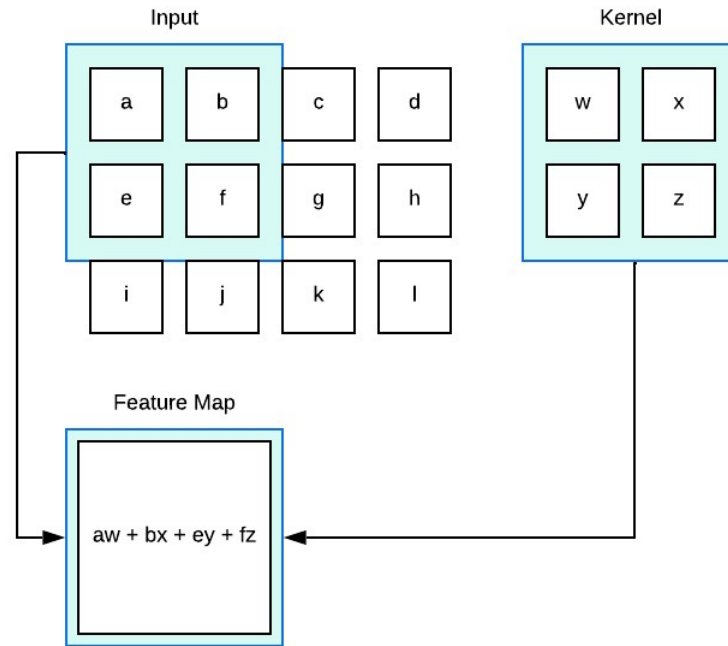


Figure 2.2. 2-D convolution.

2.3.2 Pooling Layer

The pooling layer performs the task of reducing the spatial size of the feature maps [51]. This is done using a function that either takes the maximum or the average value within the subregion of a sliding window [51]. Figure 2.3 illustrates the use of average pooling with a computed 3 x 3 matrix from a 5 x 5 matrix within a feature map. This would reduce computational time and resources during the training period [50].

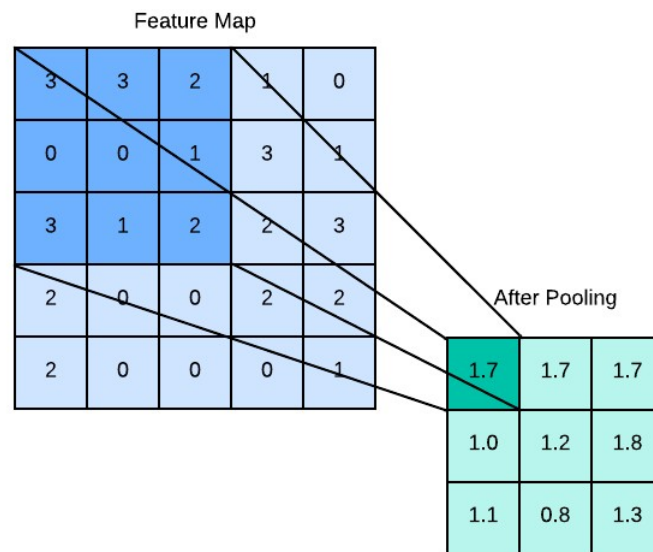


Figure 2.3. Average pooling operation on a 5 x 5 input.

2.3.3 Fully Connected Layer

The Fully Connected (FC) layer is positioned at the end of the network and performs the task of classification. This is done by categorizing the features, which are extracted after the convolutional and pooling layer for a specific class [52]. The fully connected layer is a dense layer of connected neurons [53]. The images that were retrieved after the convolutional and pooling layer would be transformed into a 1-dimensional array (flattened) before being sent through the FC layer [53]. The output FC layer would have the same number of neurons as the number of classes. The connections between the neurons are called weights and these weights are the trainable parameters that our model needs to learn. The FC layer task for classification is done through learning and associating specific features with the correct classes [53]. This is illustrated in Figure 2.4.

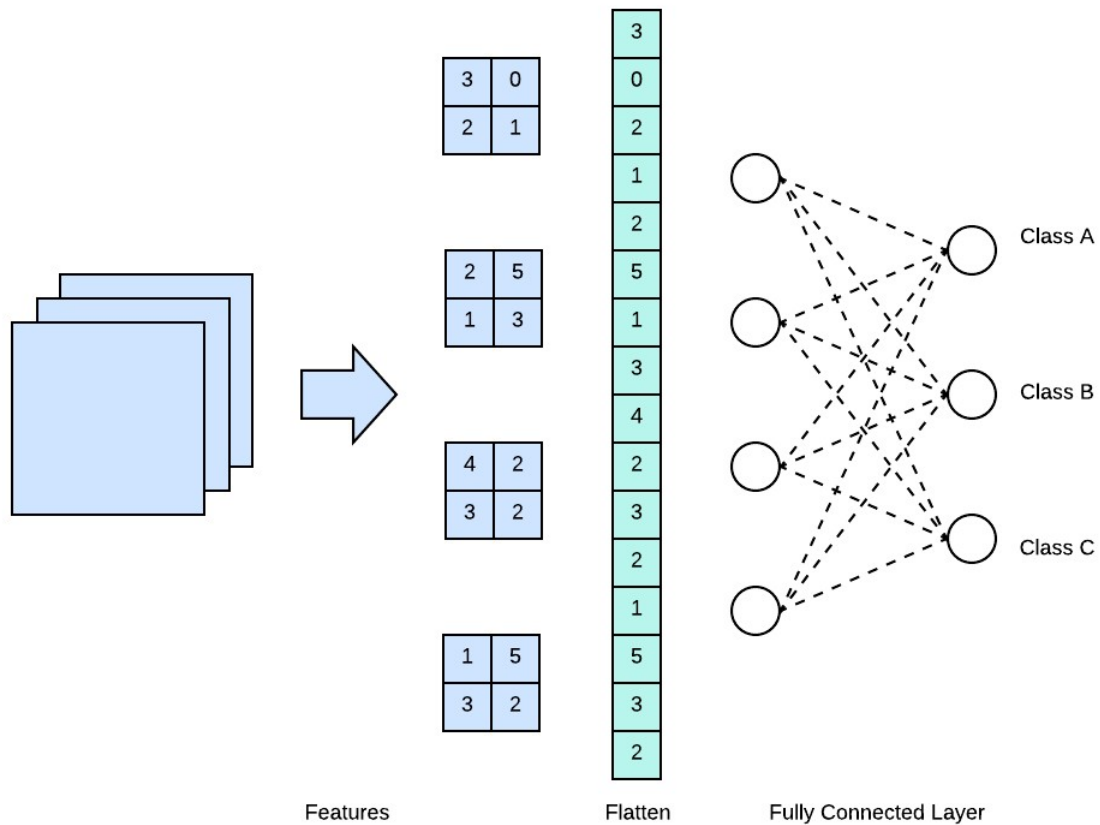


Figure 2.4. Fully connected layer.

2.4 Architecture of Convolutional Neural Networks

CNNs have undergone stages of evolution since their inception, leading to improvements in the architecture. These improvements have led to better-performing models in the computer vision field [54]. These updated architectures have focused on the ability to recognize features and patterns within an image [55].

2.4.1 LeNet-5

LeNet-5 was one of the earliest architectures used commercially as a recognition system within the banking industry, specifically document recognition [56]. The LeNet-5 architecture contained 7 layers with the layers labelled “Cx” as the convolutional layers and the layers labelled “Sx” being the subsampling layers. The FC layer is labelled as “F1”.

Figure 2.5 is the earliest introduction to the convolutional and pooling layers [56]. The LeNet-5 architecture was trained on the Modified National Institute of Standards and Technology database (MNIST) database and achieved an error rate of 0.8% [56].

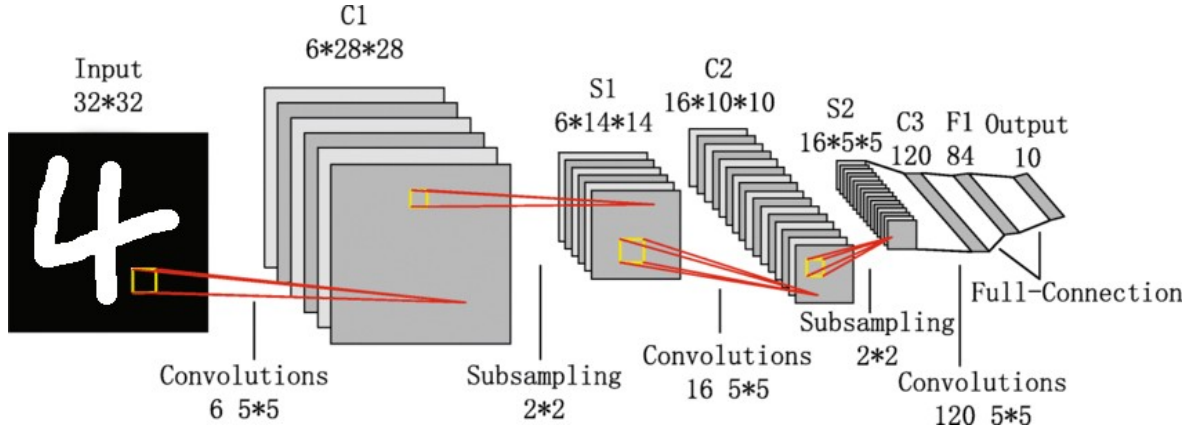


Figure 2.5. LeNet-5 architecture [56].

2.4.2 AlexNet

AlexNet was introduced in 2012, the network contains 8 learned layers with 5 convolutional layers and 3 FC layers [15]. The output of the last FC layer is fed into a 1000-way softmax that corresponds to the ImageNet dataset. The first convolutional layer has 96 filters followed by the second convolutional layer of 256 filters. The last three convolutional layers have a filter of 384, 384 and 256 respectively. The two FC layers both have a total of 4096 neurons each [15]. The architecture is illustrated in Figure 2.6.

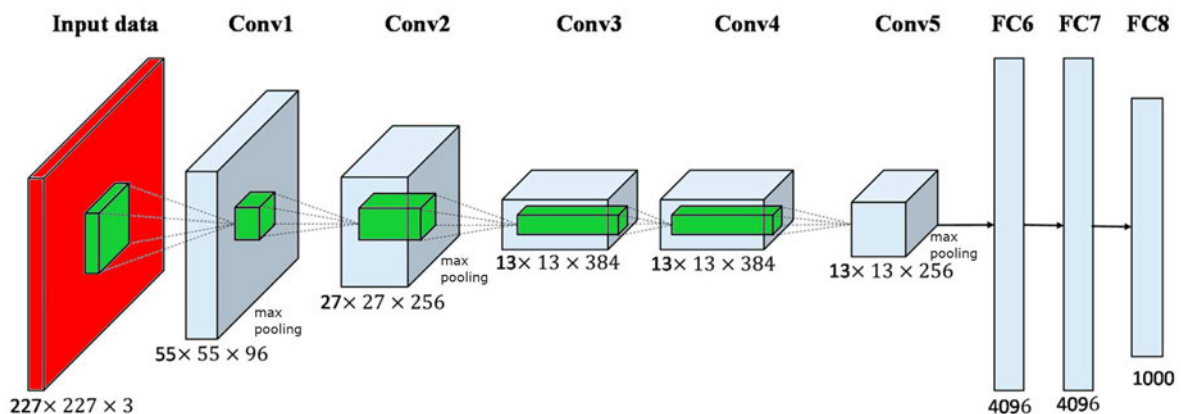


Figure 2.6. AlexNet architecture [15].

2.4.3 VGG Net

The VGG Net architecture design focused on increasing the network depth while maintaining other parameters of the network in enabling the network to learn more complex features [57]. The latest version of VGG Net had a network depth of 19 weight layers [57]. The network has 16 convolutional layers followed by three FC layers. Layer one has two convolutional layers of 64 filters each, followed by two more convolutional layers with 128 filters each in the second layer. This is followed by three layers of four convolutional networks, having 256, 512 and 512 filters respectively. The three FC layers have 4096, 4096 and 1000 neurons respectively. This is illustrated in Figure 2.7.

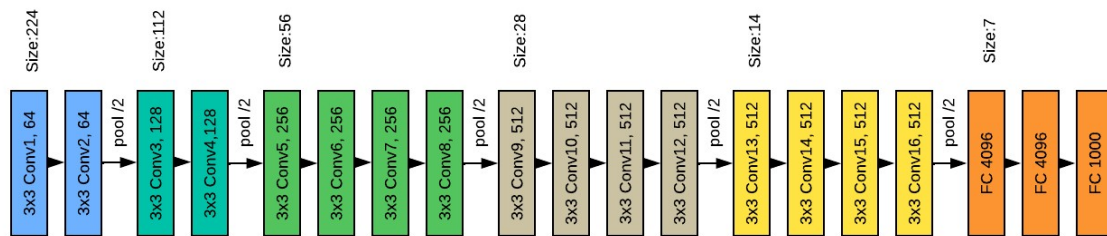


Figure 2.7. VGGNet architecture [57].

2.4.4 ResNet

Residual Networks (ResNet) is a residual learning network proposed to ease training for deeper networks [58]. The layers are formulated as learning residual functions with the input as a reference. ResNet is easier to optimize with the added accuracy gain from a deeper network [58]. Stacking more layers to a network doesn't always contribute to a better-performing network, which leads to the problem of vanishing or exploding gradients which hampers the convergence from the beginning [58]. This issue was addressed through normalized initialization layers [59]. A degradation problem was exposed as the network became deeper, leading to poor accuracy. The addition of more layers with the intention of increasing accuracy led to greater training errors [58,60,61]. The degradation problem is illustrated in Figure 2.8 with a network with 56 layers having higher training and test errors [58].

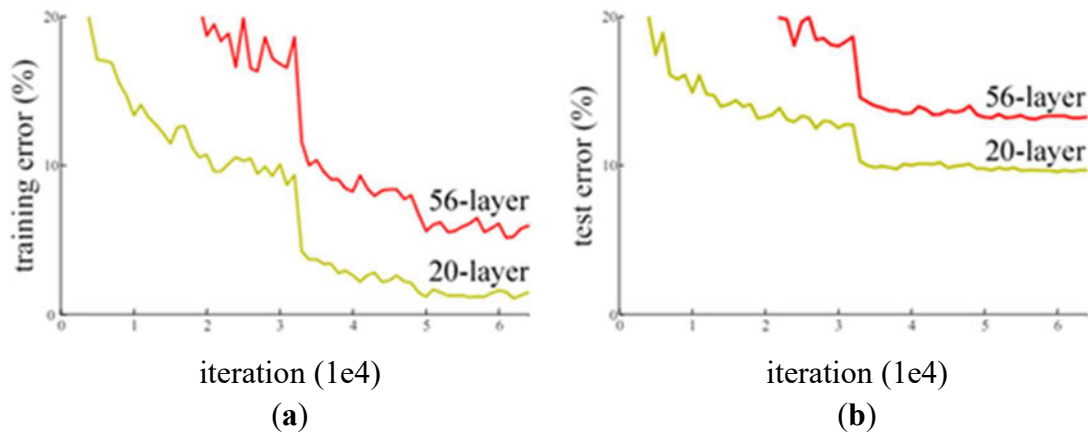


Figure 2.8. Training error and test error on CIFAR-10 with a plain network [58].

(a) Training error. (b) Test error.

The proposed ResNet aimed to address this problem with the introduction of a deep residual learning framework which has skipped connections. These skipped connections perform identity mapping that is later added to the outputs of the blocks containing the layers as illustrated in Figure 2.9.

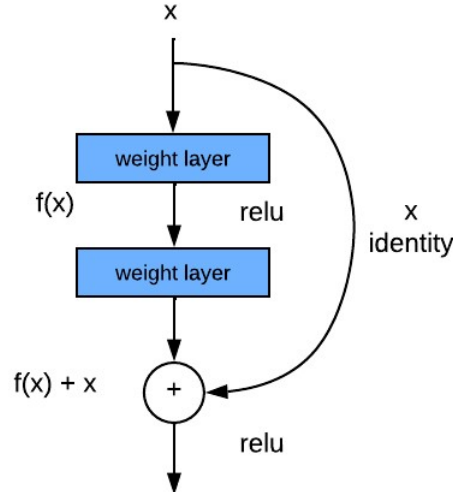


Figure 2.9. Residual learning building block.

Equation 2.2 defines the building block of Figure 2.9. x and y are the input and output respectively. The function $f(x, \{W_i\})$ defines the residual mapping that needs to be learned.

$$y = f(x, \{W_i\}) + x \quad (2.2)$$

2.5 Historical Context of Object Detection

2.5.1 R-CNN

Region-based convolutional neural networks (R-CNN) were proposed in 2014, which represented a significant improvement in the computer object detection field. The model used CNNs to perform localization and classification operations [16]. The R-CNN model takes an input image and proposes around 2000 regions. Each proposal is then sent through the CNN where features are attained, followed by the classification of each region [16]. The selective search algorithm proposes the regions. Each region proposal is then warped to a fixed size before being sent into the CNN. The features are classified using linear SVM which is trained for each class of object [16]. This model is illustrated in Figure 2.10.

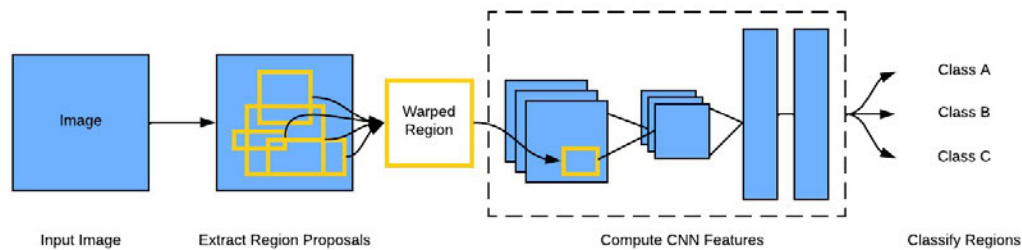


Figure 2.10. R-CNN algorithm framework [16].

2.5.2 Fast R-CNN

Fast R-CNN's improvements over R-CNN included having a single training process, which reduced the training time. Feature maps were shared, which meant that an entire image was processed through the CNN to produce the feature map. Region proposals would then be projected onto the shared feature map. This would allow for features for each region to be extracted using ROI (Region of Interest) pooling [17]. Fast R-CNN uses a multi-task loss function that combines the classification and bounding box regression losses, allowing for refinement of each loss simultaneously [17]. Region proposals are generated the same as R-CNN using the selective search algorithm. The ROI pooling layer extracts fixed-size feature maps for each region proposal. The feature maps are fed into the FC layer followed by the classification based on the class [17]. This model is illustrated in Figure 2.11.

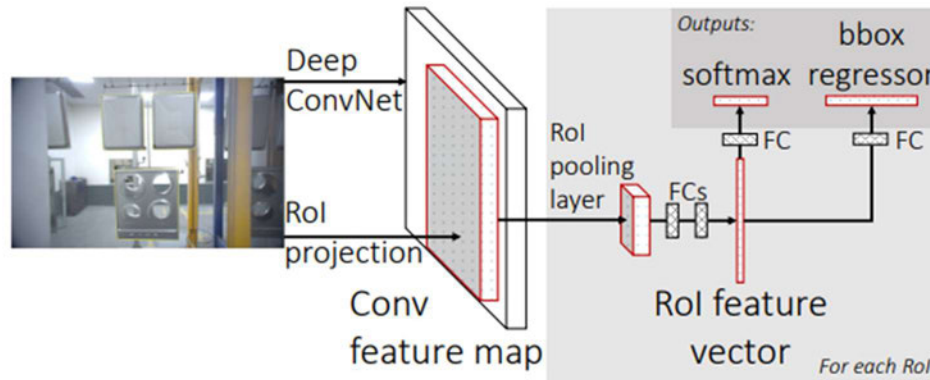


Figure 2.11. Fast R-CNN algorithm framework [17].

2.5.3 Faster R-CNN

Faster R-CNN improvements over Fast R-CNN included the introduction of Region Proposal Networks (RPN), which allowed the model to generate region proposals through the introduced network [18]. This advancement over the selective search algorithm positively contributed to computational resources and time required during training [18]. Faster R-CNN followed a two-stage detection operation due to the addition of RPN being one stage, followed by the Fast R-CNN architecture. This model is illustrated in Figure 2.12.

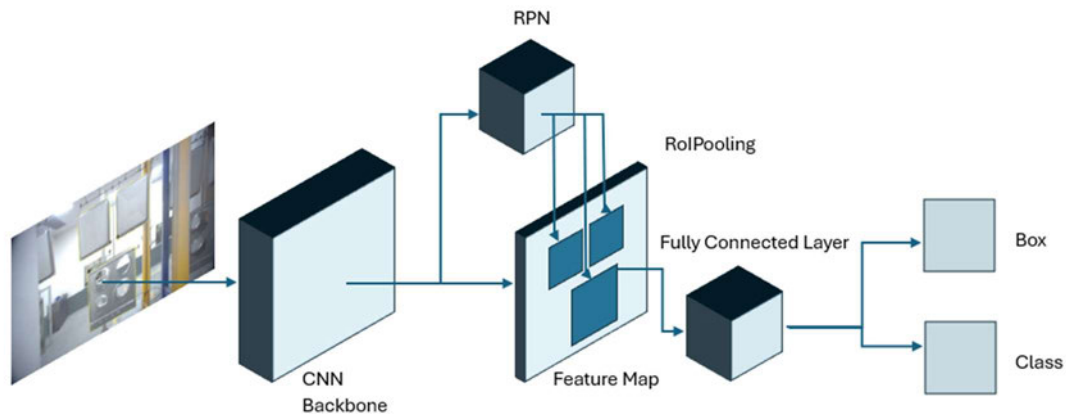


Figure 2.12. Faster R-CNN algorithm framework [18].

2.5.4 SSD

Single Shot Detector (SSD) is a single pass-through model unlike Faster R-CNN and Mask R-CNN which uses an RPN. This makes the model faster as it does not require the added stage [62]. SSD uses feature maps from multiple layers of the network to detect objects at

different scales. Each layer's operation detects objects of different sizes, which enables the model to be able to detect different-sized objects [62]. SSD uses default boxes of varying aspect ratios at different locations within several feature maps. For each of the default boxes, the shape offset, and confidence are predicted [62]. This model is illustrated in Figure 2.13.

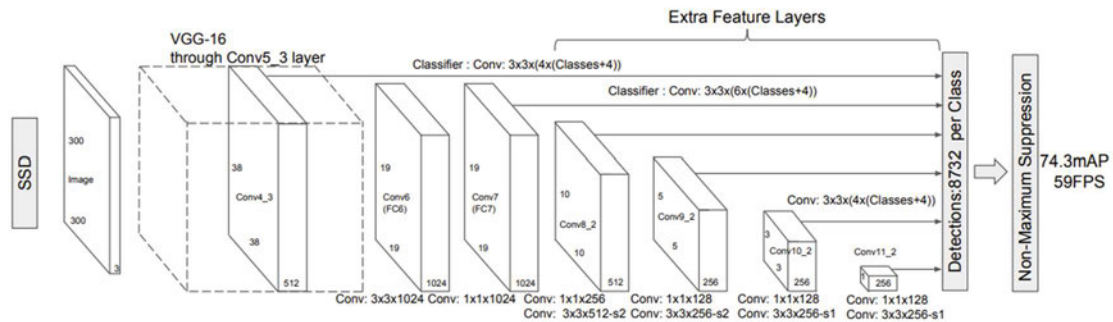


Figure 2.13. SSD algorithm framework.[62]

2.5.5 YOLO

You Only Look Once (YOLO) shares the same network concept as SSD, being a single-stage detector [19]. The YOLO architecture divides the input image into a grid format, creating blocks within the image. Confidence scores are calculated and classification probabilities are predicted within those boxes [19]. To eliminate redundant or multiple detections, YOLO implements Non-Maximum Suppression (NMS) to filter out overlapping or redundant bounding boxes [19]. This model is illustrated in Figure 2.14.

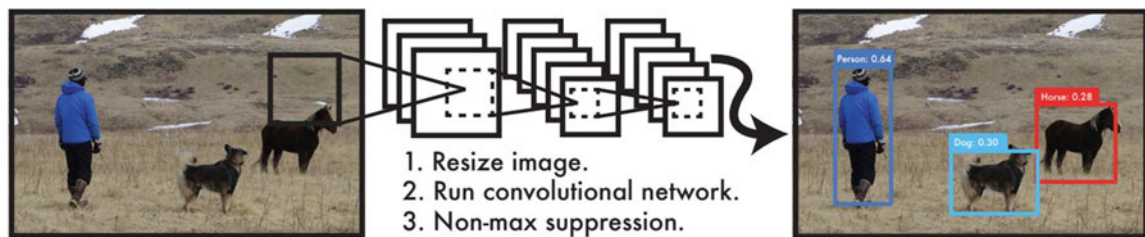


Figure 2.14. YOLO algorithm framework.[19]

2.5.6 Object Detection Impact and Application

Research conducted in [63], proposed the use of an Artificial Neural Network (ANN) model developed for fault detection on specular painted surfaces of automobiles. The authors propose Histogram of Orientated Gradients (HoG) feature vectors of a 20×20 image patch

to train the ANN model. This approach could provide a clear distinction between dirt, scratches and no defect classes. Superior results were obtained using images that had a higher contrast, namely images with a darker background and brighter defects. However, the authors did note that the ANN model could not classify images of metallic surfaces with small irregular spots or blemishes in a reliable manner further suggesting that the algorithm could be improved for noisy images. In a similar application, [64] proposed the use of deflectometry and vision-based technologies to overcome the problem of specular defects on vehicle body surfaces. They proposed a two-stage algorithm, in which the first was developed to enhance the image contrast between the image pixels with high-level and low-level intensity to distinguish between the presence of defects or lack thereof. For the second step, they present localized directional blurring to provide background exclusion.

More recently, [65] presented the results of an automatic surface defect inspection using machine vision methods. The proposed automatic inspection system extracts candidate defective regions which are classified according to dents, scratches and pseudo-defects with their relative shape, size and diver-gent features. A Support Vector Machine (SVM) was used in the classification of each class. In the work [66], computer vision techniques were used to identify defective parts on an assembly line. The captured images were pre-processed to remove background artifacts, reduce noise, correct orientation, and provide scaling for further processing. Objects of interest were then extracted based on their unique contours and centroid distance computation. Lastly, the detection results were based on the objects' similarity rating to other stored predefined model shapes and likeness matching.

In another industrial application, machine vision was used to provide robot guidance in a collaborative framework. The authors in [67] highlighted the importance of computer vision as an effective tool for providing autonomous systems that enable collaborative work between humans and industrial robots, thereby improving the robotic device positioning accuracy as tasks become increasingly complex. The authors in [68] focus on YOLO updates over the decade and illustrate the architecture of each update. The applications that were mentioned included the use of YOLO for steel surface detection. An application using Faster R-CNN was used to detect defects in chemical fibre paper tubes where human judgment was lacking in tracking the defects [69]. Authors in [70] mention the significance of having an

object detection application based on R-CNN to identify different clothes that had more complex features, contributing positively to the clothing commercial industry. The rapid development of deep learning models has been mentioned in [71], with benchmarks through the use of datasets that have been openly available to the computer vision community. The study [71], describes the importance of training these models on new datasets which may affect the results of the model. In [72], the adaptation of object detection algorithms in combination with autonomous vehicles, health care and surveillance systems applications are described. The use of neural network architecture as compared to conventional methods has come with its own set of challenges. This has prompted researchers to further investigate real-time detection and its use cases. The rise of Industry 4.0 and the concept of sustainability being the main focus in Industry 5.0, has driven research in technology to new heights as mentioned in [73]. In relation, YOLO and Faster R-CNN were models used within the transformation goals within the industry. Real-time processing takes precedence in that YOLO can achieve greater speeds as compared to Faster R-CNN for industrial applications such as robotics. However, it is mentioned that the trade would be seen in the accuracy, as Faster R-CNN is proven as more accurate.

