Automated PCB Defect Detection and Classification Using Deep Learning

A Project Report

submitted by

ROHAIL ALAM (EC20B1072)

in partial fulfilment of requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

Department of Electronics and Communication Engineering
INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, DESIGN AND MANUFACTURING KANCHEEPURAM

May 2024

DECLARATION OF ORIGINALITY

I, **Rohail Alam**, with Roll No: **EC20B1072** hereby declare that the material presented in the Project Report titled **Automated PCB Defect Detection and Classification Using Deep Learning** represents original work carried out by me in the **Department of Electronics and Communication Engineering** at the Indian Institute of Information Technology, Design and Manufacturing, Kancheepuram.

With my signature, I certify that:

- I have not manipulated any of the data or results.
- I have not committed any plagiarism of intellectual property. I have clearly indicated and referenced the contributions of others.
- I have explicitly acknowledged all collaborative research and discussions.
- I have understood that any false claim will result in severe disciplinary action.
- I have understood that the work may be screened for any form of academic misconduct.

Rohail Alam

Place: Chennai

Date: 15.05.2024

CERTIFICATE

This is to certify that the report titled **Automated PCB Defect Detection and Classification Using Deep Learning**, submitted by **Rohail Alam (EC20B1072)**, to the Indian Institute of Information Technology, Design and Manufacturing Kancheepuram, for the award of the degree of **BACHELOR OF TECHNOLOGY** is a bona fide record of the work done by him/her under my supervision. The contents of this report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. M D Selvaraj
Project Guide
Professor
Department of Electronics and Communication Engineering
IIITDM Kancheepuram, 600 127

Place: Chennai

Date: 15.05.2024

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my project guide, Professor M D Selvaraj for his unwavering support in swiftly responding to any queries I had regarding the final year project and for providing me the liberty to select the project domain of my choice. He has been very understanding with regard to my plan to pursue higher education in the same domain, which I am very grateful for.

I am highly indebted to the administration of Indian Institute of Information Technology, Kancheepuram for providing me with this wonderful opportunity of getting a more in-depth experience in my field of interest. The courses conducted by the university my four years of study has helped shape up my technical prowess and soft skills, which will be of great use in future endeavors.

The experience I have garnered from this project has helped me get an in-depth understanding of the various technicalities involved. It has also developed my project management skills and has taught me the importance of meticulous planning, time management, and task prioritization.

ABSTRACT

It is understood that automated detection of defects on electronic boards is held in high regard in the electronics manufacturing industry. The traditional manual inspection processes for defect identification are prone to human error, and is time-consuming as well as labor-intensive. Therefore, developing an automated defect detection system would greatly reduce the inconvenience of manual verification. The introduction of such a system will surge the efficiency and reliability of quality control procedures in contrast to the time-consuming and tedious process of manually inspecting every board.

The primary objectives that this project aims to accomplish are as follows :

- `Enhanced Quality Control` : By implementing an automated defect detection system, manufacturers can achieve precise and consistent identification of defects on electronic boards. This automation mitigates the risk of undetected defects, leading to improved overall quality control of the boards.
- `Improved Efficiency` : Automation of manual inspection processes will further enhance the precision of the tasks performed, and makes the system less prone to errors. As a result of this, manufacturers can experience faster cycle times, save more on costs and notice higher productivity.
- `Product Reliability` : Early detection and rectification of defects in the inspection process can ensure the production of more reliable and functional electronic boards, therefore minimizing the likelihood of faulty products reaching the end user or customer.

KEYWORDS: PCB; Object Detection Networks; Deep Learning;
Image Processing; Data Augmentation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

YOLO	You Only Look Once
PCB	Printed Circuit Board
AOI	Automated Optical Inspection
CNN	Convolutional Neural Network
CSP	Cross Stage Partial
IoU	Intersection over Union
CIoU	Complete Intersection over Union
BCE	Binary Cross Entropy
SiLU	Sigmoid-weighted Linear Unit
mAP	Mean Average Precision
TP	True Positive
FN	False Negative
FP	False Positive
GPU	Graphics Processing Unit
CUDA	Compute Unified Device Architecture
R-CNN	Region-Based Convolutional Neural Network
UI	User Interface

NOTATION

π	Pi (3.14159...)
e	Euler's Number (2.71828...)
$\circledast$	Convolution Operation
σ	Sigmoid Function
$\rho(x, y)$	Euclidean Distance between two points x and y
ms	Milliseconds
L_{CIoU}	Loss function for CIoU
v	Aspect Ratio Factor
w, h	Length and Width of Prediction box respectively
w^{gt}, h^{gt}	Length and Width of Ground Truth box respectively
c	Diagonal formed by furthest boundaries of the prediction box and true box
b, b^{b^t}	Centroid locations of prediction box and true box respectively
$^\circ$	Angle (in degrees)
$\sum$	Summation

CHAPTER 1

INTRODUCTION

1.1 Background

The printed circuit board (PCB) is considered a vital component in the electronics industry as it plays a key role in many electronic devices, providing both mechanical integrity and electrical connectivity to the device. The PCB is used in a wide variety of devices, ranging from a calculator the size of your palm to heavy-grade military equipment. It is thus understood how important it is to ensure the correct operation of these PCBs. There is a very minimal scope of error for a device this important, so it should be manufactured in the most careful and precise manner. Unfortunately, the currently available technology is not capable of ensuring 100 percent quality rate with the increasingly complex and delicate boards of today. Due to this limitation, there is a risk of occurrence of defects on the board. In order to ascertain the safety of the electronic device equipped with the PCB, it is necessary to detect and indicate any defect on the surface of the board. Even when some defects do not completely hinder the function of the board, it can be the cause of early failure due to hastened degradation.

It is understood that the inspection process should be executed after the etching stage of PCB manufacturing. The main object of the etching process is to remove excess unwanted copper deposits present on the board that deviates from the expected circuit pattern. Defective boards found after the etching process are simply thrown away, which is uneconomical since the etching process costs close to 70% of the entire process. [1]

1.2 Motivation

Before computer based non-contact inspection methods were introduced, a PCB manufacturer would test the integrity of their boards through a series of visual and electrical

tests carried out by an experienced technician, which would require a large investment for recruiting and training. The process of manually testing every board is very time-intensive and not at all efficient. Moreover, there is also scope of human error during inspection of those defects that cannot be seen by a simple glance even by the most proficient worker. As of recent times, the procedure of Automated Optical Inspection (AOI) has been gaining traction. It provides numerous advantages over manual inspection such as reduced labour requirement, lesser scope of error and faster inspection time. The problem with this method of inspection is that it has a slow detection rate and requires very costly, high-end equipment to set up. In addition to this, it is very sensitive to the illumination of the environment it is in, due to which AOI tends to give false positives even if there is a slight change in the lighting condition of the inspection station.

With the oncoming of Deep Learning and Neural Networks, the prospect of bypassing the expensive hardware costs of AOI as well as its robustness against different environments makes it a promising option to use for board inspection. With the help of data augmentation, the deep learning model can be trained on a wide range of data replicating various environmental conditions and therefore prepare the model to detect defects on the board while not heavily relying on any other dependecies like it is in the case of AOI. CNN-based deep learning models have proved to be very viable due to its high accuracy and quick inspection time. The YOLO object detection model is one such CNN which is very popularly used due to its better performance as compared against other CNN architectures [2] and strong community support. The project includes a defect detection model trained on the latest version of YOLO, YOLOv8. Detailed description of the model architecture utilized and training results are included in the subsequent chapters.

1.3 Defect Description

The defects that occur on a PCB can be divided into two categories - solder and non-solder defects. [3] As the name suggests, solder defects arise during the soldering process, which is caused due to a variety of factors which include defective solder-

ing iron tips and faulty soldering technique. Non-solder defects usually occur due to external factors such as power supply issues, incorrect placement of component and design failures. Among these defects, the most commonly occurring surface defects are : Missing Holes, Mouse Bites, Open Circuits, Shorts, Spurs and Spurious Copper. [4]. The presence of these defects on the PCB leads to a myriad of problems such as safety hazards, faulty functioning and higher manufacturing cost as a result of failing to identify the defect on time. In actuality, there are much more than just six classes of board defects. The reason why we exclusively select these classes is because they are the most commonly occurring and have the biggest impact on the performance of the board. Addressing of these defects is done through dedicated machines designed to aid the user in identification and correction of defects. So while it is important to automate the defect identification process, emphasis should also be put on the developing of an interface which is easy to use and does not require any special training or an experienced technician to identify the defects.

CHAPTER 2

LITERATURE REVIEW

The defects that may occur on a PCB board is organized on the basis of the feature of the board it impacts. Each class is further divided into sub-classes that is decided by the appearance, cause or impact of the defect on the PCB. The most significant and commonly occurring category is the *trace* defect category. A trace is a conducting path that electrically connects one or more points of a PCB layer. It is considered of high importance as the PCB is laden with these connections, malfunctioning of even one could make the difference between a functional board and a defective board. The trace defect category can be further broken down into four sub-categories: defects that are additive to the area of the conductor, subtractive to the area of the conductor, damaging to the conductor, and designed with error. [5]

Traditional methods of board inspection are currently futile, as it not only requires additional resources to be allocated towards training a designated individual but is also prone to human error. In addition, with the rapidly advancing technology used to create the PCBs, the components are becoming further diminutive in size thus making it all the more difficult to identify defects from the naked eye or with magnification equipment.

Inspection methods that utilize image processing, such as that developed by Malge et al. [6], require the board images to follow a specific template. Any deviation from the said template would yield incorrect results. Another implementation of a template based defect detection system was developed by Ismail et al. [7] , which consists of the typically used image comparison methods like image subtraction and addition. In order to replicate template conditions, there is a heavy dependency in the external factors of the inspection station, such as lighting conditions and board orientation. It is also very costly to setup due to the sophisticated equipment used. The complexities involved in an AOI system can be observed in Figure 2.1

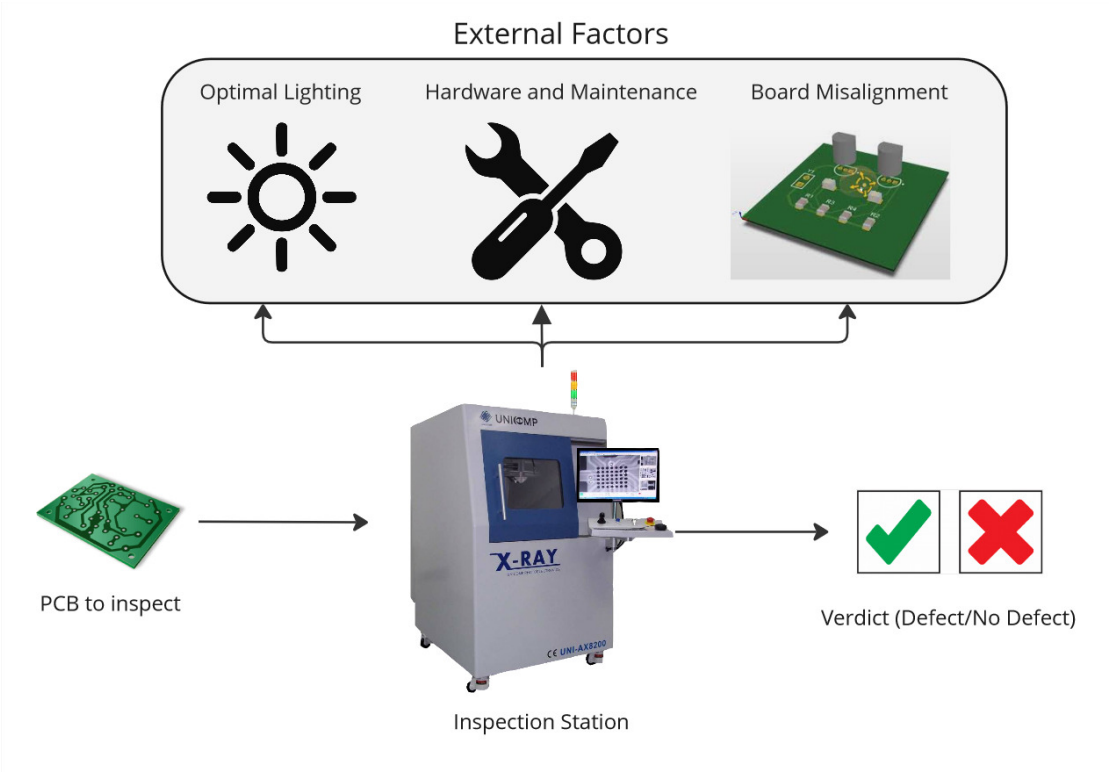


Figure 2.1: Automated Optical Inspection Flow

Modern implementations of image processing solutions make use of Machine Learning, which not only eliminates the need of a template but also significantly reduces the costs incurred in the buying of hardware and other inspection apparatus. Yun et al. [8] used an SVM classifier to carry out inspection of PCB solder joints and identify overfilling/underfilling. The issue in using such an architecture belies in how it was trained. The SVM was trained on solder joint images, and not entire board images. This would imply that the user would first have to individually segment out each solder joint on the board to carry out classification, which is a very tedious process. Furthermore, this model only deals with one specific defect - and not the many other defects that may occur on the PCB.

