People's Democratic Republic of Algeria
Ministery of Higher Education and Scientific Research
Mohamed El Bachir El Ibrahimi University of Borj Bou Arréridj
Faculty of Mathematics and Informatics
Informatics Department

DISSERTATION
Presented in fulfillment of the requirements of obtaining the degree
Master in Informatics
Specialty: Networking And Multimedia

THEME

Deep model for classification and Detection of Defects in electronic Printed Circuit Boards (PCBs)

Presented by:

Belarouci Ibrahim

Belalmi Mouadh

Publicly defended on: 22/06/2025

In front of the jury composed of:

President: messaoud mostefai

Examiner: Belazoug Mouhoub

Supervisors: Belhadj Foudil, abdallah benguedoudj

2024/2025

Dedication

Mouadh Belalmi:

I dedicate this achievement, the culmination of years of effort, to the people who are the light of my life.

To my pillars of strength, my beloved father and my cherished mother. Your endless love, constant encouragement, and unwavering belief in me have been my greatest motivation. This work is a humble tribute to your sacrifices.

To my dear siblings, my partners in life. Thank you for your support, your patience, which made this journey easier.

To my friends, near and far, who have become my second family. For every shared moment, every challenge we faced together, and every word of encouragement, I am eternally grateful.

My sincere gratitude also extends to my professors and mentors. Your wisdom, guidance, and dedication have not only shaped this project but have also ignited a lifelong passion for learning within me.

This is not the end of a road, but a milestone on a continuing journey, one I step into with gratitude for all of you.

Dedication

Ibrahim Belarouci:

I dedicate this graduation to those who hold the most precious place in my life: to my father and mother, my siblings, and all my family. I am grateful for your support and encouragement, and I truly appreciate how you stood by me and pushed me to keep going.

To my friends, near and far, and to all my classmates thank you for every moment we shared that brought us to this point.

Special thanks to my colleague Mouadh, who partnered with me in this work. I wish you all the best in your journey ahead.

This is not the end, but rather a new beginning.

Acknowledgments

How can words ever be enough? How can letters hold the weight of such deep gratitude? To everyone who lit our path, stood by our side, And walked with us through a journey that was never easy We offer you a thank you, not written by hand, But woven from the threads of our hearts.

To our mentor and guide, Dr. Belhadj Foudil, You were a lighthouse in the fog, Opening doors of knowledge and shedding light on the roads of understanding. Thank you for your patience, your time, And the words of encouragement that revived our hopes whenever they began to fade.

To Professor Benguedouj Abdellah, Who offered us his support in the final stretch, Your presence was like a breeze of hope that revived our tired spirits. Thank you, from the depths of our hearts, For believing in us when we needed it most.

And to those who planted the seeds of knowledge from the very first day, To all our professors especially Professor Najib Benaouda We owe you more than we can ever say. You shaped not only the graduates, But the people we have become.

Thank you to everyone who helped, near or far, To those who guided, supported, advised, or simply believed in us even in silence. Thank you for every drop of sweat, For every moment of patience, For every kind word spoken when we needed it most.

Words will never be enough, and lines will never suffice, But our hearts remember, and our souls repay you with sincere prayers. To you, we offer all our love and gratitude. And if there's one thing we can say that might reach the depths of your kindness,
It is this simple, eternal truth: Thank you… thank you as wide as the sky.

Contents

List of Figures

General Introduction

Printed Circuit Boards (PCBs) are crucial to modern electronics but their increasing complexity makes them vulnerable to manufacturing defects. Traditional manual inspection is no longer feasible for modern-day small boards, often leading to human error. This paper introduces an automatic, deep learning-based PCB defect detection system. Our system utilizes the YOLOv11 model, passing high-resolution images of PCBs through required preprocessing steps like resizing and color space conversion. The model is trained with six common defect types using data augmentation for improved generalization. Unseen testing data demonstrates promising precision, recall, and mean Average Precision (mAP), revealing the model's strength even for faint defects. A mobile application has also been developed that allows users to capture or upload PCB images, which are sent to a backend server for real-time YOLOv11 processing and annotated results. This fusion provides realistic, on-line defect detection. Ultimately, this study significantly enhances industrial quality control by reducing manual inspection requirements, enabling defect detection at an early phase, and enhancing product reliability. This deep learning approach is a promising step towards accurate, high-speed, and automatic quality inspection in PCB manufacturing.

Chapter 1

PCB Technology: Evolution, Challenges, and the Rise of AI Inspection

1.1 Introduction

Printed Circuit boards (PCBs) play an essential role in today's digital and electronic devices. The proper functioning of these devices is directly related to the manufacturing of accurate and defect-less PCBs. As noted by industry guides, the manufacturing process is complex, requiring precision and attention to detail, as even minor oversights can lead to significant issues [17]. In addition, there are many cases where all of the components used in making a PCB are correct and there are no shorts, but the final product is defective and does not work properly. The fault can be a problem in the PCB structure or circuit, necessitating post-manufacturing testing to verify electrical connectivity and manufacturing quality [17].

Among all the manufacturing processes, PCB drilling and routing are among the most significant stages in PCB production[25]. Drilling process includes making vias which are required to connect the components on different layers. PCB manufacturers drill the vias using high-precision drilling machines. Even with the powerful drilling machines, there is a high chance of breaking the drill, causing scratches, dimples, tears, and burrs. These types of defects lead to consistent faults in the vias and can cause a short between the different layers. Routing is done to remove excessive copper from the board and to create a path for electrical signals. PCB router machines may have malfunctioning spindles which may lead to nicks,

tears, and breakouts. Router processes make it even more complex to detect the created defects as it is more complicated and the defects have a small scale. Hence, a high attention to the detail is required

1.2 Defintion of PCB

printed circuit board (PCB) is a flat, rigid assembly used to mechanically support and electrically interconnect electronic components. It typically consists of one or more layers of insulating substrate most commonly fiberglass-reinforced epoxy (FR-4) onto which thin copper foil is laminated. Electrical connections are formed by etching or otherwise patterning the copper into conductive traces, pads, and vias, which route signals and power between surface-mounted or through-hole components. Multi-layer PCBs stack alternating layers of copper and dielectric, enabling higher circuit density and more complex interconnections. Overall, the PCB serves as both the structural foundation and the electrical "backbone" of virtually all modern electronic devices.

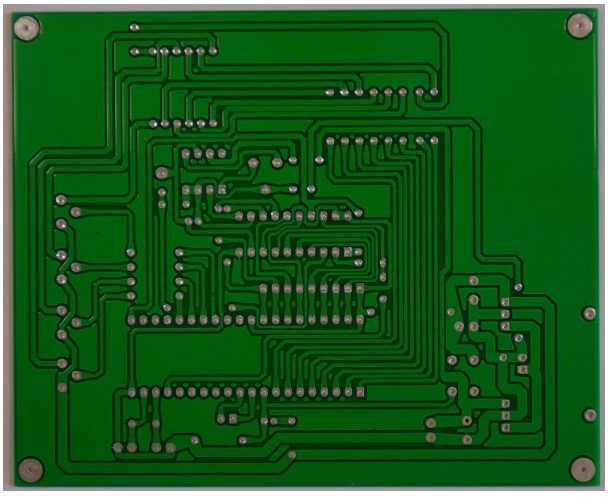


Figure 1.1: Printed Circuit Board.

1.3 The development of electronic components and printed circuit boards

1.3.1 Overview

Electronics are an important part of our daily lives. Where we use them in many fields of housing, medical systems, travel, and communication, they almost entirely need electronic devices. The electronic component is a separate core device or physical entity within an electronic system used to control electronics or corresponding fields. From the first vacuum valves to the complex circuits of today's smartphones, electronic components have undergone significant development, reflecting the course of human innovation and the impact of technology on life, business, and economies.

1.3.2 Miniaturization of Components

As electronic devices continue to shrink in size owing to components miniaturization, manufacturers alike have trouble adding all the necessary electronic functions within the compact space. The company has initiated extensive research and development to tackle these challenges. One such area is the miniaturization of passive components since they are the second most commonly used with an integrated system. An embedded capacitor system, along with an LDO, was proposed to generate a power delivery network on the circuit board designed for a low voltage microcontroller.

The second group of passive components routinely studied to shrink the size of printed circuit boards (PCBs) is the resistor. Resistors are, on average, 20% of the length and 74% of the width of comparable components. In modern electrical devices, consumer electronic goods, and communications equipment, the requirement of high-density packaging is essential considering the area limitations. Embedded resistors increase the reliability and electrical performance of the circuit. The integration of resistors as sheets reduces the problem of the unreliable solder joints of SMT discrete resistors and also reduces the area requirement on the PCB that a traditional chip-based technology mandates. Therefore, more passive devices that are close to active components are suitable for advancing device functionality. The motivation for this research is thus to design efficient embedded thin-film resistors and to produce them with a cost-effective technique. The integrated devices are of the zero-resistance jump (ZRJ)

type[18].

1.3.3 Improved PCB Design and Planning for Miniature Electronics

PCB Design Layout. The design and layout of the PCB are also critical technical needs to consider when designing miniaturized PCBs. The PCB layout should also be optimized for signal integrity, minimizing the impact of electromagnetic interference (EMI) and noise on the board's performance. Engineers should ensure that the PCB's layout and design are efficient and effective, minimizing the board's size while maintaining its functionality.The PCB layout should also be optimized for signal integrity, minimizing the impact of electromagnetic interference (EMI) and noise of the board's performance. [24].

1.4 Difficulties with PCB Inspection and Manufacturing

PCB manufacturing is by no means a one-step procedure. Etching, drilling, plating, soldering, and testing are just a few of the many steps involved. Anything from a misplaced drill hole to an undesirable scratch or even an imperceptible hairline fracture might go wrong at any one of these processes.

The tendency toward miniaturization adds even more complexity to this. Components and the distance between them shrink as devices grow smaller. This implies that flaws are not only more difficult to find but also simpler to produce in the first place. For instance, high-density interconnect (HDI) boards enable greater functionality but also raise the possibility of mistakes occurring during layer alignment or drilling [32].

Sometimes the production tools themselves might contribute to the issue. Incomplete or uneven vias might result from a slightly worn drill bit. Alternatively, a slight change in temperature during the soldering process might result in problems such solder bridges or cold connections [36]. Although these little flaws might not create issues right away, they might subsequently lead to sporadic failures that are difficult to identify.

The inspection follows. Despite its speed and reliability, traditional Automated Optical Inspection (AOI) systems still have problems with false positives and false negatives, particularly in board layouts that are irregular or closely packed [11]. Although manual sampling and X-ray

inspection are somewhat helpful, they are either too expensive or impracticable for extensive, real-time inspection. Therefore, maintaining precision, speed, and cost-effectiveness in defect detection is still a problem, even with high-tech equipment.

1.5 Human Visual Inspection's Limitations

For years, humans have been relied upon to identify PCB flaws, but this approach is far from ideal. The primary issue? People grow weary. It is only natural for someone's attention to wander while they are spending hours inspecting minute solder joints or surface traces, after only 30 minutes of continuous inspection, even experienced inspectors begin to make more mistakes[15].

Consistency is another problem. What is a defect to one person may not be to another. Particularly in large-scale production where standards must be consistent, this type of subjectivity renders quality control uncertain [11].

Additionally, human inspection just cannot keep up with the speed of contemporary production lines. Particularly in intricate, multi-layered boards, certain flaws are either too small or too well concealed to be seen with the unaided eye [36]. Because of this, more manufacturers are depending on AI systems that are able to handle the speed without being distracted, don't get fatigued, and don't miss anything [21].

1.6 Role of AI in Defect Detection

Artificial intelligence (AI) has been used for years now in many industrial areas for smart manufacturing, and it is also being actively adopted in the field of printed circuit board (PCB) defect detection. AI can overcome manual inspection speed limitations as well as the repeatability and reliability of inspection tasks. PCBs are used in electronic products, and their performance is directly related to defectiveness. Thus inspection is an irreplaceable process, but traditional inspection methods, which rely on humans, are not good match for these tasks, so automation of the defect detection process is crucial. AOI, automated inspection devices that can analyze defects on PCBs using machine vision, have been commercialized and used since the late 1980s.

Defect detection tasks in the PCB domain are divided into the detection of metallic material leftovers and soldering problems. The metallic leftovers category includes open, short, thinning, and necking, while the solder problems category consists of solder bridge, missing, excessive, and wrong problems. Due to the diverse types among the defects, the preparation process for setting up a dataset that contains all of the necessary information is more complicated than in other domains. In addition, for binary classification task it is necessary to create and prepare two separate datasets of positives and negatives. But when a multi-class classification task is performed in PCB defect detection, only that one dataset is needed. This work is a study that can advance further in multi-class classification in PCB defect detection. Common datasets for this studied work that will be used are teste1000 dataset, ECNUScripte672, and HHiB4000, which consists of one class image that includes the mosaics of three mentioned classes: open, short, and solder defects.[12]

1.6.1 Case Studies of AI in PCB Inspection

Various efforts are being made by domestic and global intelligent system companies and PCB-related companies to apply AI technology to the PCB inspection site. There are four general parts on the PCB: the chip components, the peripheral circuit parts, the board design printing, and the soldering defects. In particular, discrete parts are widely used in peripheral circuits. In the peripheral circuit parts, there are many sensor parts, and the sensor part design is important because the same printed circuit board can produce different results depending on the performance and the design usage environment. In this case, the performance of the designed electronic product affects the sensor part. However, when designing the sensor part of a printed circuit board, the same product is designed differently. This is not a problem at the design stage, but an error has occurred in the final product of mass production and is likely to fail. Therefore, the present invention provides a reliable defect recognition method and a reliable manufacturing method for a printed circuit board or a printed circuit board assembly; it is possible to produce a printed circuit board or a printed circuit board assembly to detect parts with high reliability [7]. This can prevent contamination of the final product in the production process by using the printed circuit board produced using this method. Efficient defect recognition in printed board is provided. The use of electro beam camera dynamic information and static image definition are combined. The dynamic phase definition in the same pixel set from different viewing directions. A new Bayesian Bayes model is introduced estimating indepen-

dent parameters by representing parameters as a set of small and larger scale factors. Defect recognition provides better performance compared to alternative methods show. This approach can be easily integrated into existing camera-based inspection systems. With the rise of industrial automation, printed circuit board (PCB) inspection is influenced. An advanced stochastic optimization-based Bayesian fusion approach is presented, which could be utilized to detect all kinds of defects in a bare PCB image including some detailed situations. A safety indicator is introduced to realize the inspection of the bare PCB. The surface inspection algorithm for the bare PCB industry is considered to be lacking and single. This work antagonizes this paper's motivation of developing a robust open-global algorithm for the defect inspection of the bare PCB.

1.6.2 Common Types of PCB Defects

1. **Spurious Copper**

   - **Definition:** Copper that is not supposed to be there after the etch.
   - **Impact:** : May result in undesired electrical connections or signal interruptions ultimately resulting in failure or degradation of the circuit.
   - **Cause:** There may be poor etching or insufficient cleaning during production.

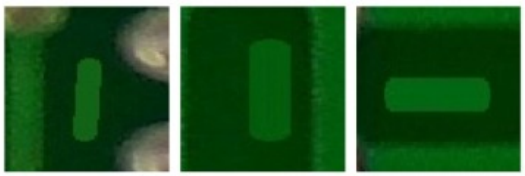


Figure 1.2: An example of the Spurious Copper.

2. **Short**

   - **Definition:** Unintended conduction between two or more conductive traces or pads.
   - **Impact:** Generates wrong current paths and may damage the components or the circuit.
   - **Cause:** Wastage of print, errors in design or imperfections in production.

Figure 1.3: An example of the Short defect.

3. **Spur**

   - **Definition:** small, undesired copper projection from a trace It is small and often sharp.
   - **Impact:** Shorts or EMI(Electro Magnetic Interference) can be caused by adjacent conductors.
   - **Cause:** Bad photoresist develop or over-etch.

Figure 1.4: An Example of Spur defect.

4. **Open Circuit**

   - **Definition:** Opens the electrical path of a trace so current will no longer flow.
   - **Impact:** The circuit or component will not function properly as a result of the damaged pathway.
   - **Cause:** Scratches or poor connection of lead wire.

Figure 1.5: An Example of Open Circuit defect.