The study by [73] states that applications relating to defect detection and analysis, show that Faster R-CNN provides superior results in accurate localization of defects ensuring high performance of the production requirements as compared to YOLO. This work [73], further explains that object tracking within a dynamic environment with moving parts on conveyor belts has seen viable results from Faster R-CNN being capable of automated material handling systems. The success of Faster R-CNN over YOLO is mentioned within inventory management, having avoided delays caused by material shortages. Material is identified and counted minimizing wastage and preventing overstocking [73]. The authors in [74], state that object detection, classification and tracking are three computer vision technologies that are important within industrial applications. The study mentions the challenges that are faced with deep learning models including acquiring good quality data, addressing overfitting and underfitting and how quantifying the uncertainty relating to prediction scores.

A review of machine learning and deep learning for object detection was conducted in [75]. The review mentions that machine vision is a field that aims to provide visual perception in

computers like humans. The use of deep learning may achieve that aim using the neural network in attaining high-level features in data. A comparative analysis was conducted in [76]. The review compares multiple one-stage and two-stage detectors based on the COCO and VOC dataset, and the conclusion outlines key takeaways from the review. There is a relationship between processing speed and accuracy. As accuracy is increased, speed is decreased due to the requirements in computational resources required for accuracy. This challenges researchers to develop more efficient networks to address these two concerns. The challenge of smaller objects within an image does affect a model's output results. This is due to the lower number of pixels found on those smaller objects. A survey [77] discussing deep learning methods for object detection within smart manufacturing, introduces a topic relating to quality control in a manufacturing assembly line. The challenge stated that errors in processing material with defects not visible to the human eye may corrupt the production line. Applying more human workers leads to an increase in manufacturing expenses. To be able to provide an increase in production while maintaining the quality of the product may be possible using intelligent manufacturing. The survey suggested that the use of computer vision techniques resulted in automated quality inspection and reduced manufacturing expenses through labour cost reductions. Deep learning models were stated as achieving more reliable and more efficient inspection in more complex environments. The use of these models may be used in environments to learn more complex features.

Work conducted by [78], describes industrial inspection as a challenging task in manufacturing. The challenge states that humans are not able to provide reliable inspection as compared to machines. The literature provides a case study for the use of deep learning neural networks to detect different classes of objects within an image. The sequence of detected objects from the model of choice was then compared to previous methods to outline any flaws detected within the process with the evidence attained through the use of a deep learning model. The authors concluded that the use of a deep learning model was applicable for industrial inspection. The work conducted by [79], provides information on the demand for robot-based depalletization systems, attributing the increase in demand to the growing sector relating to logistics, storage and supply chains. The challenge of unknown pallet layouts prompted the authors to propose a robust solution based on deep learning methods for computer vision. The term robust was suggested by the authors as the application using

a deep learning model was effective in processing variability in the data. The authors concluded that by using a CNN-based model for object detection, robust detection accuracy was achieved with the minimization of data required. The outcome resulted in a decrease in the commissioning time.

Within the mining industry, the authors [80] provide a solution to inspect belt conveyors automatically using object detection and thermal imaging. The authors concluded that based on the YOLOv2 deep learning framework, they were able to achieve a reduction in false discovery and false rates of 52.46% and 61.03% respectively as compared to previous works. The work of [81], provides valuable insight into the use of machine learning for object detection in manufacturing applications. The article states that for manufacturing industries to remain competitive, those manufacturers must adopt feature recognition technologies within the computer-aided design (CAD) analysis field. The authors mention that due to the improvement of data availability in cyber manufacturing, the use of machine learning in combination with big data generated more profits within the manufacturing sector. The authors conclude the need for the use of a machine learning model, but the challenges are noted as well. The challenges include overcoming complexity, calculation errors and ambiguity.

Research conducted by [82], states that with the increase in pace in the industrial sector, the need for a smart environment is increasing proportionally. The challenge associated with reducing downtime, safety risks and quality issues in an industrial environment prompted the authors to propose an R-CNN model to detect the faults in the product. Faster R-CNN was used with a VGGNet backbone to detect the classes of objects. The authors mentioned the challenges of the objects being small, the dataset being imbalanced using their custom dataset and the classes of objects being similar in features for a few classes. These challenges were addressed through the use of data augmentation for the data imbalances, the addition of bicubic interpolation for feature amplification and an addition of a centre loss to combat the similarities with more than one class. The authors concluded that the proposed Faster R-CNN model had performed better in comparison to R-CNN and Fast R-CNN.

CNNs used for object detection have contributed to applications outside of the industrial field that require accurate output for quick responses. The study by [83], emphasizes accurate and timely access to data is imperative relating to disaster management. Data acquisition using unmanned aerial vehicles (UAV) provides insight into disaster-affected zones. The challenge stated by the authors [83] included the processing of large amounts of data in identifying these disaster zones in real-time. The authors mention that the methods used before the study was conducted, were done manually. The authors proposed the use of CNN to detect ground objects of interest. The study concluded with achieving a mAP of 24.50% across all classes using the YOLO model.

2.6 Introduction to Mask R-CNN

The Mask R-CNN model can classify and locate multiple objects with complex features with high precision at a pixel level [84]. The focus of this model's unique feature is placed on its ability to perform instance segmentation and being able to classify and separate same-class overlapping or proximity objects [85]. The following sections will provide an overview of the architecture of the Mask R-CNN model, followed by a detailed explanation of each section within the architecture. The types of data augmentation will be explained and their function within the preprocessing step. Hardware and software influence on results will be elaborated. This is followed by case studies using Mask R-CNN in real-world and industrial applications.

2.6.1 Mask R-CNN Architecture

As shown in Figure 2.15, the Mask R-CNN is a deep learning model with the ability to achieve pixel-wise instance segmentation through the combination of object detection and semantic segmentation [20]. This model is an extension of Faster R-CNN by the addition of the mask branch. The literature review on Faster R-CNN has shown positive results in comparison with previous models. The addition of this branch performs the task of predicting the segmentation mask of each Region of Interest (RoI). This task is executed in parallel with the existing branch from Faster R-CNN that performs the task of classification and bounding box regression. The backbone used for feature extraction is Residual Neural Networks (ResNet) with the addition of the Feature Pyramid Network (FPN). The Region

Proposal Network (RPN) performs the task of generating region proposals that may contain the object of interest [18]. Once the region proposals are generated, the Region of Interest Align (ROI Align) layer performs the task of extracting features from the feature map. Mask RCNN was chosen for this study's application since it is capable of instance segmentation [86]. With instance segmentation, objects within an image can be split into their respective detected classes as a single object and not a part of a set of objects as one class [87]. This would enable the detection of individual parts from a production line. The notable change of Mask R-CNN in comparison with Faster R-CNN was the replacement of the ROI (Region of Interest) pool with ROI Align, which would handle the misalignment and data loss between the input image and feature maps [20].

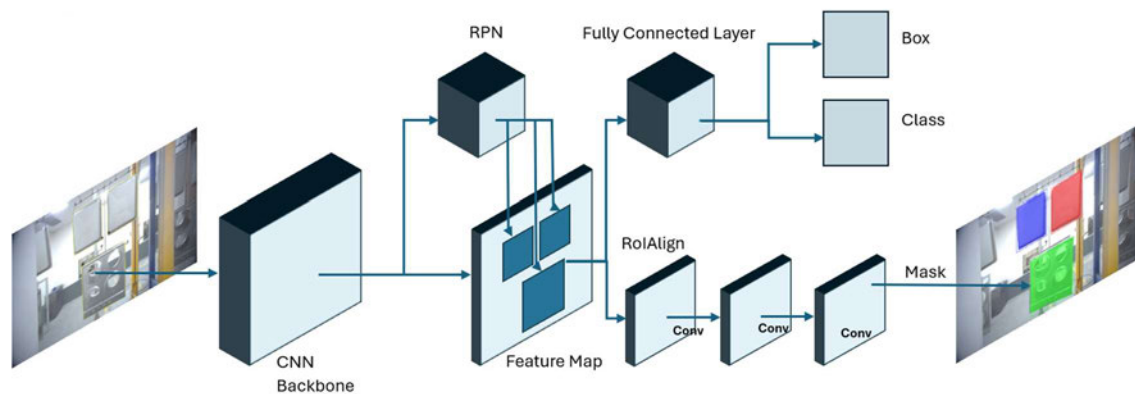


Figure 2.15. Mask R-CNN algorithm framework for parts detection [20].

2.6.2 Region Proposal Network (RPN)

The RPN was introduced in [18]. This innovative development performed the task of generating high-quality region proposals directly from the feature maps produced by CNN. The RPN is modelled with a Fully Convolutional Network (FCN) [84]. An image is taken as an input in the RPN producing multiple rectangular object proposals as the output [18]. The region proposals are generated by sliding a network over the feature maps. The network itself uses a window with a spatial window of $n \times n$ being mapped to a lower-dimensional feature. At each sliding window location or anchor point, multiple region proposals are generated. The anchor boxes which are located at each anchor point are predefined with various scales and aspect ratios. These would be the reference points for generating

proposals. The network follows a two-stage prediction; which include the objectness score associated with the prediction of whether an object is within an anchor box, and the bounding box prediction stage provides the coordinates of the anchor boxes. Through the use of the RPN, an efficiency gain is achieved over the previous selective search method. The generation of high-quality proposals directly from feature maps does improve the overall accuracy of the model [18]. The RPN structure is shown in Figure 2.16.

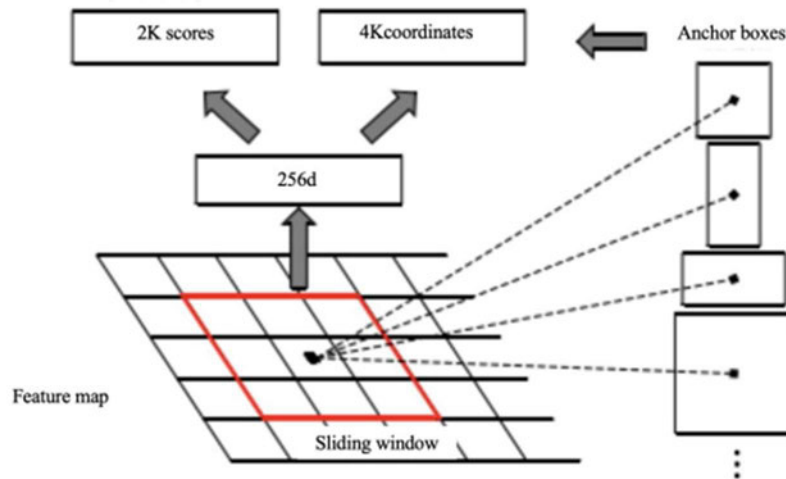


Figure 2.16. RPN structure [84].

2.6.3 Feature Pyramid Network (FPN)

The FPN plays an important role in extracting features from an image that forms part of the CNN backbone, as shown in Figure 2.17. The feature extractor would perform the task of taking a single image of an unknown scale as the input image and would output feature maps at multiple levels [88, 89]. The low-level feature maps would include information such as edges, colour and textures while high-level feature maps would include information such as the object components or the object itself. This multi-scaled feature map would include information from all the different levels forming a set that has rich semantic data [90]. The bottom-up pathway uses ResNet as its backbone which is used to structure the bottom-up pathway and is comprised of convolution modules. The ResNet module proposed in 2015 [58] has greatly improved the accuracy of image classification. The advantage of ResNet is the ability for a model to be trained at a deeper level within the neural networks [91]. A degradation problem was exposed when deeper networks started to converge [58]. With the

network depth increase, accuracy was decreased due to saturation. ResNet was proposed to solve this issue leading to better accuracy with a deeper network [58].

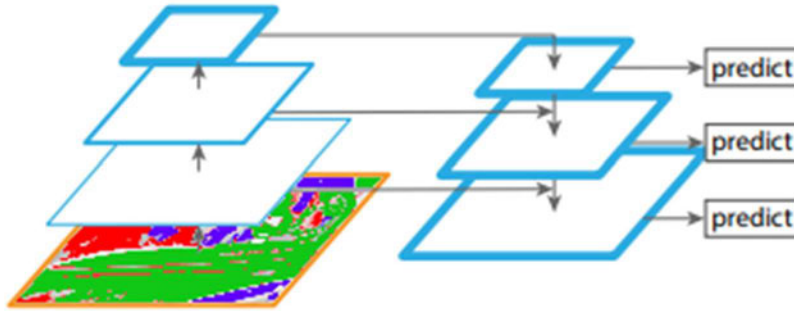


Figure 2.17. Feature Pyramid Network [90].

2.6.4 ROIAlign

In contrast to Faster R-CNN's, ROI Pooling is an operation that is used to extract feature maps from an ROI. Faster R-CNN was not designed for pixel-to-pixel alignment using ROI Pooling. This performs stride quantization resulting in misalignment and loss of data. While ROI Pooling may not affect classification, it does have a negative impact on predicting the masks generated for each pixel of the object [20]. With ROIAlign, as shown in Figure 2.18, no quantization is applied. This eliminates misalignment and data loss between the extracted features with the inputs [20]. Bilinear interpolation is used to calculate the values at four locations within the bin location of each ROI. The final value is achieved by the maximum or average of those values [20].

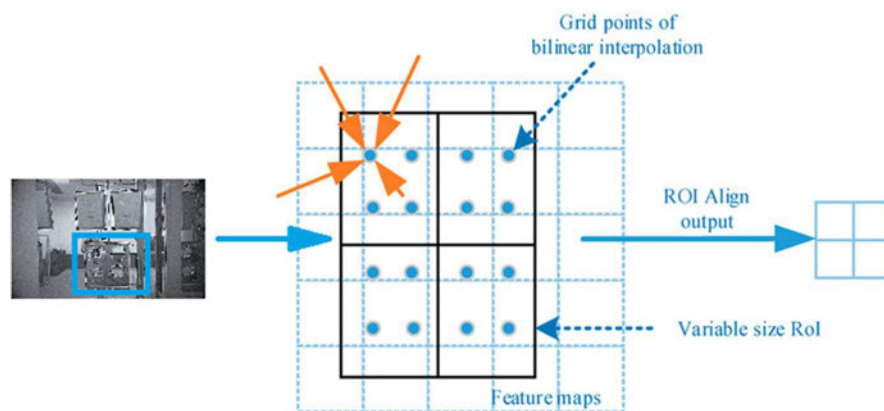


Figure 2.18. ROIAlign [20].

2.6.5 Mask Branch

Mask R-CNN extends on Faster R-CNN with the addition of the mask branch, that is able to generate segmentation masks for each instance [20]. Instance segmentation comes with its challenges as two operations need to be processed. Namely, correctly classifying the objects within the image followed by the process of the precise segmentation of that object within the image [20, 84, 85]. As Faster R-CNN was not designed for pixel-to-pixel segmentation; Mask R-CNN makes use of ROIAlign to accomplish this task. The use of ROIAlign has assisted with the issue of misalignment. Binary masks are applied to each class of object independently. This is done to ensure that each mask that has been predicted does not interfere with the remaining predictions within the image thereby allowing for high-quality precise segmentation masks [20]. The mask branch as illustrated at the bottom right in Figure 2.15, consists of multiple convolutional layers that take the features from ROIAlign as inputs. The refinements through these convolutional layers produce the required masks for each instance [84, 85]. With the addition of the parallel branch to the architecture; a multi-loss function follows the combination of the classification, bounding box, and mask losses respectively [20].

2.7 Application of Mask R-CNN

The application of Mask R-CNN has been well documented in literature for industrial, medical, environmental, construction and agricultural fields. The case studies below elaborate on the viability and its ability to be used in an industrial environment.

2.7.1 Case Studies

The use of instance segmentation models within the industry can vary in applications from quality control to counting and size estimation of an object and represents one of the classic tasks in computer vision. Essentially image segmentation can be considered as a clustering problem. In the work of [92], the focus was to detect florae in aerial images and it showed that the Mask R-CNN model was not only effective in counting but was also able to

determine the size of each vegetable based on the segmentation mask. They achieved a detection accuracy of ~78% for potatoes and ~98% for lettuces [92]. In another study [93], a Mask R-CNN model was used to identify, and segment manufactured parts in an assembly line. The parts were recognized and segmented with an accuracy of 87.5%. Recently, in a study conducted by [94], the challenge of accurate instance segmentation to determine tree density within a forest was highlighted. While the paper does not consider the approach of utilizing a Mask R-CNN model, the challenge presented shows similarity within the manufacturing industry for determining the utilization of space on an assembly line or a parts hanger as in the case of this study. In [95], the study involved counting livestock using the Mask R-CNN model. In all the defined categories, it resulted in an Average Precision (AP) of ~95% and an overall accuracy of ~93%. In comparison to other approaches, such as the You Only Look Once (YOLOv3), Faster R-CNN and Single Shot Multibox Detector (SSD), Mask R-CNN was the superior model [95]. Further, in a research study based on counting bacterial growth within a petri dish whereby the illumination was controlled throughout the experiment, the highest scoring model was achieved using Mask R-CNN. Interestingly, it achieved an AP of 93.8% and a mean absolute percentage error of 2.3% [96]. In another unique study based on pest identification and counting, a modified version of the Mask R-CNN model was used. The model was able to achieve an AP of 80.2% from 79.6% and was the highest-scoring model when compared to Faster R-CNN, YOLO and SSD500 [97].

In a study conducted by [98], the application of Mask R-CNN for defect identification in printed circuit boards (PCBs) was evaluated. Two different datasets were used to test the efficacy of the model which included missing vias and electrical short-circuits of copper tracks on the PCB layers. An impressive high accuracy within the range of 95% and 99% was achieved. Hence, based on the diverse range of applications that the Mask R-CNN model has successfully been applied; we propose its use in this study. A work [99] relating to cabin pose measurements; discusses the challenges of using traditional image segmentation methods which may be influenced negatively by the industrial environment. It is explained that this negative influence may affect the recognition effectiveness. During the study, it was explained that the choice of hyperparameter has a significant influence on the results of the model. According to the authors, the use of deeper neural networks contributed to improving

recognition accuracy. The Mask R-CNN model was able to achieve an overall accuracy of 98.84% using the ResNet101 network, which outperformed previous models.

An article [100] describes the challenges in the industrial fiber cloth process. Stains and defects are produced as the material is being processed, thus impacting sales negatively. The manufacturer relied on manual inspection and classification of defect types. The authors explain that due to visual fatigue, classification errors are prevalent. This prompted the authors to use Mask R-CNN with the combination of the FPN and the ResNet101 network to decrease the impact. A learning rate of 0.001 and 0.0001 are used within this study. The authors concluded after achieving an overall accuracy of ~87% that the model's results were acceptable to be applied in the production process of filter cloths.

The Mask R-CNN model was used in a robotic system [101]. The authors describe instance segmentation as a fundamental skill in robotic or autonomous systems. The main objective of this paper was to identify and enumerate the object that was detected in the image. The use of object detection was implemented for a pick-and-place application. The authors were able to achieve a mAP of 35.1% using a modified Mask R-CNN model. Furthermore, the model was able to achieve an error rate of 0.82%. The authors concluded that the Mask R-CNN had achieved high precision and recall rate and performed strongly in the robotic application. The main challenge noted in this study was related to occlusion due to objects being very close to one another. The use of instance segmentation was critical in addressing this challenge. A study [102], conducted on the application of Mask R-CNN used for automated identification of oil well sites using satellite imagery aimed to improve the efficiency and accuracy of detecting the sites. The main challenges included the use of images with smaller details. Atmospheric conditions did contribute to reduced performance. The oil sites themselves were able to be confused with other similar-looking features. The detection on the sites as compared to the area surrounding it did pose an additional challenge. A mAP of 63.02% was achieved which was higher as compared to Deeplab v3+ and HRNet v2.

Human intervention within a process may sometimes be difficult to implement. This challenge was noted in [103], where a robot with a computer vision system was used to

detect the stones and remove them during inspection. Artificial light was used to obtain the images during processing. ResNet101 was chosen as the network backbone. The authors mentioned a 92% average detection precision rate and a recall rate of 90%. The authors stated that through their literature review, inspection in the same process usually focused on defect detection rather than classification as it posed more challenges. These studies were based on the YOLO framework. The focus with the use of Mask R-CNN was based on robust and accurate detection, with the possibility of detecting objects in low light conditions. The study methodology provided a dataset of 280 pictures in total, where the VGG Image Annotator (VIA) tool [104], was used to annotate each image. The model was tested in a real-world environment where the authors stated that the model was able to detect objects in low lighting conditions. It was concluded that the model was able to be used in real time.

Occlusion has been a challenge in computer vision applications as noted in [105]. Moreover, accurate length estimation has been stated as a challenge in aquaculture environments. The study proposed the use of Mask R-CNN to determine the length of fish. The mask branch was used to eliminate fish which were incomplete within the image and only determine the length of fish which was fully available in the image. The results from this study showed a mean relative error rate of less than 5%. The authors concluded that the use of the Mask R-CNN model may be applied in the aquaculture environment. A similar challenge in this research is present within the industrial environment. Having objects that are incomplete within an image may contribute towards false detections. Quality assurance usually requires human intervention where sensors or hardware are not available. This is usually done through the physical inspection of the part for defects. Defects may not only be an additional cost within the process but may pose a hazard for the user or customer that may use it. This challenge has been stated in [106]. The authors mention that the use of trained workers to inspect defects is inefficient and inconsistent. Traditional image classification methods may be able to solve this problem, however, these methods lack the robust capabilities required to perform within an industrial environment. The authors state that to be able to acquire usable data from the field, a fixed camera within the process is required. The use of computer vision applications within an industrial environment should only be trained on objects that need to be detected which would economically benefit the requirements of the company. The study achieved a mAP of 87.7% using an improved Mask R-CNN model with the

ResNet101 backbone architecture. The authors did highlight that due to the limitations of the selected GPU, the tuning of all hyperparameters would have not been viable to achieve optimum results. Hence the need for further studies on the model through optimal parameter selection. This need for further studies has seen an increase in the use of deep learning models being used in different types of industrial environments. This was noted in [107], where the authors mention that the implementation of unmanned cultivation and production methods in plant factories are aimed for the near future. Automatic plant growth monitoring was stated as an important technology required for further development in large-scale plant factories. The use of Light Emitting Diodes (LEDs) were used in the environment to ensure that the enhanced area where the images were obtained would produce high-quality data. The dataset was resized to avoid complexity while being trained with the model. The model was trained with pre-trained weights through transfer learning. The study conducted a comparative analysis between various backbone architectures. This included ResNet50, ResNet101, EfficientNet_B0 and MobileNet v3. The different networks were all trained on the same dataset. The results concluded that ResNet101 achieved the lowest losses of 0.09% as compared to the other stated networks. The ResNet101 was able to achieve a mAP of 90.4% and was also the highest when compared with the other networks. The authors concluded that the use of Mask R-CNN with the ResNet101 backbone provided the most superior results in the study.

After fine-tuning the model, a mAP of 91.32% was achieved. A comparison was done with existing models, which included PolarMask, BlendMask and Segmenting Objects by Location (SOLO). The main advantages of PolarMask include high efficiency with a simplified process. The disadvantage was the lack of robustness for complex cases. BlendMask's advantages included the high precision with overall better performance relating to smaller objects while the disadvantages were noted as challenges relating to more complex applications. SOLO was noted as being highly efficient while performing poorly when being applied to detect smaller objects. The Mask R-CNN model was deemed highly precise in segmentation with the added advantage of being able to adapt to different objects that it was trained on. The disadvantage includes the high computational cost of training and testing the model. Although the food manufacturing process is not similar to the industry in this paper, the challenges are similar and may be related to general industrial challenges. The

following study conducted within the food manufacturing industry [108] indicated the use of deep learning models has seen an increase in interest within the food manufacturing industry. The main challenges for the use of such models are related to the increase in food quality requirements. This challenge is specific to insects that may damage or destroy crops. The need to monitor these insects may result in the reduction of crop destruction. The authors state that the need for manual intervention may be a time-consuming process, thus the study proposed the use of Mask R-CNN as the deep learning model for the food manufacturing field. The use of 165 images was used as the dataset in this training. A set of image augmentations were applied and included the rotation and zooming of the images. The training dataset was annotated using the VIA tool. A comparative analysis was performed between different models that ran on the same dataset. These models included the R-CNN, Faster R-CNN, RetinaNet, SSD and Mask R-CNN.

Mask R-CNN with a ResNet101 backbone network was able to achieve a mAP of 96.89%, being the highest-performing model compared to previously stated models. The study illustrated that with the use of a deeper network such as ResNet101, the detection accuracy was higher as compared to the other networks. The authors did state that Mask R-CNN does require less time to train and validate as compared to other models. Further, it was able to achieve a training time of 3.1 hours and a testing time of 0.47 seconds, which was better than the other models on average. These results are dependent on the hardware and the dataset that was used within the study. SSD was noted as being slightly faster during testing, achieving 0.36 seconds. The authors concluded that the use of the model was able to assist farmers in cost saving and reduce damage to the environment through the use of the model in the fields. A paper providing a comprehensive literature review based on the deep learning models used for product recognition [109]. The challenges and techniques used are presented in the paper. Manually locating a barcode on a product due to uncertain printing positions was stated as a challenge when trying to determine the product at the counter. According to the survey [109], 45% of customers had complained about the difficulties related to scanning products. RFID was mentioned as being used in the field to identify products. However, the use of RFID is met with its challenges that relate to scanning multiple products. Products using this technology may produce high error rates due to radio waves being blocked or signals influencing one another as stated by the authors. Another well-known problem relates

to the cost of RFID tags and the difficulty with recycling the tags. These challenges may inadvertently increase the cost per product if incorporated within the manufacturing process. The authors noted that with the advancement of camera technologies, the accessibility to image data has become more readily available. This has prompted the use of computer vision applications within industrial and retail sectors to reduce manual identification.

The process of image recognition systems according to the authors follows a step-by-step process. This process includes acquiring the image dataset, which would be through the use of cameras. The images would go through a preprocessing stage where it would be enhanced for better feature extraction. The feature classification process provides the output of the recognized object within the image. The main challenges relating to the use of deep learning models for product detection were summarized in four aspects. The first challenge included large-scale classification. The number of products needed to be identified may exceed the capabilities of the traditional object detectors. While most object detectors were evaluated on the COCO or PASCAL VOC dataset, the maximum number of categories were noted as 80 and 20 categories respectively at the time of this paper's publication [109]. The results gathered from those datasets may not have shown an accurate result to model performance to categories of products found in the retail industry as it surpassed the amount. The second challenge that was noted related to the data limitation when using deep learning models. To create a dataset with a huge, annotated image dataset is time-consuming. The authors mentioned that the majority of training data has been acquired under ideal lighting conditions where product detail is sufficient. This, however, is never the case in the practical environment.

Another challenge that is of high interest is related to intraclass variations. This is the challenge to distinguish between sub-categories due to each product sharing similar features with minimal variation in the overall appearance of the product. The authors have stated that this challenge has also been a popular topic within literature. The environment greatly affects the overall accuracy of the model when identifying intraclass objects. The final challenge that was noted by the authors related to the flexibility of the model, as new products are being added or removed from production. The question of having a pragmatic model with the ability to adapt to these changes was raised. The authors proposed techniques to address

these challenges. The main problem of image classification lies in feature extraction, through the use of extracted features. Objects can be classified into groups. This has been addressed through the use of a deep learning model and addresses the challenge of large-scale classifications. Through the use of data augmentation, the challenge of data limitations can be overcome. The authors state that gathering large amounts of data may be time-consuming and expensive. Data augmentation is a common practice in deep learning model training for trying to handle the limitations of data. The technique takes a small set of data and applies image augmentation to enhance and enlarge the small set of data initially used. The challenge of fine-grained classification relating to intraclass variations may provide positive results if managed. Furthermore, the challenge of intraclass variation may worsen due to blur, lighting, deformation and the alignment of the product as noted by the authors. This challenge was addressed with a context awareness technique. Context information may be useful for model performance increases through the ability to provide information relating to spatial and semantic information between the classes of objects in an image. These intraclass variations may be detected if the different classes are next to one another in the image training dataset. The authors state this technique addresses the challenge not only through the visual features of the classes with these intraclass variations but also through the spatial location of the objects in the image. The study compared the performance of deep learning models based on the Densely Segmented Supermarket (D2S) dataset. The dataset provided pixel-wise annotations on 21000 high-resolution images of groceries. Mask R-CNN achieved an mAP of 89.8% and was the highest-achieving model as compared to FCIS, Faster R-CNN and RetinaNet. The comprehensive literature review provided extensive information relating to the challenges. Mask R-CNN was highlighted as a model with robust capabilities being able to address these challenges while achieving high precision scores.