Recently, deep learning architectures have gained traction in the field of image processing , such as the use of YOLOv5 architecture by Tang, Liu, et al [9]. Using deep learning eliminates the requirement of a dedicated hardware setup. The goal of this project is to develop a better performing model using the new YOLOv8 architecture. The constituents of a typical deep learning based inspection system can be referred to in Figure 2.2

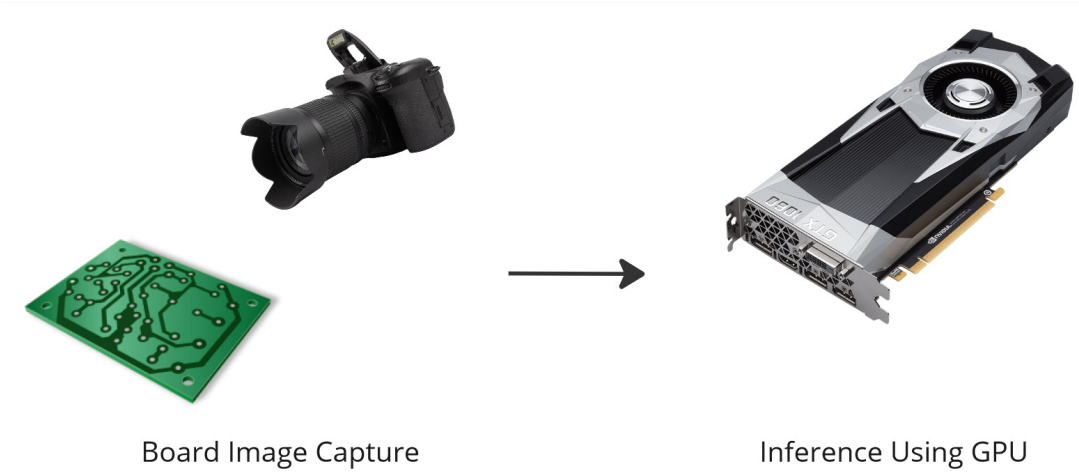
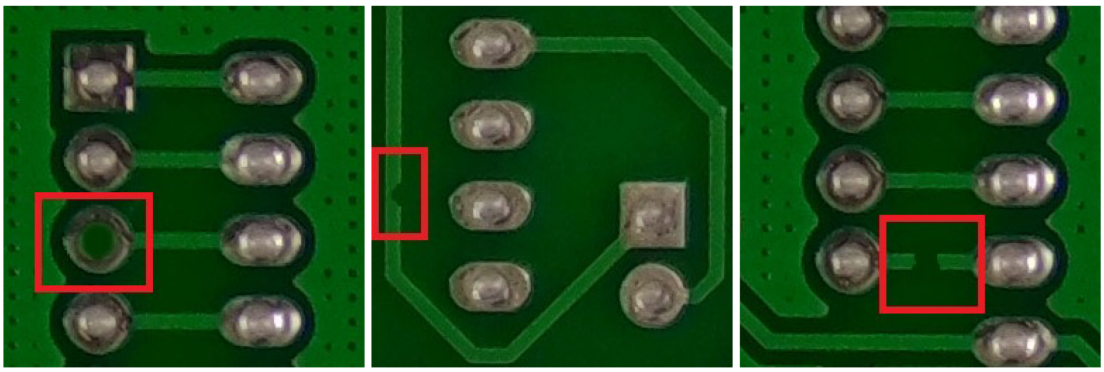
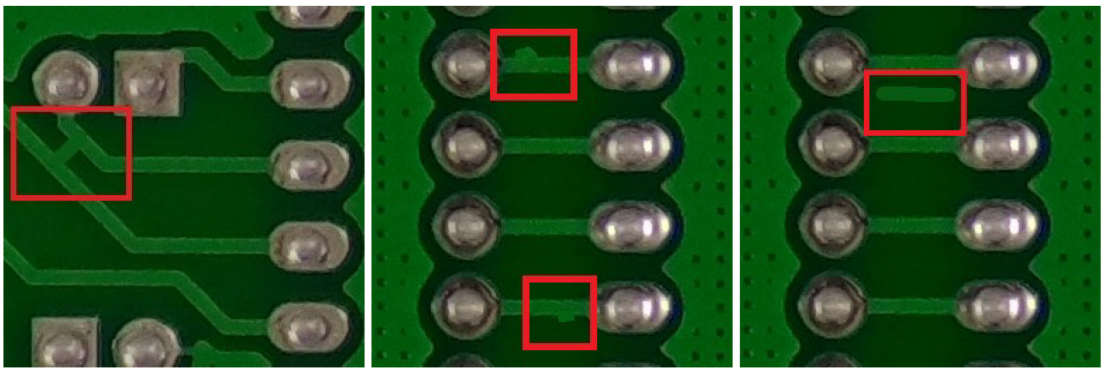


Figure 2.2: Deep Learning Inspection Flow



(a) Missing hole (b) Mouse bite (c) Open circuit



(d) Short (e) Spur (f) Spurious copper

Figure 2.3: Surface Level Board Defects. Courtesy of YOLO-RFF: An Industrial Defect Detection Method, 2022

2.1 Missing Hole

- This defect is related to the 'pad' of the PCB. Pads are the locations on the board at which the components are soldered.

- Due to manufacturing defect or any other design error, a hole may not be drilled on the pad location where it is supposed to be drilled.
- When this happens, there is a very noticeable anomaly when compared with the other parts of the board.
- Missing holes may serve as a significant obstacle to the performance of the electrical device it is fitted in, as it disrupts electrical pathways because of which the board may not function in the intended way.

2.2 Mouse Bite

- This is a subtractive trace defect, which happens in a situation where the volume of the solder joint is less than what is required.
- The defect has a variety of possible causes which include insufficient solder paste application and improper soldering technique.
- Mouse bites are also referred to as mouse nibbles, and can be identified by the occurrence of small, circular or oval shaped holes on the trace of the PCB.
- Not only does this serve as a hindrance to the performance of the board, but also negatively effects the structural strength of it.

2.3 Open Circuit

- Open circuit too is a subtractive trace defect, and occurs when there is a discontinuity between two points on a trace.
- Manufacturing errors, improper soldering or external damage on the board are the main causers of this defect.
- The presence of such a defect will break the internal circuits of the board, thus making it defective.

2.4 Short

- The Short defect is an additive trace defect. It is termed so because as the name suggests, it is additive to the surface of the conductor.
- Presence of excess solder at the solder points due to a faulty soldering technique or a design flaw.
- The excess solder acts as a short circuit, as a result of which the internal circuitry of the board is damaged.

- The non-faulty solders present on the board are of uniform size, so it is convenient to train a model to identify an inconsistently sized solder.

2.5 Spur

- Another commonly occurring additive trace defect is the Spur defect. This defect comes about during the plating and etching process of the PCB, which are processes used for the cleaning and maintenance of the board.
- Spurs refer to unwanted extensions of copper on the surface of the board. It is these extensions that interfere with other nearby circuits and thus cause malfunction.
- The copper extensions are very diminutive and are hard to view from the naked eye, which is why a deep learning model is viable to detect these minute defects.

2.6 Spurious Copper

- Spurious copper is an additive trace defect that is caused from unwanted copper deposits on the surface of the board.
- Like in the case of spur, the deposits interfere with the circuitry of the board which in turn makes it prone to malfunction.
- This defect occurs on a region on the board isolated from the trace, that makes it the contrasting factor from the Spur defect, which acts as an extension to the trace.

CHAPTER 3

METHODOLOGY

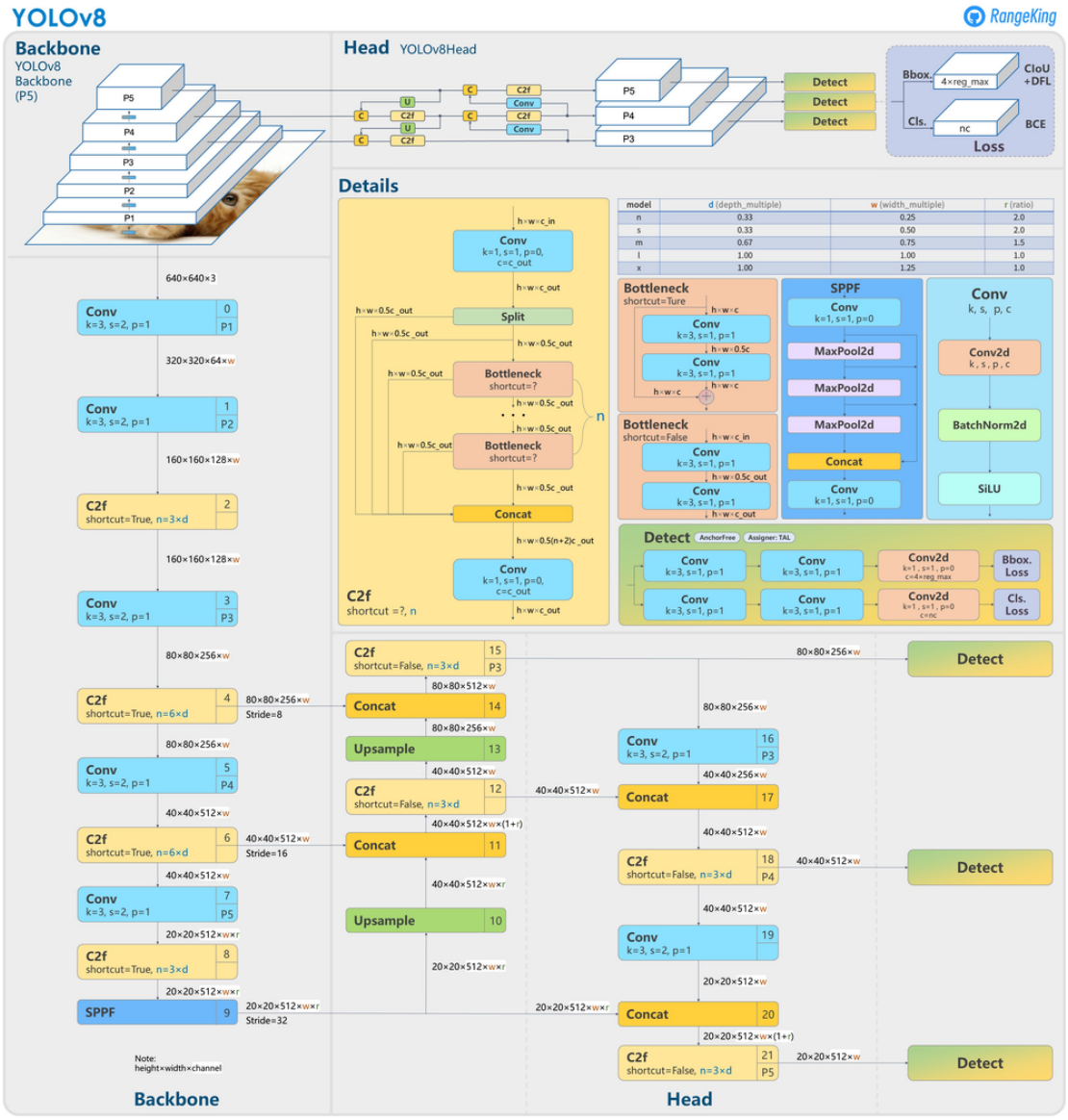


Figure 3.1: YOLOv8 Model Architecture. Courtesy of Ultralytics repository on Github https://github.com/ultralytics/ultralytics/issues/189

In order to accomplish the objective of developing a deep learning model to detect and classify defects on a PCB board, a state-of-the-art model is required in order to provide accurate and consistent predictions. To meet these requirements, the YOLOv8 object detection model is suitable. This model is the latest version of the YOLO series of

models, developed by Ultralytics. What makes using YOLO a good choice is the fact that it is able to produce a high prediction accuracy while maintaining a small model weight size. It also has the capability of being trained on a single GPU, which makes it more accessible and a less expensive affair for deployment.

The YOLO model is given preference over other object detection models primarily due to its comparatively better performance [10] and the extensive support received from a large and active community.

The architecture of the YOLOv8 model as seen in Figure 4.1 can be subdivided into three primary blocks - the backbone, the neck and the head. [11]

3.1 Primary Blocks

3.1.1 Backbone

The function of this block is to extract meaningful features from the input image. In addition to this, it also obtains the features of the image at different scales using the *conv* structure.