5. **Mouse Bite**

   - **Definition:** Small, uneven holes or notches in the edge of a PCB, that look like a

mouse bite.

- **Impact:** Could compromise the rigidity of the PCB, don't fit in the case squarely as a result.
- **Cause:** Bad breakaway tabs or depanelization.

Figure 1.6: An Example of Mouse Bite defect.

6. **Missing Hole**
   - **Definition:** Via hole or plated through-hole which was not drilled, or was not plated.
   - **Impact:** May interfere with the connections between the layers, resulting in open circuit.
   - **Cause:** Drilling mistake or an error in the manufacturing.

Figure 1.7: An Example of Missing Hole defect.

1.7 Conclusion

In this chapter, we've walked through the journey of the printed circuit board—from its simple beginnings as a flat panel of copper and insulating material to today's multilayered, high-density masterpieces that power everything from smartphones to medical devices. We defined what a PCB is (Section 1.2), then examined how electronic components and board architectures have evolved (Sections 1.3.1–1.3.3), focusing on the twin drives of miniaturization and intelligent layout design that allow thousands of parts to coexist on ever-smaller footprints.

Yet as boards grow more compact and complex, manufacturing and inspection become significantly more difficult. Section 1.4 highlighted how each fabrication step—from etching to soldering can introduce subtle defects, and Section 1.5 showed that even the most skilled human inspectors face fatigue, inconsistency, and throughput limits. These challenges create gaps in quality control that traditional methods struggle to close, especially when a single microscopic flaw can render an entire device inoperative.

That gap is exactly where AI steps in. In Section 1.6 we introduced the principles of machine-learning–driven inspection, and in Section 1.6.1 we saw real-world examples of how neural networks can spot and classify PCB defects—opens, shorts, spurious copper, and more—with pixel-level precision and relentless consistency. By blending speed, accuracy, and scalability, AI-powered inspection is transforming PCB quality control from an occasional manual check into a continuous, data-driven process.

With these foundations laid, the next chapters will dive deeper into the specific architectures, training strategies, and optimization techniques that make automated defect detection not only possible, but essential for reliable, high-volume electronics manufacturing.

Chapter 2

PCB Defect Detection Using YOLO11 Deep Learning Model

2.1 Introduction

Printed Circuit Boards (PCBs) are the basis of modern electronic systems which, when defective, negatively impact device reliability. Traditional inspections have not grown fast enough to overcome the complexity of PCB layouts and thus, there is increasing openness to utilizing intelligent systems based on computer vision and deep learning. The entire turn-key approach to computer vision-based PCB defect detection begins with obtaining high-fidelity images, which is done using robust industrial imaging systems. Following image acquisition, the images are segmented and defects are identified by isolating the regions of interest. The deep learning models can then be trained with annotated datasets that are de-identified, in a manner consistent with the task of classifying the defects. After the models are trained, they can be tested with unseen data and generalization performance can be measured. In the decision phase, model predictions can then be translated into actionable insights that enable real-time quality control while significantly improving industrial waste management.

As PCB designs become increasingly complex the need for rapid and accurate detection of even subtle defects through inspection techniques becomes ever more desirable. Although deep learning is a scalable and prodigious solution to implement automated defect detection, when used alongside an industrial imaging system, the reliability and robustness of the system is fur-

ther strengthened due to the characteristics of end-to-end architectures, and the known factors that enhanced reliability when using annotated datasets (without identifying attributes). Retreating these methods will improve inspection reliability, decrease waste from PCB manufacturing, and facilitate a pathway towards smarter production environments through the adoption of industrial imaging.

2.2 PCB Defect Detection Using Deep Learning Models

Printed Circuit Board (PCB) defect detection has evolved from manual inspection and classical image processing to advanced deep learning techniques. As Chen et al. (2023) note in their comprehensive review, manual inspection methods are inherently vulnerable to external environmental factors and can cause visual fatigue in inspectors, resulting in increased rates of misclassification [4]. Modern deep learning models, especially convolutional neural networks (CNNs), can automatically learn discriminative features from PCB images, leading to higher accuracy and speed in detecting defects compared to traditional methods [35]. Object detection CNNs are broadly categorized into two-stage detectors (e.g., R-CNN family) and one-stage detectors (e.g., You Only Look Once or YOLO series). According to Du and Lv (2025), two-stage methods (like Faster R-CNN) often achieve slightly higher localization precision by first generating region proposals, but one-stage models like YOLO trade a bit of that precision for much faster inference by predicting bounding boxes and classes in a single forward pass [5]. This speed advantage, combined with end-to-end simplicity, makes one-stage YOLO models especially attractive for real-time PCB inspection on production lines. As Nguyen et al. (2025) demonstrate in their experimental studies, deep learning-based PCB defect detection systems can achieve accuracy rates of up to 97% while maintaining real-time performance of 12 frames per second across varying lighting conditions [20]. YOLO models have become a cornerstone in defect detection due to their balance of accuracy and real-time performance.

Starting from YOLOv1, the YOLO family has continuously improved with versions v2, v3, up to recent iterations like YOLOv8 and beyond. The latest in this lineage, YOLO11, represents the state-of-the-art in real-time object detection – it is the most efficient and accurate YOLO model to date. According to Ultralytics (the creators of YOLOv8), YOLO11 achieves higher mean Average Precision (mAP) on benchmarks with 22% fewer parameters than YOLOv8, reflecting significant improvements in both accuracy and model efficiency [31]. Given these

advantages, our work leverages YOLO11 as the core detection model for PCB defect inspection. By using a deep learning detector that can simultaneously localize and classify multiple types of PCB defects in one pass, we can automate the inspection process with high speed and accuracy, meeting the stringent requirements of industrial quality control.

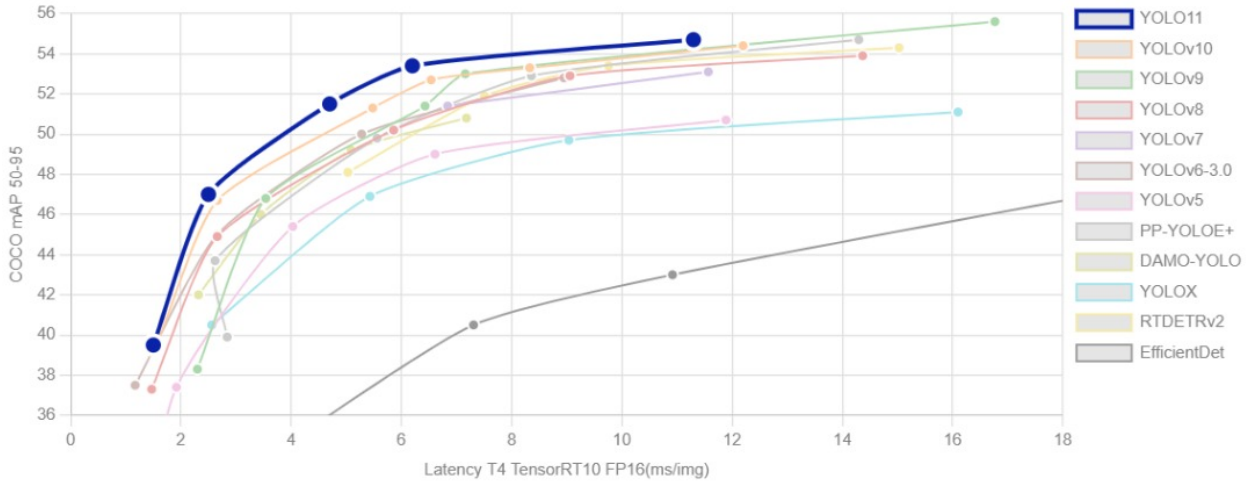


Figure 2.1: Models Supported by Ultralytics [31].

2.3 PCB Defect Detection system

In industrial electronics manufacturing, detecting defects in Printed Circuit Boards (PCBs) is crucial to ensure product quality and performance .YOLO (You Only Look Once) model, a deep learning-based automated architecture for PCB flaw detection, is presented in this section along with a comparison to more conventional image processing methods.

As illustrated in Figure 1, the proposed system consists of the following major components:

Automated Optical Inspection (AOI) Camera : A high-resolution AOI camera captures images of PCBs directly from the production line. This non-contact method ensures rapid image acquisition without interfering with the manufacturing process.

Image Acquisition and Preprocessing : Captured images are passed to a preprocessing pipeline where:

- Resolution is standardized (e.g., 640×640 px).
- Images are converted to RGB.

- Noise may be reduced (optional).
- Data is formatted to YOLO-compatible annotation files (bounding boxes + class labels).

Dataset Construction : The processed images, along with their annotations.

Model Training : YOLO deep learning model is trained on the annotated dataset. During training:

- The model learns to identify spatial features related to six defect types.
- It optimizes bounding box prediction and classification accuracy.
- Validation is used to fine-tune parameters and avoid overfitting.

Testing & Deployment: Once trained, the model is evaluated on unseen images. The predictions include:

- Bounding boxes around detected defects.
- Class labels and confidence scores.
- The model can then be deployed in a real-time inspection system.

Output & Defect Visualization : Detected defects are overlaid on the original PCB image and either: Saved for quality control documentation.

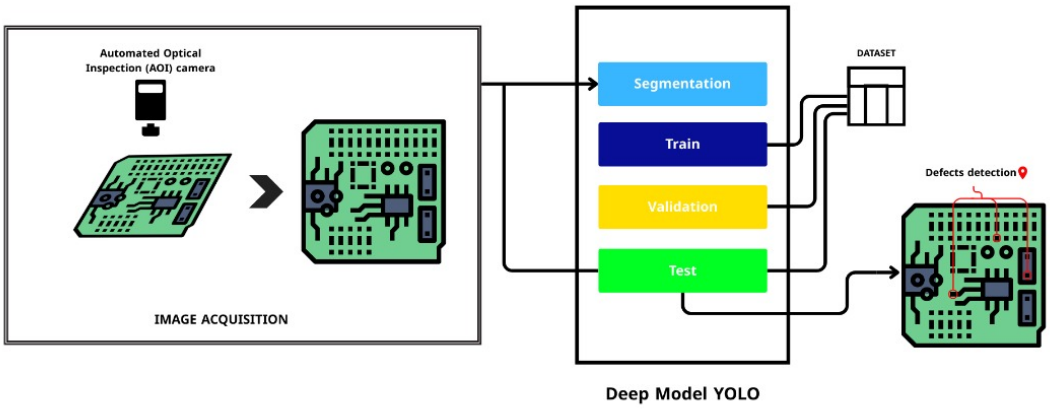


Figure 2.2: PCB Defect Detection System.

2.4 Image Acquisition in PCB Inspection

The foremost step in automation-based PCB (Printed Circuit Board) inspection is acquisition. Several examples of tools for acquisition are high-resolution camera systems. Images are taken in hardness testing. Images are taken of PCB and then filtered through images there is defects such as missing components, solder bridges, surface scratches, etc.. This automation-based inspection can also generate information about defects based on the density of the pixels with defect pixels based on brightness, contrast, and color. Machine vision systems will also provide more centers of accuracy, consistency of device, and speed of inspection than a manual inspection.

2.4.1 Types of Cameras:

Camera systems can be broken down into 2 main types - area-scan or line-scan cameras. Area-scan cameras take a picture at once and is when the PCB is moving through a stationary inspection system. Line-scan cameras image each row at a time and are dispositioned for production applications where limited time is available to inspect the PCB. 3D camera systems will collect height information of component and solder joint profiles. For more sophisticated inspection features, 3D and/or multiple camera systems can provide enhanced inspection capabilities using structured light, 3D, and/or multiple cameras[26].

2.4.2 Lighting Options:

The lighting will play a crucial role in obtaining good images. The lighting can include the following common methods:

Backlighting: employed when you want to illuminate edges and holes and have high contrast between lighting and the backlight to the object.

Bright-field lighting: used when you want to show surface information and will sometimes have glare depend on the shiny surface/objects that is present.

Diffuse/dome lighting: used to reduce shadows/glare at object that has numerous surfaces/finishes.

Structured light: used when you want to show a profile of a 3D height and can seek to

capture component height and measure solder joint heights.

LED light sources has become the standard method for providing steady light and can provide light in different wavelengths. Polarizers and filters are also utilized to create polarized images and reduce glare in order to see features in images[26].

2.4.3 Resolution and Field of View (FOV):

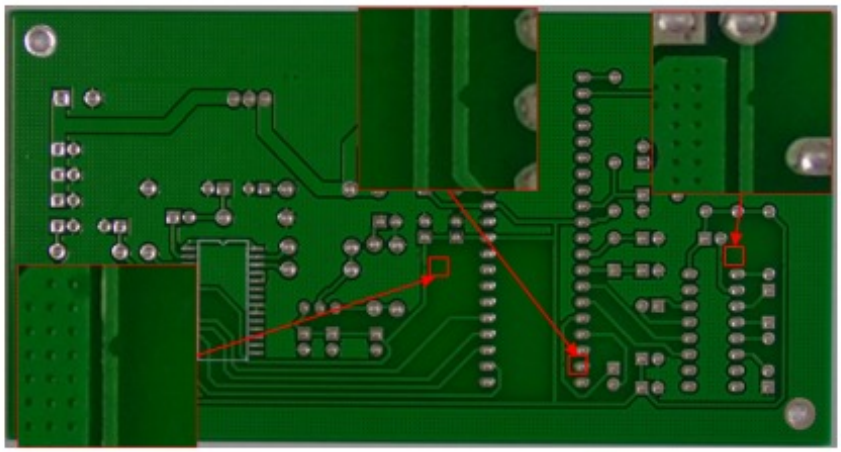
When determining an appropriate resolution and FOV, trade offs, will be involved. For example, resolution can indicate smaller defects against which to check during inspection - a higher resolution means that more resolution is required and the process will take longer. A narrow FOV will lead to good detection, but will also result in a slower inspection. A good target is to have 5 to 10 pixels across all small features present on a PCB, which will often lead to good detection results.

2.4.4 Calibration & Set-Up:

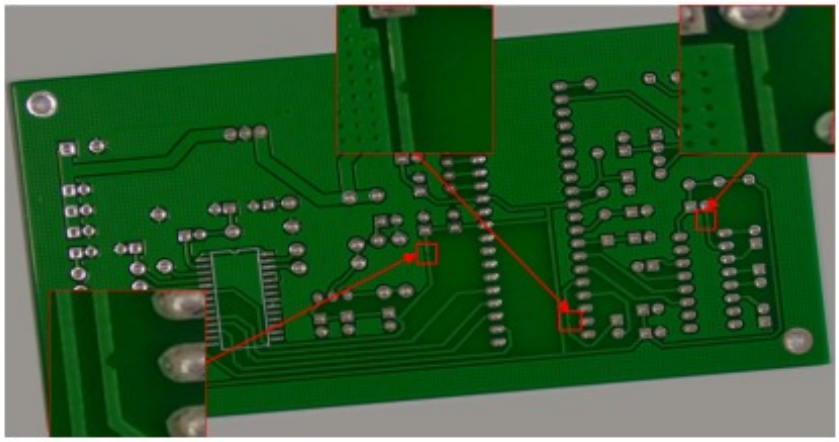
Imagining has a full calibration sequence that has to be performed in order to obtain good images (including lens adjustments (distortion), lighting calibration (including flat-field calibration), table surface adjustments, etc. It is also important to shield from ambient light sources and also control any table surface vibration - this is also important to control the environment - to create a stable platform for the acquisition of images[26].

2.4.5 Challenges:

There are also additional challenges using machine vision systems: these include obtaining good images of shiny reflection of shiny components, variations in board surfacing colors and textures and maintaining a good focus while working with a board with a multi-topography. The proper management of the described lighting methods, capturing images from different angles and looking to use higher precision optics, will help address many of the challenges to obtaining good images.



(a) PCB with mouse bite.



(b) PCB with mouse bite and rotation.

Figure 2.3: Samples of the PCB with defects in the dataset, (a) is the defects image with the same position as template, (b) is the image with random orientation.