2.8 Performance Evaluation

The performance of Mask R-CNN can be evaluated quantitatively using Equations 2.3 – 2.5. The model's performance has been compared with the previous state-of-the-art models and comparisons on key features will be elaborated on in the following section. The use of Mask R-CNN has been chosen in this study based on the performance of the model over previous models.

2.8.1 Performance Criteria

The objective of evaluating the model's performance is to provide information on its robustness capabilities. Furthermore, areas for improvements within the model based on the unique dataset used. The evaluation criteria used similar equations as mentioned in [110]. The AP determined the efficacy of the model. The precision and recall values were needed to generate a precision versus recall curve. The area under this curve would provide the percentage of the AP of the model after being trained. The AP relates to the average over all classes that were used in the model. This description is given in the COCO metrics description [111]. Thus, this would traditionally be called the mAP. Equations 2.3 – 2.5 provide the definitions relating to each criterion.

The model precision is calculated as:

$$Precision = \frac{TP}{TP+FP} \quad (2.3)$$

where True Positive (TP) represents the objects that were detected by the model and are the correctly detected class. False Positive (FP) represents objects that were incorrectly classified

The model recall is calculated as:

$$Recall = \frac{TP}{TP+FN} \quad (2.4)$$

where False Negative (FN) are objects with a positive class that were present in the image but were not detected by the model.

The area under the precision-recall curve represents the AP of the model, where p and r represents the precision and recall, respectively:

$$AP = \int_0^1 p(r)dr \quad (2.5)$$

2.8.2 Benchmarking and Datasets

Object detection has been described as a fundamental problem in [112]. This paper has highlighted that the main objective of object detectors is to detect and localize objects belonging to a predefined class. The number of applications that have been developed using deep learning models has span multiple fields. These fields include surveillance, robotics augmented reality and the use of applications that may lighten the repetitive work required by humans. The advancement of deep learning models has mostly used benchmark datasets that are readily available online. These datasets include the COCO, PASCAL VOC and ImageNet datasets. The PASCAL VOC is one of the most popular datasets available. The dataset is prevalent in the literature. The dataset has a total of 20 categories with a training and validation dataset of 11530 images containing 27450 ROIs [113]. The classes contain objects such as cars, people and various animals.

ImageNet is an image dataset that contains a total of ~1.2 million training images, 50 thousand validation images and 100 thousand test images with a total of 1000 object classes [114]. This dataset is considerably larger as compared to the other datasets. This dataset does not provide annotations for the images in the dataset as compared to the other datasets. However, it is noted in [112], that this dataset is used in pre-trained models before being finetuned on another dataset. The COCO dataset is the most recent and most popular dataset that contains both object detection, segmentation and captioning datasets. The dataset has a total of 330 thousand images for training and validation datasets with ~200 thousand labelled images. There is a total of 80 classes with a total of 1.5 million object instances. The dataset that is of importance in this work relates to the segmentation masks that have been provided for every instance in the dataset [111]. The main aim of these datasets that are readily available to the public is solely meant for the advancement in research relating to the field of object detection capabilities. These datasets are released to the public for educational and research purposes and not for commercial use [113]. The use of these datasets for benchmarking deep learning models does raise the question of whether the dataset itself has been vetted to be viable as a benchmarking tool. This question prompted the authors of [115] to investigate further. The authors state that the dataset used does contain errors and imperfections coming from the annotation process. However, the authors claim this error is common amongst large datasets done through human intervention. These errors, especially

in ground truth annotation, may pose as noise during the learning process. Throughout the study, specific types of errors were mentioned. These errors included in the study noted that the coarse boundaries aren't able to wrap around the object's edges as tight as required. This may affect the pixel-precise masks of the objects in certain classes. The masks were also noted as not having holes in objects that did require it. This was due to the lack of capability of the provided COCO annotators. Masks were not able to handle occlusion consistently as objects that were too close to one another and were annotated together. Some masks of certain objects in an image were missing or a single mask was annotated to represent the rest of the objects. Duplicate masks were also discovered in the dataset. The authors have stated that a reassessment for the datasets may be needed due to these errors made through human intervention during the annotation process. However, the results in the study showed an AP difference of 4% using Mask R-CNN based on their improved dataset as compared to the COCO dataset, with the latter being lower. This uncertainty in dataset volume and quality is an ongoing topic of research within the computer vision research community as mentioned by the authors.

2.8.3 Model Comparisons

A comprehensive comparative analysis between the use of YOLO and Faster R-CNN in smart industries highlights the applications, challenges and opportunities of each model [73]. The authors compared the key features of the models as compared to the application function. YOLO was deemed as having the capability of real-time object detection and positively contributing to collision avoidance and pedestrian safety. Faster R-CNN was described as offering high accuracy scores relating to applications that focused on precision rather than speed alone. This would assist with understanding the situation when presented with obstacle avoidance. YOLO was able to detect cars quickly in autonomous driving systems while Faster R-CNN was able to accurately identify with a higher accuracy score as compared to YOLO. YOLO can handle complex scenes effectively with multiple objects. Faster R-CNN has shown a higher performance in areas which has occlusion, crowded situations and challenging conditions as compared to YOLO. YOLO's primary advantage has been described as having a faster real-time processing capability as compared to Faster R-CNN. While YOLO might be quicker in detections, Faster R-CNN can provide a far greater score in terms of accuracy and precision as stated by the authors. Both YOLO and

Faster R-CNN may be customized for different fields or environments. YOLO's architecture allows for greater flexibility. This allows for greater adaptability and changes to be made when coming to fine tuning for an application. In terms of algorithm robustness, Faster R-CNN may be deemed more robust as compared to YOLO due to the higher accuracy scores as stated by the authors. The authors in [76], outlined the advantages and disadvantages of various backbones that may be used with the models. Alexnet brought about the introduction of having consecutive convolutional layers with the use of down sampling was seen as an advantage. The network's use of non-linearity due to the Relu function within the model resulted in better performance. With the use of fewer parameters in the network, the computational resources and complexities were decreased. The limitations that were noted by the authors related to the use of large receptive fields during the feature extraction process.

Even though the network introduced the use of consecutive convolutional layers, the accuracy through the network was still deemed as being low. The network did become memory intensive due to the overlapping of pixels from the blocks. Alexnet's wasn't as compatible as compared to other models and lacked adaptability when applied to complex tasks as described by the authors. As with technological advancements, VGGNets was introduced. The network was deeper as compared to AlexNet with the use of small convolutional filters. VGGNets were deemed as being able to generalize well with different datasets that were used in the training and testing process of the model. The network did, however, have limitations. More parameters were making the network more computationally intensive thereby leading to the network being slower to train. The use of this network did reveal a problem described as the exploding gradient due to having a deeper network. The network was stated as being specific to particular applications.

ResNet addressed the challenges related to the exploding or vanishing gradient due to deeper networks through the use of skipped connections. There were fewer parameters used hence making it faster as compared to VGGNets leading to faster training times. The network was able to generalise well with a broad range of datasets. ResNet was computationally heavier as compared to VGGNets, requiring more floating-point operations per second. The architecture was more complex as compared to VGGNets. The review in [76], provided a

detailed comparison between the deep learning models with their key advantages and limitations.

R-CNN was stated as simple to use in applications for classification and provided the foundation for future improvements to the field. The limitations were used as areas that needed to be improved going forward. This was related to the 2000 regions that needed to be classified separately, leading to high computational burden during the training process. Due to the model being slow, it was deemed not viable to be used in real-time applications. The selective search algorithm used had no learning capabilities when addressing the regional proposal process. These limitations as mentioned are investigated in the subsequent sections. SPPNet was faster as compared to R-CNN due to multi-scale allowed inputs to the feature extractor through the application of the spatial pyramid network. The limitations of SPPNet were carried over from R-CNN, namely, the use of the selective search algorithm adding to computational time resources. SPPNet does not provide an end-to-end learning architecture.

With the introduction of Fast R-CNN, model performance was greater than that of SPPNet through the introduction of ROI pooling. The addition of bounding box regression in parallel to classification was a key feature along with the ability to use RPN in conjunction with CNNs. The limitations were similar to the previously mentioned model relating to the selective search algorithm and bottlenecks due to the region proposals process. Faster R-CNN introduced the use of RPN instead of the selective search algorithm, this was a substantial achievement as compared to Fast R-CNN, hence decreasing the computation time resources needed during the training process.

Another key feature was the introduction of the anchor boxes and the multi-task losses. This made the Faster R-CNN model quicker and more accurate in comparison to Fast R-CNN. The limitation of Faster R-CNN was noted as requiring several passes of an image through the model to extract an object. The model did have difficulties in detecting smaller objects due to the use of a single deep layer for predictions. There was a limitation relating to the class imbalance of data leading to biased predictions or lower performance scores. Mask R-CNN improved performance through the replacement of the ROI Pool layer with the

ROIAlign layer which was used in the model previously. A comparative analysis of the Mask R-CNN model with multiple backbones was conducted in [116]. As previously mentioned, the backbone architecture of the Mask R-CNN mode consists of the FPN, RPN and the ROIAlign. The paper [116] investigated the CNN backbone used in the model, specifically, ResNet101, ResNet50 and MobileNetv1.

The models were compared using the PennFudan dataset [116]. The outcome results have shown that ResNet101 supersedes the other backbone CNNs having achieved a mAP of 98.3% as compared to the other backbones. ResNet was developed to provide a deep network without having the vanishing gradient problem. This was through the acceleration of convergence of the network. ResNet101 was able to outperform other networks when comparisons were made to the RPN class losses with a value of 0.60 for the training and 0.69 for the validation, which was lower in comparison to the other networks. This refers to the losses that are associated with RPN anchor classifiers providing information on whether an anchor box contains an object or not. The loss of the RPN bounding boxes were 8.03 and 7.71 for the train and validation dataset respectively, which was lower than the other networks. The loss of the Mask R-CNN class losses was lower as well with a value of 7.41 and 6.74 for the training and validation dataset. The multi-task losses were in total lower than the other networks, providing evidence that a deeper network using ResNet101 achieved a better performance. The authors concluded that ResNet101 had performed better as compared to ResNet50 and MobileNetv1 in terms of mAP, Mean Average Recall (mAR) and F1- scores. The model performance of Mask R-CNN may not only be compared with other deep learning models but also with the other backbones using the same model. The dataset does influence the outcome as mentioned. Table 2.1 provides results on the mAP based on the COCO dataset using various models and backbones following the two-stage detector configuration. The table illustrates the performance of Mask R-CNN superseding other deep learning models.

Table 2.1. COCO dataset two-stage based models.

Method	Backbone	Data	rnAP	mAP [0.5,0.95]
Fast R-CNN [18], 2015	VGGNet-16	train	39.3	19.3
Fast R-CNN [17], 2015	VGGNet-16	train	35.9	19.7
Fast R-CNN [117], 2015	VGGNet-16	train	40.3	20
Fast R-CNN [117], 2015	VGGNet-16	train	39.9	20.5
Faster R-CNN [18], 2017	VGGNet-16	train	42.1	21.5
Faster R-CNN [18], 2017	VGGNet-16	trainval	42.7	21.9
OHEM [118], 2016	VGGNet-16	trainval	42.5	22.6
SIN [119], 2018	VGGNet-16	train	44.5	23.2
ION [117], 2015	VGGNet-16	train	43.2	23.6
ION [117], 2015	VGGNet-16	train+S	46.3	24.6
OHEM++ [118], 2016	VGGNet-16	trainval	45.9	25.5
Faster+Cascade RCNN [120], 2018	VGGNet-16	train	44.3	26.9
Faster R-CNN [58], 2016	ResNet-101	train	48.4	27.2
Noc [121], 2015	ResNet101	train	48.4	27.2
R-FCN [122], 2016	ResNet-101	train	48.9	27.6
R-FCN multi-sc-train [122], 2016	ResNet-101	train	49.1	27.8
MLKP [123], 2018	ResNet101	trainval35k	52.4	28.6
R-FCN [122], 2016	ResNet-101	trainval	51.5	29.2
R-FCN multi-sc-train [122], 2016	ResNet-101	trainval	51.9	29.9
R-FCN w Cascade RCNN [120], 2018	ResNet-50	train	49.9	30.9
R-FCN multi-sc-train, test [122], 2016	ResNet-101	trainval	53.2	31.5
SMN [124], 2017	VGGNet16	trainva135k	52.2	31.6
CoupleNet [125], 2017	ResNet-101	trainval	53.5	33.1
ION [117], 2015	VGGNet-16	trainval35k+S	55.7	33.1
MPN [126], 2016	VGGNet16	train	51.9	33.2
R-FCN+Cascade RCNN [120], 2018	ResNet-101	train	52.6	33.3

ACoupleNet [127], 2019	ResNet-101	trainval+S	54.1	34.1
CoupleNet msc train [125], 2017	ResNet-101	trainval	54.8	34.4
Faster R-CNN G-RMI [128], 2017	Inception-ResNet-V2	trainval	55.5	34.7
Faster R-CNN+++ [58], 2016	ResNet-101-C4	trainval	55.7	34.9
ACoupleNet mst train [127], 2019	ResNet-101	trainval+S	55.7	35.4
RDSNet 600 [128], 2019	ResNet-101	trainval35k	55.2	36
Faster R-CNN w/ FPN [90], 2016	ResNet-101-FPN	trainval35k	59.1	36.2
FPN FRCN [90], 2016	ResNet-101	trainval	59.1	36.2
Faster R-CNN w/ TDM [130], 2016	Inception-ResNet-v2-TDM	trainval	57.8	37.3
TDM [130], 2017	Inception-ResNet-v2-TDM	trainval	57.8	37.3
Deformable R-FCN [131], 2017	Aligned-Inception-Resnet	trainval	58	37.5
RDSNet 800 [128], 2019	ResNet-101	trainval35k	58.5	38.1
Mask R-CNN [20], 2017	ResNet-101-FPN	trainval35k	60.3	38.2

2.9 Challenges and Limitations

In general, challenges relating to object detection applications and instance segmentation have been mentioned in multiple literature. A survey discussing deep learning models used for generic object detection applications [132], makes mention of the challenges relating to intra-class variations. The survey states that object instances that may be detected within intra-class variations may be considered a model robust if the model can address this challenge. Imaging conditions related to the environment may impact the results negatively of a model [132, 133]. The variability of an object may pose a challenge in itself as the object may appear with various colours, sizes and shapes. This may make it challenging for the model to be able to generalize well. Lighting conditions and occlusion may further complicate the outcome of the results as details may be missed in the image with the unfavourable conditions [13, 132]. Data annotation has been a challenge in all object detection applications as the task is time-consuming and may lead to human error due to fatigue or ambiguous requirements. The quality of data annotation has a direct impact on the model's performance [134, 135]. For applications that require live data to calculate a response, real-time performance from a model becomes crucial. However, the balance

between accuracy and speed is a well-noted challenge in literature and is dependent on the efficiency required in the application [19, 62]. Detecting small objects based on a unique dataset may pose an additional challenge due to lower resolution making it difficult to identify features [90, 136]. For certain classes in a dataset, there may be an overrepresentation. This overrepresentation may bias the model resulting in false positives and no detections. Techniques like augmentation may be used to correct this issue as mentioned earlier in the literature [137, 138].

2.10 Summary

The literature review expanded on key achievements in deep learning and computer vision. Focus was placed on the Mask R-CNN model and its capabilities in an industrial environment. Various object detection models were analysed, and their strengths and limitations were highlighted. Insights were provided into data selection, evaluation metrics, and hyperparameter tuning for enhanced model accuracy. Furthermore, challenges in real-world settings, such as lighting and occlusion, were considered to shape the foundation in the methodology chapter of this study.

CHAPTER 3 Methodology

3.1 Introduction

The current problem as mentioned previously stems from the error in data produced through human error. This data is not live and there is a delay in information travelling from one department to another. This challenge has prompted the investigation and application of the use of a deep learning model that could be used to provide live data to all relevant stakeholders in the firm. The main objective of the implementation of the deep learning model would be to provide accurate data with the lowest error rate from the model. The significance of solving this problem would not only provide the required data but this would streamline the manufacturing process relating to response management.

Through the literature review, enough evidence has been provided to support that the Mask R-CNN model has sufficient capability to meet these objectives. As the study was done in an industrial environment, environmental conditions were maintained as much as possible to provide stability to the dataset and the performance of the model during the training phase. The data collection method will be explained with the hardware used to retrieve the dataset. The dataset went through a preprocessing phase where the benefits and methodologies are explained. Elaborations are given on the calculations and decisions used in the model. The training methodology is explained with details on the hyperparameters and their effects on the model. The multitask losses is explained corresponding to the hyperparameter changes. During the evaluation phase, performance measures were used to determine whether a model was performing at a desirable level. The use of the model is not possible without correctly selected hardware. The types of libraries and the technologies used in the study will be further elaborated on. A methodology is provided for the iterative process of training and testing. The section will end with the limitations that were found during the study.

The methodology provides a critical detail on how the results were achieved. The training and testing process of the study was time-consuming. Due to this challenge, a structured methodology had to be followed to complete the study. The research methodology has

contributed to the enamel coating manufacturing process and may provide other researchers with information on the implementation of deep learning models for similar applications. The implementation of the deep learning model revolved around the question of achieving the highest possible accuracy in the environment presented. The main challenge of detecting objects with intra-class variations and object class changes in location on the conveyor line is addressed in the methodology. As with the advancement of technologies within industrial firms, traditional methods for manufacturing and production may soon become obsolete and unable to cope with the rising customer demand. Therefore the need for research in new technologies is required to be viable within such an environment. There are multiple literature studies available on deep learning models using online accessible datasets but there are few literatures available providing results within an industrial environment.

3.2 Research Design

This study follows a quantitative approach due to the reliance on data retrieved from the evaluation metrics derived from mathematical equations. The research focuses on measurable data, which includes:

- 1) Recall and precision: Precision is calculated through the number of true positive detections in comparison to the overall positive detections. Recall is calculated through the ratio of true positives in comparison to the overall number of all detected objects which would include true positive and false negatives. This equation is shown in Equation 2.3 and Equation 2.4, as previously described.
- 2) mAP: Uses the area below the precision-recall graph. This is the average precision across all classes that were used in the dataset based on a predefined IoU. This is given in Equation 2.5.
- 3) IoU: This metric measures the overlap between the bounding boxes that have been detected by the model and the ground truth that has been annotated in the dataset.
- 4) Overall accuracy: This metric determines the viability of the model in the selected environment. A confusion matrix is used to determine the overall accuracy of the model based on how many correctly classified objects were detected on the test dataset in comparison to the total number of objects annotated in the dataset.

Most literature follows this approach in producing their results as described in the literature review and illustrated in Table 2.1. Following Table 2.1, benchmarking follows the quantitative approach as well. This is based on the datasets that are available online which include challenges for the public to participate in [134, 135]. The common datasets that have been mentioned in the literature review include the COCO dataset, the VOC dataset and the ImageNet dataset. A better-performing model is represented as having a higher mAP as illustrated in Table 2.1. However, once the research is completed and the results are satisfactory, the model may follow a qualitative approach during error analysis from the model in production. This is due to the live images not being annotated while being processed through the trained model. The qualitative approach may require manual analysis of the errors made by the model such as false positives or to conduct surveys to gather feedback on the usability of the model in the selected environment. A qualitative analysis was provided in the literature review under the model comparison sub-section. For this research's methodology, the quantitative approach would be used only. The choice for this approach can be seen in the literature review whereby most researchers prefer to provide their findings through numerical values to prove a gain or loss by using a deep learning model in comparison to other models or with other selected datasets. The use of this quantitative approach allows for the object detection comparisons, testing and validations of the hypotheses through numerical evidence.

Data augmentation techniques follow the adjustment of fixed values that have a major influence on the enhanced images. This includes rotating, flipping, colour adjustments and scaling. These changes also follow a quantitative approach as changes influence the evaluated metrics. Adjustments may then be made based on these evaluation metrics. The iterative process would determine a trend from the collected results. Tuning hyperparameters has a major influence on model performance being a topic within the computer research field that doesn't have a fixed approach. The results after fine-tuning may provide more information on how the hyperparameter may affect the model based on the dataset and the backbone selected for the required application. As every application is different, the same approach may not provide the same outcome for another application. The main aim of following a quantitative approach advances the computer vision community by providing

information on achieving reliable and reproducible results in developing robust models for complex applications.

While all evaluation metrics and results are based on a quantitative approach, the methodology for achieving a greater result follows a pragmatic approach. This approach focuses on the practical outcome of the model based on the inputs. As real-world applications in this study are non-linear, an iterative approach was used to solve specific problems within the study [139]. This approach allows for the adaptability of the model in the industrial environment. The focus on solving practical problems in the study takes higher precedence compared to purely theoretical results [140]. To focus on this approach requires an iterative process with the aim of gaining on certain parameters. For this study, the focus was applied on the mAP and the overall accuracy over all classes while meeting the speed requirements of the conveyor line in providing live data. This approach integrates different disciplines to use a deep learning model to contribute positively to the industrial sector in providing a gain and calculating inefficiencies from a production line output. Rather than just providing results only based on the unique dataset. The pragmatic approach emphasizes adaptability to different conditions in various industries. The focus is placed on whether the model may be able to be fine-tuned or if object classes may be added to the existing dataset. How would the addition in data affect the performance of the model, would there be an issue with class imbalances [141]? The pragmatic approach also looks at the continuous improvement of a system through the feedback from the methodology. The collaboration with all key stakeholders may influence how challenges are solved in the deployment of the model [142].

3.3 Research Method

The data collection method followed a process that required a script to acquire the required dataset from the field through cameras that were mounted in a fixed position relative to the parts on the conveyor line. The setup consisted of a PC as the main computing resource for the training, testing and validation of the model, and to run the script to capture the images. The images were captured using a camera and were triggered using a proximity sensor that would provide a signal when the hanger's neck passed it. The same camera settings were used for all datasets used in the study. The images that were captured had a pixel size of

1920 x 1200. The study followed the CRISP-DM framework which included the collection of data, followed by data preparation to clean any unwanted images and preprocess through data augmentation. These images were then split into their individual class folders that would be used for the training, validating and testing dataset. The dataset followed a 80:10:10 ratio. Each class of images were then manually annotated using the VIA tool. The tool provided the ground truth for all the instances in the images concerning their individual classes. The Mask R-CNN model was downloaded from a GitHub repository [33]. The Mask R-CNN model was used with different backbone networks, Hyperparameters were adjusted for the model with the images themselves being augmented before being processed through the model. The trained model was processed using the validation dataset to evaluate the performance of the model providing the mAP. The efficacy of the was processed using the testing dataset to determine the overall accuracy score which was determined through the confusion matrix at different predetermined thresholds. The outcome of the system was analysed and the hyperparameters were re-adjusted, followed by an iterative process until satisfactory results were achieved. Following this outline, the framework outlined in Figure 3.1 provides the framework to this research.

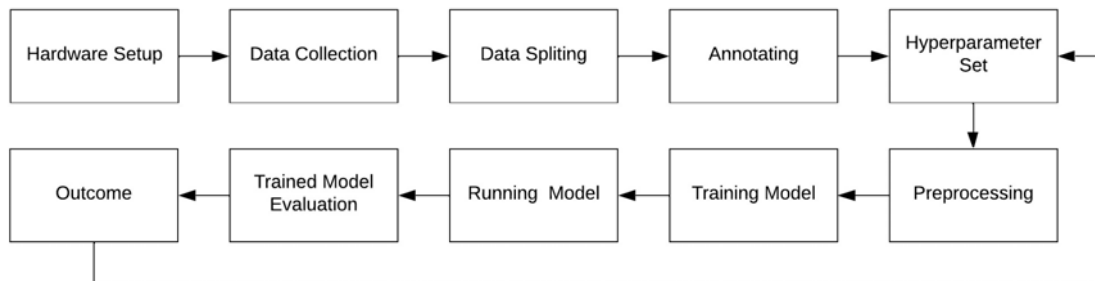


Figure 3.1. Object detection framework.

3.3.1 Hardware and Software

The hardware consisted of two cameras that were mounted on either side of the monorail conveyor, these cameras faced the objects at an elevated height to be central to the cluster of objects within the image frame. A Basler™ ACE acA1600-60gc CMOS camera was used in this study. The choice for this sensor was due to its 42 frames per second capability and

2.3MP resolution. The sensor's pixels are exposed at the same time of acquisition. At the end of exposure, the data is read out sequentially until all data has been read. This setup of the sensor assists with fast-moving objects [143]. This global shutter description is illustrated in Figure 3.2.

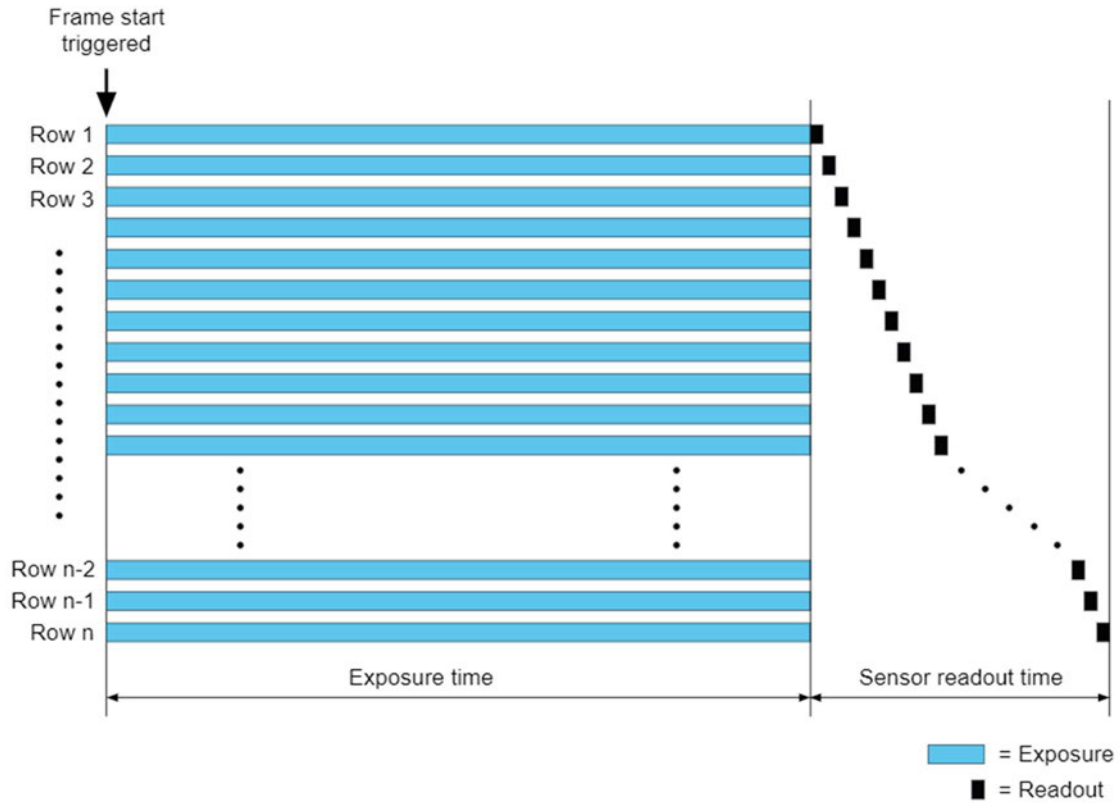


Figure 3.2. Global shutter type.

The computer used to perform the training, and validation on the Mask R-CNN model was accomplished with an Nvidia® GTX 1650 4 GB with 32 GB of RAM running an AMD® Ryzen 5 3600 6-Core processor. The need for a computer with a dedicated Graphics Processing Unit (GPU) lies in the use of the Nvidia CUDA® Deep Neural Network (cuDNN) library that utilises the GPU to accelerate the learning time related to deep learning networks [144]. A generic retroreflective sensor was used to provide the trigger signal. The sensor was connected to a Moxa® ioLogik E1200 I/O module. The module was used to translate the digital signal as a trigger in the computer. Figure 3.3 illustrates the setup that was used in the field.