3.1.2 Neck

The neck acts as an intermediary between the backbone and the head blocks. It essentially collects feature maps obtained in the backbone. These feature maps are a result of repeated convolutions, with diminishing dimensions as more convolutions are applied. It helps make the model more robust to features by concatenating feature maps of different scales. Moreover, it helps reduce the computational load by reducing dimensionality, therefore increasing the speed of training.

3.1.3 Head

This is the final part of the YOLO architecture, where the network's outputs are obtained. A YOLO prediction consists of two parts : Bounding Box and Class Confidence.

Through their respective loss functions, adjustments are made to the convolution kernel for the forthcoming epoch depending on the amount of loss in prediction, or in other words, depending on how correct/wrong the prediction is.

3.2 Computational Blocks

3.2.1 conv

Convolution is the mathematical method of combining two functions to produce a third. In the case of image processing, convolutions are used to extract features from an image. The parameters used in convolution are kernels (k), strides (s) and padding (p). [12]

- *Kernels* : Also known as feature detectors, kernels are usually two dimensional arrays whose values are updated during the training process in order to obtain minimal loss. The kernel moves across the image and performs the dot operation between the input and the values of the kernel to produce on output termed as the feature map.

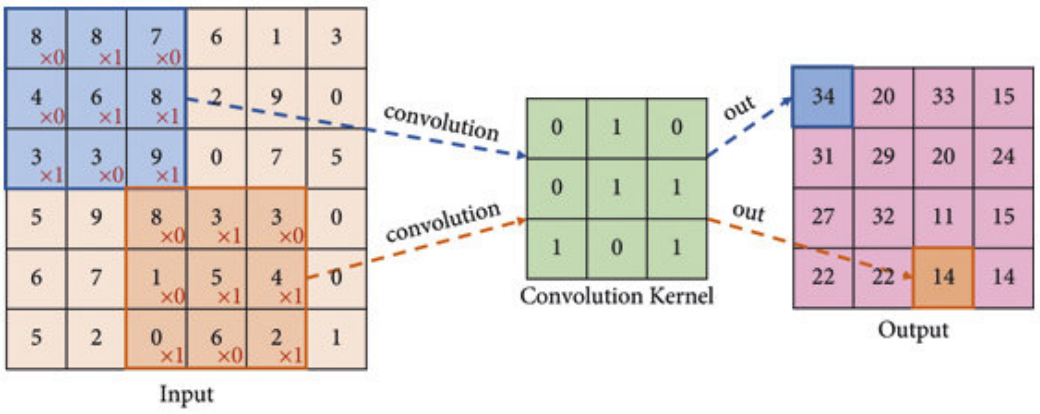


Figure 3.2: A 6 x 6 image convolved with a 3 x 3 kernel. Courtesy of Functionality-Based Processing-In-Memory Accelerator for Deep Convolutional Neural Networks - https://www.researchgate.net/figure/Convolution-operation_fig2_355656417

- *Stride* : The stride defines the displacement distance of the kernel moving across the input. There is an inverse relation between the output dimensionality and the stride value. Larger strides are used in order to avoid heavy computational costs.

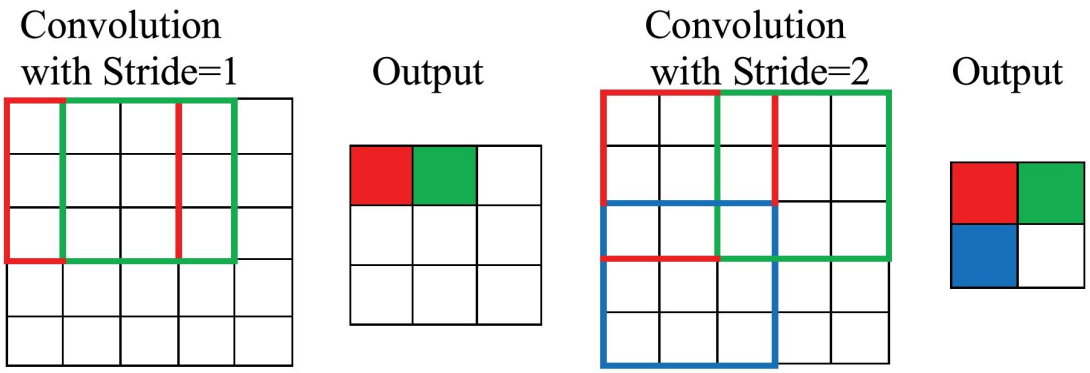


Figure 3.3: Representation of variation of output depending on stride value. Courtesy of Functionality-Based Processing-In-Memory Accelerator for Deep Convolutional Neural Networks - https://www.researchgate.net/figure/Convolution-operation_fig2_355656417

- *Padding* : Padding refers to the inclusion of additional pixels along the outermost boundary of the image. This helps avoiding loss of information along the edges of the image as it helps even out the number of neighbours of a pixel at the edge with that of a pixel embedded in the center of the image.

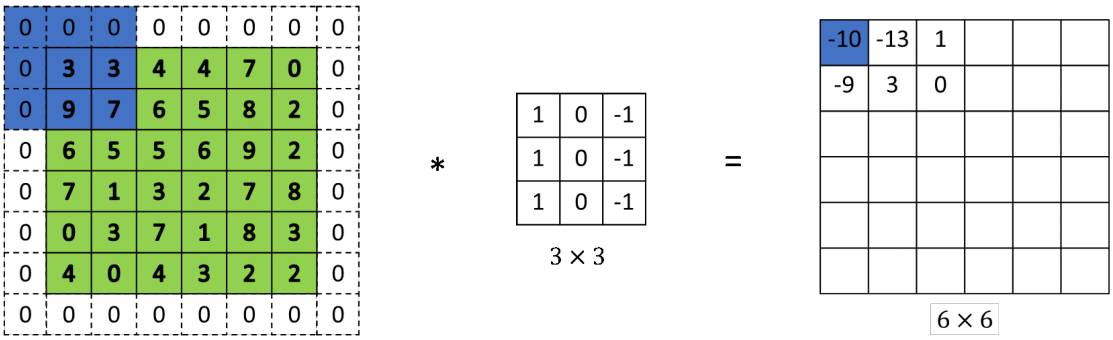


Figure 3.4: A zero-padded image convolved with a 3 x 3 kernel. Courtesy of What is Padding in Convolutional Neural Networks - https://ayeshmanthaperera.medium.com/what-is-padding-in-cnns-71b21fb0dd7

Mathematically, convolution of two functions X and Y having probability distributions $p_X(x)$ and $p_Y(y)$ respectively can be represented as such :

$$(X \circledast Y)(z) = \int_{-\infty}^{\infty} p_X(x) \,.\, p_Y(z-x)\, dx \tag{3.1}$$

After carrying out convolution, batch normalization is done through the **BatchNorm2d** block. This procedure involves finding out the mean of every feature map,

subtract by mean and then divide by standard deviation.

For the final step, the Sigmoid-weighted Linear Unit (siLU) activation function is applied. The activation function for arbitrary input x is defined as

$$SiLU(x) = x \,.\, \sigma(x) \tag{3.2}$$

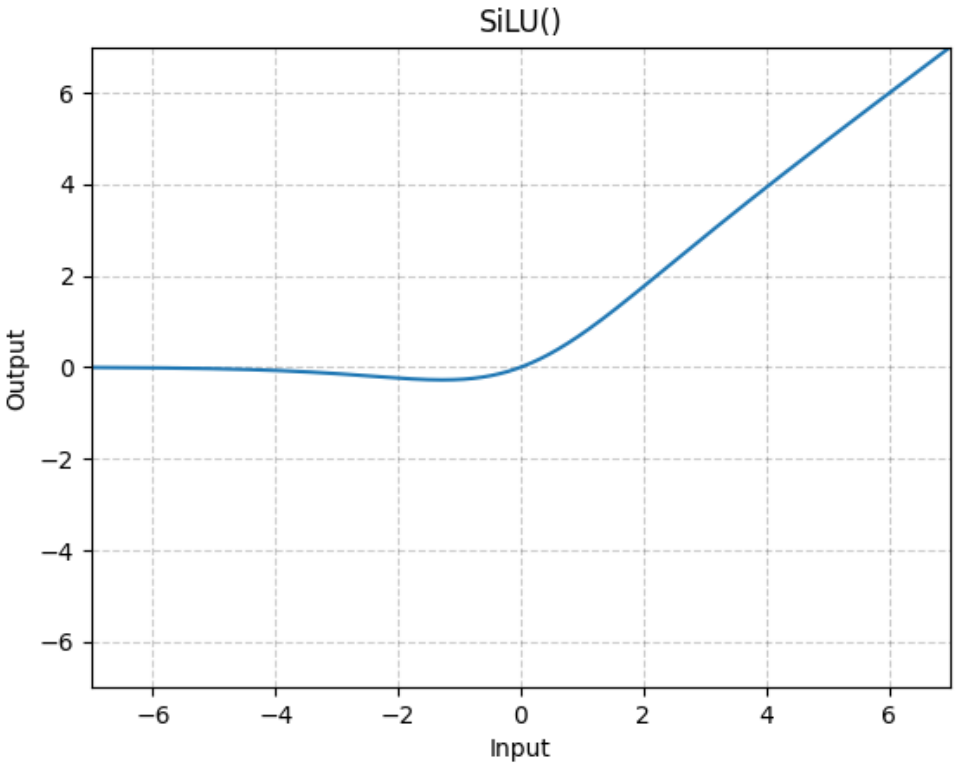


Figure 3.5: Graphical representation of the SiLU activation function

Where the sigmoid function (σ) is :

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{3.3}$$

3.2.2 C2f

- The C2 module in YOLOv8 stands for CSP (Cross Stage Partial) Bottleneck with two convolutions.
- A **Bottleneck layer** or Bottleneck block converts its input into a compressed form of representation.

- These specific blocks carry out dimensional reduction which in turn reduces the amount of computational requirement.
- The lower dimensional representation of the input data also helps get rid of any unwanted features.
- The efficiency brought about with the aid of the bottleneck layer makes it a faster implementation of the C2 module while maintaining similar performance, hence the additional 'f' denoting fast.

3.2.3 Detect

This block carries out object detection of the low, medium and high level features of the image obtained from the neck. From this block, the two results obtained are : bounding box coordinates and classifications for every predicted bounding box. These have their own respective loss functions which have been described below [9]

- *Bounding Box* : The algorithm used to find out loss is Complete Intersection over Union (CIoU). Loss function for CIoU is calculated as such

$$L_{CIoU} = 1 - IoU + \frac{\rho^2(b, b^{b^t})}{c^2} + \alpha v \tag{3.4}$$

$$\alpha = \frac{v}{(1 - IoU) + v} \tag{3.5}$$

$$v = \frac{4}{\pi^2}\left(arctan\frac{w^{gt}}{h^{gt}} - arctan\frac{w}{h}\right)^2 \tag{3.6}$$

where L_{CIoU} is the CIoU Loss, α is the parameter that controls the weight of the aspect ratio factor (v)

w, h and w^{gt}, h^{gt} are the side length and width of the prediction box and the ground truth box respectively. c is the diagonal formed by the furthest boundaries of the prediction and true box.

b and b^{b^t} are the centroid locations of the prediction box and the true box respectively, ρ is a function that calculates Euclidean distance between two points

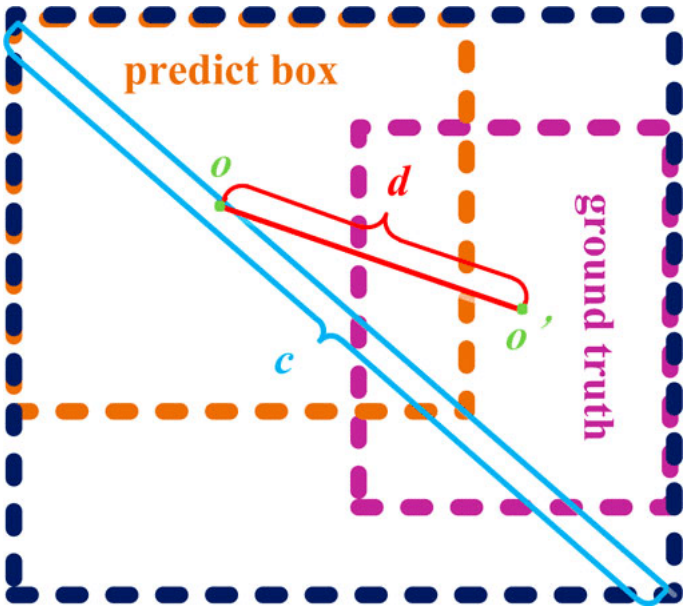


Figure 3.6: c is the length of the smallest diagonal covering the predict box and the ground truth box, and $d = \rho(o, o')$ is the euclidean distance between the centroids of the two boxes. Courtesy of Bounding box coordinate prediction with YOLO - https://www.researchgate.net/figure/Bounding-box-coordinate-prediction-with-YOLO_fig5_349664815

IoU is calculated using the formula

$$IoU(box_t, box_p) = \frac{box_t \cap box_p}{box_t \cup box_p} \tag{3.7}$$

where box_t is the true bounding box and box_p is the predicted bounding box.