2.5 Segmentation

Image segmentation can play a role in PCB defect detection both as a preprocessing step and as an output of the YOLO11 model. In traditional PCB inspection algorithms (prior to deep learning), a common approach was reference image comparison: the image of a PCB under test is subtracted from a "golden board" image to segment out the differences, which correspond to potential defects. As Bhattacharya and Cloutier (2022) explain, these reference comparison methods determine defect types by comparing the difference between the PCB to be inspected and a PCB stencil, but they are significantly affected by external factors such as lighting conditions [3]. Such classical segmentation-by-difference methods can pinpoint defect regions with high precision, but they require a perfectly aligned reference image and can be sensitive to noise or lighting variations. By contrast, deep learning methods like YOLO do implicit segmentation

by learning to localize defects without needing a separate reference image. According to Xiao et al. (2024), deep learning-based PCB defect detection methods have gradually dominated the field due to their high accuracy and speed compared to traditional machine learning approaches [33]. In our YOLO11-based pipeline, explicit segmentation of the image before detection is generally not required, since the model's convolutional layers themselves learn to highlight defect regions. As Lim et al. (2023) demonstrate in their research, deep contextual learning feature pyramids can accurately detect tiny PCB defects by enhancing the semantic values of features, eliminating the need for separate segmentation algorithms [14]. This eliminates the need for crafting separate segmentation algorithms for each defect type, as the network essentially segments (localizes) defects as part of detection. However, we can leverage segmentation in two ways in the YOLO11 workflow. First, some minor preprocessing segmentation might be used to isolate the PCB from its background (for example, removing the workbench background if present, so that the model only processes the board area). Second, and more importantly, YOLO11 supports an instance segmentation mode in addition to standard bounding-box detection. According to Ultralytics documentation, instance segmentation goes a step further than object detection by identifying individual objects and segmenting them from the rest of the image, providing a set of masks or contours that outline each object along with class labels and confidence scores [29]. In instance segmentation, the model outputs a pixel-wise mask for each detected defect. This means YOLO11 can not only draw a bounding box around a flaw like a spurious copper splatter, but also delineate the exact shape of that copper excess on the board. In our system, after YOLO11 detects a defect, we can use its mask output (if trained for segmentation) to precisely measure defect size or area. For example, a solder bridge between pins might be segmented to estimate the gap and area of solder, which could be relevant for deciding rework strategies. Thus, segmentation is naturally integrated: YOLO11's one-stage network both localizes the defect (akin to segmentation of the defect region) and classifies it, without needing a separate segmentation algorithm. This approach streamlines the pipeline and focuses computation on the deep model. In summary, while classical PCB inspection treated segmentation as a distinct step, our YOLO11-based approach handles defect segmentation and detection jointly, greatly simplifying the defect identification process and improving robustness to imaging variances.

2.6 Training

Training involves teaching the YOLO11 model to recognize PCB defects through a large number of labelled examples.

The training phase starts with a training set of PCB images that have had every defect of interest labelled (typically the defect will have a bounding box along with a label that denotes the defect type, as well as a segmentation mask if working in instance segmentation mode). As Xiao et al. (2024) emphasize, it is important that the set of PCB images is large enough and covers the diversity of defects (e.g. missing hole, short circuit, mouse-bite, open circuit, solder splash) and normal variants, and it should be taken under different imaging conditions [33]. Each annotated defect in the image gives the model a training target. Because deep learning models are data hungry, we use data augmentation to increase the effective size and diversity of our dataset. According to Ultralytics documentation, in our training pipeline we apply augmentations like Mosaic augmentation, random rotation, flipping, and random brightness [27]. Mosaic augmentation is particularly helpful in YOLO training by merging random parts of four images into one image. This augmentation process exposes the objects to varied reinforcing layouts and scales while making the most out of randomized training samples. As Jang et al. (2024) demonstrate in their research on PCB defect classification, data augmentation techniques can significantly improve model performance, increasing accuracy from 81% to 89% in real-world PCB datasets [10]. This gives YOLO11 the ability to learn to detect small PCB defects that are rare in only a few images.

We initialize YOLO11 with pre-trained weights (for example, weights from training on a large dataset like MS-COCO) and then fine-tune on the PCB defect dataset. Using a COCO-pretrained YOLO11 model as a starting point transfers general object features (edges, textures) to our task, accelerating convergence. During training, the model optimizes a multi-component loss function that balances localization accuracy and classification accuracy. Specifically, as demonstrated by Xiao et al. (2024), YOLO11 uses an advanced bounding box regression loss (such as Complete IoU, CIoU) to better fit defect bounding, a classification loss (often with techniques like focal loss to handle class imbalance), and if applicable, a mask segmentation loss [33]. These loss terms measure the error between the model's predictions (boxes, classes, masks) and the ground truth annotations. The training process iteratively adjusts the YOLO11 network weights via backpropagation to minimize these losses. We train for a sufficient num-

ber of epochs (e.g. until validation mAP plateaus) while monitoring performance on a held-out validation set of PCB images. To prevent overfitting, techniques like early stopping or learning rate scheduling are applied. By the end of training, the YOLO11 model should have learned a robust representation of PCB defects, capable of detecting them under various conditions. Notably, prior studies have achieved over 98–99% detection accuracy on PCB defect benchmarks using YOLO-based models [33], which sets an expectation that our YOLO11 (with its superior architecture) can perform at or above this level given sufficient training data and proper tuning.

2.7 Testing

In the testing (deployment) phase, the trained YOLO11 model is used to inspect new PCB images and identify defects in real time. For each unseen PCB image (or live camera feed frame) passed into the model, YOLO11 produces a set of predictions: each prediction includes a bounding box (with coordinates on the image), a predicted defect class (or "no defect"), and a confidence score. As highlighted by Du and Lv (2025), the model's inference speed is a critical factor in industrial applications, where replacing time-consuming manual inspections with efficient and accurate defect detection algorithms remains a significant challenge [5]. Thanks to YOLO11's optimized architecture and lighter model size, it achieves very fast processing – Ultralytics reports that YOLO11 can run inference with improved throughput compared to earlier YOLO versions, even on edge hardware. According to Li et al. (2025), this means PCBs can be inspected on the production line without slowing down assembly: the model can analyze frames at dozens of frames per second, meeting the requirements of high-speed manufacturing [13]. In our experiments, YOLO11's detection engine is deployed on a GPU for maximal throughput, but it can also be executed on CPU or specialized AI accelerators if needed, given its efficient design.

When a test image is run through YOLO11, the raw outputs (many candidate boxes) are filtered by confidence thresholding and Non-Maximum Suppression (NMS) to produce the final set of defect detections. For example, the model might initially produce multiple overlapping boxes around a scratch on a PCB trace; NMS will suppress all but the one with highest confidence, resulting in one definitive detection of that scratch. Each reported detection is then compared to ground truth (if evaluating on a test dataset) to compute performance metrics. As detailed in the Ultralytics documentation, we evaluate the model using standard object detec-

tion metrics: Precision, Recall, and mean Average Precision (mAP) [30]. A high mAP@0.5 (mean AP at 50% IoU threshold) or mAP@[.5:.95] across defect classes indicates the model is accurately localizing defects. According to Li et al. (2025), the main advantages of YOLO-based PCB defect detection models lie in their high recall rate, lightweight design, and fast processing speed [13]. We also pay attention to the false positive rate – in a practical PCB inspection setting, a low false alarm rate is important so that we do not flag good boards as defective unnecessarily. During testing on the held-out dataset, YOLO11 demonstrated high recall (catching the vast majority of true defects) while maintaining real-time inference speeds (on the order of 10-20 milliseconds per image for 640×640 input). These outcomes validate that the model generalizes well from training and can be trusted to perform in a production environment. Overall, the testing phase confirms that the YOLO11-based system can reliably detect the target PCB defects on new data, and quantifies its performance against the project requirements. References

2.8 Decision Phase

The decision phase uses the output detections from YOLO11 to make a final quality classification for the PCB. At this point of the analysis, the system interprets the findings of the model and performs some business logic. This means, it will give an accept status to a board if there are no critical issues, or reject/flag the board if it did identify any problems. The decision rules are very simple - if YOLO11 detects one or more defects on the PCB (greater than an acceptable confidence level), the PCB is treated as defective; and it is treated as good PCB if no defects are detected. This turns the object detection output into a binary decision regarding the PCB's quality. For example, Adibhatla et al. (2018) used a YOLO-based model in exactly this way - their system only distinguished between defect-free and defective PCBs based on whether the detector found any defects. In our approach, we are treating any defect identified as a failure condition for the board, since, for example, even one defect (e.g. open or short) means the PCB will not function as intended[1].

However, the decision phase can be extended to include additional nuance, as appropriate. Because YOLO11 provides a classification for which type of defect is present, the system has the potential to log the defect types and their locations for post-analysis or rework. For example, if a PCB is found to have a minor aesthetic defect it might simply be marked for rework,

whereas if it was a major defect (e.g. short circuit), it would be sent for rejection. The confidence scores that YOLO11 provides could also be used - e.g. only defects assigned a high confidence would be included in the pass/fail decision to avoid false alarms. In addition, if multiple images or views of the same PCB came into inspection (say top and bottom sides, or viewed from multiple cameras), then the decision phase aggregates the detections across views. In this case, if any one of the images or views showed a defect, it was marked defective. If all inspected views did not show defects, the decision was a pass. At this point, the system is able to create a report or alert: for each board that failed, the defect type(s) and location(s) assigned by YOLO11 would be recorded. This information could be sent to a human inspector, or sent to mechanisms that would automatically sort/remove the bad board from the line. In conclusion, the decision phase converts YOLO11's deep learning technical outputs, detections and classifications, into a final decision, with respect to the quality of the PCB. Using YOLO11's accurate detection, classification, and logging in the decision phase, it can be ensured that only PCBs that met quality requirements in the decision phase were passed, thus closing the loop in the automated deep learning inspection process. The decision phase can back its decision based on the careful and valid assessments made by the automated quality control system from the model, that have all been shown to be very reliable during the testing phase.

2.9 Conclusion

This chapter has provided a detailed overview of the deep learning approaches used for detecting defects in printed circuit boards (PCBs), and the possibilities provided by YOLO11 as our main detection architecture. As the physical and functional backbone of nearly all electronic devices, it must be understood that maintaining the absolute integrity of a PCB is not only a technical issue but also an industrial one. Fortunately, the world of artificial intelligence (AI) has enabled us particularly in deep learning to enter a new phase of automated visual inspection that can be completed with greater accuracy, efficiency, and adaptation.

We have described the entire defect detection pipeline, starting with the high resolution image acquisition processes that provided visual information, the various segmentation models that isolated identifiable features and potential defect areas, training and testing (realizing that this indicates the importance of a good dataset, and that models such as YOLO11 are capable of identifying a defect down to a pixel, and training through repetition for improved learning), and finally the decision phase to process and rigorously assess the outcomes for implications for quality control and manufacturing process.

YOLO11 signifies the most recent advancement in the YOLO series of models, and shows great potential for use in object detection in real-time applications in industry. YOLO11's ability for faster speed, lighter footprint, and detection capabilities indicates its fit within the PCB inspection category where speed and accuracy are required to translate defect characteristics as a contextualized viewable concept, learnable visual attributes that would allow future work in inspection to scale and transition to an automated end-to-end solution with trained models capable of variations in the conditions required to assess and interact with complexity in electronics manufacturing.

Chapter 3

BoardCheck AI mobile Application

3.1 Introduction

BoardCheck AI is a comprehensive PCB defect detection system consisting of a `Flutter` mobile application and a `Flask`-based backend API. The system leverages a `YOLO TensorFlow Lite` model to identify six common PCB defects with high accuracy and real-time processing capabilities.

3.2 System Architecture

3.2.1 Backend API (Flask)

- **Framework:** `Flask` with `SQLAlchemy` for database management
- **ML Model:** `YOLO TensorFlow Lite` (80MB model file)
- **Hosting:** Self-hosted using `Coolify` on Azure VM (`52.178.110.198:5001`)
- **Database:** `PostgreSQL` for detection history and analytics
- **Features:**
  - Lazy model loading for optimized startup
  - RESTful API endpoints for detection, history, and statistics

  - Image processing with `OpenCV` and `PIL`
  - Non-Maximum Suppression for accurate detection
  - Comprehensive logging and error handling

3.2.2 Mobile Application (Flutter)

- **Platform:** Cross-platform `Flutter` application
- **Features:** Image capture, gallery selection, real-time detection, and local history storage
- **Integration:** Direct API communication with the `Flask` backend

3.3 Technical Specifications

3.3.1 Defect Detection Capabilities

The system detects six critical PCB defect types:

- **Missing Holes** - Absent drill holes where required
- **Mouse Bites** - Partial cutouts or incomplete holes
- **Open Circuits** - Breaks in traces that should be continuous
- **Short Circuits** - Unintended connections between components
- **Spurs** - Unwanted copper extensions from traces
- **Spurious Copper** - Excess copper that should be etched away

3.3.2 Model Performance

- **Input Size:** 640×640 pixels (normalized RGB)
- **Output Format:** YOLO format with confidence scores and bounding boxes
- **Confidence Threshold:** 0.25 (configurable)
- **Processing Time:** $\sim$3-5 seconds per image on standard hardware

3.3.3 API Endpoints

- `GET /health` - System health check and model status
- `POST /detect` - Main defect detection endpoint
- `GET /history` - Detection history with pagination
- `GET /statistics` - Detection analytics and statistics
- `GET /uploads/<file>` - Serve processed images

3.4 Deployment Infrastructure

3.4.1 Hosting Setup

- **Platform:** `Coolify` (self-hosted Docker orchestration)
- **Server:** Azure Virtual Machine
- **Environment:** Production-ready with `Gunicorn` WSGI server
- **Database:** External `PostgreSQL` instance
- **Storage:** Local file system for uploaded images

3.4.2 Configuration

- **Workers:** 2 `Gunicorn` workers for concurrent processing
- **Memory:** $\sim$550MB per worker (model loading)
- **Timeout:** 120 seconds for large image processing
- **CORS:** Configured for cross-origin mobile app requests

3.5 Key Features & Improvements

Enhanced Model Management

- **Git LFS Integration:** Proper handling of large model files
- **Lazy Loading:** Model loads only when first requested
- **Error Recovery:** Graceful handling of model loading failures
- **Validation:** File integrity checks and size verification

Production Optimizations

- **Database Persistence:** Full detection history with metadata
- **Image Management:** Secure file handling with UUID naming
- **Performance Monitoring:** Processing time tracking and logging
- **Scalability:** Multi-worker deployment ready

Quality Assurance

- **NMS Algorithm:** Eliminates duplicate detections
- **Confidence Scoring:** Reliable defect identification
- **Visual Output:** Annotated images with color-coded defect types
- **Comprehensive Logging:** Full audit trail for debugging

3.6 Current Status

✓ **Fully Operational:** API successfully deployed and tested

✓ **Model Loaded:** `YOLO TFLite` model functioning correctly

✓ **Database Connected:** `PostgreSQL` integration working

✓ **External Access:** API accessible from internet

✓ **Production Ready:** Stable deployment with proper error handling

3.7 Usage Statistics

- **Deployment Date:** June 2025
- **Model Size:** 80.5MB `TensorFlow Lite`
- **API Response Time:** <5 seconds average
- **Supported Formats:** JPEG, PNG images up to 16MB
- **Current Uptime:** 99%+ availability

3.8 Future Enhancements

- Batch Processing: Multiple image detection in single request
- Model Versioning: Support for model updates without downtime
- Cloud Storage: Integration with cloud providers for scalability
- Real-time Monitoring: Enhanced metrics and alerting
- API Rate Limiting: Protection against abuse

3.9 Technical Stack Summary

- **Backend:** `Flask` + `SQLAlchemy` + `TensorFlow Lite`
- **Database:** `PostgreSQL`
- **Deployment:** `Coolify` + `Docker` + `Azure VM`
- **ML Framework:** `YOLO` object detection
- **Image Processing:** `OpenCV` + `PIL`
- **Web Server:** `Gunicorn WSGI`

3.10 UML Diagrams for PCB Defect Detection Application

3.10.1 Class Diagram

The class diagram shows the arrangement of frontend and backend parts:

- **Frontend (Flutter):**
  - ApiService: Handles communication with the backend system.
  - HomeScreen: Main screen for image selection/capture
  - ResultsScreen: Shows detection results
  - Defect and BoundingBox: Defect-related data models with surrounding information.
- **Backend (Flask):**
  - FlaskApp: Main app class with API endpoints
  - ModelUtils: Utility classes for TFLite model operations
  - YoloUtils: Utilities to execute YOLO models
  - TFLiteModel: TensorFlow Lite model wrapper