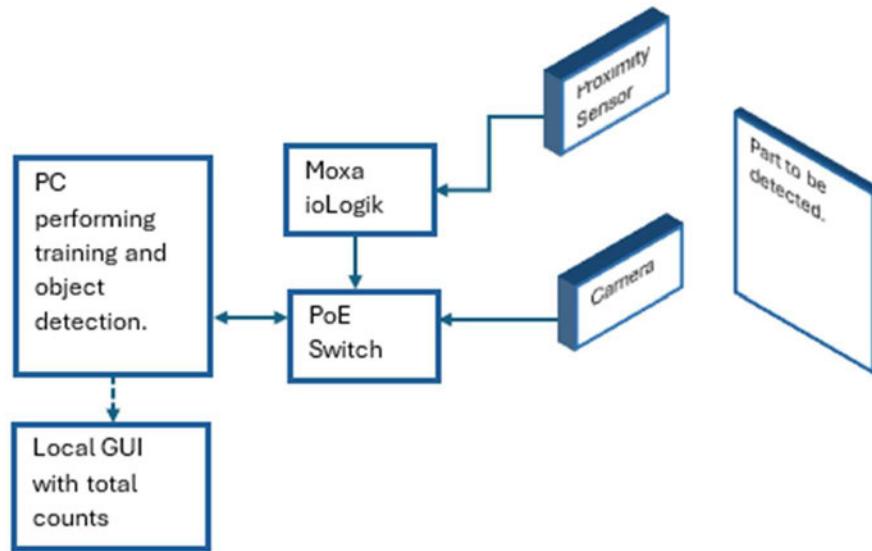


Figure 3.3. Hardware configuration of field devices.

The script used to capture the images initially was developed using Python using the ‘Pypylon’ library. The library allowed for the custom-created GUI to communicate with the cameras in the field. When the trigger signal was received, the grab command was activated which captured the images of the objects in the field. The script saved all images into the installed hard drives on site as they were captured. These images were displayed on the GUI as they were captured to fine-tune the focus of the camera lenses and make adjustments to any feature values that were available on the Pylon viewer software [145].

The Mask R-CNN model found on Matterport’s GitHub repository [33] is used in a wide variety of open-source projects with more than a few forums explaining how the model may be used for unique applications. The implementation of the model is achieved through the use of the Keras and TensorFlow libraries. The Keras library is a neural network Application Programming Interface (API) which is written in Python. This library can be able to run on top of TensorFlow in implementing neural networks during the prototyping phase [146]. TensorFlow is a platform for data scientists, software developers and educators to use the capabilities of machine learning through the use of data flow graphs. The architecture allows for algorithms to be described as graphs of connected mathematical operations. This framework allows for the model to make predictions through the preprocessing of data followed by the developed model being trained [147]. The Mask R-CNN model has key sections that need to be understood in the codebase. The configuration of the model is

defined by the hyperparameters and the settings that were used to train, validate and test. This includes the number of classes, learning rate, steps per epoch and epochs. The model definition of the code applies the Mask R-CNN architecture parameters. This includes the backbone, RPN and the head used for the classification relating to the bounding box regression and mask predictions. For the training process of the model, the dataset is first loaded with all the correctly annotated objects attached to each image. Data augmentation is then applied according to the predefined parameters. This step would define the loss functions and the optimization steps. For the inference process, the trained model would then be processed through a new set of annotated test images to predict the objects with their respective classes. The utilization of the utilities function provides the user the ability of the model to visualize the results of the model during each step of the process [33].

3.3.2 Data Collection Method

The dataset had 5 classes and contained a “train”, “validate” and “test” section. The dataset split following the 80:10:10 ratio. The number of images that contained the classes of objects is presented under the images column while the total number of objects within all the images is presented under the Region of Interest (ROI) column as illustrated in Table 3.1.

Table 3.1. Dataset split.

Objects	Physical dimensions	Train (80%)		Validate (10%)		Test (10%)	
Class	(mm)	Images	ROI	Images	ROI	Images	ROI
A	300×210	88	188	11	22	13	26
B	400×300	64	77	8	13	8	12
C	600×600	26	51	3	6	3	6
D	600×350	50	50	6	6	6	6
E	550×300	18	51	2	6	3	6

The data that was collected from the field used in the model was selected individually through a qualitative approach whereby each image was manually selected through observation by human means. Each image was scrutinized for occlusion, shadows, object misplacement, incorrect lighting conditions and obstructions. This method of selecting the dataset used for the training, validating and testing process was critical and time-consuming. The need for a good dataset as mentioned in the literature review, has a major impact on the model's performance. Once the images were selected, class folders were created with objects that had similar classes in the images. This was done to speed up the process of manually annotating each object within the object. This also allowed for quicker counting of objects that had been annotated in a certain class making our dataset that has been illustrated in Table 3.1. This method of defining the size of the dataset followed a quantitative approach. The difference in objects per class was done intentionally to follow the percentage of objects per class from the production line in one full production cycle. This choice of method would have provided more information on how the model can handle class imbalances.

A second set of data was added to the model after all results with the main dataset were collected. The second dataset had images that captured the object from different angles not seen on the production line. The idea behind this method was to provide more details of the object that may have not been seen from the fixed cameras. However, this method provided poor results when compared to the dataset from the fixed cameras only.

3.3.3 VIA Annotation Tool

The use of an annotation tool is a critical step in the object detection process. The main task of providing the correctly labelled data with its corresponding annotation of that object instance in the image has an impact on the performance and accuracy outcome after training and testing the model. Since the Mask R-CNN model provides the bounding box and mask prediction after inference, the need for good quality annotations relating to the mask is crucial in predicting the correct masks attached to the object. High-quality annotations ensure the model can learn accurately [132]. The use of high-quality annotations does allow for the model to better generalize in attaining a higher accuracy score with unseen data [134]. The tool that was used in this study was VIA is a manual annotation tool. The software is capable of annotating images, videos and audio. This software runs on a web browser and

was not installed, making it a lightweight tool. The tool is open-source and available for academic and commercial use. Once annotation is manually completed, a 'JSON' file can be saved. The JSON files are used in the Mask R-CNN model. The file provides the coordinates of the annotation that would serve as the ground truth in the dataset during the training phase. Figure 3.4 illustrates the annotation process completed using the VIA tool.



Figure 3.4. VIA annotation tool.

3.3.4 Field Setup

Figure 3.5 illustrates the setup in the field. An aluminium frame was used to mount the cameras and proximity sensors. Due to the enamel process requiring enamel powder to coat the parts before being transferred into the furnace, a computer cabinet was used to protect the computer. The location of data capturing and testing was done before the enamel coating process. This was chosen to control the part's surface as the parts after the enamel coating would cause variations due to the powder coating thickness on the surface of the part. The area after the furnace was not chosen due to significant heat leaving the part and the addition of reflection caused by the baked part.

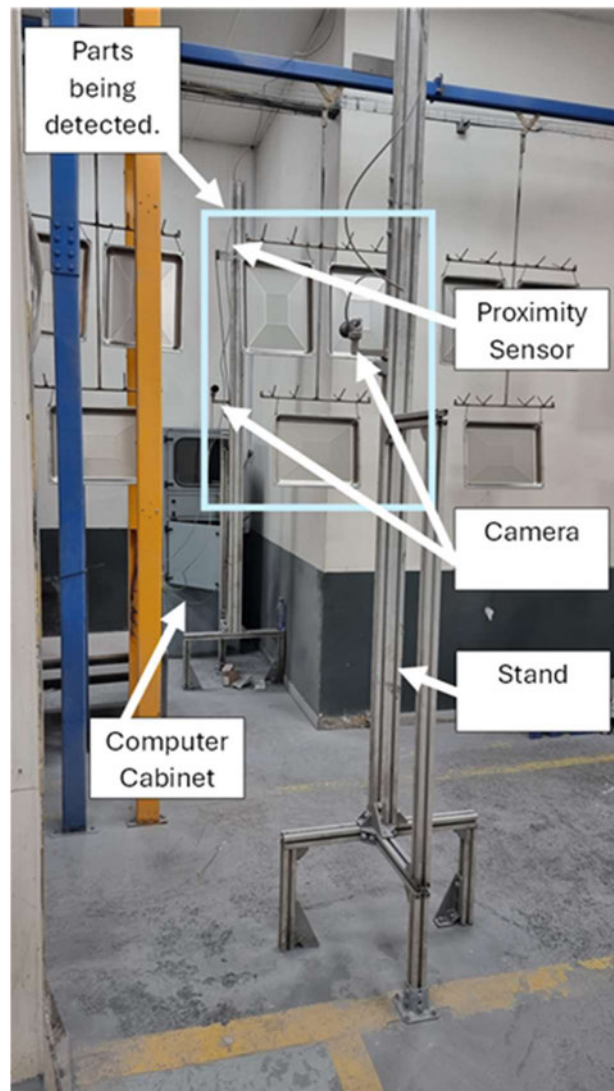


Figure 3.5. Field Setup.

Figure 3.6 illustrates the different classes that were used in the dataset during the training, validating and testing process. Figure 3.7 illustrates the group of classes that were used in the dataset.

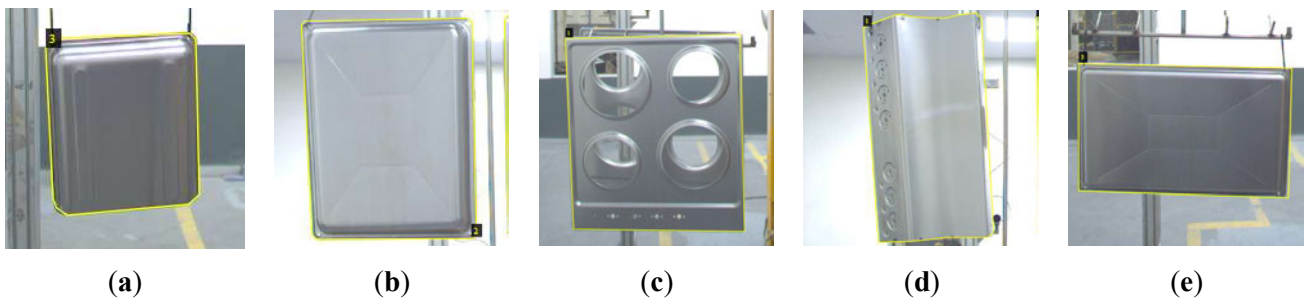


Figure 3.6. Classes used in the dataset with the alphabets representing the classes.



Figure 3.7. Random set of images from the dataset used in the model.

3.4 Data Preprocessing

Data preprocessing is required for training and is critical in maintaining the pixel value ranges in an image. The standardization relating to pixel values ensures that the model is not biased towards specific ranges of pixel values when being processed [15]. Techniques most used in data augmentation include rotating, flipping, colour enhancement and scaling. For this study, colour enhancements were used only. This enhancement does assist in the model's

ability to generalize well with new unseen datasets [148]. Improving the generalization ability in a model is regarded as a difficult task. This is referred to as the ability of the model to generalize with previously seen data versus unseen data. The difference between these metrics provides performance indications of the model on whether the model is overfitting or underfitting [148].

The method to discover if a model is overfitting or underfitting would be to plot the graph between the training and validation losses at each epoch. A model that generalizes well and is usable is a model where both the training and validation losses continue to both decrease with the increase in epochs. Data augmentation techniques focus on reducing these overfitting models to allow for better generalization. As mentioned in [148], focus on generalization may also be accomplished through looking at the model's architecture. This path has also brought about major improvements in the computer vision field which include dropout regulation, batch normalization, transfer learning and pretraining. However, data augmentation is said to focus on the root cause of the problem at hand which is the dataset itself. This idea stems from the assumption that more information may be able to be retrieved from the current dataset that is available for use in the model.

Data wrapping augmentation performs the task such as colour transformation. This process is done where the labels and annotations are kept in their original configuration. Image data for an RGB image is a stacked matrix comprising pixel values associated with an RGB value. Lighting conditions are one of the major issues which may cause biases during the training process. This challenge is addressed through colour space transformations also known as photometric transformations is common practice in the computer vision field. This is done by increasing or decreasing the pixel values per image by a constant value. This effect of manipulating the pixel values for the environmental condition is illustrated in Figure 3.8. The image dataset had gone through multiple preprocessing techniques, which included manipulating data augmentation parameters, as illustrated in Table 3.2. The preprocessing parameters were applied sequentially to each image. Images were augmented in a fixed arrangement to produce and ensure stability with results after testing. This defines the sequence of augmentation techniques that need to be applied to each image. The sequence of augmentation was applied according to Figure 3.8. The lighting conditions were

controlled during the collection of training, validating and testing datasets; however, due to the slight angle formed as the object was suspended from the conveyor hanger, the brightness was increased to minimize the loss of detail due to this challenge. Saturation, grayscale and sharpness were increased until defined metal folds and edges from the objects of interest were noticeable. The final values from the experimental results are shown in Table 3.2

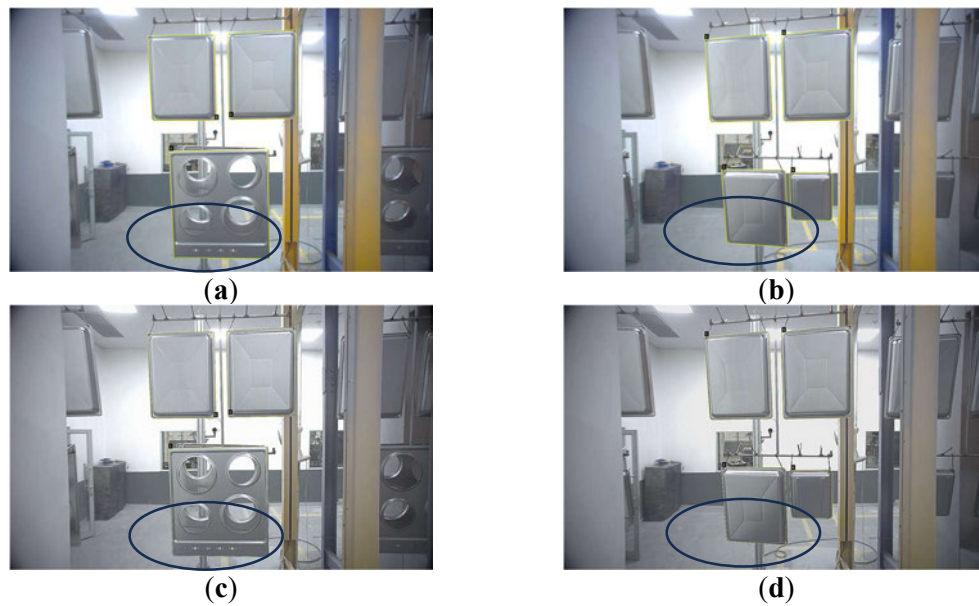


Figure 3.8. (a,b) Raw images with shadows circled; (c,d) augmented images with improved object details.

Table 3.2. Image Preprocessing.

Augmentation Applied	Units
Grayscale	0.75
Saturation	10
Brightness	5
Sharpen	0.5

3.5 Function Selection

3.5.1 Learning Rate

The learning rate is a critical hyperparameter that determines the step size change at each iteration while travelling towards a loss function minimum. This essentially determines the rate of change the model would make to the estimated error whenever the model's weight is updated. The selection of the learning rate has a direct impact on the learning performance of the model with the input dataset. The learning rate is a scaling mechanism for the magnitude of weight updates during the training process. The path towards an optimal loss function is calculated through the gradient of the loss function. This means that the learning rate dictates how big the step jump is in the following path. The learning rate is defined in the range between 0 and 1. The learning rate determines the convergence speed, the training stability and the model performance [50]. If the learning rate is too high during the training process, the optimal point may be overshoot. This may lead to oscillation or divergent outcomes from the model. If the learning rate is too low during the training process, the model may be too slow to learn. This may require more epochs to be able to allow the model to converge if it had not gotten stuck at a local minimum. The training stability of the model is determined by the learning rate as well, therefore a well-chosen learning rate may lead to the model being able to converge more efficiently [149]. This may reduce the requirement for more epochs, reducing the overall training time [50,149]. This affects the final accuracy of the model due to the ability to converge optimally. Through understanding the effect of the learning rate on the model, the focus to explore deeper how that may affect the results in the selected environment was conducted. Using the back-propagation method, the weight of the model is updated using the gradient descent optimization algorithm. The equation used to update the weights is given in Equation 3.1. Where W represents the weight parameter, α represents the learning rate and $\frac{\partial L}{\partial b}$ is the partial derivative of the loss function concerning the weight.

$$W = W - \alpha \frac{\partial L}{\partial b} \quad (3.1)$$

The Mask R-CNN model uses a fixed learning rate algorithm that is combined with a decay. This approach takes into consideration a stable fixed learning rate while reducing the

learning rate over some time to allow for a more finely tuned model. The fixed value that was manipulated did show to influence the model. These values were set at 0.2, 0.02, 0.002, 0.0002 and 0.00002. The results will be discussed in further detail in the following section. The learning rate was reduced at specific epochs during the training process. The value of 10 was used as the reducing factor. The learning rate decay is given in Equation 3.2, where α represents the learning rate and d and e represent the decay rate and epoch number respectively. This also allows a larger setpoint initially to be able to be decreased over time reducing the changes of the weight while training is proceeding.

$$\alpha = \frac{1}{1+(d+e)} * \alpha \quad (3.2)$$

The need for this approach assists in achieving a balance between larger and rapid changes initially in the training process while allowing for fine-tuning as the training process continues. In selecting the different learning rates, the losses were affected which would be shown in the results. As mentioned in [150], a learning rate that is too big may result in overshooting the minimum thus having a negative effect on the accuracy. Due to this problem of overshooting, the model may take longer to train and may result in over learning of the required features. This may cause oscillations as seen in the results relating to poor generalization. The selection of learning rates provided evidence to confirm the findings in the literature review. Similarly, this method of selecting various learning rates illustrated the fine-tuning capabilities of the model. In summary, the learning rate may be adjusted pragmatically until desired outcomes are achieved. This method has been commonly used in the literature.

3.5.2 Learning Momentum

The learning momentum of the model assists with accelerating the gradient vectors leading to faster convergence of the model. The improvement to the convergence rate is done by smoothing out the updates based on the past gradient values. This is useful as losses within the model may be erratic [50]. The equation for this is given in Equation 3.3. Where $v(t)$ represents the weights after being updated at iteration t . β represents the momentum constant, which was set to 0.9 for this study. $\delta(t)$ represents the gradient at iteration t .

$$V(t) = \beta * v(t - 1) + (1 - \beta) * \delta(t) \quad (3.3)$$

3.5.3 Activation Function

The ReLu activation function uses a mathematical operation that is added to the input value. If this input value is greater or equal to zero, the output value would be equal to the input value. If the value is less than zero, the output value would be equal to zero. The need for a ReLu function in the Mask R-CNN model allows for non-linearity, giving the model the capability to be able to learn complex tasks. This function assists with overcoming the vanishing gradient problem, which posed a big challenge in previous state-of-the-art models mentioned in the literature. It is more efficient as compared to other activation functions [50]. The function of the ReLu function is given in Equation 3.4.

$$F(x) = \max(0, x) \quad (3.4)$$

3.5.4 Weight Decay Regularization

A trained model may show signs of overfitting or underfitting. An underfit model is unable to understand the relationship between the features in the test dataset as compared to the training dataset. An overfitting model may generalize well with the training dataset, however, struggles to generalize well with a new test dataset. To achieve a model that both performs well on the training and test datasets, regularization techniques may be applied. A weight decay regularization was applied in the Mask R-CNN model. To reduce the overfitting of the model, weight parameters are kept small through the regularization technique to allow the model to generalize better with the new dataset. The weight decay regularization causes the weights to decay and is described by Equation 3.5. The $Loss_0$ refers to the original loss function, λ represents the regularization parameter, w_i^2 represents the coefficients of the model. Using the regularization parameter, the coefficients are reduced. This reduction ensures that the model doesn't overfit while becoming reliant on certain

features in the dataset [151]. The weight decay regularization parameter was set to 0.9 in this study.

$$Loss_{L2} = Loss_0 + \lambda \sum |w_i^2| \quad (3.5)$$

3.5.5 Images per GPUs

The images per GPU refers to the total amount of images that would be processed. Higher values did give an error relating to low memory availability from the GPU and was reduced to 2. A GPU with more memory would be more beneficial as more images may be processed in one cycle. This does add to the cost of the total project, so the GPU was not changed and an acceptable performance to cost ratio was achieved.

3.5.6 Epochs

An epoch is one complete pass through the entire dataset, the choice of epoch value selection has an impact on the model performance. This value would also affect the model's ability to generalize well with new data. Selecting the correct number is crucial in determining a balance between underfitting and overfitting the model. A larger dataset generally does require a larger epoch number as this requires more iterations of the model to learn more features from the dataset [50]. This is also true for a more complex dataset. Conversely, a dataset that is simpler concerning its features may require fewer epochs to learn. As mentioned before the learning rate may influence the epochs number too. A learning rate too low may require more epochs to converge. The opposite is true, where a larger learning rate may require fewer epochs. The practical approach used in this study considered the loss curve relating to the train and validation dataset. The curve was visualized to determine whether the model was overfitting or underfitting. This practical approach assisted in determining the epoch number. As the dataset was not complex, due to the controlled environment. The number of epochs was set to 10 which showed a converging curve on both the training and validation dataset without overfitting and underfitting. The gaps between

each curve were not diverging from one another, which indicated that the model was generalizing well with the given dataset. This curve is illustrated in Figure 3.9.

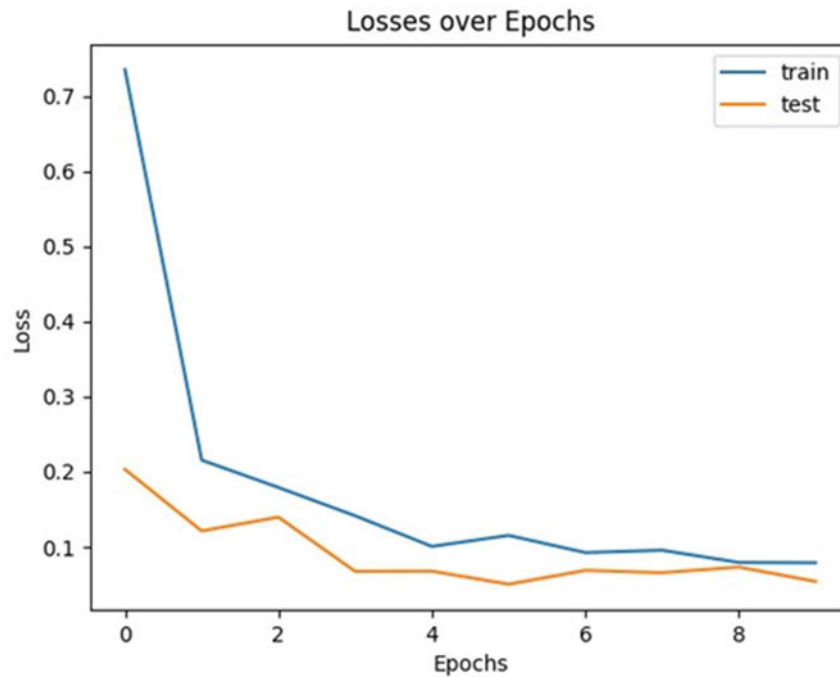


Figure 3.9. Losses over epochs.

3.5.7 Steps per Epochs

The steps per epoch are referred to as the total number of batches of samples before the finishing of one epoch. During the training process, the value would be calculated by the total number of training samples in the dataset divided by the batch size [152]. This is illustrated in Equation 3.6. The batch size is determined by the number of images selected per GPU multiplied by the number of GPUs available for use. The size was set to 2 for this study, with a dataset that is over 200, and the selected steps per epoch were set to 100. This selection has a direct effect on the training duration of the model and memory allocation from the processing unit. A balance between hardware allocation and time resources needs to be satisfied.

$$\text{Steps per Epochs} = \frac{\text{Training Examples}}{\text{Batch Size}} \quad (3.6)$$

3.5.8 Validation Steps

The validation steps run at the end of every epoch training interval [152], while a greater number may improve accuracy, it may slow down the training process. This step is used to evaluate the model's performance based on unseen data. This tool is essential in monitoring the model's performance while being trained relating to the generalization capabilities of the model. As this is more of an assessment and not used to update the weights of the model, this forms a separate function within the model. The validation step is dependent on the dataset amount that has been selected. Similarly to the steps per epoch, the validation steps are calculated based on the validation batch size used in the model. The number was set to 50 in this paper. Equation 3.7 illustrates what has been mentioned.

$$\text{Validation Steps} = \frac{\text{Validation Examples}}{\text{Validation Batch Size}} \quad (3.7)$$

3.5.9 Backbone

The backbone used in this paper was set to ResNet101 [20]. The choice for this selection was dependent on the information and results that were provided in the literature review. Table 2.1 has provided a clear idea of which is the superior backbone to select based on previous state-of-the-art models. The literature review adequately provided details on the use of a deeper network and its advantages and disadvantages. As the network does have more layers, the model would have the capability to extract more detailed and complex features from the dataset. Making the model more robust as compared to the other network. This would lead to a better output regarding object detection and instance segmentation. ResNet50 has been shown to perform well based on the literature review. However, this network is shallower, having 50 layers as compared to 101 layers. This may not have the capability to extract higher detailed features. However, it would be quicker during the training process requiring less computational resources. The ResNet101 network was selected as the purpose of this study was to have a model that was more robust with the selected application and the environment. According to [20], the ResNet101 network did provide a higher accuracy score as compared to the same Mask R-CNN model being used

with the ResNet50 network. A higher AP was preferred with the sacrifice being placed on the training time.

3.5.10 Backbone Strides

The backbone stride down samples the input image as the image passes through the convolution blocks in the ResNet backbone. The use of different stride values is related to the different features that are extracted at different levels of the network. With the maximum stride value being set to 64 for this study. The values per convolution block were set to [4, 8, 16, 32, 64]. The first convolutional layer would have a stride of 4, which would reduce the spatial resolution by 4. This would be followed by a max pooling function which would further reduce the resolution of the image [90]. The cumulative strides of the model would down sample the image with a spatial resolution calculated by the image resolution being divided by the final stride value after passing the convolutional layers. The final feature map would have a lower value as compared to the start of the model. Equation 3.8 shows this concept. As mentioned, the images used in this paper has a pixel size of 1920 x 1200. After using Equation 3.8, the final feature map would be 30 x 18. The smaller resolution feature map allows the model to be able to capture high-level features without excessive computational resources required to perform the task.

$$\text{Feature map Resolution} = \frac{\text{Input image Resolution}}{\text{Stride Value}} \quad (3.8)$$

3.5.11 ROI Positive Ratio

The ROI positive ratio relates to the percentage of positive ROIs used to train the mask heads. Positive ROIs are those that overlap with the predefined ground truths within the dataset. This overlap percentage that determines whether an ROI is positive or not is determined through an IoU value that has been predefined as well. Once the ROIs are determined in the RPN determining if an object may potentially contain the object of interest, the IoU would determine the positive or negative ROIs. The thresholds for this study were varied between 0.5, 0.75 and 0.9. The positive ROIs would have a value greater than these selected thresholds. Conversely, if the value of the ROIs is less than the predefined threshold

values, this would make this a negative ROI. The ROI positive ratio is calculated through the number of total positive ROIs divided by the total amount of ROIs that were generated. The importance of having the ROI positive ratio in the model assists with the model not being biased towards background objects. This is relevant in this study as the background does pose additional noise in the model's training process. The chosen value in this paper was set to 33% which would ensure that the model can distinguish between objects that are present in the background versus the objects that are of interest. This selection would have assisted in training stability by ensuring that the model received enough positive samples to learn the required features efficiently.