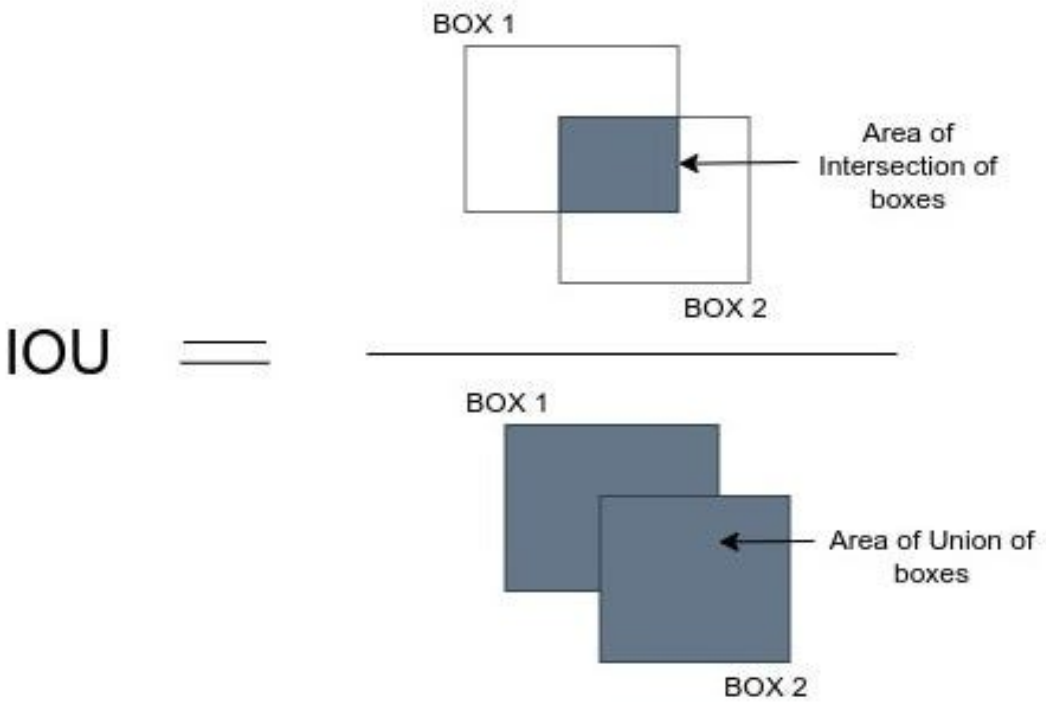


Figure 3.7: A graphical representation of Equation (3.7). Courtesy of Intersection over Union (IoU): Definition, Calculation, Code - https://www.v7labs.com/blog/intersection-over-union-guide

- *Classification* : The loss in this case is measured using BCE - Binary Cross Entropy. The loss function for BCE is also referred to as $logloss$. The classification problem of all categories is reduced to whether or not it belongs to said category. This way there is mutual exclusivity of classes.

$$logloss = \frac{-1}{N}\sum_{i=1}^{N} y_i \,.\, log(p_i) + (1 - y_i) \,.\, log(1 - p_i) \tag{3.8}$$

where N is the number of predicted probabilities, y_i is the binary indicator (1 if it belongs to class 1, 0 otherwise), p_i is the probability of class 1, $(1 - p_i)$ is the probability of class 0

CHAPTER 4

USER INTERFACE

4.1 Dependencies

The user interface designed to interact with the model and to carry out defect inference on a board image is implemented using the **Tkinter** library. The **ultralytics** package is also required to load the trained model and carry out inference. In addition, **OpenCV** is used to read and draw bounding boxes over the image. **CustomTkinter** is a visual overhaul of the Tkinter library which provides a more modern styling of customizable widgets, which constitutes most of the UI.

4.2 Solution Application

- Initially, the user is provided with two choices: to upload an image to be inspected upon, and to begin the inspection process.
- Once the inference is triggered, the image is fed to the trained model which in turn returns the bounding box and class predictions.
- The defect classes are color-coded, and their individual counts are displayed on a sidebar that pops out when the inference is complete. For every prediction, a label indicating the confidence is printed along with the bounding box.
- A threshold dial at the bottom of the sidebar can be used to configure the minimum limit of the prediction confidence. Any prediction with a confidence value lesser than that of the threshold will not appear. This can be used to filter out unnecessary or incorrect predictions.

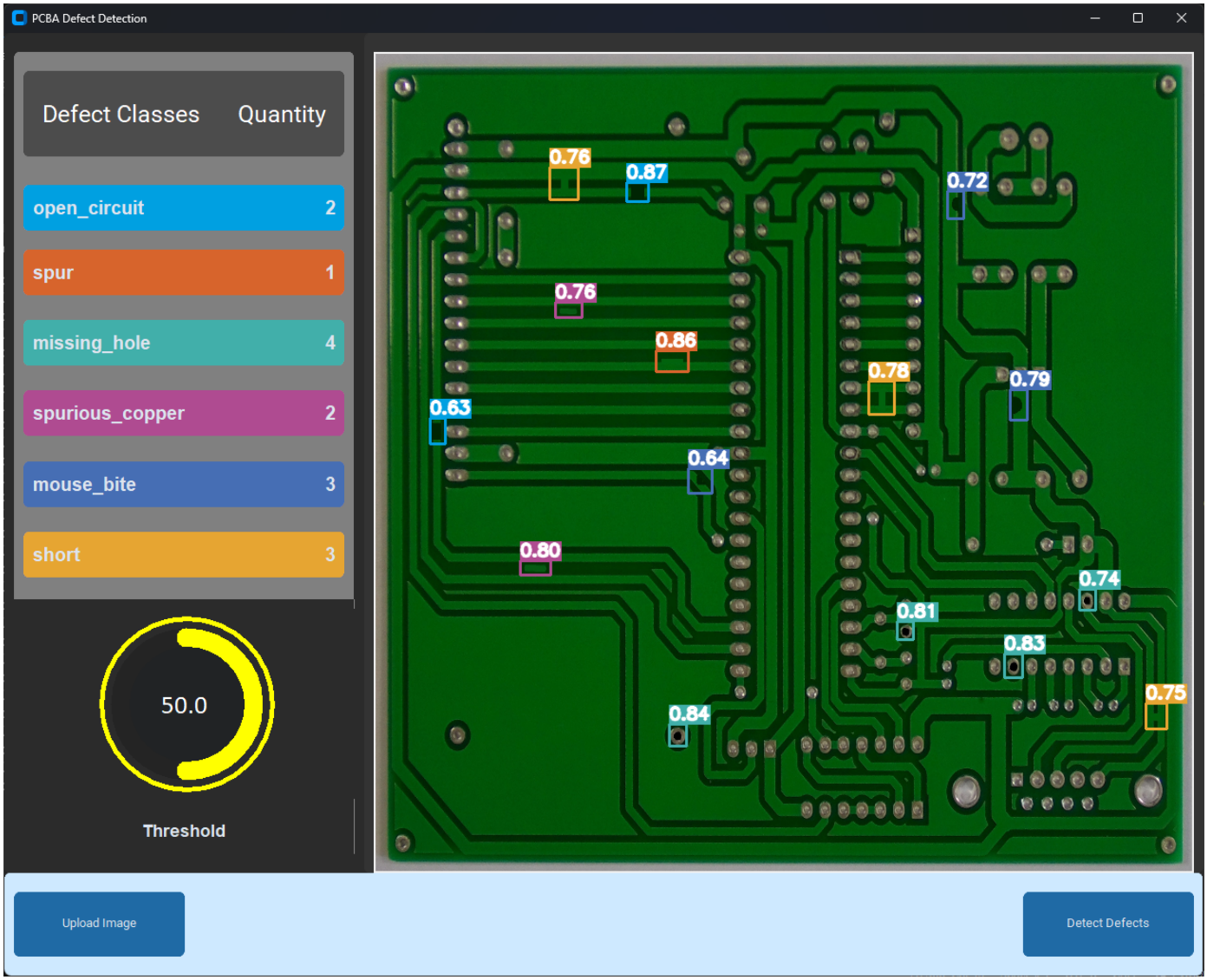


Figure 4.1: Output representation after inference

CHAPTER 5

RESULTS

5.1 Dataset and Augmentation

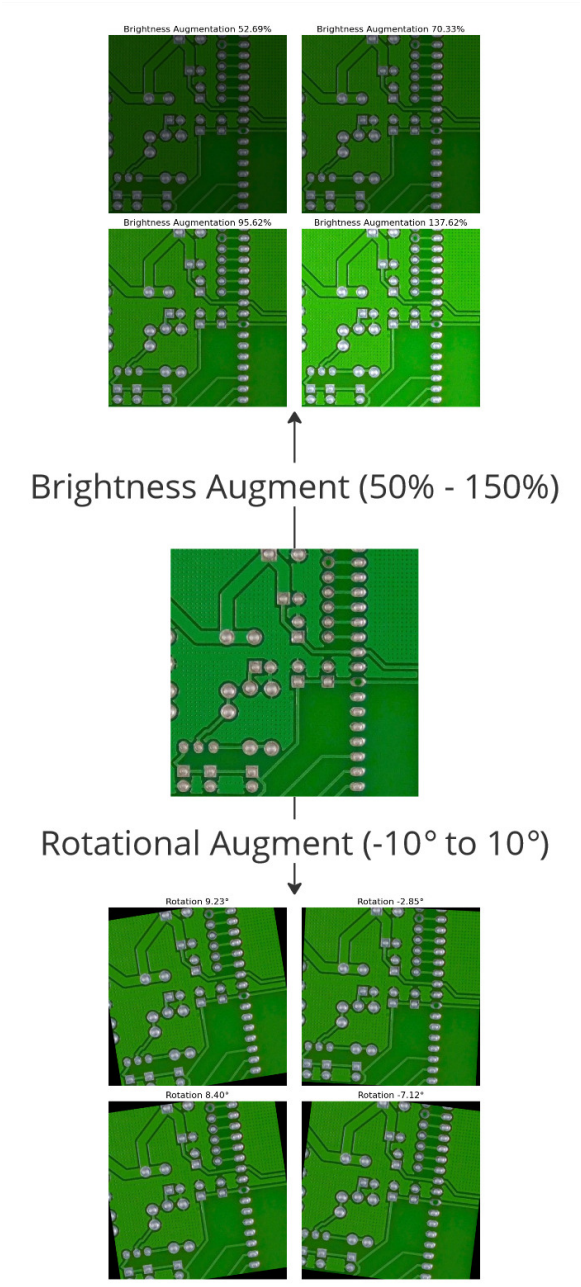


Figure 5.1: Illustration of the augmentations applied to the board image

The original dataset was obtained from the HRIPCB dataset [13]. This dataset consists of images of 600 x 600 dimension. As a whole, it has 693 images with the six defects previously discussed : Missing Hole, Mouse Bite, Open Circuit, Short, Spur and Spurious Copper. Due to the limited size of the dataset, there is a risk of problems such as low prediction accuracy, overfitting and low robustness to new data. This issue of insufficient training samples can be rectified with the help of data augmentation. Data augmentation refers to the increasing of the size of a dataset by applying modifications to the original dataset, thus creating new data for the model to train on. For this particular scenario, rotational and brightness augmentations are used in order to make the data more conditioned to varying environmental conditions. The window of rotation is between -10°(Anti-clockwise) and +10°(Clockwise). The range of brightness augmentation is between 50% and 150%. These augmentations are applied to randomly selected samples in the dataset. Through this method, the size of the dataset was increased to 10,668 where the ratio of split of training data, validation data and testing data is (0.7, 0.2, 0.1). The number of images per defect class is provided in Table 5.1. Refer to Table 5.2 for the comparison between the original dataset and the augmented dataset.