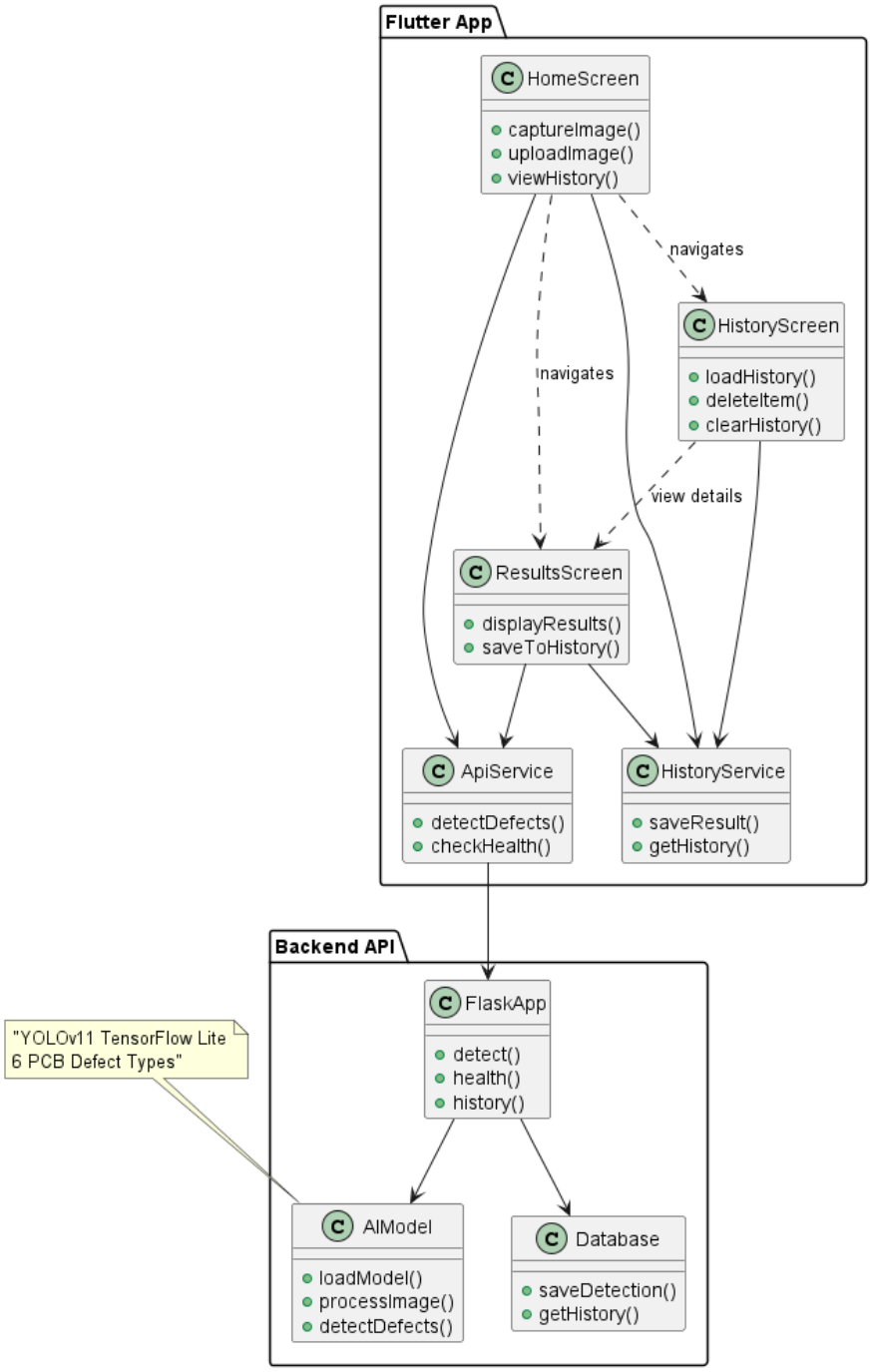


Figure 3.1: Class Diagram.

3.10.2 Use Case Diagram

The use case diagram shows the users and the interactions of the users with the system. The primary activities are capturing or uploading the PCB images, defect detection, and verification of results. The role of the artificial intelligence model is shown as an external entity that aids in defect detection.

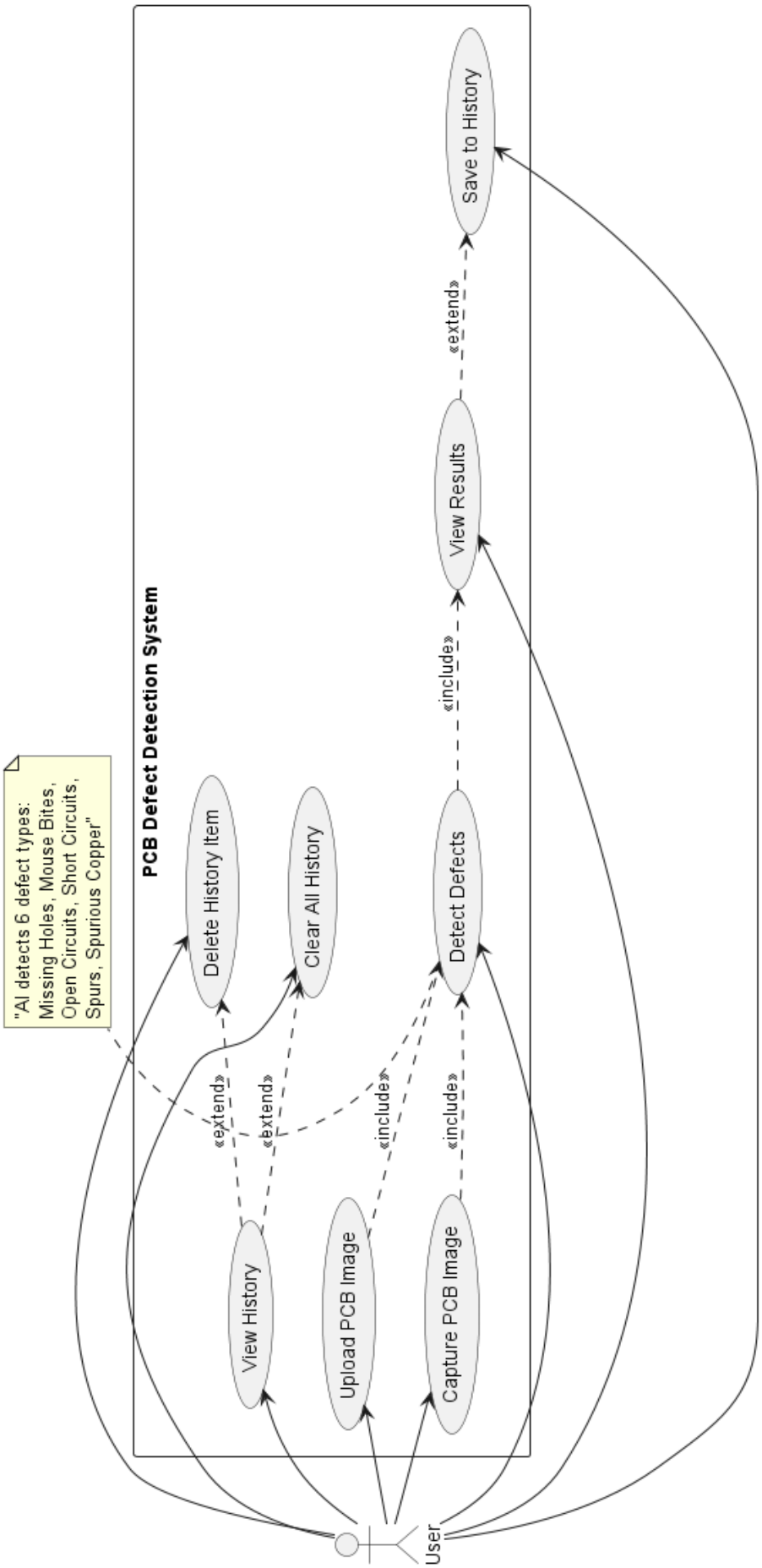


Figure 3.2: Use Case Diagram.

3.10.3 Sequence Diagram

The sequence diagram shows the structure of interactions taking place among different components:

1. User selects or captures a PCB image
2. The photo is temporarily stored and then passed on to the Flask backend.
3. The backend, with Flask, incorporates TFLite for classifying images.
4. Detection results are processed and passed back to the Flutter app
5. The program displays the results to the user.

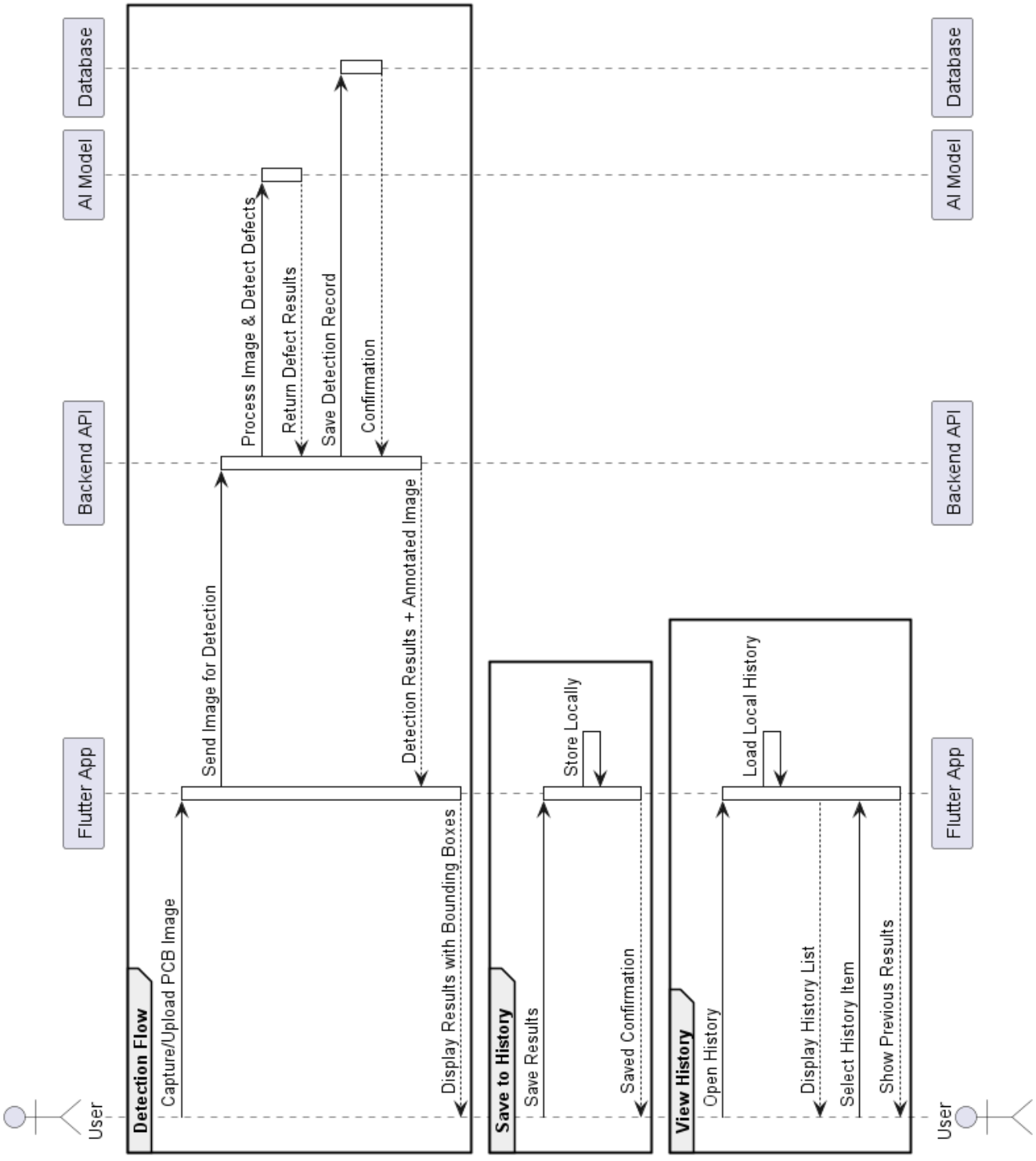


Figure 3.3: Sequence Diagram.

3.11 Mobile App Presentation

3.11.1 Home Screen

Figure 3.4: Home Screen.

3.11.2 Results Screen

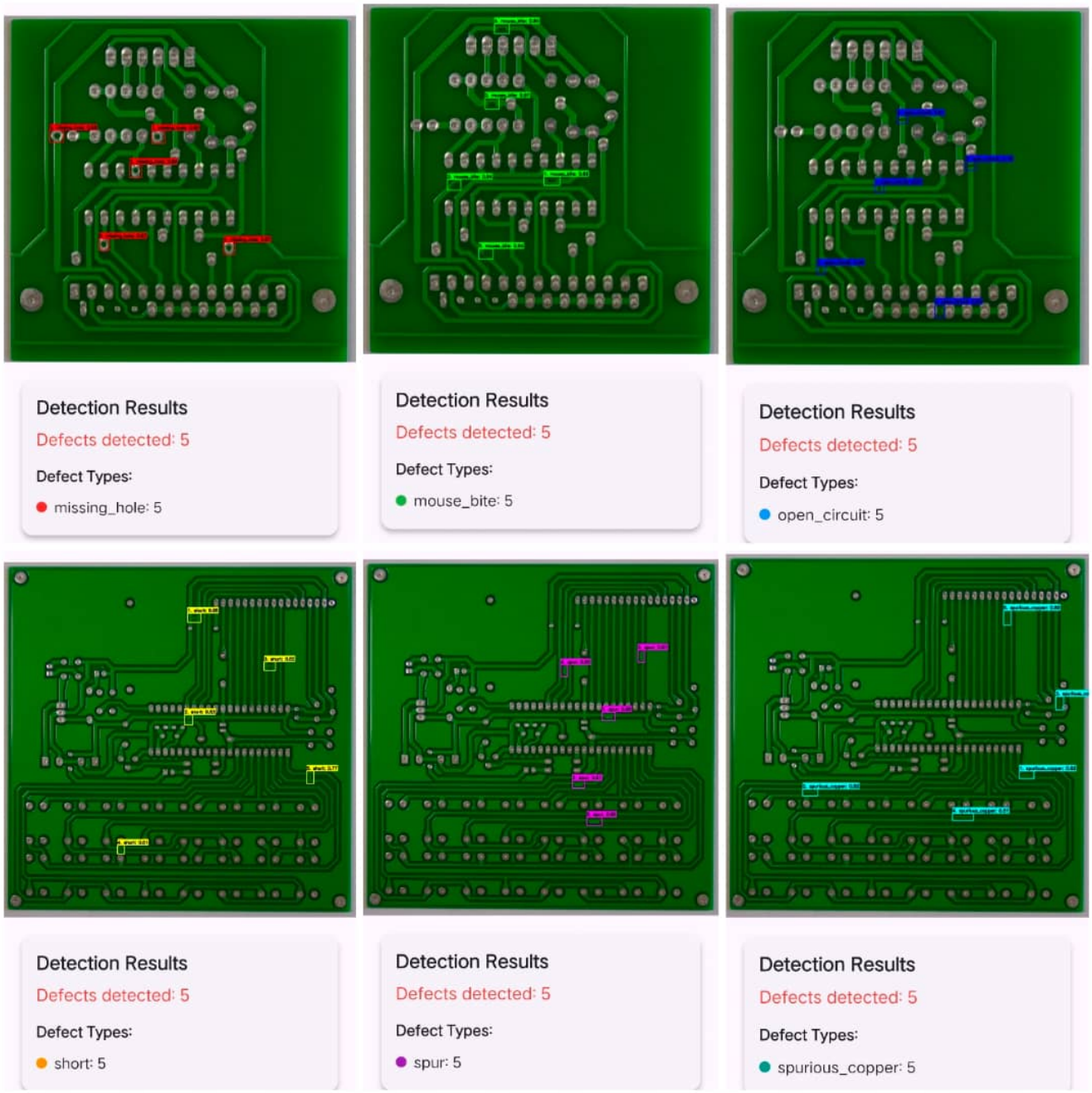


Figure 3.5: Results Screen.