3.5.12 Detection Minimum Confidence

In the Mask R-CNN model, the detection minimum confidence value is related to the IoU threshold value. As mentioned in the literature review, the threshold values play a significant role in the ROIs that would be used in the model. The threshold values determine whether a detected object may be considered to be valid or not. The confidence score is a probability output of the model concerning whether the object detected is an object relating to a specific class in the dataset. If the value of the detection is greater than the threshold, it is kept. If the value is less than the threshold, it is discarded. The use of this threshold is useful in filtering out all detections that have a low probability score that may be related to false positives. This selection in threshold values does have an impact on the model's AP across all class values and the overall accuracy. The selected values in this paper were set to 0.5, 0.75 and 0.9. With these values, a difference in AP values was noted and will be described in the results section. The setting of different values for the threshold does impact the output concerning the number of false positives. Having a higher threshold value may eliminate false positives, however, it may also eliminate true positives in the process. Conversely, if the threshold is too low, the chances of including more false positives increase as with a reduced precision score. The choice for selecting the final value for the model followed an iterative process which included the review of the results of the once the model went through the inference process. The IoU is determined through Equation 3.9.

$$IoU = \frac{\text{area of overlap}}{\text{area of union}} \quad (3.9)$$

This experimental process included the review of the results of the precision-recall curve and the analysis of the confusion matrix.

3.5.13 RPN NMS Thresholds

The RPN Non-Maximum Suppression (NMS) refers to the threshold that is used in the RPN stage of the model. The function of the NMS filters out redundant or overlapping bounding boxes that are generated by the RPN. This ensures that the most relevant bounding boxes are further processed by the model [153]. The function selects the highest confidence score of the overlapping bounding boxes. The bounding boxes with a confidence score that is lower than the maximum bounding box is suppressed. The NMS threshold is predefined, the value selected in this study was set to 70%. Based on the NMS threshold, the train ROIs per image may then be adjusted. This refers to the ROIs used during the training process for each image. As the model would not use all ROIs that were generated, it would be fixed to the ROIs allowed per image. Through the limiting of ROIs used during the training process, the memory and computational resources are regulated, allowing the model to be more efficient.

3.5.14 Loss Function

The Mask R-CNN model generates classes, bounding boxes and masks as the output. This model utilizes multitask losses relating to classification, localization and mask generation, which is illustrated in Equation 3.10. Where L refers to the loss.

$$L_{Total} = L_{class} + L_{bbox} + L_{mask} \quad (3.10)$$

The L_{class} measures the error between the ground truth class and predicted classes. Equation 3.11 illustrates the class loss as a log loss function between the ground truth class and the predicted class. This loss function is used to determine whether the predicted class is related to the ground truth class. Where p_i represents the predicted class and g_i represents the ground truth class.

$$L_{class}(p_i, g_i) = -g_i \log p_i - (1 - g_i) \log(1 - p_i) \quad (3.11)$$

The L_{bbox} measures the difference between the ground truth bounding box and the predicted bounding box through a smoothing function. The smoothing function reduces the sensitivity of the loss function to outliers. Equation 3.12 and Equation 3.13 represents the bounding box losses and smoothing function respectively. Where t_i represents the predicted bounding box and v_i represents the ground truth bounding box.

$$L_{bbox} = \sum_{i \in x, y, w, h} L_1^{smooth}(t_i - v_i) \quad (3.12)$$

$$L_1^{smooth} = \begin{cases} 0.5x^2, & \text{if } |x| < 1 \\ |x| - 0.5, & \text{else} \end{cases} \quad (3.13)$$

The L_{mask} measures the error between the ground truth mask and the predicted mask. Since there is no competition in mask generation relating to classes as the model learns each mask that relates to a specific class. Equation 3.14 provides the average binary cross-entropy loss including the mask if the region of the mask is associated with the class ground truth. Where m^2 is the size of the mask, i, j is the cell, y_{ij} is the label of the cell and p_{ij}^k is the predicted value of the cell.

$$L_{mask} = -\frac{1}{m^2} \sum_{1 \leq i, j \leq m} [y_{ij} \log p_{ij}^k + (1 - y_{ij}) \log(1 - p_{ij}^k)] \quad (3.14)$$

3.6 Training Methodology

3.6.1 Experimental Training Setup

After fine-tuning the final process flow is shown in Figure 3.10. If a change was observed in False Positives (FP) or AP or Overall Accuracy (OA), the dataset and hyperparameters would be investigated and adjusted. The process would be iterative until results were considered satisfactory.

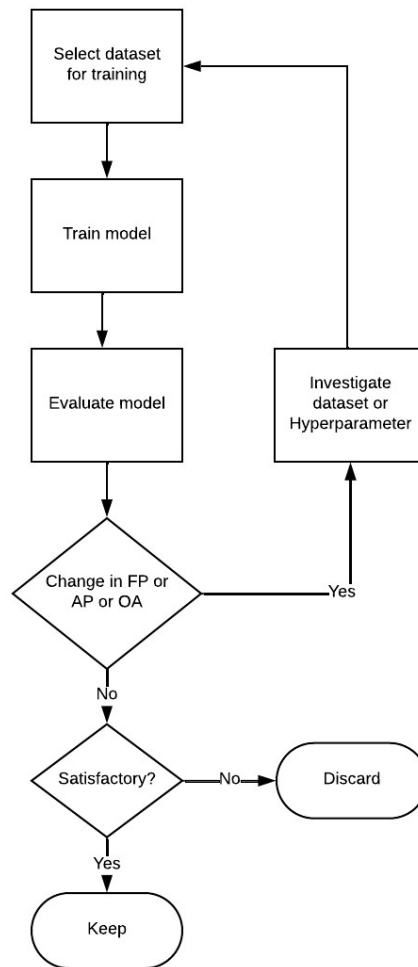


Figure 3.10. Training process flow.

3.6.2 Types of Datasets

There were two types of datasets used in this study. The main dataset that produced stable results was illustrated in Table 3.1 and produced the highest AP and OA scores. The second dataset that was used in an experimental attempt to add more images from different angles other than what was seen on the production line provided a unique set of results that were lower as compared to the first set of data. The second set of data was gathered using a mobile smartphone's camera. The dataset was captured off the main production line. This addition of data of 130 ROIs on top of the 188 ROIs gathered for Class A in Table 3.1 was only

gathered for the class that was the highest produced part in one production shift. Figure 3.11 illustrates samples from the second dataset used in this paper.



Figure 3.11. Sample of an additional dataset.

Figure 3.12 illustrates the segmentation of objects within an image into its respective classes. Each class belongs to its group when isolating objects and their corresponding masks. This operation is applied to every annotated image in the dataset and is fed into the model for processing during the training process. The isolation of each mask relating to each class is crucial for the model to be able to learn the features of each object class. The isolation of each class is used to calculate the mask losses of each class independently as illustrated in Equation 3.14.

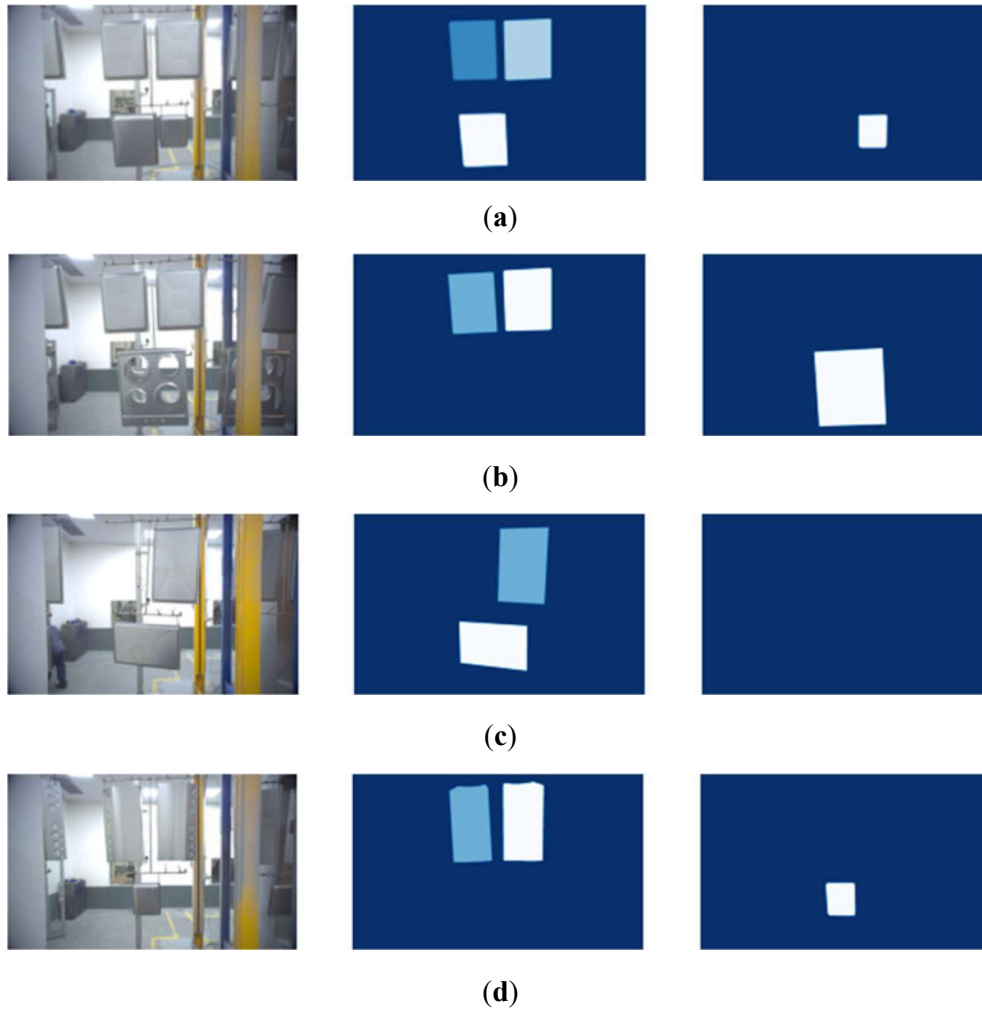


Figure 3.12. a) The second panel contains class B and the third panel contains class A; (b) the second panel contains class B and the third panel contains class C; (c) the second panel contains class E and the third panel contains no class; (d) the second panel contain class D and the third panel contains class A.

3.6.3 Practical Methodology

According to [50], successfully applying deep learning techniques does require more than understanding the algorithms or principles associated with how the model works. The additional requirements related to how to analyse and respond to the feedback and results from the model is another important function. The choice of whether to gather more data, add or remove the regularization feature, increase or decrease the model's capacity, debug the software or tune hyperparameters is dependent on the study and its requirements. Since

every method mentioned is time-consuming, testing each method may prolong the desired output of the model. A planned-out approach must be made on which method needs to be selected. In this study, dataset and hyperparameter manipulation were used as the approach to output refinement. The incremental changes made to hyperparameters and the dataset were recorded. The need to increase the model capacity was not a requirement as the model performed sufficiently being able to provide information about production requirements that met the speed of the line.

The code was refined to provide the required visual results. However, modifications to the structure of the model were not necessary as the dataset used in the model was minimal. The need to make changes to the instruments that were used on the line was not required as the cameras were able to keep up with the FPS requirements of the line. The resolution of the images was also considered sufficient as smaller objects were still being detected after training. The performance of the model was deemed satisfactory if the overall accuracy was above 97.5%. This would have been higher than the output related to human observation as mentioned previously.

The goal of this study relating to performance metrics was to increase the AP of the model while reducing the FP which would lead to higher OA of the model. This was achieved through the utilization of the manipulation of the hyperparameters of the model. With enough information gathered from the literature, ranges on the selection of the epochs, steps per epochs and learning rates were attained in training an improved Mask R-CNN model that was different to the original values used in previous studies [20]. Due to the hardware configuration, a simpler ReLu activation function was used to learn complex tasks. Since the initial dataset did provide close to satisfactory results, there was no initial need to increase the dataset. However, it was noted that there were instances during the testing phase of detecting FPs. The idea of adding more unique data to the dataset was prompted. However, this did not eliminate the FPs that were detected in the background of the images. The model did fail to generalize well with the new dataset that was illustrated in Figure 3.11.

Hyperparameter selection was made on the observation of training and validation losses and the AP of the model after changes were made. Adjustments made to epochs, steps per epochs

and the learning rate did influence the computational resources available for this study. If the changes were observed to have a positive influence on the losses and the AP of the model, the results were kept and the direction of the hyperparameter path was adjusted until undesired values were released. The values of certain hyperparameters were then fixed once a balance between performance and computational resources was achieved. These values are illustrated in Table 3.3.

Table 3.3. ResNet101 hyperparameter values.

Hyperparameters	Experiment Value
Epochs	10
Steps per Epoch	100
Weight Decay	0.0001
Learning Momentum	0.9

3.7 Evaluation Metrics

The performance evaluation of the Mask R-CNN model took place in two instances. After the model had been trained using the training and validation dataset, the model was run with the test dataset which was the unseen data that included the annotated ground truth. The average precision and average recall were calculated from the results that were retrieved after the test results. The AP was calculated using the area underneath the precision-recall curve which provided insight on how the model was performing with quantitative results. This however provided the information about the AP across different IoU thresholds. The model was then evaluated using the confusion matrix, this result provided more information on how well the trained model was able to classify unseen data. This information represented the prediction summary in a matrix format. The results from the confusion matrix illustrate the number of predictions being placed in the correct class or the incorrect class. An example of the confusion matrix with the column and row titles is illustrated in Figure 3.13.

		Actual class	
		P	N
Predicted class	P	TP	FP
	N	FN	TN

Figure 3.13. Confusion matrix.

The AP across all classes is commonly used as a metric for object detection applications as mentioned in the literature review. The measure provides details relating to the precision and recall of the model across multiple classes and the predefined IOU threshold, providing insight into how well the model can perform with the various selected object overlaps. As described in the literature review this metric was used in the challenges that included the readily available datasets. This was important in this research study and was also used as part of the evaluation step. The average precision and average recall values are important in determining how well the model may be able to preserve a high precision while recalling the required objects. This value is important when FPs are of concern during the testing phase. These metrics were selected as the evaluation criteria as they provide different insights related to different aspects of the model. A holistic approach to the evaluation of the model is needed to determine the effectiveness of the model in general.

3.7.1 Losses

The classification losses may be defined through a cross-entropy loss function for multi-task losses. The regression losses are defined in Equation 3.15. Where R represents the robust loss function, t and t^* represents the four vertices of the vector of the predicted and ground truth bounding box.

$$L = R(t - t^*) \quad (3.15)$$

$(t_x, t_y, t_w, t_h, t_x^*, t_y^*, t_w^*, t_h^*)$ are computed as defined using Equation 3.16 to Equation 3.23. Where x, y, w and h represents the x-axis coordinates of the box centre, y-axis coordinates of the box centre, width of the box and height of the box respectively. This forms part of the total losses of the model and can be visualized as seen in Figure 3.9.

$$t_x = \frac{(x - x_a)}{w_a} \quad (3.16)$$

$$t_y = \frac{(y - y_a)}{h_a} \quad (3.17)$$

$$t_w = \log\left(\frac{w}{w_a}\right) \quad (3.18)$$

$$t_h = \log\left(\frac{h}{h_a}\right) \quad (3.19)$$

$$t_x^* = \frac{(x^* - x_a)}{w_a} \quad (3.20)$$

$$t_y^* = \frac{(y^* - y_a)}{h_a} \quad (3.21)$$

$$t_w^* = \log\left(\frac{w^*}{w_a}\right) \quad (3.22)$$

$$t_h^* = \log\left(\frac{h^*}{h_a}\right) \quad (3.23)$$

3.7.2 Average Recall

The average recall is defined as the model's ability to detect objects at varying IoU thresholds. Equation 2.4 illustrates this definition for a single class. In this study, the micro

averaging technique was used as compared to the macro-averaging technique. The macro-averaging technique is given in Equation 3.24 representing the average recall over all classes with N representing the total number of classes. The micro-averaging technique takes the total number of TP across all classes and divides it by the total number of TP and FN over all classes. This is given in Equation 3.25.

$$Recall_{macro\ av.} = \frac{Recall_a + Recall_b + \dots Recall_N}{N} \quad (3.24)$$

$$Recall_{micro\ av.} = \frac{TP_a + TP_b + \dots TP_N}{TP_a + FN_a + TP_b + FN_b + \dots TP_N + FN_N} \quad (3.25)$$

3.7.3 Average Precision

The average precision for a single class is calculated in Equation 2.3. The average precision is the model's ability to identify objects that are in the correct class over all classes. In this study, the micro averaging technique was used as compared to the macro-averaging technique. The macro-averaging technique is illustrated in Equation 3.26 representing the average precision over all classes with N representing the total number of classes. The micro-averaging technique takes the total number of TP across all classes and divides it by the total number of TP and FP over all classes. This is illustrated in Equation 3.27.

$$Precision_{macro\ av.} = \frac{Precision_a + Precision_b + \dots Precision_N}{N} \quad (3.26)$$

$$Precision_{micro\ av.} = \frac{TP_a + TP_b + \dots TP_N}{TP_a + FP_a + TP_b + FP_b + \dots TP_N + FP_N} \quad (3.27)$$

3.7.4 Precision Recall Curve

The precision-recall curve is a useful metric to be used when the class from the dataset is imbalanced as conducted in this research [154]. The precision-recall curve illustrates the trade-off between the precision and recall of the model at different thresholds. The area under the curve closer to the value of one shows us that the model is performing well with both metrics. In other words, the model can return accurate results (high precision) while providing many positive results (high recall). An example of a model with high recall and low precision values may result in the majority of it being incorrectly classified. During the testing phase of the model, a script takes the calculated micro-averaged precision and recall values over all classes and tabulates it to generate a curve which would be used for analysis. Table 3.4 illustrates how the calculation is entered. Where the 'Image' column represents all the images in your test dataset, 'Detection' is the total detection made in the test dataset, 'TP' and 'ACC'. TP is the TP detected in that instance and the accumulated TP across the entire dataset. FP and ACC. FP is the FP detected in that instance and the accumulated FP across the whole dataset. The recall and precision values are calculated using Equation 3.25 and Equation 3.27, with N representing an image and their respective values.

Table 3.4. Precision- Recall table.

Image	Detection	Confidence	TP	FP	ACC. TP	ACC. FP	Precision	Recall
1	A	98%	1	0	1	0	1	0.0666
2	B	95%	0	1	1	1	0.5	0.0666
N	...	...	...	...	...	...	Eqn. 3.27	Eqn. 3.25

The detection with the highest value would be at the top of the table followed by the confidence score that is lower in value. This list of values would continue until all values have been tabulated from the test dataset.

3.7.5 Mean Average Precision

The mean average precision is calculated from the area under the precision-recall curve. Once all precision and recall values have been tabulated as shown in Table 3.4, the precision-

recall curve may be drawn out. In Figure 3.14, each transparent blue circle represents a point found on a table that follows the same column setup as Table 3.4. Equation 2.5 would be applied to generate a value relating to the AP across all classes also referred to as the mAP, an area value closer to one is preferred and the better model as compared to a model with a value lower.

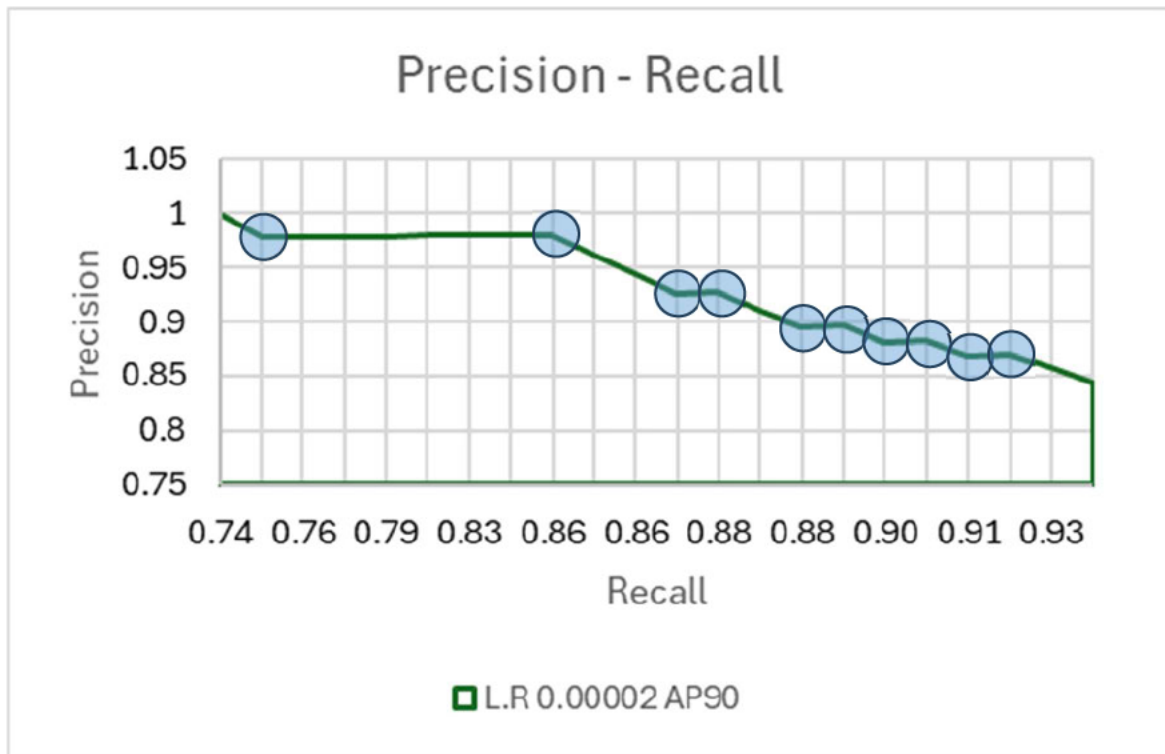


Figure 3.14. Precision–Recall curve.

3.7.6 F1 Score

The F1 score represents the harmonic mean of precision and recall. The metric balances both precision and recall. As compared to the just the precision and recall metric, The F1 score ensures that both the precision and recall metric would need to be high to ensure a high F1 score. The F1 score is useful metric when considering an unbalanced multi-class dataset.

3.7.7 Overall Accuracy

The confusion matrix provides import information such as TP, TN, FP, FN. From this information, Overall Accuracy may be measured which would measure the proportion of

correctly classified objects, sum of TP and TN, relative to the total amount of objects, sum of TP, TN, FP and FN, that were classified. This metric provides an indication on the model ability to classify the correct objects per class.

3.7.8 Confusion Matrix

The confusion matrix was used to determine the overall accuracy of the model, which is calculated by summing the number of correctly classified objects and dividing that by the total number of objects that were detected. This calculation is illustrated in Equation 3.28.

$$\text{Overall Accuracy} = \frac{\text{Correctly detected objects}}{\text{Total objects detected}} \quad (3.28)$$

Following Figure 3.15, the correctly detected objects are found highlighted going diagonally from the top left corner to the bottom right corner belonging to each class.

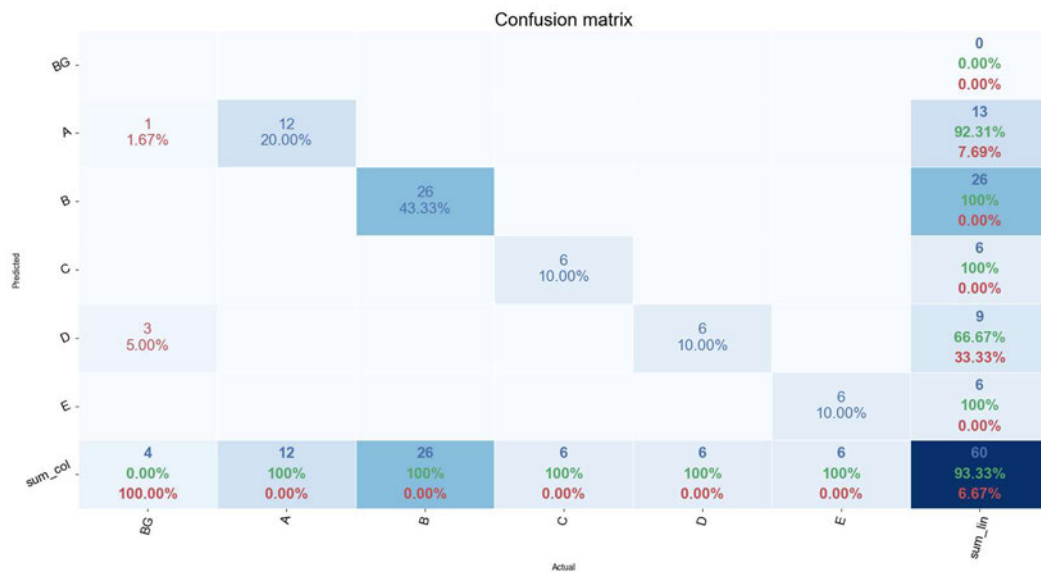


Figure 3.15. Confusion matrix layout.

The kappa coefficient may have been used as another method to determine the overall accuracy other than the use of the confusion matrix. For this study, the confusion matrix was used only. Equation 3.29 illustrates the use of the Kappa's coefficient. Where P_o represents the correctly observed agreement and P_e represents the expected agreement.

$$K = \frac{P_o - P_e}{1 - P_e} \quad (3.29)$$

Using the confusion matrix in Figure 3.15, there are additional metrics that may be attained. The error of commission represents the objects that were predicted to be in the wrong class when referring to one class. This is the measure of the false positive values attained from the model and it is represented as the total number of incorrectly detected objects divided by the total number of predicted objects per class. This definition is represented in Equation 3.30.

$$\text{Error of Commission} = \frac{\text{Incorrectly detected objects}}{\text{Total predicted objects}} \quad (3.30)$$

3.8 Experimental Environment

The lighting conditions were maintained at a level where the edges of the objects of interest were identified. The cameras were installed in a controlled environment where the illuminations were maintained at 8000 lumens per 20 square meters with a colour temperature of 4000K. Lighting conditions needed to be maintained so as not to overexpose an image and introduce blur. If the lighting conditions were inadequate, this may not allow for adequate detail to be captured by the camera. The objects were at a height where human interference from encountering the objects was at a minimum. Figure 3.16 illustrates objects that were detected after the model went through the training and testing phase. All objects that have been detected have been shaded in a different colour in the images from Figure 3.16 (a) to Figure 3.16 (d).

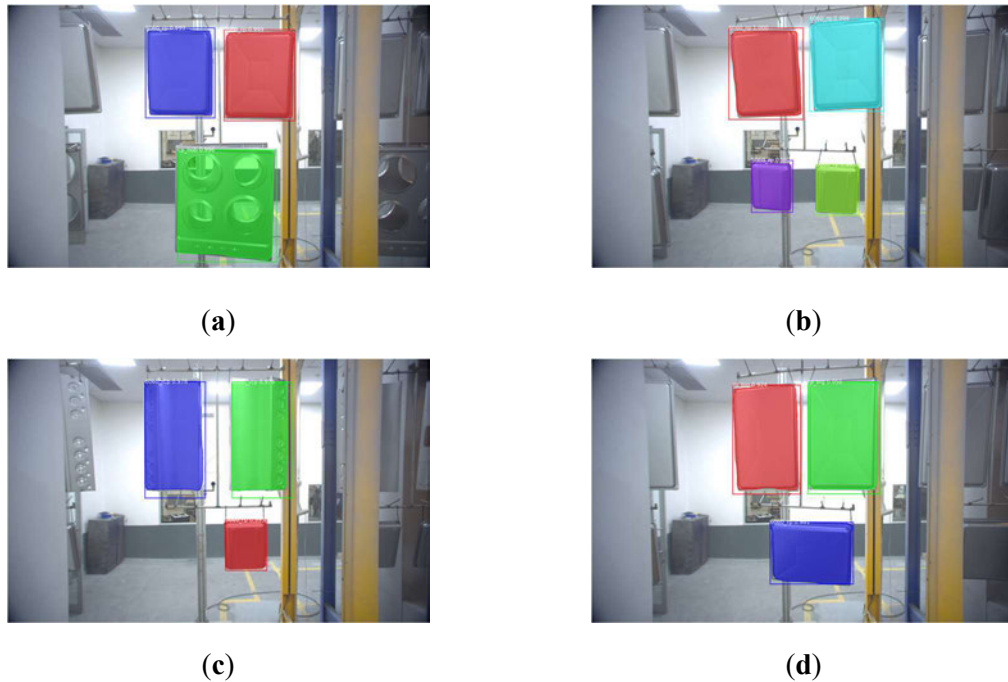


Figure 3.16. (a) Positive detection of class B and C; (b) positive detection of class A and B; (c) positive detection of class A and D; (d) positive detection of class B and E

3.9 Limitations

Several limitations and challenges were encountered during the study. These challenges did have an impact on the overall effectiveness and feasibility of the deployment. The majority of the limitations were related to the industrial environment. In this study, the environment lighting was well controlled considering that the majority of studies based on object detection did not focus on having a dataset with good lighting conditions. This however did not mean that this environmental challenge was eliminated in total. The environment did still provide unpredictable conditions relating to dust, occlusion and blur. As the room where the cameras were installed was near the enamel powder coating booth, this dust travelled from the booth to the area under consideration. This dust laid on the lens of the camera, reducing the quality of the image feed into the model. This challenge needed to be addressed through general cleaning at the start of every shift.