Table 5.1: All defect classes and their frequencies

Defect Class	**Quantity**	**Number of Defects**
Missing Hole	1832	3612
Mouse Bite	1852	3684
Open Circuit	1740	3548
Short	1732	3508
Spur	1752	3636
Spurious Copper	1760	3676

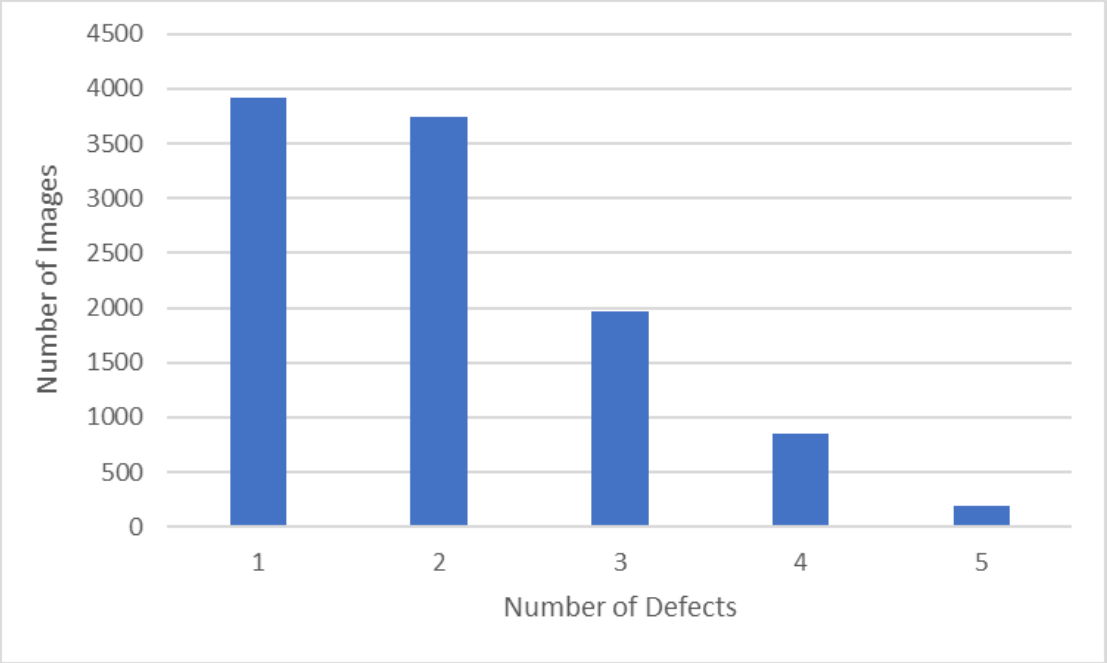


Figure 5.2: Distribution of number of defects on the images of the dataset

Table 5.2: Comparison of frequencies of original dataset and augmented dataset

	Original Dataset	**Augmented Dataset**
Number of Images	693	10688
Number of Defects	2953	21664

5.2 Model Evaluation Metrics

There are a total of five metrics utilized by the YOLO model to give the user an idea of the performance of the model [14] :

5.2.1 Recall

Recall is the measure of how correctly the model is able to predict the positive class. In other words, it is an indication of how capable the model is in identifying true positives. With respect to this scenario, it is the models ability to identify all defects in a given image.

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

TP is the number of true positives and FN is the number of false negatives in the prediction.

5.2.2 Precision

Precision refers to the accuracy of the positive predictions made by the model. It assesses whether or not all of the defect detections are in fact actually defects.

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

FP indicates the number of false positives.

5.2.3 F1 Score

F1 score combines the Precision and Recall metrics by using their harmonic mean. Maximizing F1 score would imply the maximization of Precision and Recall, so a high F1 score indicates good prediction accuracy of the model.

$$F1 = 2 \,.\, \frac{Precision \,.\, Recall}{Precision + Recall}$$

5.2.4 AP

AP, which stands for Average Precision is defined as the area under the Precision-Recall curve (also called PR Curve). Mathematically, AP for a given precision-recall curve $p(t)$ can be represented as

$$AP = \int_0^1 p(t).dt$$

5.2.5 mAP

mAP stands for mean average precision, and is obtained by calculating the mean of the AP scores of all the concerned classes. It is a good metric to get the overall assessment of the accuracy of the model for each class. For n classes, the mAP score is

$$mAP = \frac{1}{n}\sum_{i=1}^{n} AP_i$$

A popular notation of the mAP used is mAP**X**, where *X* indicates the IoU threshold required for correct prediction. For example, mAP95 is the accuracy value when the IoU is 95%. This means that there should be a 95% overlap between the predicted bounding box and the ground truth bounding box for the detection to be considered successful.

5.3 Comparison of Model Architectures

Each of the following models were trained on the same dataset with the following system specifications :

Table 5.3: Configuration of the device used to train the model

Specification	Size/Version
Operating System	Windows 11 Home
CPU	Intel Core i7-10750H
RAM	16GB
GPU Memory	4GB
Graphics	NVIDIA GeForce GTX 1650 Ti
CUDA	12.4
Python	3.12.1
Torch	2.2.0

5.3.1 YOLOv8

The YOLOv8 model was trained using the yolov8s pretrained weights and was trained for 100 epochs. The loss and accuracy curves of the trained model can be referred to in

Figure 5.3 and Figure 5.4

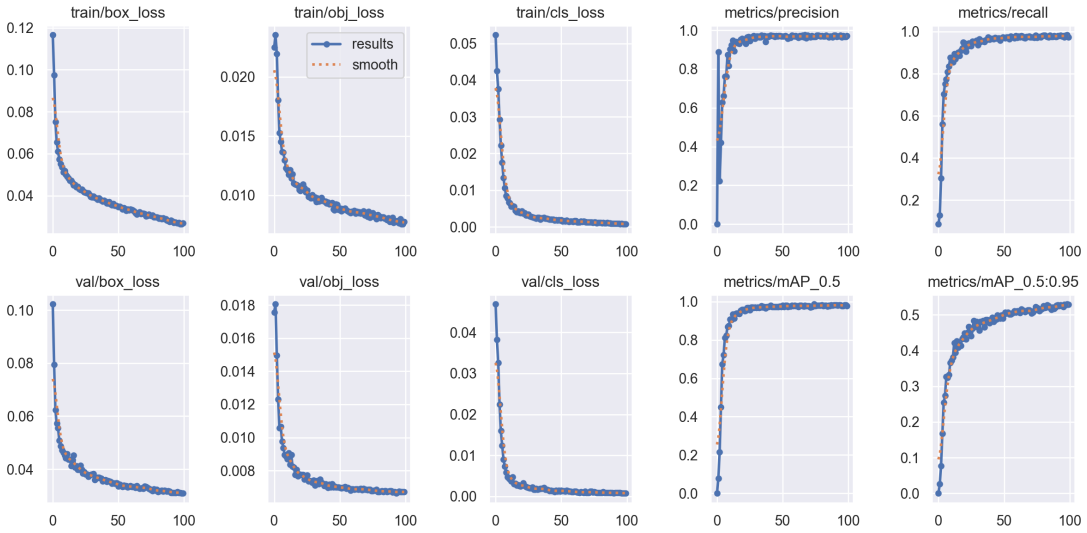


Figure 5.3: YOLOv8 training results

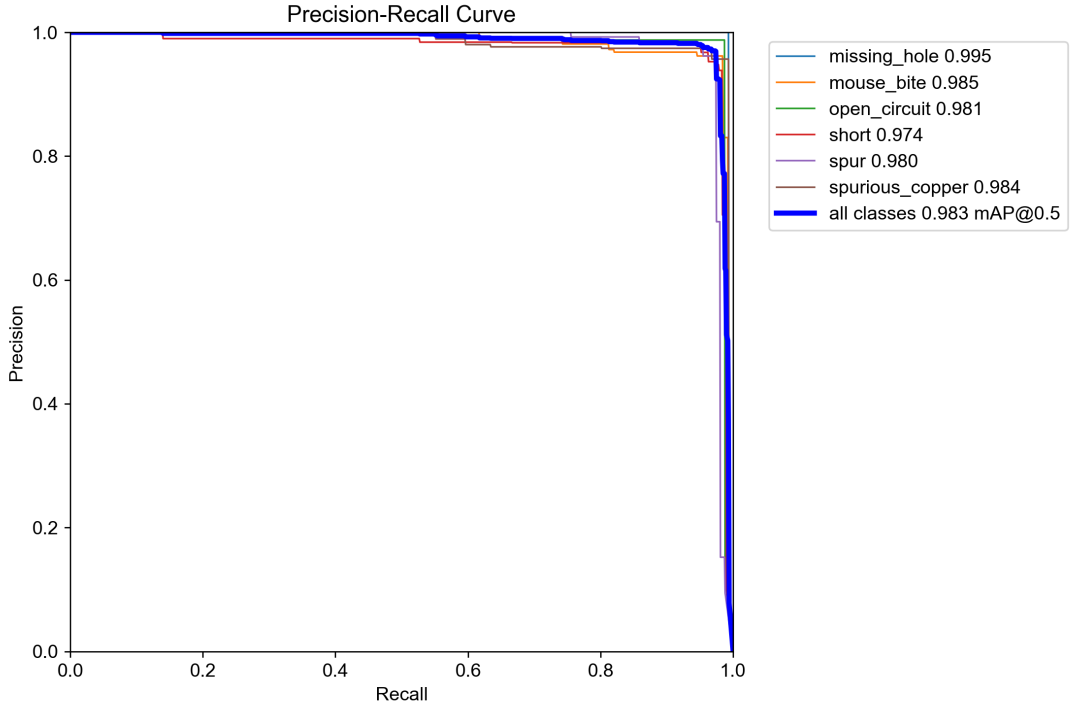


Figure 5.4: PR Curve for the YOLOv8 model

5.3.2 YOLOv5

Similar to YOLOv8, this model was trained on the small version of the pretrained weights (yolov5s) for the same 100 epochs. The performance of this model is slightly

worser than its successor as indicated by Figure 5.5 and Figure 5.6

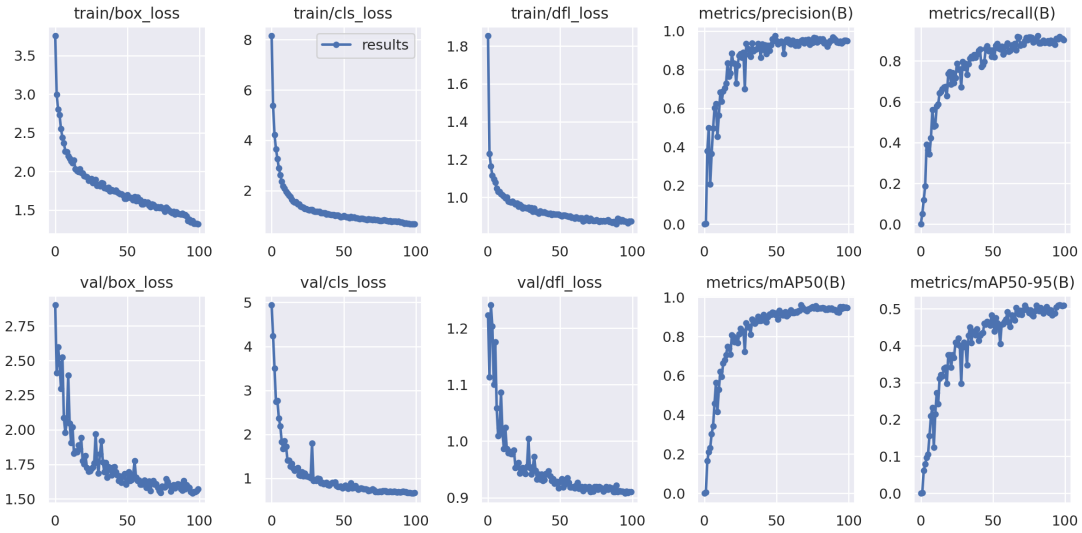


Figure 5.5: YOLOv5 training results

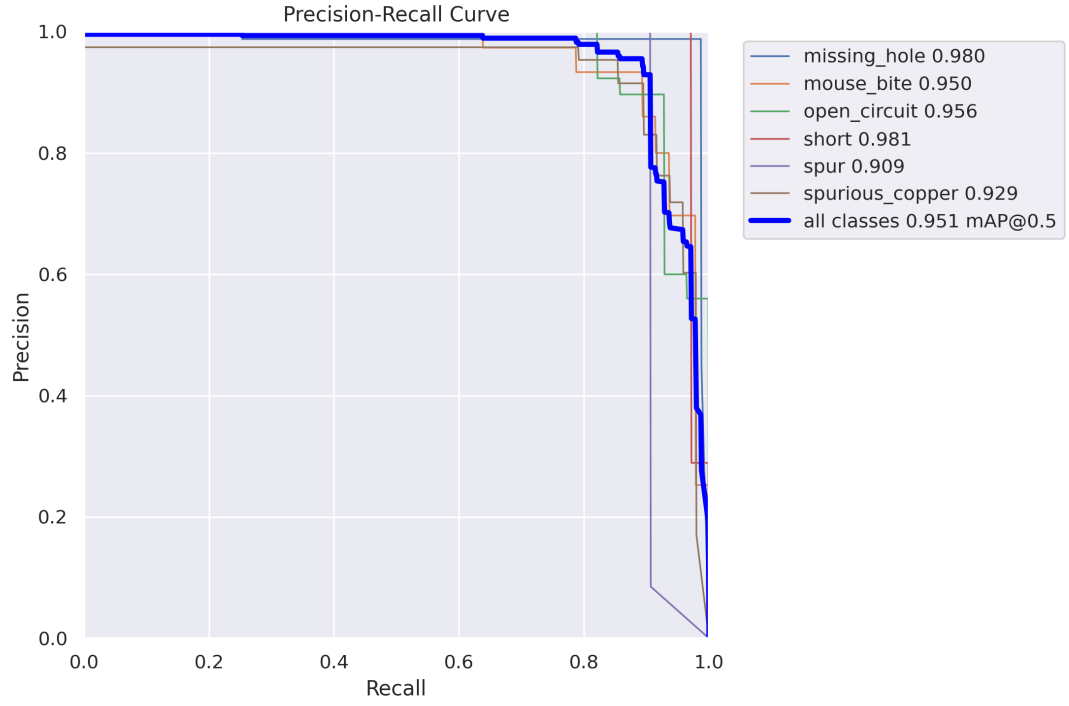


Figure 5.6: PR Curve for the YOLOv5 model

5.3.3 Detectron2

Detectron2 is an object detection library developed by Facebook AI. It is regarded as the successor Detectron and Mask RCNN. It has been trained for 10000 epochs on a

faster R-CNN based object detection model. The loss curve for the trained model can be referred to in Figure 5.7. The improvement in classification accuracy can be noted in Figure 5.8