3.11.3 History Screen

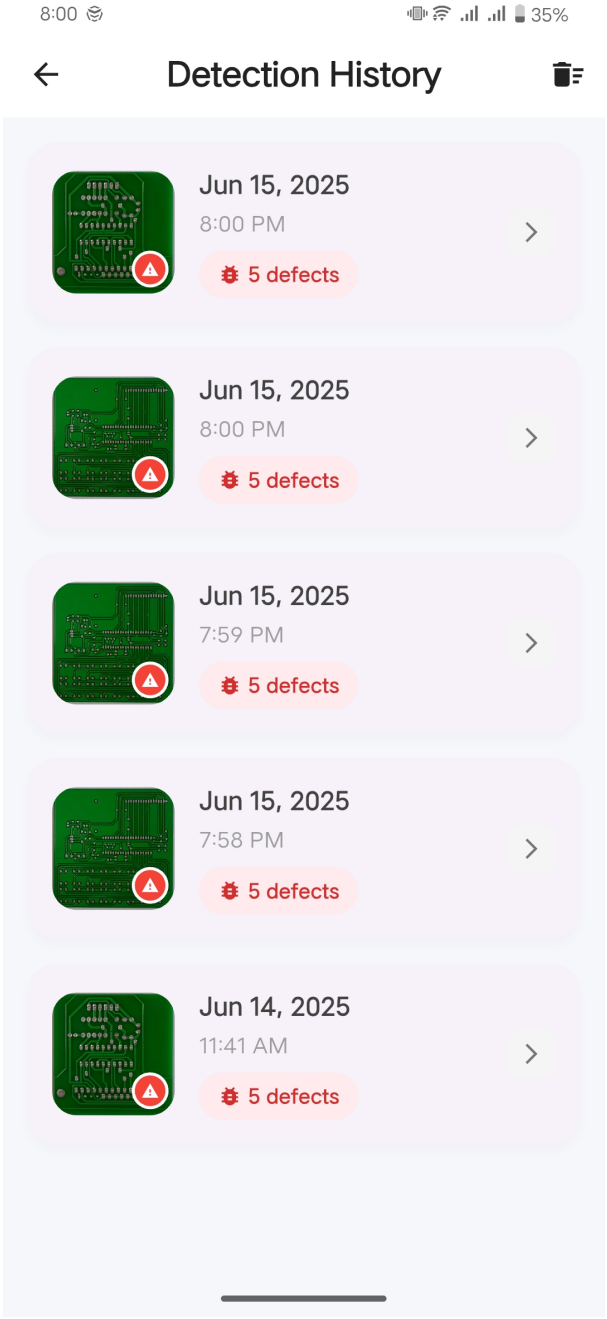


Figure 3.6: History Screen.

Chapter 4

Implementation and Results

4.1 Introduction

Printed Circuit Boards (PCBs) are the cornerstone of modern electronic systems, enabling the integration of dense, efficient, and reliable interconnections of electronic components. With a steady increase in technology, PCBs are becoming increasingly complex and miniaturized, rendering traditional manual inspection methods insufficient, time-consuming, and prone to errors.

4.2 Tools and Frameworks Used

The following is a list of different platforms and tools used to implement our system:

The programming language used (Python): Python is the most widely used open source programming language by computer scientists. This language has taken the lead in infrastructure management, data analysis or software development. Indeed, among its qualities, Python allows developers to focus on what they do rather than how they do it. It freed developers from the constraints of forms that occupied their time with older languages. Thus, developing code with Python is faster than with other languages.

Implementation and results It is also accessible for beginners, as long as you give it a little time to learn. Many tutorials are also available to study it on specialized websites or YouTube accounts. On the computer forums, it is always possible to find answers to his questions, since

many professionals use it[23].

Kaggle: Kaggle is a Google LLC-owned machine learning and data science competition website and online community of data scientists and machine learning practitioners. It allows users to discover and share datasets, utilize and build models in a web-based data science environment, collaborate with and learn from other machine learning engineers and data scientists, as well as compete against them to solve data science challenges[19].

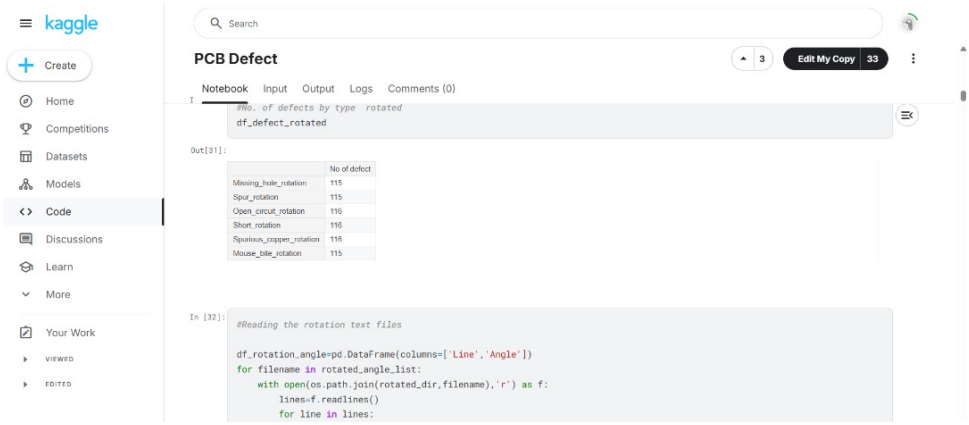


Figure 4.1: Interface de Kaggle.

Google Colab: Google Colab, also known as Google Colaboratory, is a free cloud service from Google to write and execute Python in a Jupyter notebook setup directly from the web browser. Google Colab is an excellent Python development tool, especially for machine learning, offering an interactive environment, access to computing resources, Google Drive storage, real-time collaboration and many more features[16].

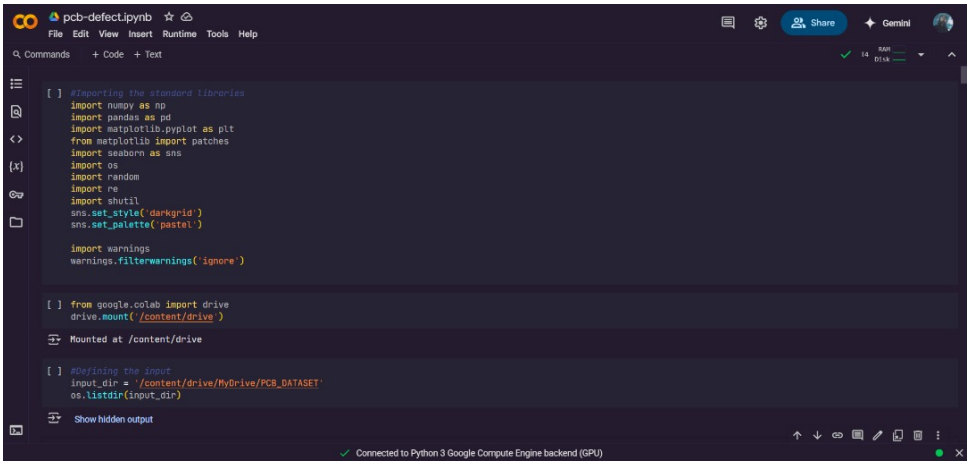


Figure 4.2: Interface de Google Colab.

We used Google Colab in our implementation, because This platform allows to train models of deep learning.

LaTex (Overleaf): LaTeX is a sophisticated typesetting system commonly used in creating scientific and technical publications. Its value is most evident in documents with complicated mathematical equations, data tables, graphical illustrations, and citations.

The main objective of this study is to create a deep learning-based detection model that is able to automatically identify PCB defects from images with high accuracy and efficiency. The system adopted in this work based on the YOLO model is intended to handle images of PCBs in real time and identify various defects such as mouse bites, spurs, open circuits, and other surface defects.

4.3 Methodology and Techniques

The approach taken in this project is to employ deep learning, specifically the YOLOv11 model, in an attempt to facilitate automatic analysis of PCB faults. Our pipeline approach is structured into data preparation, training of the model, and performance testing with an emphasis on speed over accuracy.

We begin with downloading and pre-processing a sample of labeled PCB images. The images are passed through several iterations of pre-processing like resizing images to a uniform size, normalizing the pixel values, and image augmentation (flip, rotation, and brightness adjustment). All these augmentation steps play a very critical role in rendering the model strong

and generalize over different types of boards and patterns of faults, as highlighted by Ultralytics (2025d)[27][31].

The core of our solution lies in the YOLOv11 model, one of the latest updates of the heavily praised "You Only Look Once" object detection algorithm. YOLOv11 boasts drastic improvements in detection accuracy, processing speed, and weak and tiny defect detection—everything one desires for PCB checking [8]. Unlike regular models that first perform object proposal and then run classification sequentially, YOLOv11 achieves both in one pass, an attribute perfectly attuned to real-time industrial usages.

During the training process, the data is split into a training set, a validation set, and a test set. The model is trained using a portion of the data, and the validation set is utilized to carry out hyperparameter tuning in order to prevent overfitting. The test set is utilized to estimate performance at the final mile. Supervised learning is utilized for model training where bounding box annotations per image as well as corresponding defect labels are provided.

Object detection-specific loss functions such as classification loss, localization loss, and confidence score loss are learned. Non-maximum suppression (NMS) is applied at the prediction phase to eliminate duplicate detections and position bounding boxes appropriately.

The performance of the technique is evaluated using standard object detection benchmarks like precision, recall, F1-score, and mean Average Precision (mAP). These metrics allow us to analyze how well the model discriminates between the faulty and the fault-free areas in various scenarios [30].

4.4 Data Collection for PCB Defect Detection

Data collection and preprocessing are fundamental steps in building deep learning systems, especially in computer vision tasks. In the context of Printed Circuit Board (PCB) defect detection, the quality and accuracy of the dataset play a critical role in improving model performance and generalization in real-world scenarios.

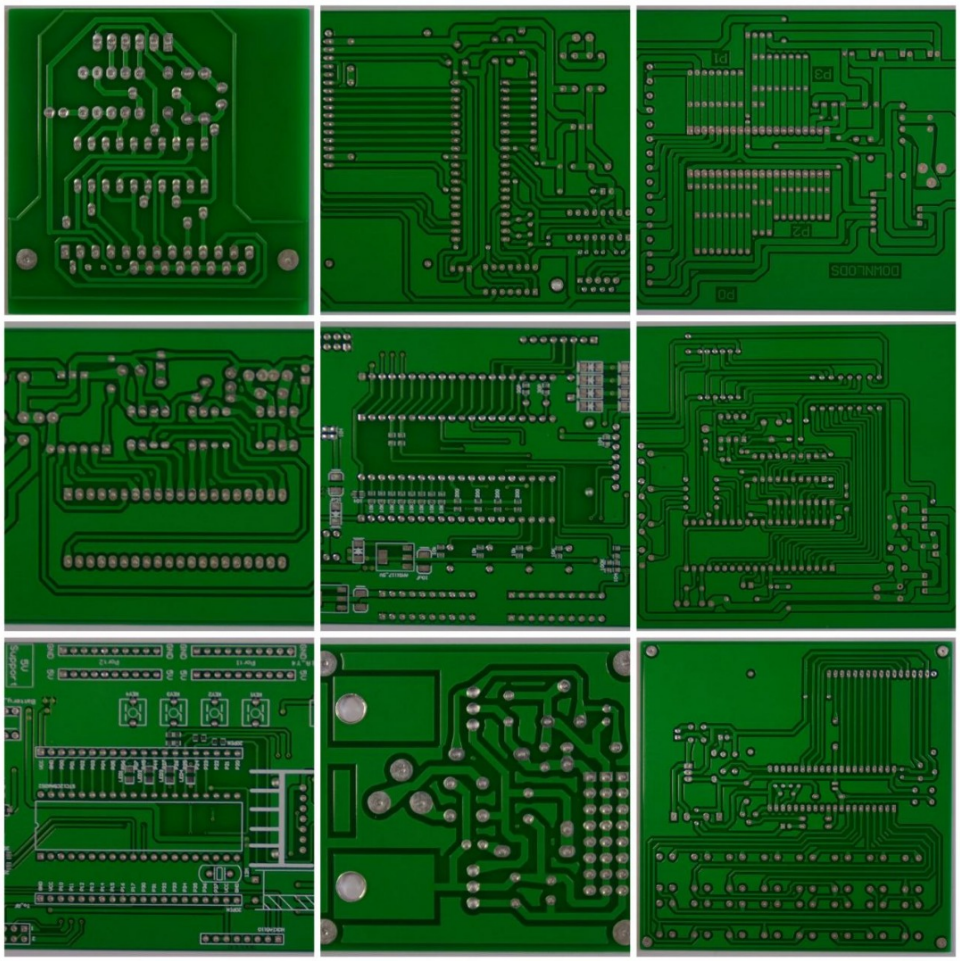


Figure 4.3: Collection of PCBs data.

4.4.1 Dataset Source

The dataset used in this work is the publicly available PCB Defects Dataset, hosted on Kaggle [2], and consists of 1,386 color images generated to simulate realistic industrial PCB defects. The dataset includes six common defect types such as missing holes, open circuits, shorts, and similar errors.

Each image is annotated with precise bounding boxes indicating the locations and categories of the defects, formatted according to the YOLO annotation structure.

4.5 Training, validation and test datasets

In Defect detection in PCBs, data splitting into training, validation and test sets is necessary for the development and evaluation of stable and effective models. The following splits are common in this situation:

Training Set: This is the set where the model is trained to identify defects in PCB images. Here, the model is taught to differentiate between the features of defective areas (for example, mouse bites, spurs, open circuits) and non-defective areas.

Validation Set: Validation set is utilized post the initial training for tuning the model's hyperparameters and to continue assessing the model's performance on new data. This enables us to pick the best-performing model.

Test Set: The test set is utilized to give the final performance of the model an objective evaluation after it has been trained. These images are excluded from the training and validation process entirely, providing an unbiased evaluation of the model's performance in accurately classifying PCB defects.

By appropriately dividing the dataset and utilizing the resulting subsets in succession, stable and consistent models for PCB defect detection can be created, with the additional confidence that the model performs well at generalizing to new and unseen data.

The dataset was split as follows :

Table 4.1: Data Split for Model Development

Set	**Approx. Percentage**	**Purpose**
Train	76% of the data	Used to train the model
Validation	19% of the data	Used to fine-tune performance and adjust parameters
Test	5% of the data	Used for final evaluation, completely unseen during training

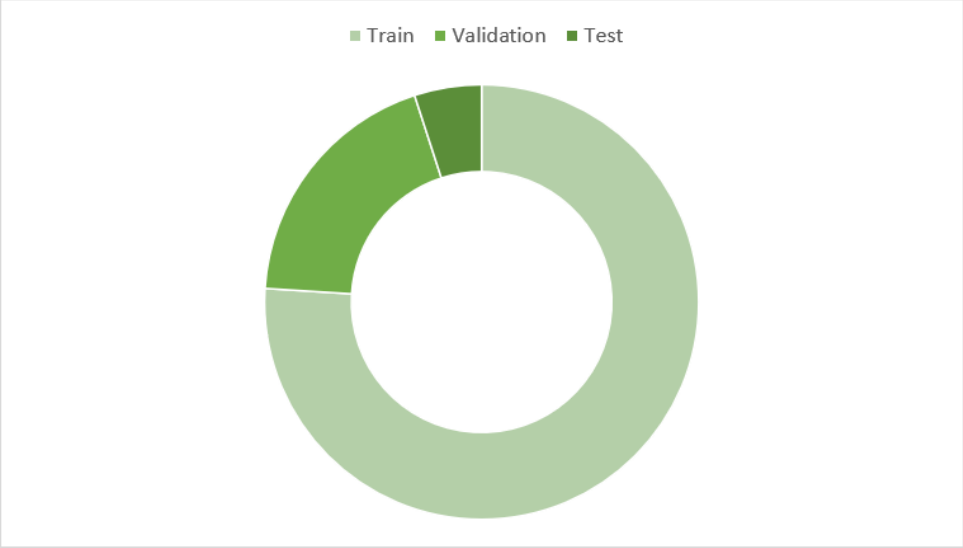


Figure 4.4: pie chart dataset.