Due to power-saving initiatives, the lighting around the area close to the study was switched off close to the end of the shift while the line was still moving. This challenge did pose a risk of losing data from the production line due to the objects not being able to be detected in the dark. This challenge was addressed by adjusting the power-saving schedule to be adjusted after the shift ended. The extreme changes in the lighting conditions had a direct impact on the accuracy of the model. There were instances where objects on the hanger were not placed correctly by the operator. These objects were near one another, causing an issue of occlusion with one another. Occlusion did affect the model's accuracy due to the unknown shape not being able to be detected or segmented.

There were instances that due to the system being unfamiliar to the operators in the firm, human presences were detected in the images that cancelled all detections in the model. These challenges were addressed through training the operators on the floor on object placement on the line and to avoid any interruption of the study due to humans. The paint and enamel area was close to the heavy press area. This area did have a lot of vibration due to the moulding of parts from a heavy press. This constant motion that was detected in some images did add time to data sorting as these images were discarded. This issue would have had an impact on the robustness of the model. This challenge was addressed by adding more support to the stand that held the cameras, lowering the vibration from the nearby area. Due to the dust and vibration as mentioned above. This prompted the choice to select robust hardware that was able to handle these harsh conditions. This included well-known industrial-grade hardware that was available.

The deep learning Mask R-CNN model was computationally heavy, and a balance needed to be met between the hardware and performance criteria. As in the literature review, the processing speed is heavily dependent on the selection of the hardware. A GPU was one of the requirements within this study to be able to complete this study promptly. The Nvidia GPU proved to be sufficient for this application. Usually, industrial environments operate on existing systems that may pose a risk in being compatible with the implementation of new systems. The data received from this system would be integrated into the existing ecosystem being able to communicate over Open Platform Communications Unified

Architecture (OPCUA) to the relevant dashboards. The data would be saved to an SQL database which was not a concern in this study. The effectiveness of an object detection application is highly dependent on the quality of the dataset as mentioned in the literature review. This task was highly time-consuming in the study requiring manually annotating each object within an image. Each object was then assigned to a specific class after being annotated. This labour-intensive process did pose a huge risk if not done correctly. Each object that was annotated was checked multiple times to confirm if the annotations included all edges of the objects and excluded any background objects due to occlusion. The object's classes were also checked multiple times to confirm whether each object in the image was correctly assigned the ground truth classification. The incorrect assignment of classes would have resulted in noise being added to the training process, effectively producing incorrect results on the model's capabilities.

In the industrial environment, some objects were seen more often than others while being captured through the camera. This was due to the production requirements of the business and the firm. This frequency of a certain object may pose a risk relating to imbalanced data. Scaling may be seen as a challenge to come after this study with the addition of more classes as the business requirements change. However, with the results achieved in this study, training on a smaller dataset has been shown to produce good results if all environmental conditions are met.

The last challenge and one of the aims of this study was to implement an object detection application that could produce results that superseded the human requirements in producing data manually. This task posed a challenge as the industry standards require the outputs of any system to be stable, viable and feasible when used as the truth when being referenced. The study provided great insight into this challenge as it is an ongoing challenge and investigation within the computer vision community.

3.10 Summary

The methodology of this study provided a comprehensive understanding of the research design approach being pragmatic. The framework provided the path to achieving the

required results promptly. The method of the topic was demonstrated through the hardware selection, software selection and annotation tool. The importance of the data selection and sorting process was highlighted explaining its impact on the model's performance relating to the mAP and OA. Once the data was cleaned, the preprocessing requirements were described, illustrating how the controlled environment still has an impact on the object. Each function used in the model was discussed and the influence it plays on the model training and testing capabilities. The experimental application of the object detector was developed and studied on two datasets after being tuned with the mentioned limitations of applying the model in an industrial environment.

CHAPTER 4 Results

4.1 Introduction

In this chapter, the results are provided from the experiments using the Mask R-CNN model following the methodology mentioned in chapter three. The focus is placed on the observation of the actual output from the model and the evaluation of the performance of the model based on selected parameters and methods. The performance of the model is based on the various IoU thresholds and learning rates that were selected. This selection provided an understanding of the impact it has on the model's performance. The quantitative approach in this study was achieved through the use of key metrics such as precision, recall and average precision over all classes. The results were split into their respective IoUs and learning rates which were further evaluated based on the ResNet101 and ResNet50 architecture. The IoUs were set from a common baseline of 50% to a stricter value of 75% and 90%. The best-modified model was also subject to two datasets to evaluate how the model would perform. The initial dataset split, which is illustrated in Table 3.1, contained images that were similar to what would be detected in the field by the fixed cameras. The second dataset had additional images from different angles related to the objects of interest. The second dataset did have more detail of the object as this captured the features of the objects that would have not been detected by the fixed cameras. The overall accuracy of the model was determined using the confusion matrix after the testing process. Through this chapter, the aim is to discover the strengths and weaknesses of the proposed modified model being used in an industrial environment based on an imbalanced dataset that matched the production line's requirements per shift.

4.2 Dataset Anomalies

The types of datasets used have been explained in section 3.6.2, each having five distinct classes marked as A, B, C, D and E. A few anomalies were observed and are illustrated in Figure 4.1. This included false positive detection, no detection, multiple detections, incorrect mask detections or a combination of the mentioned challenges on some of the test datasets.

This was found to be more prevalent using Mask R-CNN and ResNet50 as compared to Mask R-CNN and ResNet101. Images (a) to (d) illustrate false positive detections where objects are detected belonging to a specific class which is not true as compared to the ground truth. Figure 4.1 (a) illustrates an object being detected which is physically not in the image. Figure 4.1 (b) illustrates the toolbox in the background being detected as belonging to a specific class which is false. Figure 4.1 (c) illustrates an object with an unknown mask that has been falsely identified where no object is physically visible. Figure 4.1 (d) illustrates an object being detected in the middle of two other objects. These detections may be related to similarities in features that may result in the detection of false positives. Figure 4.1 (e) to (h) illustrates the challenge of multiple detection on a single object within an image. Class A and B both share similar features and only differ in size, the similarity in features may confuse the model in determining which class the object may belong to. Figure 4.1 (i) to (l) illustrates the challenge of no detection being made on certain objects. This may relate to more training required and the use of higher IoU thresholds as mentioned in the literature review. Figure 4.1 (m) to (p) illustrates a combination of false positives, no detections, multiple detections on one object and irregular masks. These errors may be present due to feature-level similarities that would exist with the background and the object itself. The errors that were more prevalent in the ResNet50 backbone, compared to the ResNet101 suggests that a deeper neural network provides more robust results and should be considered in an industrial environment. Mitigation strategies include refinement of anchor points used to create bounding boxes, an increase in training iterations and to reduce or eliminate background objects that are not part of training samples.



(a)

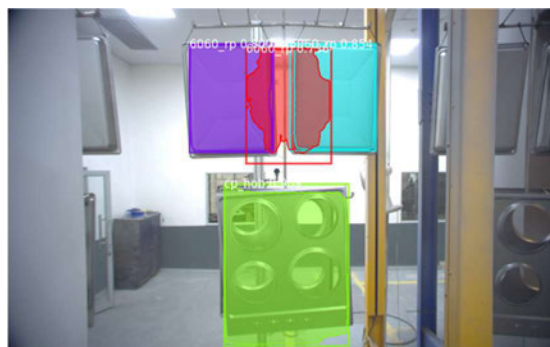


(b)

Continued...



(c)



(d)



(e)



(f)



(g)



(h)

Continued...



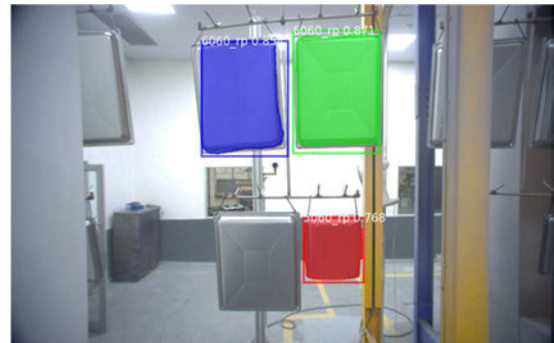
(i)



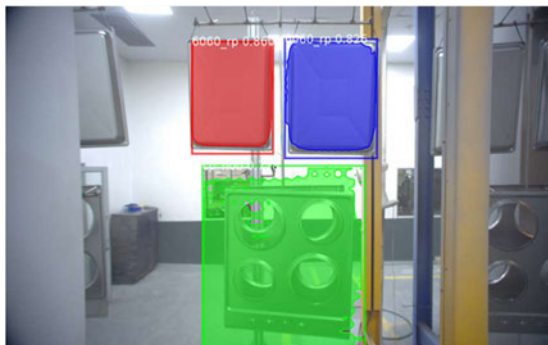
(j)



(k)



(1)



(m)



(n)



(o)



(p)

Figure 4.1. Challenges faced during the testing process.

4.3 Performance Evaluation

The performance of a model may be evaluated quantitatively using the precision–recall curve in determining the AP per IoU over all classes. This metric provides an understanding of how the model is performing concerning the recall and precision capabilities. Equation 3.25 and Equation 3.27 were used to determine each metric. This was followed by the use of Table 3.4 to get all the data required to generate the curve. The model with an area closer to one is considered better in comparison to a lower value. Figures 4.2 and 4.3 illustrate all the curves used in calculating the AP of each model at various learning rates and IoUs.

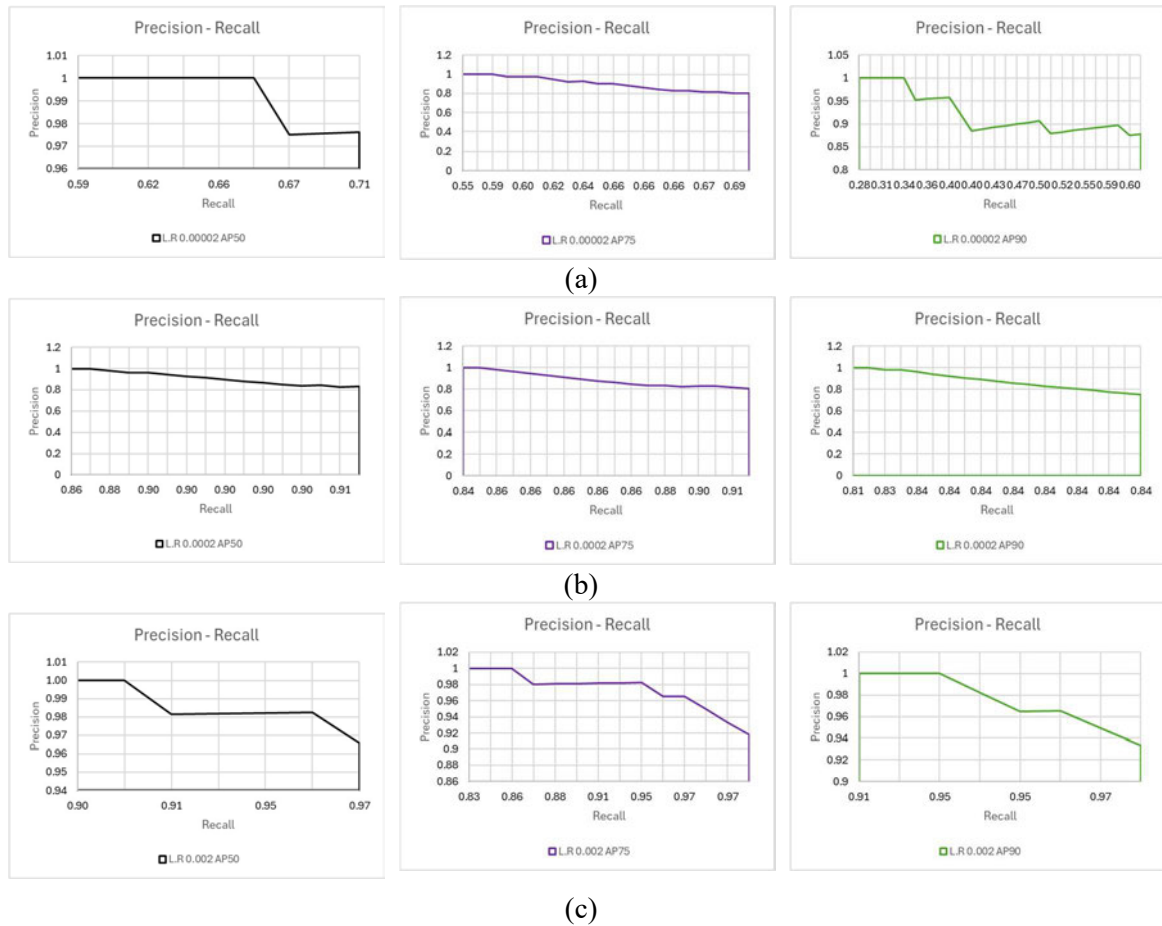


Figure 4.2. Mask R-CNN + ResNet50: (a) learning rate 0.00002 with an IoU of 50%, 75% and 90%; (b) learning rate 0.0002 with an IoU of 50%, 75% and 90%; (c) learning rate 0.002 with an IoU of 50%, 75% and 90%.

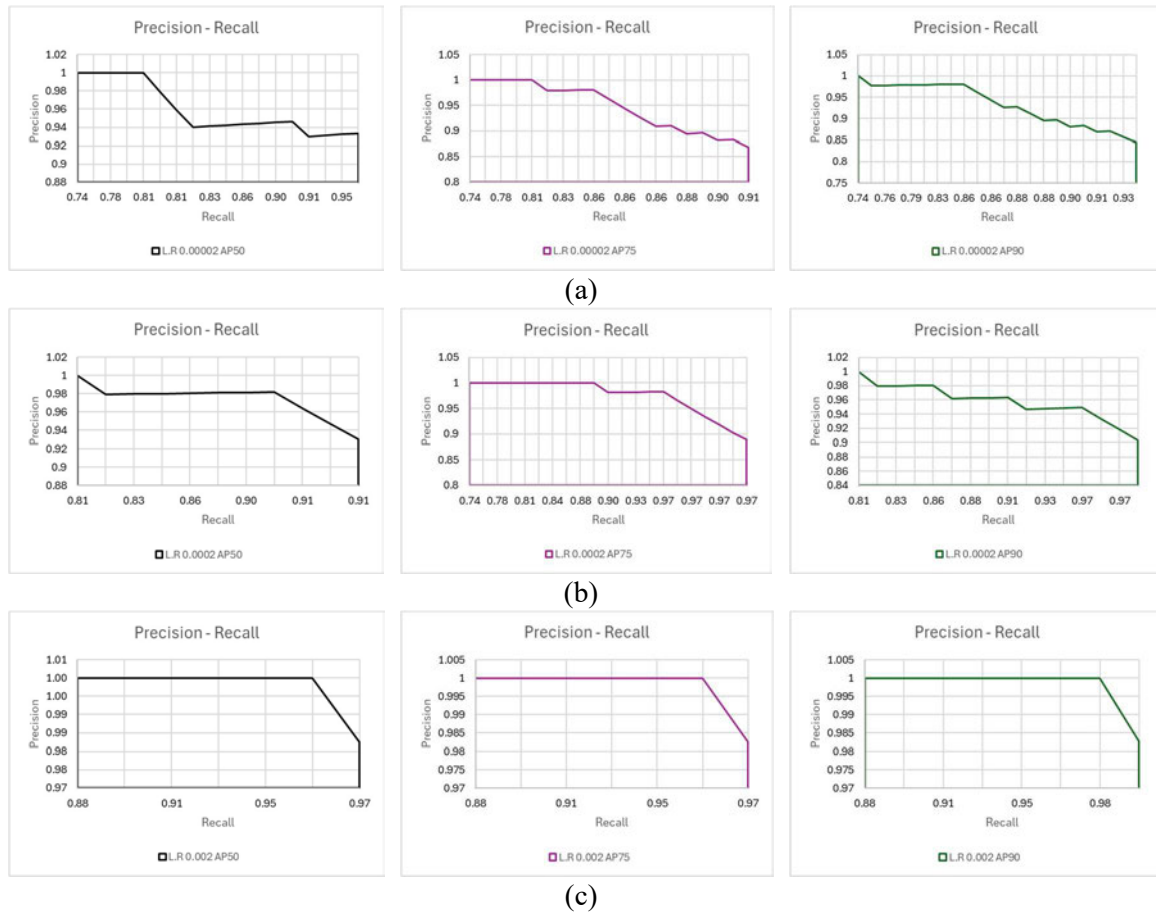


Figure 4.3. Mask R-CNN + ResNet101: (a) learning rate 0.00002 with an IoU of 50%, 75% and 90%; (b) learning rate 0.0002 with an IoU of 50%, 75% and 90%; (c) learning rate 0.002 with an IoU of 50%, 75% and 90%.

The losses of a model during the training process indicate whether a model can generalize well with the dataset it has been trained with and the new dataset that it is tested on. Figures 4.4 and 4.5 illustrate the loss characteristics of the models at learning rates of 0.00002, 0.0002, 0.002 and IoU thresholds at 50%, 75% and 90%. The blue line indicates how the value of the losses after each epoch relates to the training dataset, while the orange line indicates the losses after each epoch relating to the validation dataset that the model was tested on during the training process. As the model is trained, the losses are preferably decreased. If both values are decreasing as training continues. This illustrates that the model is generalizing well with the training and validation dataset.

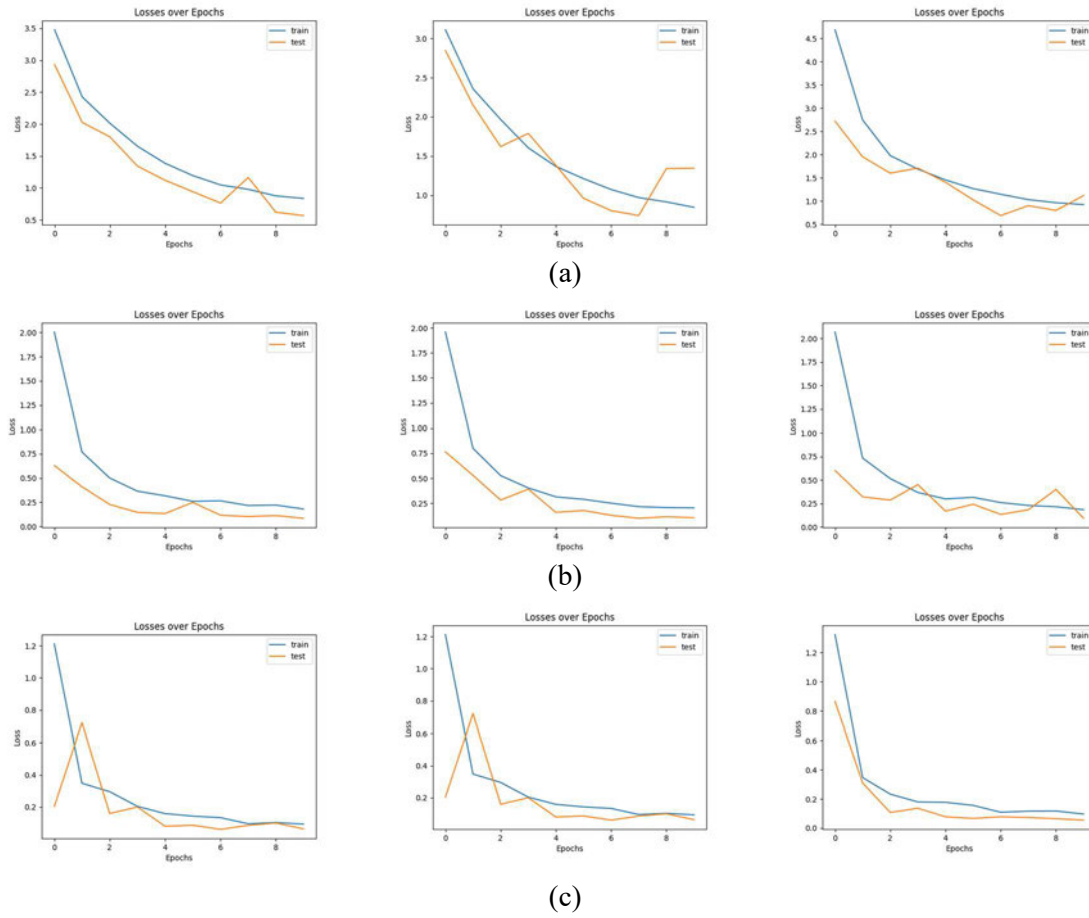


Figure 4.4. Mask R-CNN + ResNet50: (a) losses of the model with a learning rate 0.00002 and an IoU of 50%, 75% and 90%; (b) losses of the model with a learning rate 0.0002 and an IoU of 50%, 75% and 90%; (c) losses of the model with learning rate 0.002 and an IoU of 50%, 75% and 90%.

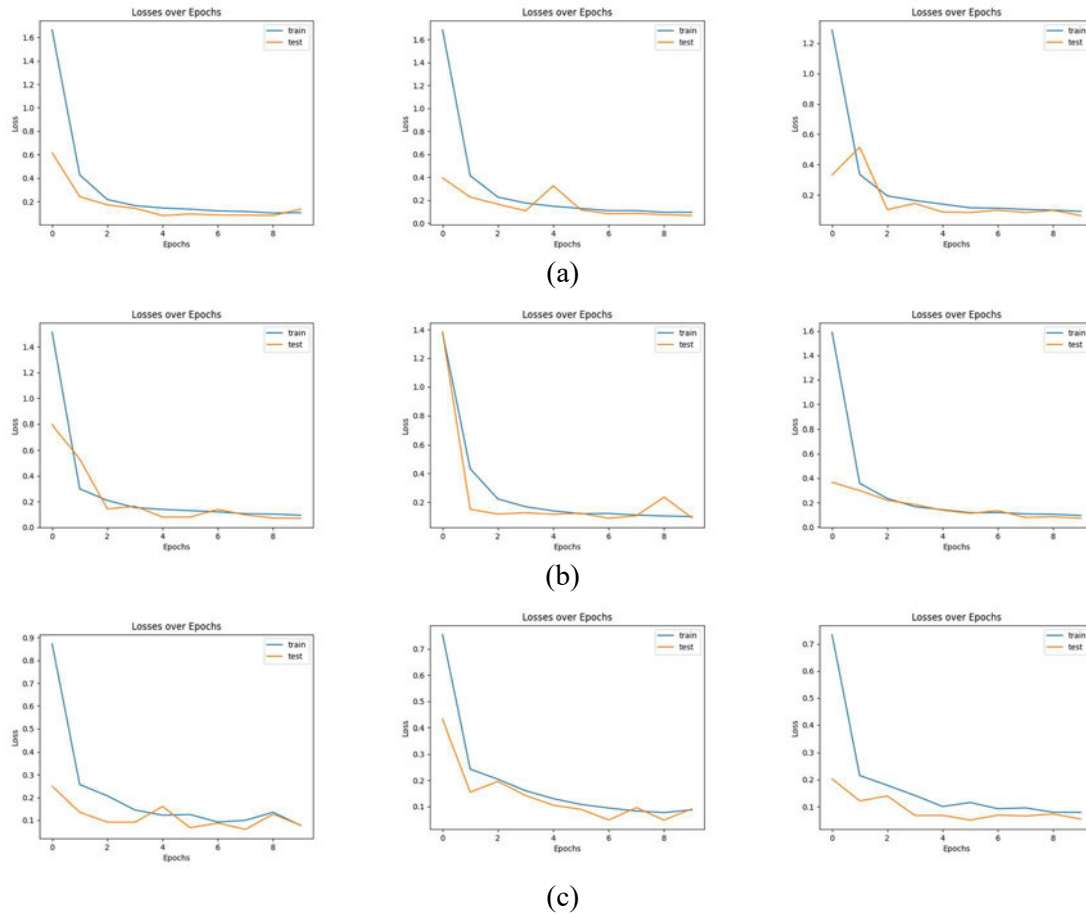


Figure 4.5. Mask R-CNN + ResNet101: (a) losses of the model with a learning rate 0.00002 and an IoU of 50%, 75% and 90%; (b) losses of the model with a learning rate 0.0002 and an IoU of 50%, 75% and 90%; (c) losses of the model with learning rate 0.002 and an IoU of 50%, 75% and 90%.

4.4 Overall Accuracy

The confusion matrix was used to assess how well the model was able to identify objects belonging to a certain class based on new unseen data. The output of the confusion matrix is the overall accuracy of the model. The training process provided information based on two backbone architectures used with Mask R-CNN. This was ResNet50 and ResNet101. Figures 4.6 to 4.8 illustrate the overall accuracy of the model with a learning rate of 0.00002 over 50%, 75% and 90% IoU thresholds using ResNet50 architecture. The model was not able to detect class D and E with multiple false positive detections. The model could not detect all objects related to classes B and C. Figures 4.9 to 4.11 illustrate the overall accuracy of the model with a learning rate of 0.0002 over 50%, 75% and 90% IoU thresholds using ResNet50 architecture. The model struggled to detect class E with multiple false positives.

Figures 4.12 to 4.14 illustrate the overall accuracy of the model with a learning rate of 0.002 over 50%, 75% and 90% IoU thresholds using ResNet50 architecture. The model's performance has superseded the previous results with a learning rate of 0.002. The model was able to detect all objects belonging to the correct class. False positive detections of classes A, B and E belonged to the background in all IoU thresholds.

Predicted	BG	0	0	3	1	6	6	16
	A	0	12	0	0	0	0	12
	B	1	0	23	0	0	0	24
	C	1	0	0	5	0	0	6
	D	0	0	0	0	0	0	0
	E	0	0	0	0	0	0	0
	SUM	2	12	26	6	6	6	58
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.6. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.00002 and an IoU of 50%.

Predicted	BG	0	0	5	0	6	2	13
	A	0	12	0	0	0	0	12
	B	1	0	21	0	0	4	26
	C	6	0	0	6	0	0	12
	D	0	0	0	0	0	0	0
	E	0	0	0	0	0	0	0
SUM		7	12	26	6	6	6	63
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.7. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.00002 and an IoU of 75%.

Predicted	BG	0	3	3	2	6	1	15
	A	0	9	0	0	0	0	9
	B	0	0	23	0	0	5	28
	C	0	0	0	4	0	0	4
	D	0	0	0	0	0	0	0
	E	0	0	0	0	0	0	0
SUM		0	12	26	6	6	6	56
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.8. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.00002 and an IoU of 90%.

Predicted	BG	0	0	0	0	0	0	
	A	1	12	2	0	0	15	
	B	1	4	26	0	0	37	
	C	0	0	0	6	0	6	
	D	0	0	0	0	6	6	
	E	1	0	0	0	0	1	
	SUM	3	16	28	6	6	65	
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.9. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.0002 and an IoU of 50%.

Predicted	BG	0	0	0	0	0	1	1
	A	2	12	2	0	0	0	16
	B	1	4	26	0	0	2	33
	C	0	0	0	6	0	0	6
	D	2	0	0	0	6	0	8
	E	0	0	0	0	0	3	3
	SUM	5	16	28	6	6	6	67
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.10. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.0002 and an IoU of 75%.

Predicted	BG	0	0	0	0	0	0	
	A	2	12	1	0	0	15	
	B	1	4	26	0	0	37	
	C	0	0	0	6	0	6	
	D	1	0	0	0	6	7	
	E	0	0	0	0	0	0	
	SUM	4	16	27	6	6	6	65
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.11. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.0002 and an IoU of 90%.