Figure 5.7: Total Loss Curve for Detectron2 Model

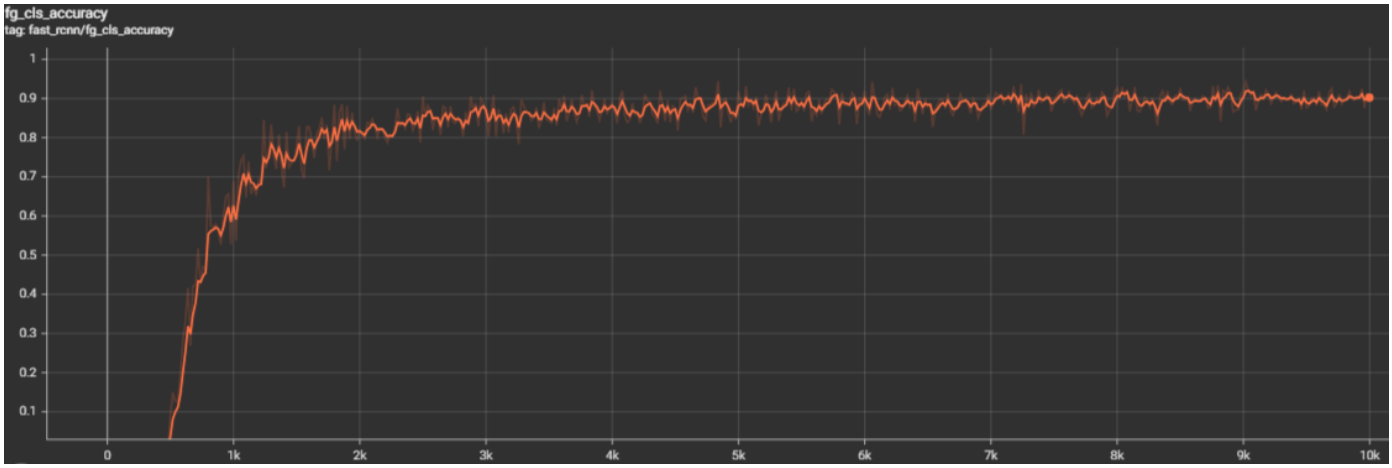


Figure 5.8: Classification Accuracy Curve for Detectron2 Model

5.3.4 EfficientDet

EfficientDet utilizes an optimized version of the EfficientNet architecture and a weighted bi-directional Feature Pyramid Network (BiFPN) which enables faster multi-scale feature fusion. This tensorflow based model was trained for 100 epochs. The training and validation curves for the trained model may be observed in Figure **??**

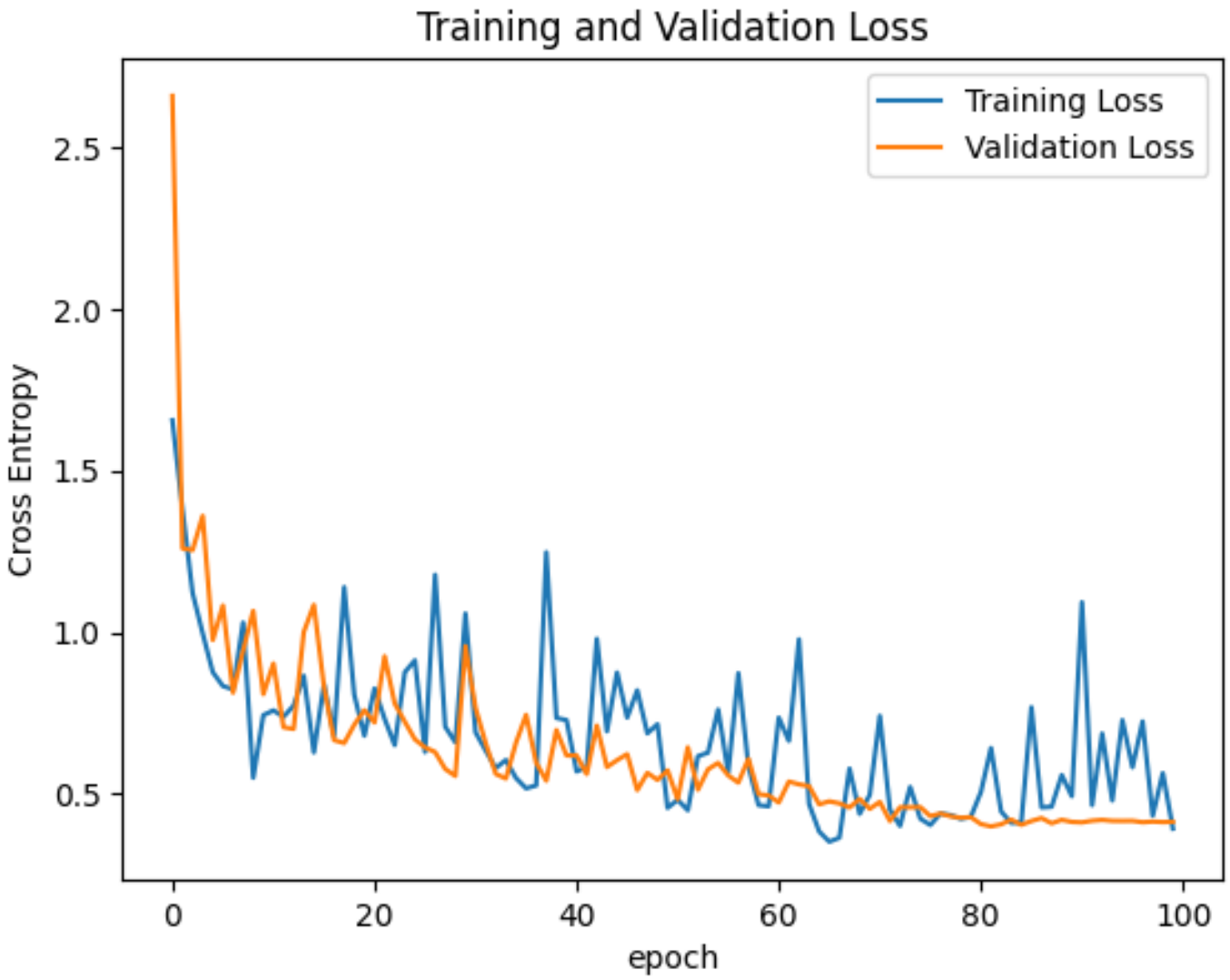


Figure 5.9: Validation and Training Loss Curves for Faster R-CNN Model

5.3.5 Faster R-CNN

A Faster R-CNN architecture developed on the torchvision pretrained weights is trained for 100 epochs to obtain the loss curves in Figure 5.10

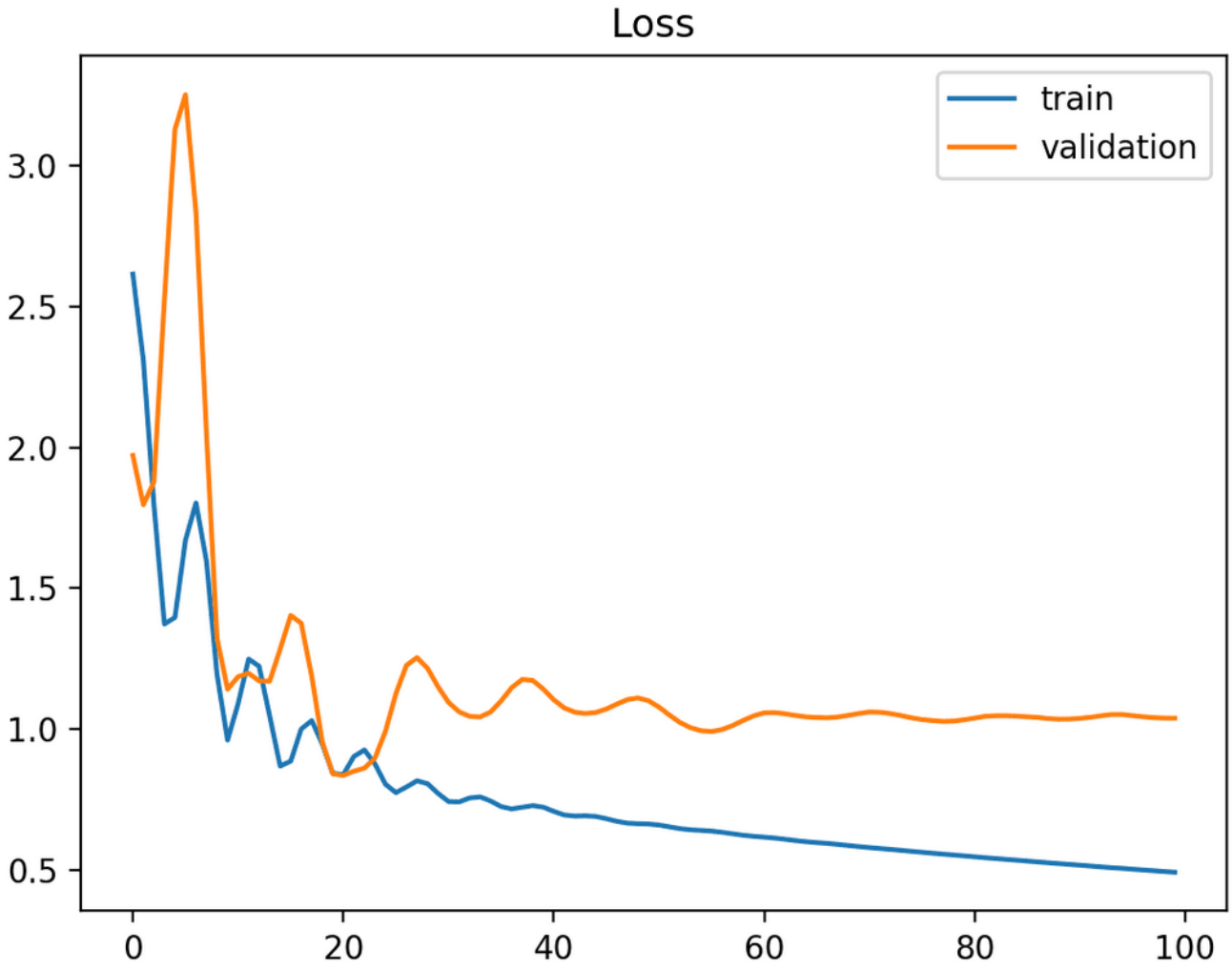


Figure 5.10: Validation and Training Loss Curves for Faster R-CNN Model

5.4 Comparison Summary

On comparing the preciseness of the models, it can be concluded that the model trained on the YOLOv8 architecture performs the best. A detailed description of the model performances is listed in Table 5.4. The total loss is calculated by the finding out of the summation of bounding box loss and classification loss.