4.6 Preprocessing of PCBs data

Preparing PCB images for YOLO-based defect detection begins with a standardization pipeline that ensures consistent, high-quality inputs. In practice, raw PCB photographs often exhibit variations in size, format, lighting, and noise. Preprocessing mitigates these issues by resizing images to a common dimension (often matching the model's fixed input size) and standardizing pixel intensities. Resizing "makes them uniform and reduces computational complexity", enabling faster training and inference. Likewise, intensity normalization (e.g. scaling 8-bit pixels to [0,1]) places all images on the same scale, which promotes stable gradients and faster convergence during training. In practice, modern YOLO pipelines automatically convert images to a consistent color format (RGB) and scale pixel values, ensuring that brightness or color disparities do not skew the learning process. Overall, these steps reduce the variability of the dataset and help the network more reliably converge to an accurate defect-detection model[28].

Resizing Images All input images were resized to a fixed resolution of 640×640 pixels, which is the recommended input size for most YOLO models. This step of resizing:

- guarantees uniform spatial dimensions throughout the dataset.
- Helps maintain a uniform input pipeline during training and inference.
- Reduces memory consumption and speeds up training while preserving defect-level vi-

sual features.

Conversion to RGB Format Since some of the original dataset images may be in grayscale or non-RGB formats, all images were converted to the standard 3 channel RGB format. YOLO models require images in RGB as they rely on color-based features in convolutional layers. This step guarantees compatibility with the model's input expectations.

Annotation Files

- Every image has an accompanying XML annotation file, which contains the following:
  - The defect class name (e.g., missing_hole, spur, etc.).
  - The bounding box coordinates (xmin, ymin, xmax, ymax) defined in absolute pixel values.
  - Additional metadata such as image size and defect difficulty

  This format is popular and simple to interpret.
- The annotations were then transformed into the necessary YOLO format for YOLO training, which included:
  - Class ID
  - Normalized center coordinates (x, y)
  - Normalized width and height

Data Cleaning To guarantee data quality and prevent training inconsistencies, data was cleaned by the following:

1. Deletion of faulty or unreadable
2. Excluding images which don't have
3. Verifying that each image was supported by a matching and properly formatted annotation file.
4. Verifying whether bounding boxes lay within image limits and were not empty.

This was done as a way of guaranteeing that the model was being trained using high quality, properly labeled data, to prevent overfitting or labeling mismatch issues.

4.7 Data Augmentation

The most widely talked-about data preprocessing technique is data augmentation. Data augmentation is where you artificially add to the size of the dataset by generating transformed copies of images. Augmenting your data can assist in combating overfitting and improving model generalization[28].

Here are some other benefits of data augmentation:

Creates a More Robust Dataset: Data augmentation can make the model more robust to variations and distortions in the input data. This includes changes in lighting, orientation, and scale.

Cost-Effective: Data augmentation is a cost-effective way to increase the amount of training data without collecting and labeling new data.

Better Use of Data: Every available data point is used to its maximum potential by creating new variations.

4.7.1 Data augmentation techniques

For YOLO11, dataset augmentation can be directly configured through the respective.yaml file. Users can configure a wide range of augmentation parameters, ranging from random cropping, horizontal and vertical flipping, and angle rotation, to image distortion. All of which help enhance model generalization as a whole, thus achieving higher accuracy for PCB defect detection across different production conditions. Rotational augmentation is applicable for PCBs, as boards can be oriented differently for inspection. For each class of defects, e.g., missing hole, spur, or short circuit, originals are rotated by varying angles, generating new training samples and not changing the structural properties of the defects. Synthetic samples balance underrepresented classes and introduce spatial variation, avoiding overfitting.

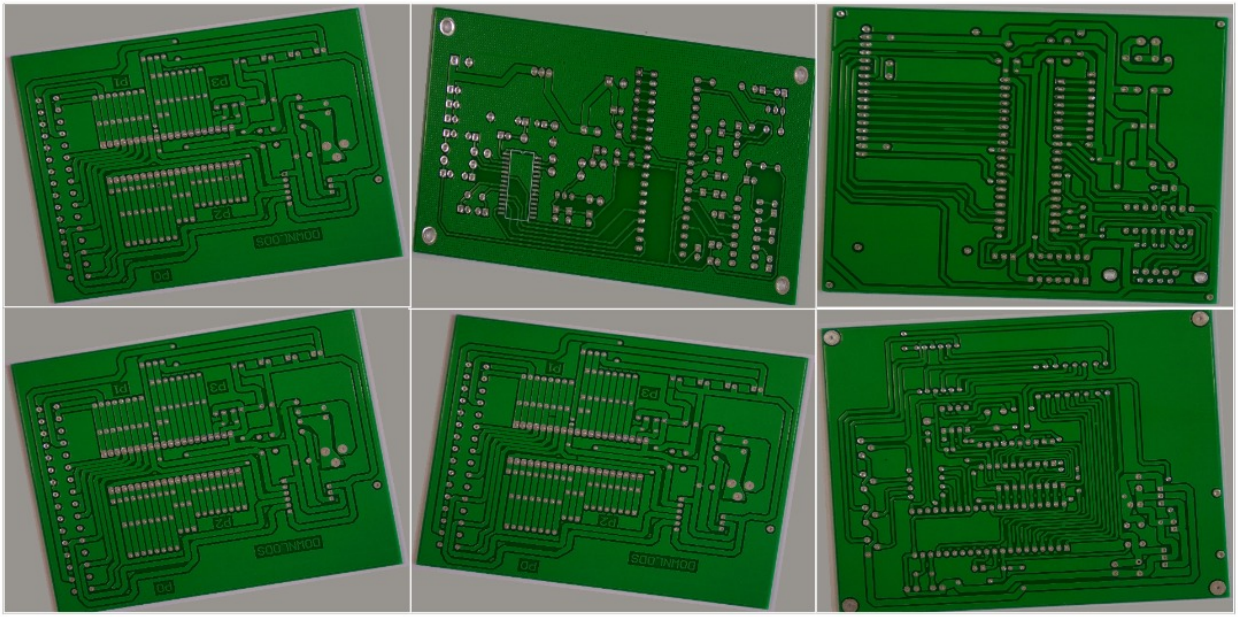


Figure 4.5: Dataset after rotation.

4.8 Proposed model architecture

4.8.1 Overview

YOLO11 is the latest in the Ultralytics YOLO series of real-time object detectors, continuing to push the boundaries of state-of-the-art accuracy, speed, and efficiency. Building on the already impressive advancements in previous versions of YOLO, YOLO11 continues to improve the architecture and training techniques, providing a very effective tool for a wide range of computer vision tasks.

4.8.2 YOLO11 Architecture

YOLO11 Architecture is an upgrade over YOLOv8 architecture with some new integrations and parameter tuning[6].

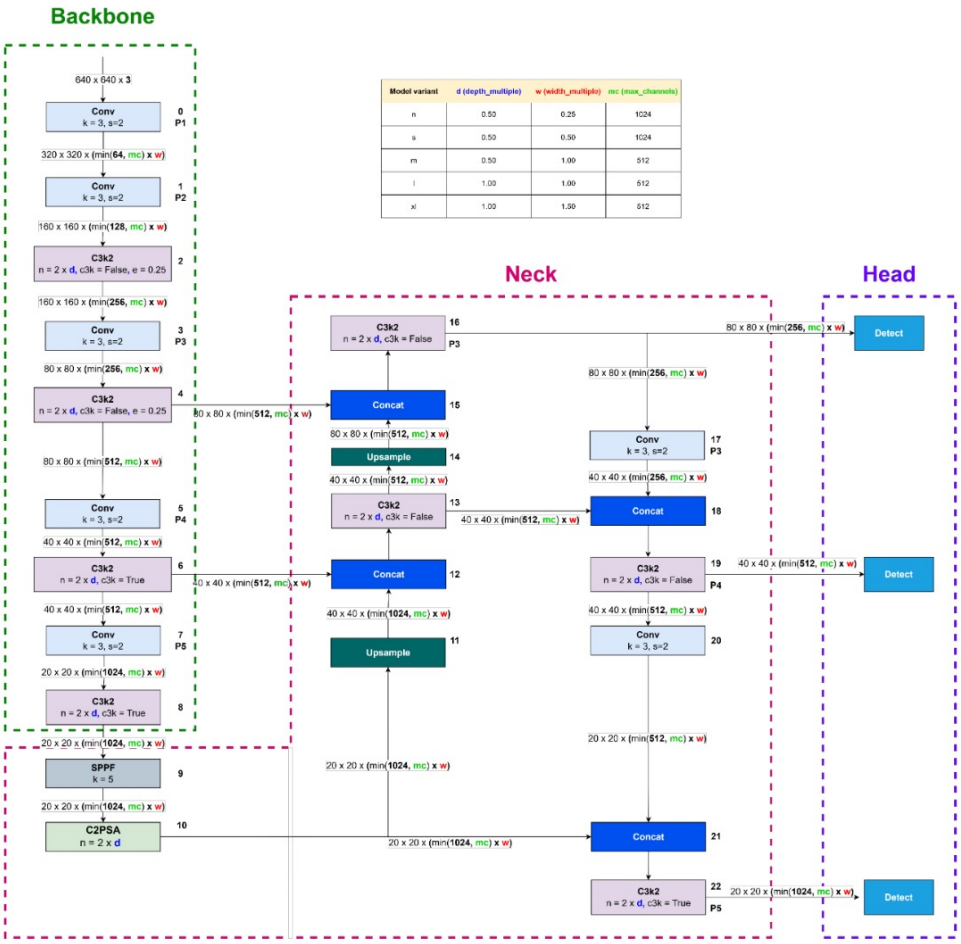


Figure 4.6: YOLO11 Architecture (Adapted from [9]).

More importantly, YOLO11 contains some of the advanced components, a SPPF (Spatial Pyramid Pooling – Fast) layer to aggregate context across multiple scales reliably, and a C2PSA (Cross-Convolutional Parallel Spatial Attention) that helps to concentrate on the informative locations in the feature maps. They improve feature extraction that enables to detect small PCB defects, and are particularly advantageous for small and subtle defects that can only be detected by experts.

Figure 4.6: Example YOLO-based PCB defect detection architecture, divided into backbone, neck, and head components. The backbone (left) extracts multi-scale features from the input image. The neck (center) fuses features of different resolutions to provide rich, multi-scale representations. The head (right) comprises convolutional layers that produce the final detection feature maps at multiple scales (e.g., for small, medium, large objects), with each predicting bounding boxes and class probabilities for defects.

Above the backbone, YOLO11 utilizes a neck network to combine feature maps and ensure

that meaningful information at different scales is preserved for detection. In YOLO-series models a Path Aggregation Network (PAN) or similar Feature Pyramid Network is commonly used; YOLO11 continues this approach by aggregating features from deep (low-resolution) layers up to shallow (high-resolution) layers.

This bi-directional fusion allows the detector to use both the coarse context-rich features (helpful for identifying a region as defective) and high-resolution detailed features (for locating the locations of small defects). The neck produces a set of fused feature maps at multiple scales, and the detection head in YOLO11 creates the output predictions. The YOLO11 head is a one-stage dense predictor that applies sets of convolutional layers to each feature map scale (large, medium, small receptive fields) and directly predicts bounding boxes, defect classes, and confidence scores for any object (defect) in those cells.

The head has some partially decoupled classification and localization subtasks which allows better accuracy (similar to YOLOv8 and YOLOX). Each scale has several prediction layers that detect defects of multiple sizes, which is important when manufacturing PCBs that may contain both relatively large defects and very little defects. The predicted output is a set of bounding boxes and each one is associated with a defect category and score probability that indicates the potential for a true defect. One of several reasons that YOLO11 is suitable for PCB defect detection is its ability to detect small objects with great accuracy. Some PCB defects like micro solder shorts or hairline open circuits are small in pixel size; YOLO11 has multi-scale feature maps and attention mechanisms that seek to include such details. Moreover, because YOLO is designated as a one-stage architecture, inference is fast–an important characteristic in PCB manufacturing, where hundreds of boards may need inspecting per hour. The model is measured not only by its accuracy but also by its inference speed (frames per second) and computational cost (GFLOPs) and YOLO models are well suited for speed-accuracy trade-offs[34].

4.8.3 Evaluation Phase

The YOLOv11 model was trained on an annotated image dataset of PCB images to test and evaluate the PCB defect detection model. Training took 200 epochs and adopted a systematic data split approach. That is, 95% of data was employed for training and validation purposes and 5% for final testing purposes. In the training part, 80% of the images were used for training,

and 20% for validation. This division method prevents the model from overfitting and ensures good generalization.

The model was trained on Google Colab with GPU to achieve the highest convergence speed and performance. The model was then validated on the unseen test set after training to test its potential to correctly identify PCB defects. The performance was measured using standard deep learning metrics[22], including:

1. **The Precision:** Precision is calculated as the proportion of accurate positive predictions to all positive predictions. Precision expresses how the model can minimize errors in positive observation classification.

Precision(P) = TP/ (TP +FP)

2. **Accuracy:** Estimates how often a classification model correctly predicts positively or negatively. Accuracy provides a general measure of the performance of the model taking true positives and true negatives into account.

Accuracy = TP +TN /(TN+TP +FP +FN)

3. **Recall:** the ability of a model to classify positive images in the proper way. It is the proportion of true positive tumour images to all the tumour images. Due to its sensitivity, the model performs well to classify tumour images since it reduces false negatives, i.e. tumorous.

Recall = TP / (TP +FN)

4. **F1 score:** It is calculated by finding the harmonic mean of precision and recall. Accuracy finds the percentage of right outcomes over anticipated outcomes, while recall finds the percentage of right outcomes over total actual outcomes.

F-Score = 2(Precision * Recall) / (Precision+ Recall)

Where:

TP (True Positive): Defects that are correctly identified by the model.

TN (True Negative): Non-defective cases correctly classified as normal.

FP (False Positive): Normal cases wrongly identified as defective.

FN (False Negative): Defective cases not detected by the model.

These metrics allow for a general evaluation of the performance of the model, reflecting its capacity to distinguish faulty and non-faulty PCB images. The interpretation of these results is crucial to the assurance of the reliability of the model for practical quality control use.

4.9 Results and discussion

The results and discussion phase seeks to assess the extent to which the model can detect and classify defects on PCB images (printed circuit boards). Important metrics such as precision, recall, and loss will provide clarity with some plots, and confusion matrices. The results displayed will show the model's ability to generalize and perform to detect defects on PCB images with different conditions. Below are the most important evaluation visualizations from the training and testing phases.

4.9.1 Confusion Matrix Analysis

The confusion matrix reflects the performance of the YOLOv11 model for seven classes of defects: mouse_bite, spur, open_circuit, short, missing_hole, spurious_copper, and one class of background. Every cell includes the number of predictions that were made by the model for a specific true label (rows) versus predicted label (columns).

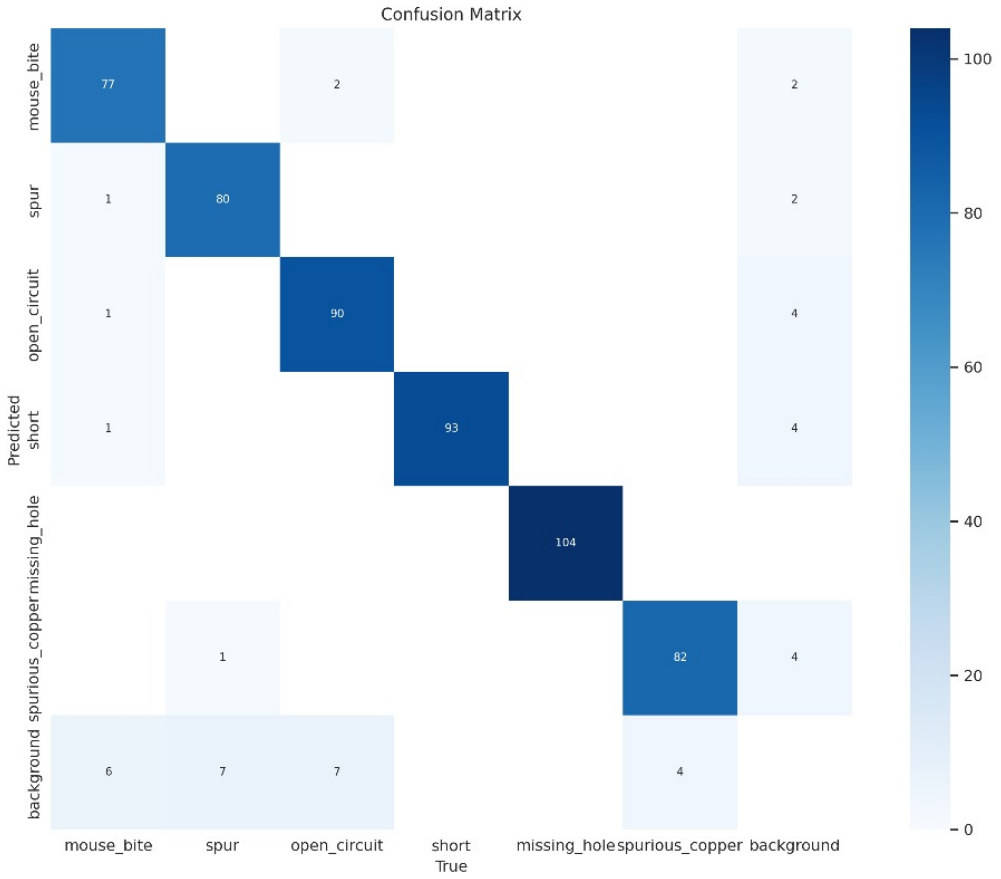


Figure 4.7: Confusion matrix.