Predicted	BG	0	0	0	0	0	0
	A	1	12	0	0	0	13
	B	1	0	26	0	0	27
	C	0	0	0	6	0	6
	D	0	0	0	0	6	6
	E	0	0	0	0	0	6
	SUM	2	12	26	6	6	58
		BG	A	B	C	D	E
Actual							

Figure 4.12. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.002 and an IoU of 50%.

Predicted	BG	0	0	0	0	0	0	0
	A	1	12	0	0	0	0	13
	B	1	0	26	0	0	2	29
	C	0	0	0	6	0	0	6
	D	0	0	0	0	6	0	6
	E	1	0	0	0	0	6	7
	SUM	3	12	26	6	6	8	61
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.13. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.002 and an IoU of 75%.

Predicted	BG	0	0	0	0	0	0	0
	A	1	12	0	0	0	0	13
	B	1	0	26	0	0	0	27
	C	0	0	0	6	0	0	6
	D	0	0	0	0	6	0	6
	E	1	0	1	0	0	6	8
	SUM	3	12	27	6	6	6	60
		BG	A	B	C	D	E	SUM
Actual								

Figure 4.14. Confusion matrix: Mask R-CNN + ResNet50 with a learning rate of 0.002 and an IoU of 90%.

The second group of results were achieved from Mask R-CNN using ResNet101 as the architecture. This is illustrated in Figures 4.15 to 4.23. Figures 4.15 to 4.17 illustrate the overall accuracy of the model with a learning rate of 0.00002 over 50%, 75% and 90% IoU thresholds using ResNet101 architecture. The model struggled to identify all objects

belonging to class E and had multiple false positive detections in classes A, B and D. Figures 4.18 to 4.20 illustrate the overall accuracy of the model with a learning rate of 0.0002 over 50%, 75% and 90% IoU thresholds using ResNet101 architecture. The model performed better detecting all objects in comparison to the previous model. Figures 4.21 to 4.23 illustrate the overall accuracy of the model with a learning rate of 0.002 over 50%, 75% and 90% IoU thresholds using ResNet101 architecture. The set of results belonging to this model's parameter selection performed the best in comparison to all previous results mentioned. All objects were correctly detected as belonging to the correct class. There are instances of false positives being detected in the background only. These false positives may be removed in future works by sizing the detection window to only include the objects of interest.

Predicted	BG	0	0	0	0	0	0
	A	1	12	0	0	0	13
	B	0	0	26	0	0	26
	C	0	0	0	6	0	6
	D	3	0	0	0	6	9
	E	0	0	0	0	6	6
	SUM	4	12	26	6	6	6
	BG	A	B	C	D	E	SUM
	Actual						

Figure 4.15. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.00002 and an IoU of 50%.

Predicted	BG	0	0	0	0	0	1	1
	A	1	12	0	0	0	0	13
	B	5	0	26	0	0	2	33
	C	0	0	0	6	0	0	6
	D	0	0	0	0	6	0	6
	E	0	0	0	0	0	3	3
SUM		6	12	26	6	6	6	62
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.16. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.00002 and an IoU of 75%.

Predicted	BG	0	0	0	0	0	1	1
	A	1	12	0	0	0	0	13
	B	5	0	26	0	0	1	32
	C	0	0	0	6	0	0	6
	D	3	0	0	0	6	0	9
	E	0	0	0	0	0	4	4
SUM		9	12	26	6	6	6	65
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.17. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.00002 and an IoU of 90%.

Predicted	BG	0	0	0	0	0	0	0
	A	1	12	0	0	0	0	13
	B	2	0	26	0	0	2	30
	C	0	0	0	6	0	0	6
	D	0	0	0	0	6	0	6
	E	0	0	0	0	0	6	6
	SUM	3	12	26	6	6	8	61
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.18. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.0002 and an IoU of 50%.

Predicted	BG	0	0	0	0	0	0	0
	A	1	12	0	0	0	0	13
	B	6	0	26	0	0	0	32
	C	0	0	0	6	0	0	6
	D	0	0	0	0	6	0	6
	E	0	0	0	0	0	6	6
	SUM	7	12	26	6	6	6	63
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.19. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.0002 and an IoU of 75%.

Predicted	BG	0	0	0	0	0	0
	A	2	12	0	0	0	14
	B	4	0	26	0	0	30
	C	0	0	0	6	0	6
	D	0	0	0	0	6	6
	E	0	0	0	0	0	6
	SUM	6	12	26	6	6	62
		BG	A	B	C	D	E
Actual							

Figure 4.20. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.0002 and an IoU of 90%.

Predicted	BG	0	0	0	0	0	0	
	A	1	12	0	0	0	13	
	B	3	0	26	0	0	29	
	C	0	0	0	6	0	6	
	D	0	0	0	0	6	6	
	E	0	0	0	0	0	6	
	SUM	4	12	26	6	6	6	60
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.21. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.002 and an IoU of 50%.

Predicted	BG	0	0	0	0	0	0
	A	1	12	0	0	0	13
	B	2	0	26	0	0	28
	C	0	0	0	6	0	6
	D	0	0	0	0	6	6
	E	0	0	0	0	0	6
	SUM	3	12	26	6	6	59
		BG	A	B	C	D	E
Actual							

Figure 4.22. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.002 and an IoU of 75%.

Predicted	BG	0	0	0	0	0	0	
	A	1	12	0	0	0	13	
	B	0	0	26	0	0	26	
	C	0	0	0	6	0	6	
	D	0	0	0	0	6	6	
	E	0	0	0	0	0	6	
	SUM	1	12	26	6	6	6	57
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.23. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.002 and an IoU of 90%.

An additional dataset was used as illustrated in Figure 3.11. The addition of class A objects were added. However, using the best-performing hyperparameter, the model performed slightly worse as compared to the initial dataset that was used, this was due to the additional training dataset not matching the orientation of the validation dataset. All objects were

detected, however, there were a few instances where false positives were detected that belonged to the background of the object. This is illustrated in Figure 4.24.

Predicted	BG	0	0	0	0	0	0	
	A	1	12	0	0	0	13	
	B	1	0	24	0	0	25	
	C	1	0	0	6	0	7	
	D	1	0	0	0	6	7	
	E	0	0	2	0	0	6	
	SUM	4	12	26	6	6	6	60
		BG	A	B	C	D	E	SUM
		Actual						

Figure 4.24. Confusion matrix: Mask R-CNN + ResNet101 with a learning rate of 0.002 and an IoU of 90% trained on additional data.

4.5 Comparative Analysis

Table 4.1 contains values relating to the average precision over all classes based on the learning rate of 0.00002, 0.0002 and 0.002 across IoU thresholds of 50%, 75% and 90%. The AP is calculated from the area under the precision-recall curve as illustrated in Figure 4.2 and Figure 4.3. The precision and recall values at IoU = 50% are generally higher across all learning rates as this threshold is lenient when coming to bounding box overlap requirements. IoU = 75%, has a slight drop in average precision values as compared to IoU = 50%. This can be related to the model finding it more difficult to detect the objects accurately. IoU = 90% does have a lower score in comparison to the other mentioned thresholds. However, the opposite is noticeable when observing results using a learning rate of 0.002, which also was the best-performing model in both uses of ResNet50 and ResNet101. The third loss curve illustrated in Figure 4.4 (c) and Figure 4.5 (c) both show that the models with a learning rate of 0.002 at IoU = 90% seem to generalize better with the training and validation dataset during the training process showing no signs of overfitting or

underfitting. The AP values belonging to each learning rate do seem consistent irrespective of IoU values selected. Having attained a better value at a higher learning rate may be attributed to faster convergence. The quality of annotation used in the study followed a process of including all edges of objects within the dataset, allowing the model to learn to distinguish finer details between classes more effectively. The best-performing model based on the precision-recall curve was achieved through Mask R-CNN using ResNet101 based on a learning rate of 0.002 at an IoU = 90% as illustrated in Table 4.1.

Table 4.1. Average precision over all classes.

	Learning Rate	AP50(%)	AP75(%)	AP90(%)
Mask R-CNN + ResNet50	0.00002	70.60	68.95	59.59
	0.0002	92.22	90.93	84.05
	0.002	96.45	96.33	96.49
Mask R-CNN + ResNet101	0.00002	95.01	91.45	91.28
	0.0002	96.01	96.43	95.98
	0.002	95.55	96.55	98.27

Table 4.2 illustrates the overall accuracy of the Mask R-CNN model using the ResNet50 backbone based on the learning rates of 0.00002, 0.0002 and 0.002 across IoU thresholds of 50%, 75% and 90%. Class A had a constant error rate across learning rates 0.0002 and 0.002 while class B's error rate was higher in IoU thresholds 75% and 90%. Class C showed an instance of an error rate at a lower learning rate of 0.00002 while classes D and E showed error rates at higher learning rates of 0.0002 and 0.002.

Table 4.2. Overall accuracy using Mask R-CNN + ResNet50.

Mask R-CNN + ResNet50	Metric	Classes	Learning Rate 0.00002			Learning Rate 0.0002			Learning Rate 0.002		
			AP50 (%)	AP75 (%)	AP90 (%)	AP50 (%)	AP75 (%)	AP90 (%)	AP50 (%)	AP75 (%)	AP90 (%)
	OA		68.97	61.90	64.29	76.92	79.10	76.92	96.55	91.80	93.33
	Error of commission	A	0	7.69	0	20	25	20	7.69	7.69	7.69
		B	4.17	32.26	17.86	29.73	21.21	29.73	3.70	10.34	3.70
		C	16.67	0	0	0	0	0	0	0	0
		D	0	0	0	0	25	14.29	0	0	0
		E	0	0	0	100	0	0	0	14.29	25

Table 4.3 illustrates the overall accuracy of the Mask R-CNN model using the ResNet101 backbone based on the learning rates of 0.00002, 0.0002 and 0.002 across IoU thresholds of 50%, 75% and 90%. The model performed better at a higher learning rate of 0.002 with an increase in AP values at IoU = 90%. The model has learnt effectively at this threshold allowing for better object localization and mask segmentation. The best-performing model of 98.25% with relation to the overall accuracy was achieved at a learning rate of 0.002 at an IoU = 90% with an inference time of 400ms. The model was able to obtain real-time parts classification at a high overall accuracy. In comparison to the use of ResNet50, there is a notable increase in OA values across all learning rates. Suggesting that the use of a deeper neural network provides more robust results in an industrial environment. Class A showed a consistent error rate through all learning rates. This, however, was an object that was detected in the background and not a misidentification of other objects. Class B's error rate decreased as the learning rate increased. Class C, D and E had no error rates except class D at a lower learning rate. The results from both Tables 4.2 and 4.3 suggest that a model with a dataset given in an industrial environment tends to perform better at a higher learning rate. The analysis suggests that for the task required in an enamel coating line requiring a high IoU threshold, the use of Mask R-CNN using ResNet101 performed better with an optimized

learning rate of 0.002. An interesting observation based on the additional dataset used as illustrated in Figure 3.11 suggests that the model performed worse when using the parameters of the optimized model due to the variation in orientation of the objects between the training and validation dataset. The model with the additional data achieved an OA value of 90% as compared to 98.25%. This suggested that the model did not require additional data for better feature extraction.

Table 4.3. Overall accuracy using Mask R-CNN + ResNet101.

Mask R-CNN + ResNet101	Metric	Classes	Learning Rate 0.00002			Learning Rate 0.0002			Learning Rate 0.002			
			AP50 (%)	AP75 (%)	AP90 (%)	AP50 (%)	AP75 (%)	AP90 (%)	AP50 (%)	AP75 (%)	AP90(%)	
	OA		93.33	85.48	83.08	91.80	88.89	90.32	93.33	94.91	98.25	90
	Error of commission	A	7.69	7.69	7.69	7.69	7.69	14.29	7.69	7.69	7.69	7.69
		B	0	21.21	18.75	13.33	18.75	13.33	10.34	7.14	0	4
		C	0	0	0	0	0	0	0	0	0	14.29
		D	33.33	0	33.33	0	0	0	0	0	0	14.29
		E	0	0	0	0	0	0	0	0	0	25

CHAPTER 5 Analysis and Conclusion

5.1 Introduction

In this chapter, a comprehensive analysis is provided based on the results obtained from chapter four using the Mask R-CNN model following the prescribed methodology. Key findings are further discussed and limitations of the method are highlighted. Recommendations for further improvements in the industrial settings is given.

5.2 Summary of Research Objectives and Methods

The research aimed at solving the issue of incorrect data being received from the process line by incorporating an automated system that would add redundancy to a human-performed task that was currently being performed. This research aimed to provide a unique image processing solution compared to what is currently available in the coating industry. Mask R-CNN was used for this study with the choice of various learning rates and IoU thresholds. The methodology included the use of Mask R-CNN with two different ResNet backbones. The research objectives included the literature review of objectors used within an industrial environment in delivering robust results from the model while being fast enough to provide live data. After hardware configuration, the focus was placed on collecting data and building high-quality datasets. This dataset comprised of a train, validate and test portion which followed an 80:10:10 ratio split. Once the dataset was built, data augmentation was applied to further enhance the quality of the dataset to be used in the model during the training process. During the training process, the hyperparameters were adjusted and analysed to determine an optimized model. The trained models were then tested and evaluated to determine which model performed better. The evaluation was conducted through the use of the AP and the OA of each model. The models that had a higher AP and OA were determined as the superior model as compared to lower values. The dataset had 5 classes. Each portion was separated, and data was not shared amongst the portions. The number of images and their classes are presented in Table 3.1.

5.3 Key Findings and Analysis

The literature review in chapter two provided a base on model selection and establishing a balance between performance and speed. Table 2.1 provides a comparative analysis of previous state-of-the-art models and how they compare with the Mask R-CNN model based on the COCO dataset. Mask R-CNN had the highest mAP of 60.3% in 2017 as compared to other models. The backbone selection within a deep learning neural network has a notable impact on the results of the mAP as seen in Table 2.1. Where Faster R-CNN with ResNet101/FPN had achieved a mAP of 59.1% as compared to Faster R-CNN with VGGNet-16 achieving a lower mAP of 42.7%. This observation has also been investigated in this study with two backbone selections. These backbone selections included ResNet50 and ResNet101. The analysis of the model's performance based on the unique dataset can be split into sections that include the IoU thresholds, learning rates, backbones and error analysis. The IoU threshold selection when training the model provides crucial information on how well the model can generalize with all datasets relating to the overlap between the detected objects and the ground truth as illustrated in Table 4.1. The AP performance of a deep learning model tends to decrease as the IoU thresholds are increased as stricter criteria are applied to the model. This is true for the learning rates of 0.00002 and 0.002. However, it must be noted that for a larger learning rate, the model was able to achieve a higher AP using ResNet50 and ResNet101 of 96.49% and 98.27% respectively at IoU = 90%. The model's ability to attain a high AP at a high IoU suggests that the model is capable of producing robust detections with relation to the precision and recall metrics. With the chosen dataset size and maintained industrial environment conditions, the model achieved higher APs with a learning rate of 0.002. This suggests that the model can learn more complex patterns and features at an aggressive learning rate with the selected dataset and architecture. With the lower learning rates of 0.00002 and 0.0002, the model was able to perform moderately well. However, these AP values were lower as compared to the higher learning rate. Observing a learning rate of 0.00002, the AP values were lower with smaller increases. This suggests that the model was slower at learning not being able to provide the required optimal recall and precision performance. The performance across different model backbones like ResNet50 and ResNet101 illustrated in Table 4.1, provided insight into the model's robust capabilities. Having achieved a higher AP in both architectures at a learning

rate of 0.002. These results were consistent across all IoUs, being able to balance computational resources and detection performance. The ResNet101 provided a deeper network for training, allowing the model to learn more features from the objects in the dataset. This deeper network had a significant impact concerning multiple detections on one object, false detection on actual objects and no detection as illustrated in Figure 4.1. The detailed explanation regarding Figure 4.6 to Figure 4.23, illustrates the reduction of false positives on multiple object classes to only having false positives being detected in the background of the image. The initial challenges of multiple objects being detected on one object and no detections of certain objects were eliminated using Mask R-CNN with ResNet101. The overall accuracy as illustrated in Table 4.2 and Table 4.3 provides insight on the model's detection capability with unseen data and provides error analytics data. The error of commission, which is the false positive detection of the model, is another crucial metric displaying the model's mistakes. From the observation of both models' architectures, there has been a low error rate across all classes' overall learning rates and IoU thresholds not exceeding 33.33% except for the one instance of 100% for class E for a learning rate of 0.0002 at IoU = 50%. Suggesting that the model has a high sensitivity rate on average across all metrics. Based on the error rate related to class-specific performance in Table 4.2 and Table 4.3, classes A and B show the most instances of errors attained. This suggests that the model has had more of a challenge regarding intra-class variances between the classes.

5.4 Contributions of Findings

The main contribution of this study is the introduction of an automated technique for detecting and classifying manufactured metal parts on a conveyor line into their respective classes. The classification of these parts was done on a unique area of the manufacturing plant, which was the enamel paint coating line, thus eliminating the need for manual inspection. The contribution of this study resulted in the reduction of error rate as compared to manual human intervention. The adoption of Mask R-CNN, for this specific case study was highlighted. Mask R-CNNs of segmentation and classification allowed for detailed analysis with detected objects and the ground truth. The study also addressed the challenge of objects being detected in the background of the area of interest. This is a significant challenge in a conveyor-based inspection system. This challenge was minimized through

determining an optimized model that performed better than initial training iterations. Another challenge which was evident in the literature was the use of imbalanced data that replicated the ratio of the production line in one running shift. The imbalanced data provided insight into model performance based on the error analysis and the adaptability of intra-class variability as seen in classes A and B. With this variability, the optimized model was able to achieve a high AP and OA, suggesting that the Mask R-CNN model is capable of flexibility and scalability based on the dataset and environment used in this study. Finally, the optimized model met the stringent requirements needed within an industrial firm, achieving a high AP and OA relating to data quality requirements.

5.5 Conclusions and Future Research Recommendations

In this research, a novel application was presented using the Mask R-CNN model for detecting and classifying parts on an enamel coating line which has contributed to the field of image processing in industrial automation. The main objective of this research was to create a system that could be used locally to identify and count each part on a conveyor line after passing a certain process. This system was to be reliable in data capturing from the field and to provide information on production performance throughput. The aim included the reduction or elimination of repetitive human intervention needed to manually count the parts on the line.

The Mask R-CNN model using ResNet50 and ResNet101 as backbones, illustrated its capability to provide reliable and robust data in a controlled industrial environment. The Mask R-CNN model using ResNet101 as its backbone was able to achieve an impressive AP of 98.25% and an OA of 98.27% with a real-world image dataset from the field. The results highlight the model's ability to deliver reliable data with the challenges faced in computer vision applications. The model's ability to deliver high precision and recall values suggests that the model is well-suited for use in an industrial environment where precision and reliability are critical. The main finding during the analysis is that increasing the learning rate of the model resulted in a better AP and OA. This was true across all IoU thresholds used in the model. However, it must be noted that when the learning rates were set to 0.02 and 0.2, the model wasn't able to generalize well with the dataset which led to the model's

unstable convergence during the training process. The model's inference time of approximately 400 ms per frame made this application a viable solution for real-time data capturing on the production line. The line speed operates at 0.1m/s. The required capturing of 10 frames required a capture window of 4 seconds, the averaging of the objects in the 10 frames allowed for a reduced false positive detection. This made for a travel distance of 0.4 m, which was subsequently within the capture window period. The model was able to provide high accuracy values, however, there were instances in class A having a constancy false positive in all thresholds of 7.69%. This was due to the model predicting an object of class A in the background. When observing the results between classes B and E, which share the same features with differences in size only, the optimized Mask R-CNN model was able to differentiate between both classes even near one another.

Future improvements include the isolation of the area of interest where the objects of interest may be. This improvement would eliminate all false positives and reduce noise in the background. Another future improvement based on the accurate localization of the masks generated by the mask branch would be to incorporate the segmentation mask localization feature in determining spray gun localization to reduce enamel powder consumption and increase manufacturing quality during the spraying process. This research has successfully illustrated the potential of applying a deep learning model, specifically Mask R-CNN, to improve the efficiency relating to data quality through the automation of industrial processes. The insight in this research contributes to the future development of industrial automation and process optimization using deep learning techniques. The objectives met from this study are listed as follows:

1. A comprehensive review was conducted into state-of-the-art deep learning models with specific attention to Mask R-CNN, the review provided information on model selection, backbone selection, environment consideration and hyperparameter considerations.
2. Hardware was installed and successfully captured data used for training and validation processes. The hardware was capable enough to complete the study.
3. Image augmentation was applied to images to improve generalisation within the model. This contributed to the robustness of the model.

4. Comparative analysis after hyperparameter tuning illustrated performance improvements through an iterative process.
5. The results of the model were quantitatively evaluated using the confusion matrix, providing information on model detection accuracy.
6. Identification of future research recommendations was recognised with practical suggestions.

CHAPTER 6 References

- [1] V. Fremont, ‘The Digital Transformation of the Manufacturing Industry: Metamorphic Changes and Value Creation in the Industrial Network’, PhD dissertation, Acta Universitatis Upsaliensis, Uppsala, 2021.
- [2] D. Libes, S.-J. Shin, and J. Woo, “Considerations and Recommendations for Data Availability for Data Analytics for Manufacturing,” Oct. 2015. doi: 10.1109/BigData.2015.7363743.
- [3] B. Weber, H. Yarandi, M. Rowe, and J. Weber, “A comparison study: Paper-based versus web-based data collection and management,” *Applied nursing research: ANR*, vol. 18, pp. 182–5, Sep. 2005, doi: 10.1016/j.apnr.2004.11.003.
- [4] J. Wang, W. Zhang, Y. Shi, S. Duan, and J. Liu, *Industrial Big Data Analytics: Challenges, Methodologies, and Applications*. 2018.
- [5] Ts. Dr. A. Arumugam, A. Bhaumik, and S. Rangaraju, “International Journal of Research Publication and Reviews Digitalization and its Impact in Manufacturing Sector -Leveraging Digital Technology to Improve Efficiency and Productivity Background of Industry 4.0,” vol. 4, pp. 1149–1151, Jan. 2023.
- [6] M. M. Wahi, D. V. Parks, R. C. Skeate, and S. B. Goldin, “Reducing Errors from the Electronic Transcription of Data Collected on Paper Forms: A Research Data Case Study,” *J Am Med Inform Assoc*, vol. 15, no. 3, pp. 386–389, 2008, doi: 10.1197/jamia.M2381.
- [7] R. Panko, “Thinking is Bad: Implications of Human Error Research for Spreadsheet Research and Practice,” Feb. 2008.
- [8] I. Atkinson, “Accuracy of data transfer: Double data entry and estimating levels of error,” *Journal of Clinical Nursing*, vol. 21, pp. 2730–5, Oct. 2012, doi: 10.1111/j.1365-2702.2012.04353.x.
- [9] O. Haffner, E. Kučera, and D. Rosinová, “Applications of Machine Learning and Computer Vision in Industry 4.0,” *Applied Sciences*, vol. 14, no. 6, Art. no. 6, Jan. 2024, doi: 10.3390/app14062431.

- [10] L. D. Viveros Escamilla, A. Gómez-Espinosa, J. A. Escobedo Cabello, and J. A. Cantoral-Ceballos, “Maturity Recognition and Fruit Counting for Sweet Peppers in Greenhouses Using Deep Learning Neural Networks,” *Agriculture*, vol. 14, no. 3, Art. no. 3, Mar. 2024, doi: 10.3390/agriculture14030331.
- [11] S. Makris et al., “Artificial Intelligence in Manufacturing White paper Prepared by the Artificial Intelligence in Manufacturing Network -AIM-NET,” May 2023.
- [12] S. Balasubramanian and R. Scholar II, “Integration of Artificial Intelligence in the Manufacturing Sector: A Systematic Review of Applications and Implications,” *International Journal of Production Technology and Management*, vol. 14, pp. 1–11, Mar. 2023, doi: 10.17605/OSF.IO/3XPWN.
- [13] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05), Jun. 2005, pp. 886–893 vol. 1. doi: 10.1109/CVPR.2005.177.
- [14] Q. Zhu, M.-C. Yeh, K.-T. Cheng, and S. Avidan, “Fast Human Detection Using a Cascade of Histograms of Oriented Gradients,” in 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06), Jun. 2006, pp. 1491–1498. doi: 10.1109/CVPR.2006.119.
- [15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.
- [16] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation.” *arXiv*, Oct. 22, 2014. doi: 10.48550/arXiv.1311.2524.
- [17] R. Girshick, “Fast R-CNN.” *arXiv*, Sep. 27, 2015. doi: 10.48550/arXiv.1504.08083.
- [18] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.” *arXiv*, Jan. 06, 2016. doi: 10.48550/arXiv.1506.01497.
- [19] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection.” *arXiv*, May 09, 2016. doi: 10.48550/arXiv.1506.02640.
- [20] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN.” *arXiv*, Jan. 24, 2018. doi: 10.48550/arXiv.1703.06870.