Table 5.4: Comparison of Model Training Metrics

Architecture	mAP	Precision	Recall	Total Loss
YOLOv8	**0.986**	**0.97**	**0.977**	2.412
YOLOv5	0.951	0.944	0.948	4.85
EfficientDet	0.924	0.911	0.908	1.313
Detectron2	0.902	0.899	0.9	**0.322**
Base FR-CNN	0.851	0.843	0.839	0.497

The anomaly in the relatively high loss value despite higher accuracy can be better explained by the fact that the loss is not bounded in any manner, which means that a single 'very wrong' prediction can make a significant difference to the loss value. This implies that the total loss quantity is not entirely indicative of the performance of the respective model architecture.

5.5 Training Analysis

Depending on the number of trainable parameters, YOLOv8 pre-trained weights come in varying sizes - yolov8n (Nano), yolov8s (Small), yolov8m (Medium), yolov8l (Large) and yolov8x (Extra Large). To balance out the computational cost and model performance, the yolov8s weight is considered most suitable for this use case. The hyper parameters set for training are as follows :

- Image size (imgsz) : 640 x 640
- Epochs : 100
- Patience (For early stopping) : 15, with validation loss as the monitoring function
- Batch size : 16
- Learning Rate : 0.001

Since there was no point during the training where the validation loss did not improve for the patience value (15 epochs), thus the model was trained for all 100 epochs.

From the loss plots in Figure 5.4 and Figure 5.3, it can be inferred that there is no underfitting or overfitting. If the model was underfitted, the loss values would have a loss value significantly greater than zero even after 100 epochs. On the other hand, if the model was overfitted, there would be a point where the validation loss spikes back up.

The metric curves indicate high accuracy of the model, as the final value is above 0.9. This implies that the overall prediction accuracy is above 90%

Table 5.5: Estimated duration of different stages of prediction

Procedure	Time taken (ms)
Model Loading	57.8
Preprocessing	11.3
Inference	163.9
Postprocessing	15.1

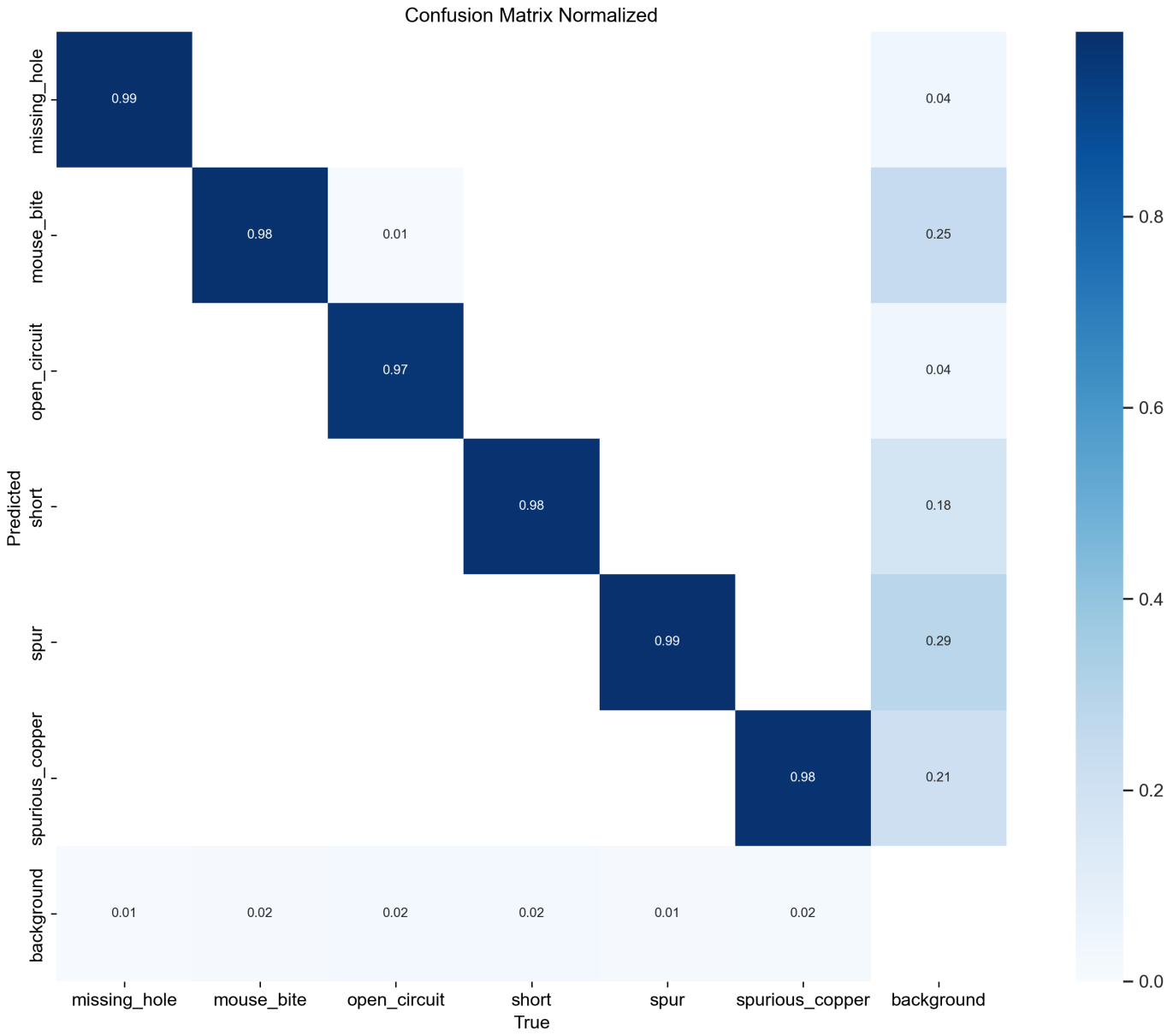


Figure 5.11: Confusion matrix for the defect classes that the model was trained on

Ideally, all the diagonal values of the confusion matrix should be 1. The current confusion matrix is very close to achieving that state, which indicates that the model is able to give the correct prediction for the available classes for nearly all cases.

5.6 Model Testing

A randomly selected image belonging to each class has been selected from the testing data and passed through the model. The results obtained by the model along with their

confidence values (with the exception of those which are going beyond the boundaries of the image) are shown below :