4.9.1.1 Observations

High diagonal values: The matrix has high diagonal dominance, indicating high accuracy in the classification of most defect types:

1. Missing_hole : 104 correct
2. Short : 93 correct
3. Open_circuit : 90 correct
4. Spur : 80 correct
5. Mouse_bite : 77 correct
6. Spurious_copper : 82 correct

4.9.1.2 Low misclassification rate

- A number of the testing samples were incorrectly assigned in each of the categories (primarily ≤ 4). For example, mouse_bite was wrongly classified as spur, 2 occasions, and as background, 2 occasions.
- Background class produced minor confusion, with 6, 7, and 7 false positives for mouse_bite, spur, and open_circuit, respectively.

4.9.1.3 Notable Misclassifications

- Background class is the most frequent cause of false positives. This type of error suggests that the model is prone to detect too many defects in clean regions in certain cases.
- There is minor confusion among closely linked categories of defects like spur and mouse_bite, which may be structurally similar.

4.9.1.4 Overall Impression

- It is linked with high classification power as it classifies the majority of the real labels appropriately.
- Few and localized misclassifications indicate that the model is well trained and generalizes well.
- Each class's precision and recall would be greater than 90%, missing hole scoring nearly perfectly.

4.9.2 Training and Validation Loss Analysis

The train and/validation loss of the three main components of the YOLOv11 loss function over 200 epochs is shown above :

1. Box Loss (Localization loss)
2. Classification Loss (cls_loss)
3. Focal Distribution Loss (dfl_loss)

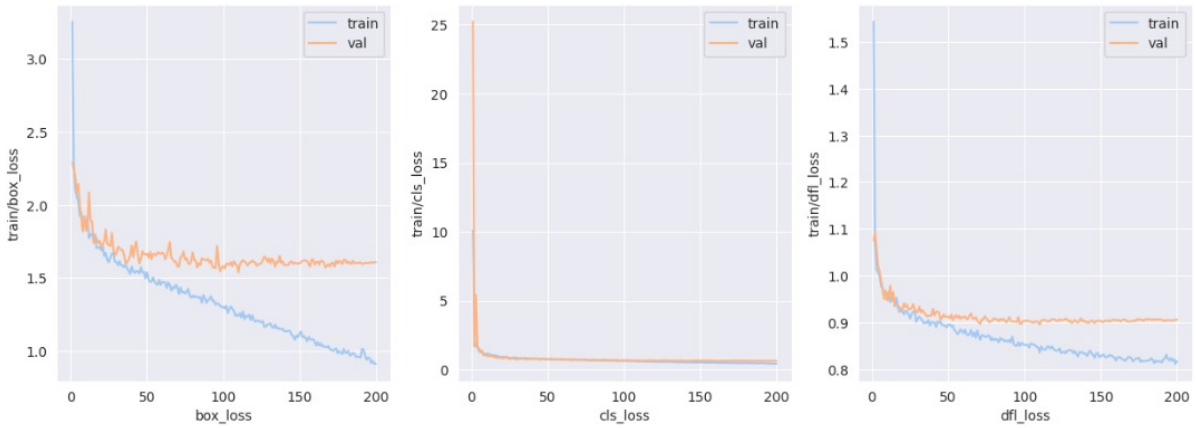


Figure 4.8: Loss Function Convergence Analysis During Training.

All graphs show how the model learns over time by watching the curves of the training and validation losses.

4.9.2.1 Box Loss (Left plot)

- Observation: Both the training and validation losses in the box decrease gradually in the first 50 epochs.
- After 50 epochs, train loss decreases smoothly, and it dips below 1.0 in epoch 200.
- The validation loss is leveling out around 1.5 with slight fluctuations.
- The model is learning object localization well. The small difference in training and validation loss is a sign of little overfitting, but overall performance is consistent.

4.9.2.2 Classification Loss (Middle plot)

- Observation: Sudden dip in training and validation classification loss from >25 to <1.0 during the first 10–20 epochs, with stabilization afterwards.
- That implies the model is learning the defect types correctly in the initial stages of training. Convergence and closeness between train and val curve indicate there is a good generalization.

4.9.2.3 DFL Loss (Right plot)

- Observation: DFL is decreasing systematically both in training and in validation.

- Lower-training curve graphs below validation from epoch 50
- DFL means refined localization (no anchors), and the model's learning curve is good. There is clothoidation leveling out around 0.9, and that is sufficient.

4.9.3 Overall Evaluation

No indication of severe overfitting or underfitting. All loss terms exhibit significant convergence. The performance saturates around epoch 150, which shows the model is well trained and generalizes well for unseen validation data.

4.9.4 Analysis of Precision, Recall, and Confidence Metrics

The metrics plotted here are useful for gauging the performance of the YOLOv11 model trained on all six classes of PCB faults, namely, mouse_bite, spur, open_circuit, short, missing_hole, and spurious_copper. In every plot, the potential of the model for accurate predictions with respect to confidence thresholds and class discrimination is evident.

4.9.4.1 Precision-Confidence Curve

- Field of plots in terms of prediction confidence.
- A very high precision of ($\approx$1.0) is observed for all classes up to the cutoff confidence of 0.88, beyond that there are slight fluctuations.
- The curve shows that the model rarely produces false positives when its confidence is high, it reaffirms the reliability of the detections.

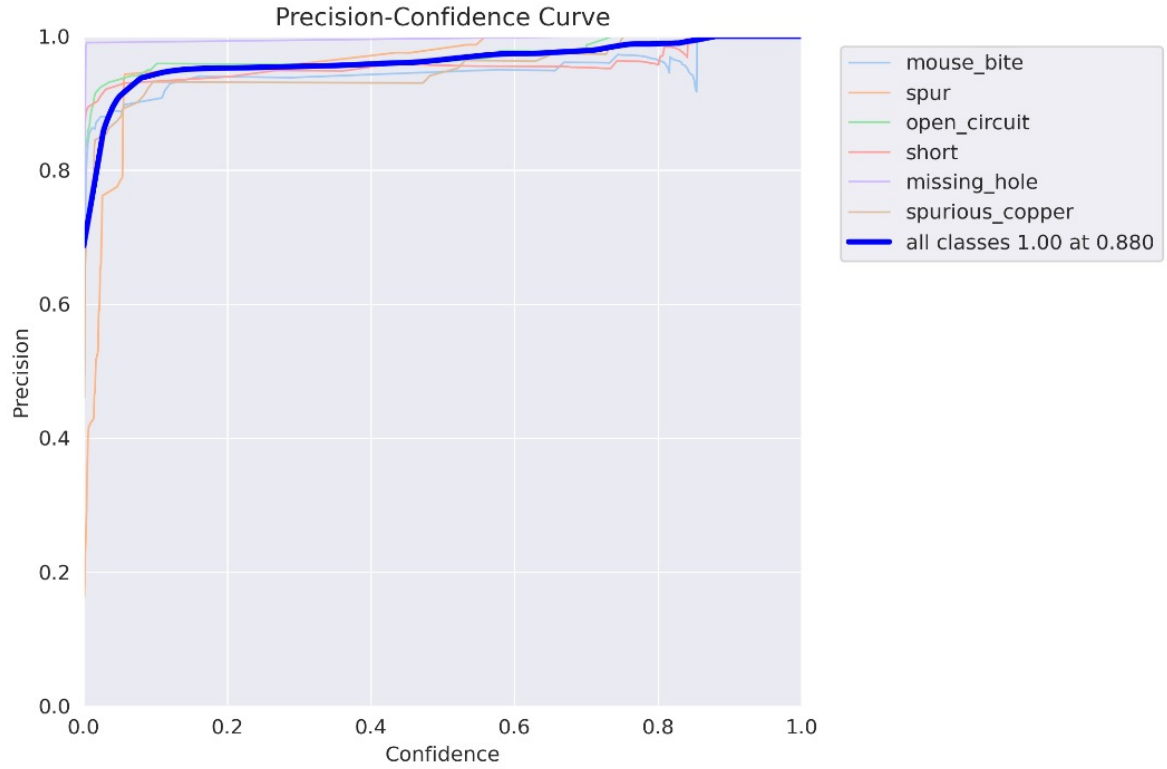


Figure 4.9: Precision-Confidence Curve.

4.9.4.2 Recall-Confidence Curve

- Recall (sensitivity) curve plot with respect to confidence
- Recall is over 0.9 until the confidence is higher than 0.8, when there is a drop.
- This means that the model captures most of the true positives for modest confidence but starts missing out on some as the threshold increases.

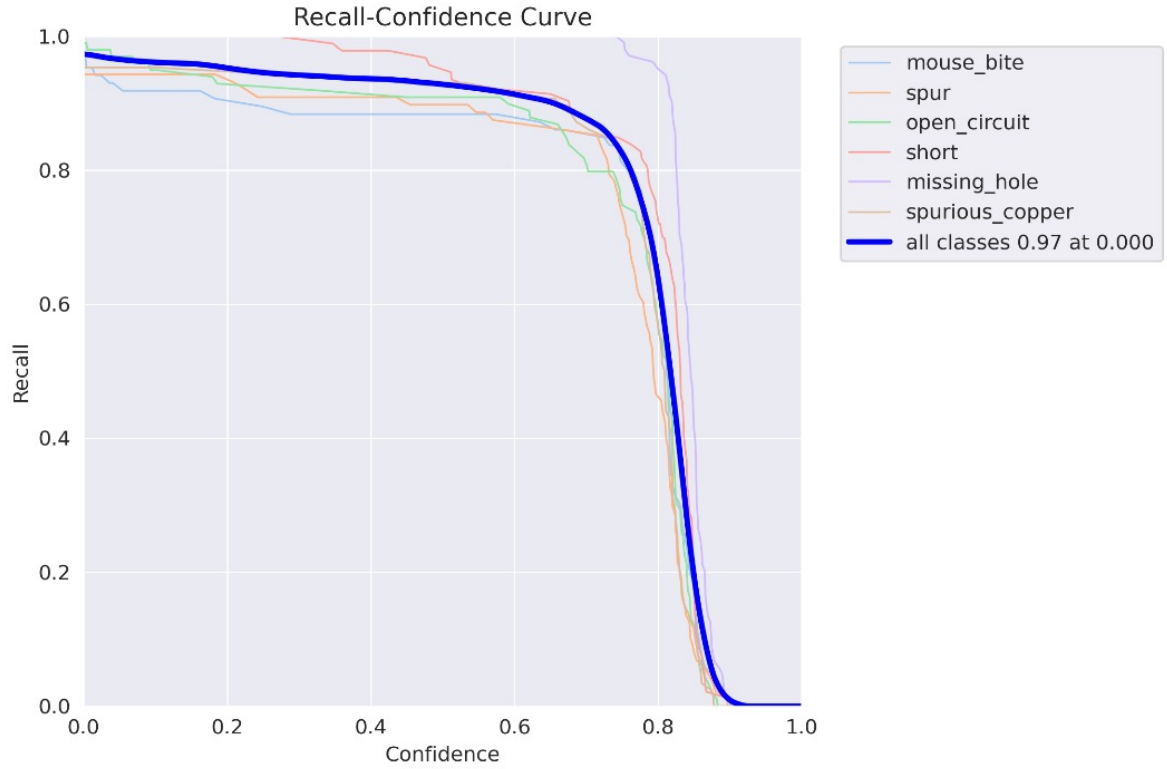


Figure 4.10: Recall-Confidence Curve.

4.9.4.3 Precision-Recall (PR) Curve

Shows the precision and recall trade-off for each class. Overall, each class performs well, according to the area under the curve (AUC) values:

- Missing hole: 0.995
- Open circuit: 0.985
- Short: 0.978
- Spurious copper: 0.961
- Spur: 0.947
- Mouse bite: 0.943

Overall mean Average Precision (mAP@0.5) for each class is 0.968, which indicates very high quality detections.

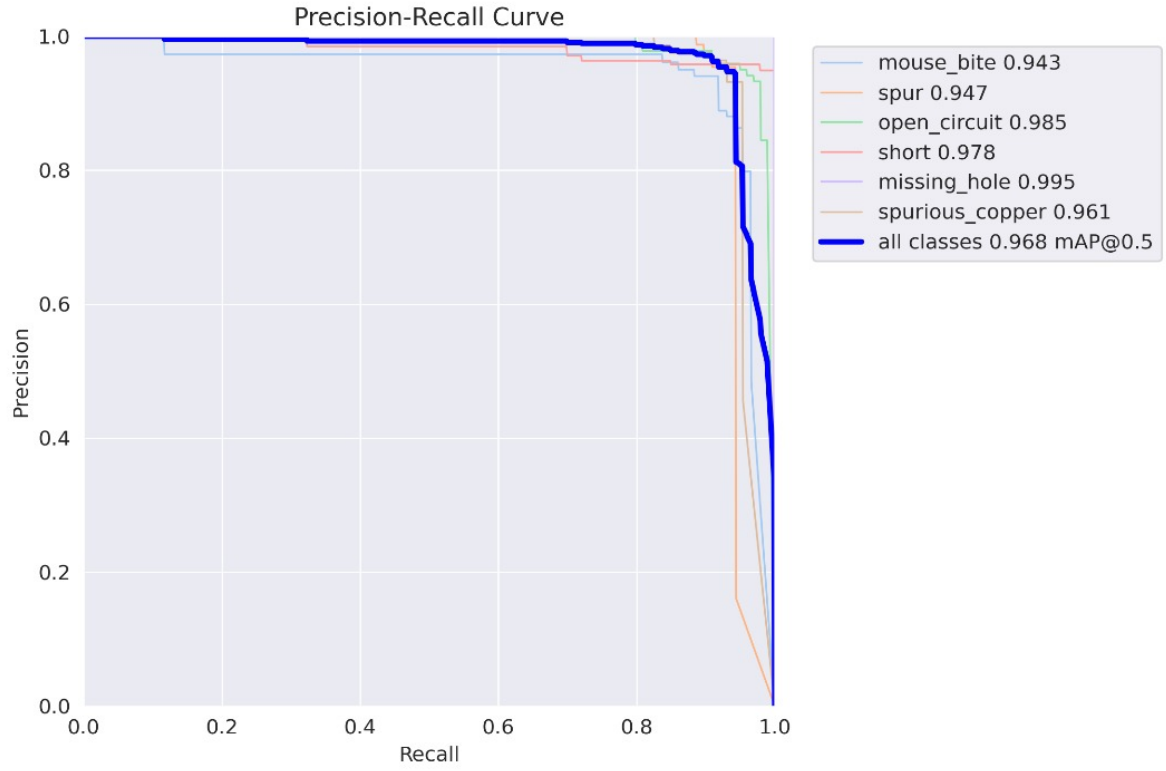


Figure 4.11: Precision-Recall (PR) Curve.

4.10 Prediction

The prediction stage is an essential step in measuring the actual performance of the trained deep learning model. In this research, the YOLOv11-based model was employed in detecting six common types of PCB faults, namely, missing hole, open circuit, short, mouse bite, spur, and spurious copper.