- [21] H. Yaka and S. Durmus, "Experimental and numerical analysis of distortions after enamel coating process on oven catalytic side panel sheets," *Therm sci*, vol. 27, no. 4 Part B, pp. 3061–3069, 2023, doi: 10.2298/TSCI2304061Y.
- [22] D. Banabic, *Sheet Metal Forming Processes*. Berlin, Heidelberg: Springer, 2010. doi: 10.1007/978-3-540-88113-1.
- [23] W. F. Hosford and J. L. Duncan, "Sheet metal forming: A review," *JOM*, vol. 51, no. 11, pp. 39–44, Nov. 1999, doi: 10.1007/s11837-999-0221-5.
- [24] M. A. Ablat and A. Qattawi, "Numerical simulation of sheet metal forming: a review," *Int J Adv Manuf Technol*, vol. 89, no. 1–4, pp. 1235–1250, Mar. 2017, doi: 10.1007/s00170-016-9103-5.
- [25] L. Bragina et al., "Powder electrostatic enamelling of household appliances," *Iop Conference Series: Materials Science and Engineering*, vol. 25, Dec. 2011, doi: 10.1088/1757-899X/25/1/012012.
- [26] Guskov S 1996 *Proc. Powder Coating 96* (Indianapolis, USA, 18 September 1996)
- [27] "Batch Manufacturing - an overview | ScienceDirect Topics." Accessed: Jun. 22, 2024. [Online]. Available: <https://www.sciencedirect.com/topics/engineering/batch-manufacturing>
- [28] E. R. Larson, "2 - Why Use Plastic?," in *Thermoplastic Material Selection*, E. R. Larson, Ed., William Andrew Publishing, 2015, pp. 19–56. doi: 10.1016/B978-0-323-31299-8.00002-7.
- [29] P. Scallan, "1 - Introduction to manufacturing," in *Process Planning*, P. Scallan, Ed., Oxford: Butterworth-Heinemann, 2003, pp. 1–34. doi: 10.1016/B978-075065129-5/50002-7.
- [30] D. R. Kiran, "Chapter 13 - Types of production situations," in *Production Planning and Control*, D. R. Kiran, Ed., Butterworth-Heinemann, 2019, pp. 181–193. doi: 10.1016/B978-0-12-818364-9.00013-5.
- [31] F. Sgarbossa, E. Grosse, W. Neumann, D. Battini, and C. Glock, "Human factors in production and logistics systems of the future," *Annual Reviews in Control*, vol. 49, May 2020, doi: 10.1016/j.arcontrol.2020.04.007.
- [32] "RFID tags improve tracking, quality on Ford line in Mexico," *Control Engineering*. Accessed: Apr. 01, 2024. [Online]. Available: <https://www.controleng.com/articles/rfid-tags-improve-tracking-quality-on-ford-line-in-mexico/>

- [33] “GitHub - matterport/Mask_RCNN: Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow.” Accessed: Sep. 2, 2022. [Online]. Available: https://github.com/matterport/Mask_RCNN
- [34] P. Bharati and A. Pramanik, “Deep Learning Techniques—R-CNN to Mask R-CNN: A Survey,” in *Computational Intelligence in Pattern Recognition*, A. K. Das, J. Nayak, B. Naik, S. K. Pati, and D. Pelusi, Eds., Singapore: Springer, 2020, pp. 657–668. doi: 10.1007/978-981-13-9042-5_56.
- [35] M. Karthi, C. Niroshini Infantia, G. Subhashini, and V. Shyam Sundar, “Comparative Studies with Random Datasets Using Enhanced Faster R-CNN, Mask R-CNN, and Single Shot Detector,” in *International Virtual Conference on Industry 4.0*, R. J. Kannan, S. Geetha, S. Sashikumar, and C. Diver, Eds., Singapore: Springer Nature, 2023, pp. 221–232. doi: 10.1007/978-981-19-9989-5_19.
- [36] “Beyond track-and-trace: Using RFID on the factory floor,” *Control Engineering*. Accessed: Jun. 27, 2024. [Online]. Available: <https://www.controleng.com/articles/beyond-track-and-trace-using-rfid-on-the-factory-floor/>
- [37] “RFID tags: a powerful assistant for intelligent production line management | HUAHAO RFID,” Xuzhou Huahao Information Technology Co., Ltd. |. Accessed: Jun. 27, 2024. [Online]. Available: <https://www.huahaorfid.com/rfid-tags-a-powerful-assistant-for-intelligent-production-line-management/>
- [38] “The Top Uses Cases for RFID in Manufacturing,” SpotSee. Accessed: Jun. 27, 2024. [Online]. Available: <https://spotsee.io/blog/rfid-in-manufacturing-spotsee/>
- [39] CYBRA, “RFID in Manufacturing [Updated for 2024],” CYBRA. Accessed: Jun. 27, 2024. [Online]. Available: <https://cybra.com/rfid-in-manufacturing/>
- [40] E. Pereira et al., “RFID Tags for On-Metal Applications: A Brief Survey,” *Technologies*, vol. 12, no. 5, Art. no. 5, May 2024, doi: 10.3390/technologies12050058.
- [41] X. Li, B. Du, Y. Li, and K. Zhuang, “RFID-based tracking and monitoring approach of real-time data in production workshop,” *Assembly Automation*, vol. 39, no. 4, pp. 648–663, Jan. 2019, doi: 10.1108/AA-06-2018-080.
- [42] “Light curtains for efficient automation and more safety in the material flow ” SICK. Accessed: Jun. 02, 2024. [Online]. Available: <https://www.sick.com/at/en/sick->

- sensor-blog/light-curtains-for-efficient-automation-and-more-safety-in-the-material-flow/w/blog-efficient-safe-material-flow/
- [43] “A look inside light curtains,” Control Engineering. Accessed: Jun. 29, 2024. [Online]. Available: <https://www.controleng.com/articles/a-look-inside-light-curtains/>
- [44] “Light Curtains for Detection, Measurement And Object Recognition,” Norstat Safety, Automation & Connectivity Solutions. Accessed: Jun. 29, 2024. [Online]. Available: <https://norstat.com/light-curtains-for-detection-measurement-and-object-recognition/>
- [45] “Transforming the Light Curtain: Using Pattern and Object Recognition for Safer Material Transfer” SICK. Accessed: Jun. 02, 2024. [Online]. Available: <https://cdn.sick.com/media/content/h2f/h3d/9692995420190.pdf>
- [46] W. Rawat and Z. Wang, “Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review,” Neural Computation, vol. 29, pp. 1–98, Jun. 2017, doi: 10.1162/NECO_a_00990.
- [47] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.
- [48] P. Kumar, S. Luo, and K. Shaukat Dar, “A Comprehensive Review of Deep Learning Approaches for Animal Detection on Video Data,” International Journal of Advanced Computer Science and Applications, vol. 14, Dec. 2023, doi: 10.14569/IJACSA.2023.01411144.
- [49] P. Purwono et al., “Understanding of Convolutional Neural Network (CNN): A Review,” vol. 2, pp. 739–748, Jan. 2023, doi: 10.31763/ijrcs.v2i4.888.
- [50] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016. [Online]. Available: <http://www.deeplearningbook.org>
- [51] V. Dumoulin and F. Visin, “A guide to convolution arithmetic for deep learning.” arXiv, Jan. 11, 2018. doi: 10.48550/arXiv.1603.07285.
- [52] S. H. 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, vol. 378, pp. 112–119, Feb. 2020, doi: 10.1016/j.neucom.2019.10.008.

-
- [53] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights Imaging*, vol. 9, no. 4, Art. no. 4, Aug. 2018, doi: 10.1007/s13244-018-0639-9.
- [54] M. A. Saleem, N. Senan, F. Wahid, M. Aamir, A. Samad, and M. Khan, "Comparative Analysis of Recent Architecture of Convolutional Neural Network," *Mathematical Problems in Engineering*, vol. 2022, no. 1, p. 7313612, 2022, doi: 10.1155/2022/7313612.
- [55] S. Cong and Y. Zhou, "A review of convolutional neural network architectures and their optimizations," *Artif Intell Rev*, vol. 56, no. 3, pp. 1905–1969, Mar. 2023, doi: 10.1007/s10462-022-10213-5.
- [56] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998, doi: 10.1109/5.726791.
- [57] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition." *arXiv*, Apr. 10, 2015. doi: 10.48550/arXiv.1409.1556.
- [58] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition." *arXiv*, Dec. 10, 2015. doi: 10.48550/arXiv.1512.03385.
- [59] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *Journal of Machine Learning Research - Proceedings Track*, vol. 9, pp. 249–256, Jan. 2010.
- [60] K. He and J. Sun, "Convolutional Neural Networks at Constrained Time Cost." *arXiv*, Dec. 04, 2014. doi: 10.48550/arXiv.1412.1710.
- [61] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway Networks." *arXiv*, Nov. 03, 2015. doi: 10.48550/arXiv.1505.00387.
- [62] W. Liu et al., "SSD: Single Shot MultiBox Detector," vol. 9905, 2016, pp. 21–37. doi: 10.1007/978-3-319-46448-0_2.
- [63] Augustian, M. Neural Network Based Fault Detection on Painted Surface. Master's Thesis, Umeå University, Umeå, Sweden, 2017.
- [64] J. Molina, J. Solanes, L. Arnal, and J. Tornero, "On the detection of defects on specular car body surfaces," *Robotics and Computer-Integrated Manufacturing*, vol. 48, pp. 263–278, Dec. 2017, doi: 10.1016/j.rcim.2017.04.009.

- [65] Q. Zhou, R. Chen, H. Bin, C. Liu, J. Yu, and X. Yu, "An Automatic Surface Defect Inspection System for Automobiles Using Machine Vision Methods," *Sensors*, vol. 19, p. 644, Feb. 2019, doi: 10.3390/s19030644.
- [66] A. Paramarthalingam and M. T. T, "Machine parts recognition and defect detection in automated assembly systems using computer vision techniques," *Revista Técnica de la Facultad de Ingeniería Universidad del Zulia*, vol. 39, pp. 71–80, Jan. 2016.
- [67] L. Perez, Í. Rodríguez, N. Rodríguez, R. Usamentiaga, and D. García, "Robot Guidance Using Machine Vision Techniques in Industrial Environments: A Comparative Review," *Sensors*, vol. 16, p. 335, Mar. 2016, doi: 10.3390/s16030335.
- [68] A. Vijayakumar and S. Vairavasundaram, "YOLO-based Object Detection Models: A Review and its Applications," *Multimed Tools Appl*, Mar. 2024, doi: 10.1007/s11042-024-18872-y.
- [69] Y. Shi, Y. Li, X. Wei, and Y. Zhou, "A Faster-RCNN Based Chemical Fiber Paper Tube Defect Detection Method," in *2017 5th International Conference on Enterprise Systems (ES)*, Sep. 2017, pp. 173–177. doi: 10.1109/ES.2017.35.
- [70] J. Xiang, T. Dong, R. Pan, and W. Gao, "Clothing Attribute Recognition Based on RCNN Framework Using L-Softmax Loss," *IEEE Access*, vol. 8, pp. 48299–48313, 2020, doi: 10.1109/ACCESS.2020.2979164.
- [71] X. Zhou, W. Gong, W. Fu, and F. Du, "Application of deep learning in object detection," in *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*, May 2017, pp. 631–634. doi: 10.1109/ICIS.2017.7960069.
- [72] Md. T. Hosain, A. Zaman, M. Abir, S. Akter, S. Mursalin, and S. Khan, "Synchronizing Object Detection: Applications, Advancements and Existing Challenges," *IEEE Access*, vol. 12, pp. 54129–54167, Apr. 2024, doi: 10.1109/ACCESS.2024.3388889.
- [73] N. Rane, "YOLO and Faster R-CNN object detection for smart Industry 4.0 and Industry 5.0: applications, challenges, and opportunities," *SSRN Electronic Journal*, Jan. 2023, doi: 10.2139/ssrn.4624206.
- [74] D. Wang, J.-G. Wang, and K. Xu, "Deep Learning for Object Detection, Classification and Tracking in Industry Applications," *Sensors*, vol. 21, p. 7349, Nov. 2021, doi: 10.3390/s21217349.

- [75] N. Manakitsa, G. S. Maraslidis, L. Moysis, and G. F. Fragulis, "A Review of Machine Learning and Deep Learning for Object Detection, Semantic Segmentation, and Human Action Recognition in Machine and Robotic Vision," *Technologies*, vol. 12, no. 2, Art. no. 2, Feb. 2024, doi: 10.3390/technologies12020015.
- [76] A. Benali Amjoud and M. Amrouch, "Object Detection Using Deep Learning, CNNs and Vision Transformers: A Review," *IEEE Access*, vol. PP, pp. 1–1, Jan. 2023, doi: 10.1109/ACCESS.2023.3266093.
- [77] H. M. Ahmad and A. Rahimi, "Deep learning methods for object detection in smart manufacturing: A survey," *Journal of Manufacturing Systems*, vol. 64, pp. 181–196, Jul. 2022, doi: 10.1016/j.jmsy.2022.06.011.
- [78] A. Rahimi, M. Anvaripour, and K. Hayat, "Object Detection using Deep Learning in a Manufacturing Plant to Improve Manual Inspection," in *2021 IEEE International Conference on Prognostics and Health Management (ICPHM)*, Jun. 2021, pp. 1–7. doi: 10.1109/ICPHM51084.2021.9486529.
- [79] D. Buongiorno et al., "Object Detection for Industrial Applications: Training Strategies for AI-Based Depalletizer," *Applied Sciences*, vol. 12, no. 22, Art. no. 22, Jan. 2022, doi: 10.3390/app122211581.
- [80] T. D'Angelo et al., *Deep Learning-Based Object Detection for Digital Inspection in the Mining Industry*. 2019. doi: 10.1109/ICMLA.2019.00116.
- [81] H. Yun, E. Kim, D. M. Kim, H. W. Park, and M. B.-G. Jun, "Machine Learning for Object Recognition in Manufacturing Applications," *Int. J. Precis. Eng. Manuf.*, vol. 24, no. 4, pp. 683–712, Apr. 2023, doi: 10.1007/s12541-022-00764-6.
- [82] F. Saeed, M. J. Ahmed, M. J. Gul, K. J. Hong, A. Paul, and M. S. Kavitha, "A robust approach for industrial small-object detection using an improved faster regional convolutional neural network," *Sci Rep*, vol. 11, no. 1, p. 23390, Dec. 2021, doi: 10.1038/s41598-021-02805-y.
- [83] Y. Pi, N. D. Nath, and A. H. Behzadan, "Convolutional neural networks for object detection in aerial imagery for disaster response and recovery," *Advanced Engineering Informatics*, vol. 43, p. 101009, Jan. 2020, doi: 10.1016/j.aei.2019.101009.
- [84] Y. Chang, "Waste Detection Based On Mask R-CNN," laurea, Politecnico di Torino, 2022. Accessed: Jul. 06, 2024. [Online]. Available: <https://webthesis.biblio.polito.it/23504/>

- [85] L. Valdivielso Capell, “Realistic data augmentation in EM nanoparticle images for deep learning training enhancement,” masters, E.T.S. de Ingenieros Informáticos (UPM), 2022. Accessed: Jul. 06, 2024. [Online]. Available: <https://oa.upm.es/71424/>
- [86] Mohanapriya S, Natesan P, Mohana Saranya S, Sumithra P, Swathi M, and Yoga Priya P, “Instance Segmentation using Mask RCNN for Surveillance,” in 2022 6th International Conference on Computing Methodologies and Communication (ICCMC), Mar. 2022, pp. 235–241. doi: 10.1109/ICCMC53470.2022.9753924.
- [87] A. M. Hafiz and G. M. Bhat, “A Survey on Instance Segmentation: State of the art,” *Int J Multimed Info Retr*, vol. 9, no. 3, pp. 171–189, Sep. 2020, doi: 10.1007/s13735-020-00195-x.
- [88] J. Liu, B. Yang, M. Li, and D. Xu, “Assessing Forest-Change-Induced Carbon Storage Dynamics by Integrating GF-1 Image and Localized Allometric Growth Equations in Jiangning District, Nanjing, Eastern China (2017–2020),” *Forests*, vol. 15, no. 3, Art. no. 3, Mar. 2024, doi: 10.3390/f15030506.
- [89] Q. Chen et al., “Combining Cylindrical Voxel and Mask R-CNN for Automatic Detection of Water Leakages in Shield Tunnel Point Clouds,” *Remote Sensing*, vol. 16, no. 5, Art. no. 5, Jan. 2024, doi: 10.3390/rs16050896.
- [90] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature Pyramid Networks for Object Detection.” *arXiv*, Apr. 19, 2017. doi: 10.48550/arXiv.1612.03144.
- [91] Y. Li, B. Song, X. Kang, X. Du, and M. Guizani, “Vehicle-Type Detection Based on Compressed Sensing and Deep Learning in Vehicular Networks,” *Sensors*, vol. 18, no. 12, Art. no. 12, Dec. 2018, doi: 10.3390/s18124500.
- [92] M. Machefer, “Mask R-CNN Refitting Strategy for Plant Counting and Sizing in UAV Imagery,” *Remote Sensing*, vol. 12, Sep. 2020, doi: 10.3390/rs12183015.
- [93] H. Huang, Z. Wei, and L. Yao, “A Novel Approach to Component Assembly Inspection Based on Mask R-CNN and Support Vector Machines,” *Information*, vol. 10, p. 282, Sep. 2019, doi: 10.3390/info10090282.
- [94] S. Puliti et al., FOR-instance: a UAV laser scanning benchmark dataset for semantic and instance segmentation of individual trees. 2023.
- [95] B. Xu et al., “Automated cattle counting using Mask R-CNN in quadcopter vision system,” *Computers and Electronics in Agriculture*, vol. 171, p. 105300, Apr. 2020, doi: 10.1016/j.compag.2020.105300.

- [96] T. Naets, M. Huijsmans, P. Smyth, L. Sorber, and G. Lannoy, A Mask R-CNN approach to counting bacterial colony forming units in pharmaceutical development. 2021.
- [97] M. Rong, Z. Wang, B. Ban, and X. Guo, “Pest Identification and Counting of Yellow Plate in Field Based on Improved Mask R-CNN,” *Discrete Dynamics in Nature and Society*, vol. 2022, pp. 1–9, Mar. 2022, doi: 10.1155/2022/1913577.
- [98] M. Calabrese, L. Agnusdei, G. Fontana, G. Papadia, and A. del prete, Application of Mask R-CNN for Defect Detection in Printed Circuit Board manufacturing. 2023. doi: 10.21203/rs.3.rs-3491517/v1.
- [99] Y. Fu, J. Fan, S. Xing, Z. Wang, F. Jing, and M. Tan, “Image segmentation of cabin assembly scene based on improved RGB-D Mask R-CNN,” *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–1, Jan. 2022, doi: 10.1109/TIM.2022.3145388.
- [100] G. Liu, F. Wang, and H. Hui, “Application of Mask R-CNN in defect detection of industrial filter cloth,” in *2022 Global Conference on Robotics, Artificial Intelligence and Information Technology (GCRAIT)*, Jul. 2022, pp. 794–798. doi: 10.1109/GCRAIT55928.2022.00171.
- [101] Y. Liu, N. Mishra, P. Abbeel, and X. Chen, “Distributional Instance Segmentation: Modeling Uncertainty and High Confidence Predictions with Latent-MaskRCNN,” May 03, 2023, arXiv: arXiv:2305.01910. doi: 10.48550/arXiv.2305.01910.
- [102] H. He et al., “Mask R-CNN based automated identification and extraction of oil well sites,” *International Journal of Applied Earth Observation and Geoinformation*, vol. 112, p. 102875, Aug. 2022, doi: 10.1016/j.jag.2022.102875.
- [103] H. Kabir and H.-S. Lee, “Mask R-CNN-Based Stone Detection and Segmentation for Underground Pipeline Exploration Robots,” *Applied Sciences*, vol. 14, p. 3752, Apr. 2024, doi: 10.3390/app14093752.
- [104] A. Dutta and A. Zisserman, *The VIA Annotation Software for Images, Audio and Video*. (Oct. 21, 2019). Nice, France. [Online]. Available: <https://doi.org/10.1145/3343031.3350535>
- [105] H. Li, X. Ma, and H. Liu, “Cross-modal feature fusion Mask R-CNN and point cloud normalization segmentation transformation for fish length estimation,” *Aquacult Int*, Jul. 2024, doi: 10.1007/s10499-024-01610-4.

- [106] H. Zhu, Y. Wang, and J. Fan, “IA-Mask R-CNN: Improved Anchor Design Mask R-CNN for Surface Defect Detection of Automotive Engine Parts,” *Applied Sciences*, vol. 12, p. 6633, Jun. 2022, doi: 10.3390/app12136633.
- [107] F. Huang, Y. Li, Z. Liu, L. Gong, and C. Liu, “A Method for Calculating the Leaf Area of Pak Choi Based on an Improved Mask R-CNN,” *Agriculture*, vol. 14, p. 101, Jan. 2024, doi: 10.3390/agriculture14010101.
- [108] T. Kasinathan and U. S. Reddy, “Detection of fall armyworm (*spodoptera frugiperda*) in field crops based on mask R-CNN,” *Signal, Image and Video Processing*, vol. 17, pp. 1–7, Jan. 2023, doi: 10.1007/s11760-023-02485-3.
- [109] Y. Wei, S. Tran, S. Xu, B. Kang, and M. Springer, “Deep Learning for Retail Product Recognition: Challenges and Techniques,” *Computational Intelligence and Neuroscience*, vol. 2020, no. 1, p. 8875910, 2020, doi: 10.1155/2020/8875910.
- [110] T. Citlak and N. Pillay, “Real-Time Object Classification on an Enamel Paint Coating Conveyor Line Using Mask R-CNN,” *Automation*, vol. 5, pp. 213–229, Jun. 2024, doi: 10.3390/automation5030013.
- [111] “COCO - Common Objects in Context.” Accessed: Jun. 01, 2024. [Online]. Available: <https://cocodataset.org/#detection-eval>
- [112] H. Patel, *A Comprehensive Study on Object Detection Techniques in Unconstrained Environments*. 2023.
- [113] “The PASCAL Visual Object Classes Challenge 2012 (VOC2012).” Accessed: Jul. 25, 2024. [Online]. Available: <http://host.robots.ox.ac.uk/pascal/VOC/voc2012/>
- [114] “ImageNet.” Accessed: Jul. 25, 2024. [Online]. Available: <https://www.image-net.org/download.php>
- [115] S. Singh, A. Yadav, J. Jain, H. Shi, J. Johnson, and K. Desai, “Benchmarking Object Detectors with COCO: A New Path Forward,” Mar. 27, 2024, arXiv: arXiv:2403.18819. doi: 10.48550/arXiv.2403.18819.
- [116] R. Rubin, C. Jacob, S. M. Anzar, and A. Panthakkan, “Mask R-CNN with Multi-Backbones - A Comparative Analysis,” in *2022 5th International Conference on Signal Processing and Information Security (ICSPIS)*, Dec. 2022, pp. 55–60. doi: 10.1109/ICSPIS57063.2022.10002546.
- [117] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick, “Inside-Outside Net: Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks,” Dec. 13, 2015, arXiv: arXiv:1512.04143. doi: 10.48550/arXiv.1512.04143.

- [118] A. Shrivastava, A. Gupta, and R. Girshick, “Training Region-based Object Detectors with Online Hard Example Mining,” Apr. 12, 2016, arXiv: arXiv:1604.03540. doi: 10.48550/arXiv.1604.03540.
- [119] Y. Liu, R. Wang, S. Shan, and X. Chen, “Structure Inference Net: Object Detection Using Scene-Level Context and Instance-Level Relationships,” in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Jun. 2018, pp. 6985–6994. doi: 10.1109/CVPR.2018.00730.
- [120] Z. Cai and N. Vasconcelos, “Cascade R-CNN: Delving Into High Quality Object Detection,” in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Jun. 2018, pp. 6154–6162. doi: 10.1109/CVPR.2018.00644.
- [121] S. Ren, K. He, R. Girshick, X. Zhang, and J. Sun, “Object Detection Networks on Convolutional Feature Maps,” Aug. 17, 2016, arXiv: arXiv:1504.06066. doi: 10.48550/arXiv.1504.06066.
- [122] J. Dai, Y. Li, K. He, and J. Sun, “R-FCN: Object Detection via Region-based Fully Convolutional Networks,” Dec. 11, 2023, arXiv: arXiv:1605.06409. doi: 10.48550/arXiv.1605.06409.
- [123] H. Wang, Q. Wang, M. Gao, P. Li, and W. Zuo, “Multi-scale Location-aware Kernel Representation for Object Detection,” Apr. 02, 2018, arXiv: arXiv:1804.00428. doi: 10.48550/arXiv.1804.00428.
- [124] X. Chen and A. Gupta, “Spatial Memory for Context Reasoning in Object Detection,” Apr. 13, 2017, arXiv: arXiv:1704.04224. doi: 10.48550/arXiv.1704.04224.
- [125] Y. Zhu, C. Zhao, J. Wang, X. Zhao, Y. Wu, and H. Lu, “CoupleNet: Coupling Global Structure with Local Parts for Object Detection,” Aug. 09, 2017, arXiv: arXiv:1708.02863. doi: 10.48550/arXiv.1708.02863.
- [126] S. Zagoruyko et al., “A MultiPath Network for Object Detection,” Aug. 08, 2016, arXiv: arXiv:1604.02135. doi: 10.48550/arXiv.1604.02135.
- [127] Y. Zhu, C. Zhao, H. Guo, J. Wang, X. Zhao, and H. Lu, “Attention CoupleNet: Fully Convolutional Attention Coupling Network for Object Detection,” IEEE Transactions on Image Processing, vol. 28, no. 1, pp. 113–126, Jan. 2019, doi: 10.1109/TIP.2018.2865280.
- [128] J. Huang et al., “Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors,” in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jul. 2017, pp. 3296–3297. doi: 10.1109/CVPR.2017.351.

- [129] S. Wang, Y. Gong, J. Xing, L. Huang, C. Huang, and W. Hu, “RDSNet: A New Deep Architecture for Reciprocal Object Detection and Instance Segmentation,” Dec. 10, 2019, arXiv: arXiv:1912.05070. doi: 10.48550/arXiv.1912.05070.
- [130] A. Shrivastava, R. Sukthankar, J. Malik, and A. Gupta, “Beyond Skip Connections: Top-Down Modulation for Object Detection,” Sep. 19, 2017, arXiv: arXiv:1612.06851. doi: 10.48550/arXiv.1612.06851.
- [131] J. Dai et al., “Deformable Convolutional Networks,” Jun. 05, 2017, arXiv: arXiv:1703.06211. doi: 10.48550/arXiv.1703.06211.
- [132] L. Liu et al., “Deep Learning for Generic Object Detection: A Survey,” Aug. 22, 2019, arXiv: arXiv:1809.02165. doi: 10.48550/arXiv.1809.02165.
- [133] D. G. Lowe, “Object recognition from local scale-invariant features,” in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Sep. 1999, pp. 1150–1157 vol.2. doi: 10.1109/ICCV.1999.790410.
- [134] T.-Y. Lin et al., “Microsoft COCO: Common Objects in Context,” Feb. 20, 2015, arXiv: arXiv:1405.0312. doi: 10.48550/arXiv.1405.0312.
- [135] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The Pascal Visual Object Classes (VOC) Challenge,” *Int J Comput Vis*, vol. 88, no. 2, pp. 303–338, Jun. 2010, doi: 10.1007/s11263-009-0275-4.
- [136] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely Connected Convolutional Networks,” Jan. 28, 2018, arXiv: arXiv:1608.06993. doi: 10.48550/arXiv.1608.06993.
- [137] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal Loss for Dense Object Detection,” Feb. 07, 2018, arXiv: arXiv:1708.02002. doi: 10.48550/arXiv.1708.02002.
- [138] F. Perronnin, J. Sánchez, and T. Mensink, “Improving the Fisher Kernel for Large-Scale Image Classification,” in *Computer Vision – ECCV 2010*, K. Daniilidis, P. Maragos, and N. Paragios, Eds., Berlin, Heidelberg: Springer, 2010, pp. 143–156. doi: 10.1007/978-3-642-15561-1_11.
- [139] A. Esteva et al., “Dermatologist-level classification of skin cancer with deep neural networks,” *Nature*, vol. 542, no. 7639, pp. 115–118, Feb. 2017, doi: 10.1038/nature21056.
- [140] P. Domingos, “A few useful things to know about machine learning,” *Commun. ACM*, vol. 55, no. 10, pp. 78–87, Oct. 2012, doi: 10.1145/2347736.2347755.

- [141] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008, doi: 10.1145/1327452.1327492.
- [142] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. C. Martin, S. Mellor, K. Schwaber, J. Sutherland, and D. Thomas, "Manifesto for Agile Software Development," 2001. [Online]. Available: <https://agilemanifesto.org/>. [Accessed: 25-Jul-2024].
- [143] "Electronic Shutter Types | Basler Product Documentation." Accessed: Aug. 01, 2024. [Online]. Available: <https://docs.baslerweb.com/electronic-shutter-types#global-shutter>
- [144] "CUDA Deep Neural Network," NVIDIA Developer. Accessed: Aug. 01, 2024. [Online]. Available: <https://developer.nvidia.com/cudnn>
- [145] "pylon Viewer - pylon Software Suite." Accessed: Aug. 01, 2024. [Online]. Available: <https://www.baslerweb.com/en/software/pylon/pylon-viewer/>
- [146] "Keras: Deep Learning for humans." Accessed: Aug. 01, 2024. [Online]. Available: <https://keras.io/>
- [147] "What is TensorFlow?," NVIDIA Data Science Glossary. Accessed: Aug. 01, 2024. [Online]. Available: <https://www.nvidia.com/en-us/glossary/tensorflow/>
- [148] C. Shorten and T. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *Journal of Big Data*, vol. 6, Jul. 2019, doi: 10.1186/s40537-019-0197-0.
- [149] Y. Bengio, "Practical Recommendations for Gradient-Based Training of Deep Architectures," in *Neural Networks: Tricks of the Trade: Second Edition*, G. Montavon, G. B. Orr, and K.-R. Müller, Eds., Berlin, Heidelberg: Springer, 2012, pp. 437–478. doi: 10.1007/978-3-642-35289-8_26.
- [150] D. Wilson and T. Martinez, The need for small learning rates on large problems, vol. 1. 2001, p. 119 vol.1. doi: 10.1109/IJCNN.2001.939002.
- [151] F. J. Dijkstra, "Explaining L1 and L2 regularization in machine learning," Medium. Accessed: Aug. 07, 2024. [Online]. Available: <https://medium.com/@fernando.dijkinga/explaining-l1-and-l2-regularization-in-machine-learning-2356ee91c8e3>
- [152] K. Team, "Keras documentation: Model training APIs." Accessed: Aug. 09, 2024. [Online]. Available: https://keras.io/api/models/model_training_apis/

-
- [153] Z. Luo, Z. Fang, S. Zheng, Y. Wang, and Y. Fu, “NMS-Loss: Learning with Non-Maximum Suppression for Crowded Pedestrian Detection,” in Proceedings of the 2021 International Conference on Multimedia Retrieval, Aug. 2021, pp. 481–485. doi: 10.1145/3460426.3463588.
- [154] “Precision-Recall,” scikit-learn. Accessed: Aug. 11, 2024. [Online]. Available: https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html