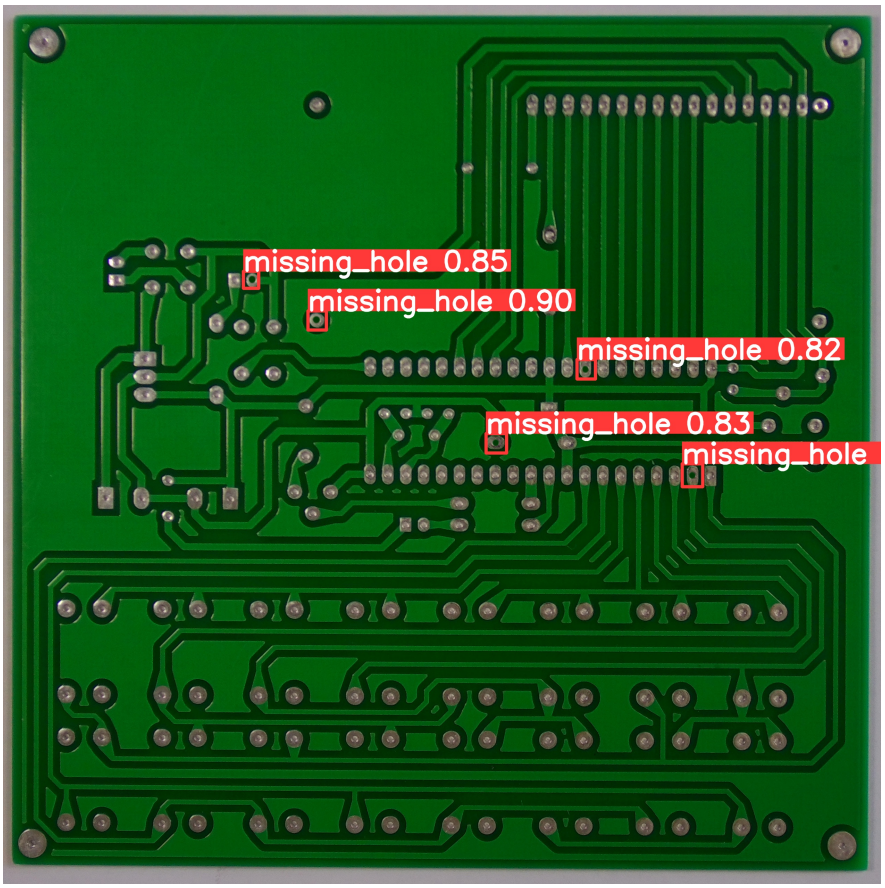


Figure 5.12: Missing hole predictions

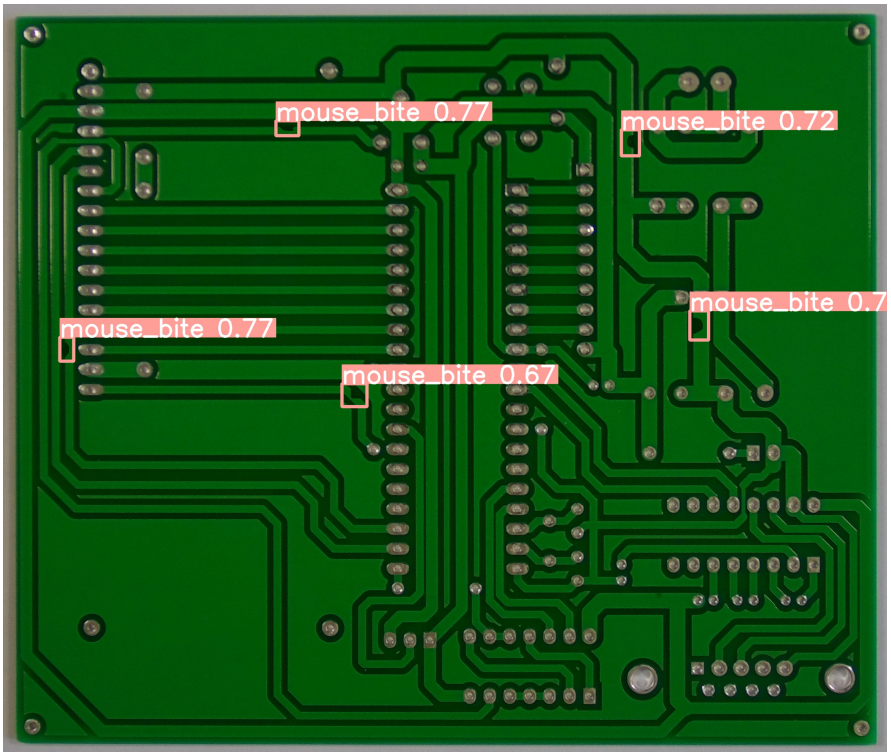


Figure 5.13: Mouse Bite predictions

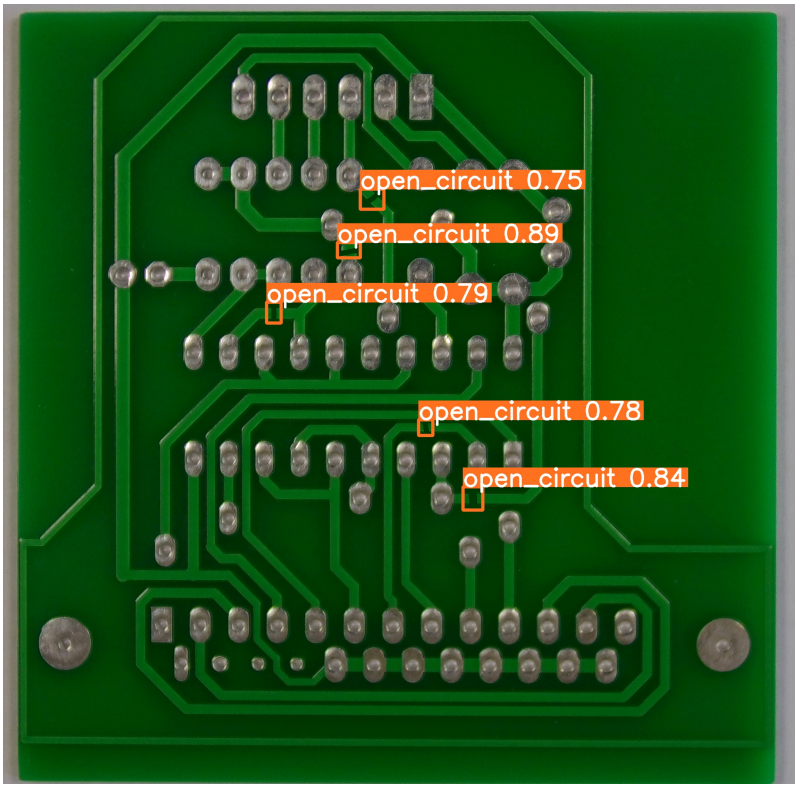


Figure 5.14: Open Circuit predictions

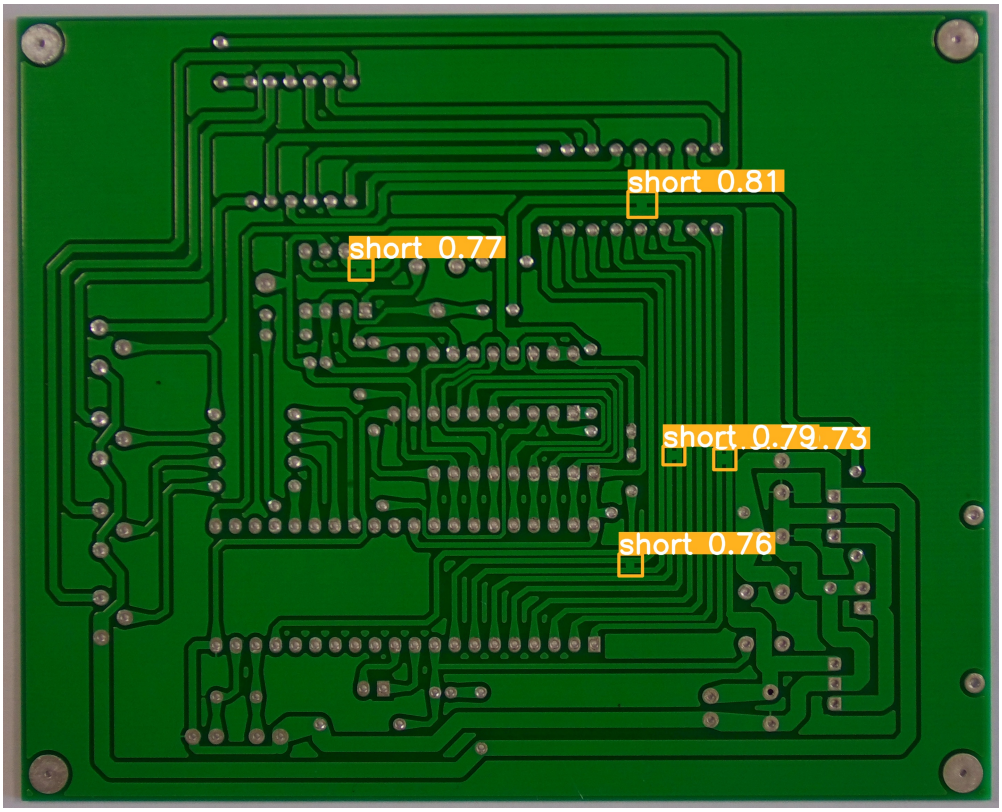


Figure 5.15: Short Circuit predictions

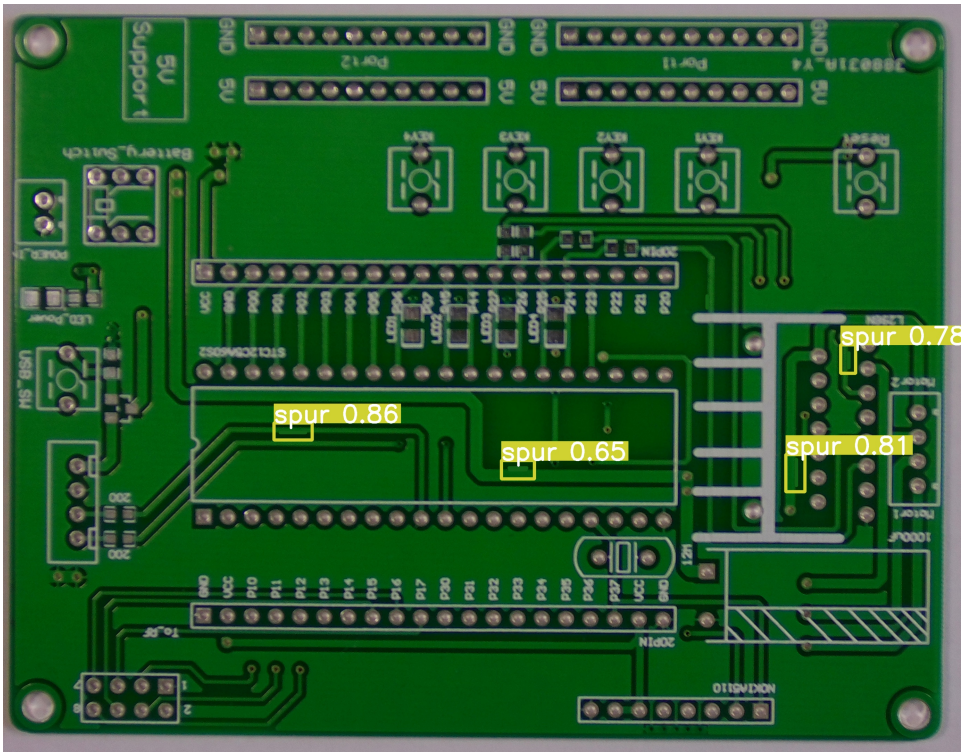


Figure 5.16: Spur predictions

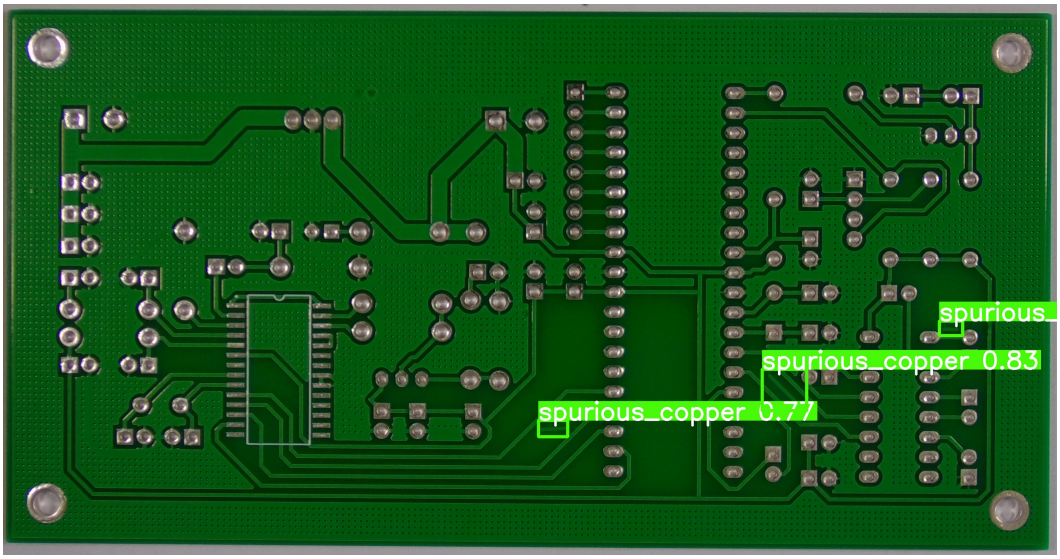


Figure 5.17: Spurious Copper predictions

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

A deep learning approach to solving the issue of board defect detection has been discussed through this paper. The most significantly occurring defects on a PCB board have been discussed about and explained in detail. A dataset has been synthesized using data augmentation with images of the six most significant defects on a PCB - Missing Hole, Mouse Bite, Open Circuit, Short, Spur and Spurious Copper. Using this dataset, an object detection model with the YOLO architecture has been trained with optimal prediction accuracy. The architecture of the model utilized has been discussed about thoroughly, with explanation of the various functions applied on the input image and its effects on the same. The loss and metric graphs obtained from training gives us an idea about how the model has adjusted its parameters to return the correct predictions. When an image of a defective board is passed through this model, it produces an image with bounding boxes around the defective regions along with their appropriate classification. With the convenience provided by the trained model, the hardships of manual inspection and the fallacies of AOI is mitigated. Surface defects on PCBs can be detected with high precision in very less time. The model is also made robust to miniature defects that may be hard to notice from other visual inspection methods. A user interface has also been developed with the help of which a clear-cut representation of the occurrence of defects is shown to the end user, making the inspection process much less of a hassle.

6.2 Future Scope

Future work in this domain can focus on the inclusion of other defects that may occur on a PCB, while making sure that the performance of the model in classifying other defects

is not affected. As of now, the dataset used to train the model contains images of defects on only one specific kind of board. A greater variety of data, especially data from a wide range of boards will enable the model to carry out correct defect detection without any major environment or board dependency. A superior model architecture would help reduce the time and computational power required to train a working instance of a model, as well as bring down the time taken to carry out inference of an input image. Further research on quick identification of defects on a PCB and singling out the source of the defect is of key importance in the electronic board manufacturing industry, especially because of how delicate the process of manufacturing of these boards are. Minimizing the occurrence of these defects would go a long way in saving the costs and raw materials used in the production of the boards. In addition, a UI overhaul to the current interface can help improve the robustness of the developed solution application, such as manual training of data on a custom dataset, an annotation window and zooming in and out of the inferred image.

REFERENCES

[1] M. Moganti, F. Erçal, C. H. Dagli, and S. Tsunekawa, "Automatic pcb inspection algorithms: A survey," *Comput. Vis. Image Underst.*, vol. 63, pp. 287–313, 1996. [Online]. Available: https://api.semanticscholar.org/CorpusID:8540351

[2] R. Mirchandani, C. Yoon, S. Prakash, A. Khaire, A. Naran, A. Nair, and S. Ganti, "Comparing the architecture and performance of alexnet faster r-cnn and yolov4 in the multiclass classification of alzheimer brain mri scans," 2021.

[3] V. Sankar, G. Lakshmi, and Y. Sankar, "A review of various defects in pcb," *Journal of Electronic Testing*, vol. 38, pp. 481–491, 2022.

[4] R. Ding, L. Dai, G. Li, and H. Liu, "Tdd-net: a tiny defect detection network for printed circuit boards," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 2, pp. 110–116, 2019. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/trit.2019.0019

[5] D. S. Koblah, O. P. Dizon-Paradis, J. Schubeck, U. J. Botero, D. L. Woodard, and D. Forte, "A comprehensive taxonomy of visual printed circuit board defects," *Journal of Hardware and Systems Security*, vol. 7, no. 2, pp. 25–43, Sep 2023. [Online]. Available: https://doi.org/10.1007/s41635-023-00132-4

[6] N. R. S. Malge P. S., "Pcb defect detection, classification and localization using mathematical morphology and image processing tools," *International Journal of Computer Applications*, vol. 87, no. 9, pp. 40–45, February 2014. [Online]. Available: https://ijcaonline.org/archives/volume87/number9/15240-3782/

[7] I. Ibrahim, Z. Ibrahim, K. Khalil, M. Mokji, S. Abdul, R. Syed, S. A. R. Abu Bakar, N. Mokhtar, and W. Khairunizam, "An improved defect classification algorithm for six printing defects and its implementation on real printed circuit board images," *International journal of innovative computing, information control: IJICIC*, vol. 8, pp. 3239–3250, 05 2012.

[8] T. Yun, K. Sim, and H. J. Kim, "Support vector machine-based inspection of solder joints using circular illumination," *Electronics Letters*, vol. 36, pp. 949 – 951, 2000.

[9] J. Tang, S. Liu, D. Zhao, L. Tang, W. Zou, and B. Zheng, "Pcb-yolo: An improved detection algorithm of pcb surface defects based on yolov5," *Sustainability*, vol. 15, no. 7, 2023. [Online]. Available: https://www.mdpi.com/2071-1050/15/7/5963

[10] S. Srivastava, A. V. Divekar, C. Anilkumar, I. Naik, V. Kulkarni, and V. Pattabiraman, "Comparative analysis of deep learning image detection algorithms," *Journal of Big Data*, vol. 8, no. 1, p. 66, May 2021. [Online]. Available: https://doi.org/10.1186/s40537-021-00434-w

[11] R.-Y. Ju and W. Cai, "Fracture detection in pediatric wrist trauma x-ray images using yolov8 algorithm," *Scientific Reports*, vol. 13, 11 2023.

[12] Y. Dong, Y. Zhuang, and X. Yan, "Data-driven quality prediction of batch processes based on minimal-redundancy-maximal-relevance integrated convolutional neural network," *Mathematical Problems in Engineering*, vol. 2021, pp. 1–13, 11 2021.

[13] W. Huang, P. Wei, M. Zhang, and H. Liu, "Hripcb: a challenging dataset for pcb defects detection and classification," *The Journal of Engineering*, vol. 2020, no. 13, pp. 303–309, 2020. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/joe.2019.1183

[14] *Academic Journal of Science and Technology*, vol. 7, no. 3, p. 297–304, Oct. 2023. [Online]. Available: https://drpress.org/ojs/index.php/ajst/article/view/13420