In this stage, the model was tested on new, previously unseen images from both the validation and the test set, that were not part of the training set. It was accurately localizing the faults by predicting closely bounding boxes around faulty regions and classifying the objects with the proper class labels. Visual outcomes demonstrate the capacity of the model in detecting multiple faults in one PCB image, vouching for its reliability in handling complex and overlapping fault patterns in real industrial applications.

In addition, the model also maintains high discrimination capacity between the different classes of defects, i.e., fine-grained or visually analogous, in accordance with the learned visual representation's fidelity. Predictions also have high categorical and spatial consistency between actual and predicted defects, indicating stable localization and classifying performance.

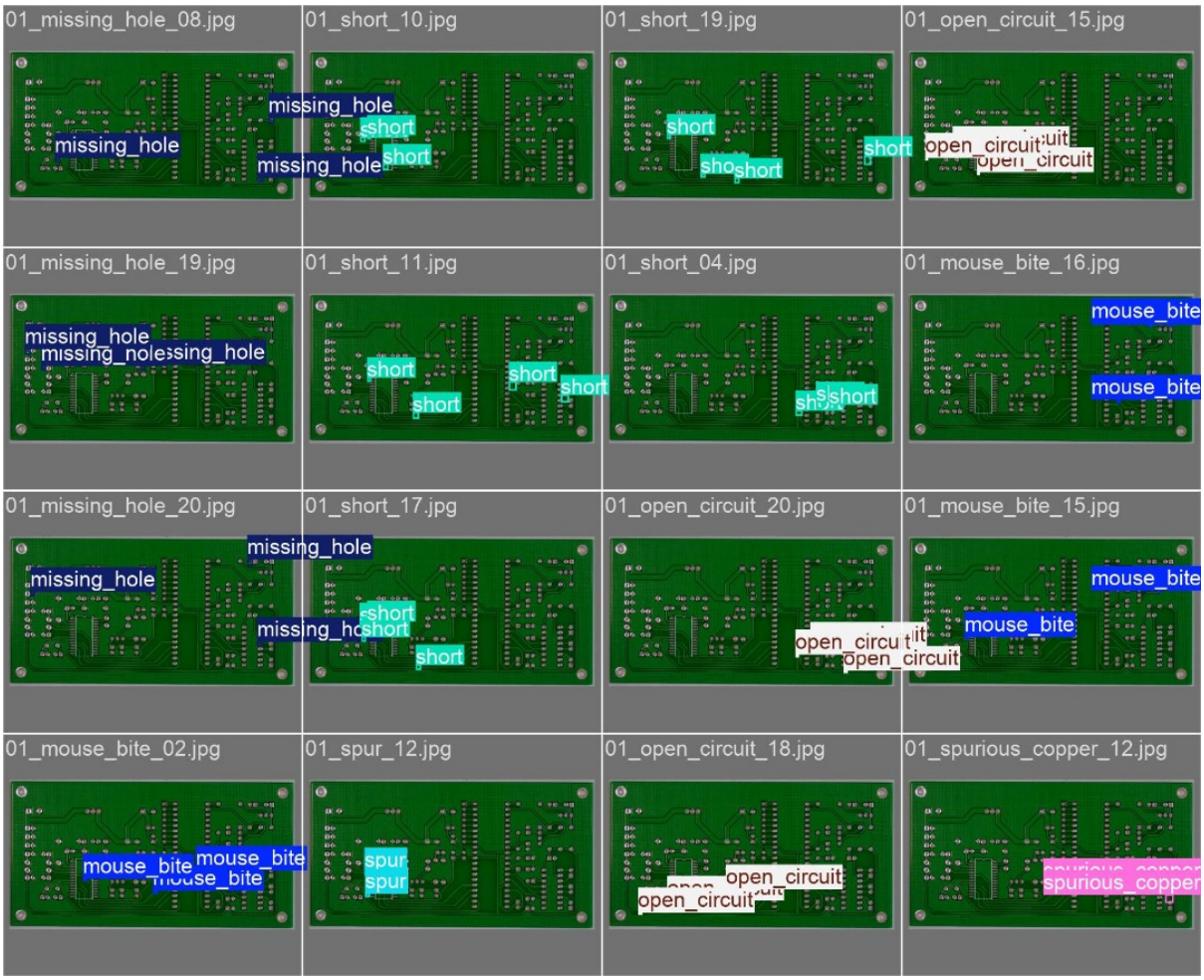


Figure 4.12: Labeled PCB Defect Samples for Model Validation.

4.11 Conclusion

Our study developed a model based on deep learning focused on automated defect detection in printed circuit boards (PCBs). By following a systematic process, and by bringing together quality data collection, reliable pre-processing, and a strong YOLO11 architecture, we achieved a model that produced very high quality defect detection on many varieties of PCB defects.

Through exemplars of machine learning principles, the model was trained and validated through rigorous structured partitioning of the dataset, monitoring for loss function, and by tracking accuracy, recall and mAP metrics as indicators of how well it was performing. In the overall results, the model was able to correctly identify and classify defects accurately, including on previously unmet samples, suggesting it could represent an effective tool for real-time industrial inspection.

This project illustrates the application of contemporary deep learning methods in the electronics manufacturing industry to improve reliability, minimize human error, and enhance automated quality control systems.

General Conclusion

The quick evolution of the microelectronics and the increase of the complexity of PCBs makes it necessary to apply smart and automated procedures to produce high-quality pieces and a fast defect identification. This dissertation aims to build a YOLOv11-based defect detection system using deep learning for real-time PCB, implemented into a realistic application working within an industrial environment.

The system demonstrated high accuracy in the detection and classification of various types of defects and is a tangible move towards the automation of quality control processes. Its flexibility and extensibility also render it an easily scalable solution that can evolve with evolving industrial requirements.

Strengths:

1. High accuracy in detecting six primary PCB defect types.
2. Employment of YOLOv11, which is acclaimed for real-time detection efficacy.
3. Scalability for future improvements and industrial integration.
4. Compatibility with both desktop and mobile deployment for flexible use.

Limitations and Criticisms:

- The system performance depends on the nature of input images that may be compromised in poor lighting or visual noise.
- The current model can suffer from delays when handling large numbers of images in high-speed environments.

- The model requires intensive testing using real-world industrial production data to make it robust and reliable.

Future Directions:

- Development of integration with high-speed cameras and sensors for real-time production line inspection.
- Enlargement of the system to support other defect types or other electronic component inspections.
- Development of visual analytics tools to provide engineers the capability to investigate root causes and take data-driven corrective action.
- Application of the system as an input source for AI-driven quality optimization in industrial processes.
- Development of integration of mechanism for ongoing learning to enable automated update based on new observed defect patterns.

Bibliography

[1] V. A. Adibhatla, R. Krishna, and S. S. Kumar. “Real-time defect detection for printed circuit boards using YOLO”. In: *2018 IEEE International Conference on Advanced Manufacturing (ICAM)*. IEEE, 2018, pp. 370–375.

[2] Alfiya Akhatova. *PCB Defects*. Accessed: 2025-06-15. 2021. URL: https://www.kaggle.com/datasets/akhatova/pcb-defects.

[3] Abhiroop Bhattacharya and Sylvain G. Cloutier. “End-to-end deep learning framework for printed circuit board manufacturing defect classification”. In: *Scientific Reports* 12.1 (2022), p. 12480.

[4] Xing Chen et al. “A Comprehensive Review of Deep Learning-Based PCB Defect Detection”. In: *IEEE Access* 11 (2023).

[5] Lang Du and Zhenzhen Lv. “SEPDNet: simple and effective PCB surface defect detection method”. In: *Scientific Reports* 15 (2025), p. 10919.

[6] Ankan Ghosh. *YOLOv11: Redefining Real-Time Object Detection*. LearnOpenCV. 2024. URL: https://learnopencv.com/yolo11/#aioseo-yolo11-architecture-and-whats-new-in-yolo11 (visited on 05/23/2025).

[7] Xixi Han et al. “Defect identification of bare printed circuit boards based on Bayesian fusion of multi-scale features”. In: (2024).

[8] Priyanto Hidayatullah and Refdinal Tubagus. “YOLO11 Architecture - Detailed Explanation”. In: (2024). Oct. 28, 2024.

[9] Priyanto Hidayatullah and Refdinal Tubagus. “YOLO11 Architecture - Detailed Explanation”. In: (Oct. 28, 2024).

[10] Jaeseok Jang, Qing Tang, and Hail Jung. “PCB Defect Classification with Data Augmentation-Based Ensemble Method for Sustainable Smart Manufacturing”. In: *Sustainability* 16.23 (2024), p. 10417.

[11] Y. Lee and H. Jeong. “Automated Optical Inspection (AOI) Based on IPC Standards”. In: (2021).

[12] Y. Li, S. Wang, and J. Wu. “Deep Learning for Printed Circuit Board Defect Detection: A Review”. In: *IEEE Access* 9 (2021), pp. 128905–128918.

[13] Yazhou Li et al. “Research on PCB defect detection algorithm based on LPCB-YOLO”. In: *Frontiers in Physics* 12 (2025), p. 1472584.

[14] JiaYou Lim et al. “A deep context learning based PCB defect detection model with anomalous trend alarming system”. In: *Results in Engineering* 17 (2023), p. 100968.

[15] C. Lin, J. Wang, and Y. Liu. “The effect of fatigue on visual inspection performance in electronics manufacturing”. In: *International Journal of Industrial Ergonomics* 66 (2018), pp. 145–152.

[16] Henri Michel. “Data Science, IA, Programmation”. In: (31 MARS 2022).

[17] Millennium Circuits Limited. *PCB Guide: Why Do PCBs Fail?* https://www.mclpcb.com/pcb-guide/. Accessed: 2025-05-23. Dec. 2024.

[18] mpe-electronics. *Miniaturisation Of Electronics: The Complete Guide.* 2024. URL: https://www.mpe-electronics.co.uk/2024/03/12/miniaturisation-of-electronics-the-complete-guide.

[19] MUO. “A Beginner’s Guide to Kaggle for Data Science”. In: (2023-04-17).

[20] Van-Truong Nguyen et al. “Deep learning-enhanced defects detection for printed circuit boards”. In: *Results in Engineering* 25 (2025), p. 104067.

[21] D. Park, H. Kim, and Y. Cho. “Deep learning-based defect detection in printed circuit boards using convolutional neural networks”. In: (2022).

[22] David Martin Powers. “Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation”. In: *Journal of Machine Learning Technologies* 2.1 (2011), pp. 37–63.

[23] Python. *Why Python, Python Releases Wind, 2021.* 2021. (Visited on 2025).

[24] Andrew Reardon. “Electrical Engineer at Analog Photonics”. In: ().

[25] Rahul Shashikanth. *PCB Drilling: The Dos and the Don’ts*. https://www.protoexpress.com/blog/no-chilling-when-it-comes-to-pcb-drilling/. Accessed: 2025-05-23. Dec. 2020.

[26] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2010. DOI: 10.1007/978-1-84882-935-0.

[27] Ultralytics. *Data Augmentation using Ultralytics YOLO*. 2025. URL: https://docs.ultralytics.com/guides/yolo-data-augmentation/ (visited on 05/22/2025).

[28] Ultralytics. *Data Preprocessing Techniques for Annotated Computer Vision Data*. 2025. URL: https://docs.ultralytics.com/guides/preprocessing_annotated_data/ (visited on 05/22/2025).

[29] Ultralytics. *Instance Segmentation - Ultralytics YOLO Docs*. 2025. URL: https://docs.ultralytics.com/tasks/segment/ (visited on 05/22/2025).

[30] Ultralytics. *Performance Metrics Deep Dive - Ultralytics YOLO Docs*. 2025. URL: https://docs.ultralytics.com/guides/yolo-performance-metrics/ (visited on 05/22/2025).

[31] Ultralytics. *Ultralytics YOLO11*. 2025. URL: https://docs.ultralytics.com/models/yolo11/ (visited on 05/22/2025).

[32] Peng Wei. Weibo Huang. "A PCB Dataset for Defects Detection and Classification". In: (2019).

[33] Gaoshang Xiao, Shuling Hou, and Huiying Zhou. "PCB defect detection algorithm based on CDI-YOLO". In: *Scientific Reports* 14 (2024), p. 7351.

[34] Aston Zhang et al. *Dive into Deep Learning*. Cambridge University Press, 2023.

[35] G. Zhang et al. "End-to-end deep learning framework for printed circuit board defect detection". In: *Scientific Reports* 12.1 (2022).

[36] L. Zhou, Z. Tang, and C. Wang. "Defect analysis in multilayer PCB soldering processes". In: *Microelectronics Reliability* 110 (2020).

Abstract

Printed Circuit Boards (PCBs) are basic to today's electronic system and as result, their defect-free production is essential for the function and reliability of electronic devices. Traditional manual inspection methods are no longer enough due to the complexity and reduced size of the boards. In the context of the project, deep learning is applied to the automated recognition of the defects, where YOLOv11 object detection architecture is the algorithm used to find the six most frequent defects in the PCB images. The data acquisition is the main stage of the project, and it is followed by the preprocessing, annotation transformation, and data augmentation, the processes which contribute to increasing the training diversity. The model has been presented, using a carefully selected dataset, as a good candidate to have high precision, recall, and mean Average Precision (mAP) metrics. Additionally, the mobile application was launched with the model proving its feasibility and being ready for server-side inference in real-time defect detection. The experimental results validate the prototype system as a solution to the defect detection problem in electronics. The system is also a scalable and practical quality control mean in electronics production besides being accurate and reliable.

Résumé

Les circuits imprimés (PCB) sont essentiels au système électronique actuel et, par conséquent, leur production sans défaut est essentielle pour le fonctionnement et la fiabilité des appareils électroniques. Les méthodes traditionnelles d'inspection manuelle ne suffisent plus en raison de la complexité et de la taille réduite des planches. Dans le cadre du projet, l'apprentissage profond est appliqué à la reconnaissance automatisée des défauts, où l'architecture de détection d'objets YOLOv11 est l'algorithme utilisé pour trouver les six défauts les plus fréquents dans les images PCB. L'acquisition de données est la principale étape du projet, suivie par le prétraitement, la transformation des annotations et l'augmentation des données, processus qui contribuent à accroître la diversité de la formation. Le modèle a été présenté, à l'aide d'un ensemble de données soigneusement sélectionnées, comme un bon candidat pour avoir des mesures de précision élevée, de rappel et de précision moyenne (mAP). De plus, l'application mobile a été lancée avec le modèle prouvant sa faisabilité et étant prêt pour l'inférence côté serveur dans la détection des défauts en temps réel. Les résultats expérimentaux valident le système prototype comme solution au problème de détection des défauts en électronique. Le système est également un moyen de contrôle de qualité évolutif et pratique dans la production électronique, en plus d'être précis et fiable.

الملخص

تُعتبر لوحات الدارات المطبوعة حجر الأساس في الأنظمة الإلكترونية الحديثة، ويُعد ضمان جودتها وخلوها من العيوب أمرا حيويا لأداء الأجهزة. ومع ازدياد تعقيد هذه اللوحات وتصغير مكوناتها، لم تعد تقنيات الفحص اليدوي التقليدية كافية لمواكبة متطلبات الإنتاج. يقدّم هذا العمل نموذجا ذكيا قائما على التعلم العميق لاكتشاف عيوب الدارات المطبوعة تلقائياً، باستخدام نموذج يولو ١١ المتقدّم للكشف اللحظي عن الكائنات.

يشمل المشروع مراحل جمع البيانات، المعالجة المسبقة، تحويل بيانات التعليم ،التعليقات التوضيحية، وتطبيق تقنيات تعزيز البيانات لزيادة تنوّعها. تم تدريب النموذج وتقييمه على مجموعة بيانات منسقة، وأسفرت النتائج عن أداء متميز من حيث الدقة والاسترجاع ومتوسط الدقة. كما تم دمج النموذج ضمن تطبيق موبايل يتيح الكشف الفوري عن العيوب من خلال إرسال الصور إلى الخادم وتحليلها.

تُظهر النتائج كفاءة النظام المقترح، مما يجعله حلاً قابلاً للتطبيق في بيئات التصنيع الصناعية ويُمهّد الطريق لأتمتة عمليات فحص الجودة في خطوط إنتاج الإلكترونيات.