Software Requirement Specifications
Automated PCB Defect Detection Using Deep Learning

Submitted by

Nimra Akram (F22BINFT1E02045)

Submitted to

Dr. Musarat Karim

Department of Information Technology

Faculty of Computing

The Islamia University of Bahawalpur

Meeting Details

Sr No	Details	Date	Supervisor Signature

Table of Contents

Summary

This Software Requirements Specification (SRS) describes the requirements for the project 'Automated PCB Defect Detection Using Deep Learning'. The project aims to automate the quality inspection process of Printed Circuit Boards (PCBs) by developing a deep learning–based system capable of identifying defects such as open circuits, shorts, mousebites, missing holes, spurs, and copper defects.

1. Introduction

Printed Circuit Boards (PCBs) are a core component in almost all electronic devices.During mass production, PCB defects such as shorts, opens, missing holes, copper excess, and spurs can occur, resulting in faulty devices, financial loss, and safety risks. Traditional PCB inspection is done manually or through rule-based vision systems that often fail to detect complex defects.

The **Automated PCB Defect Detection System** aims to replace human-dependent visual inspection with an intelligent deep learning system. This module uses YOLOv8 for defect detection and classification and serves as a building block for a fully automated Industry-4.0-based inspection pipeline. The system takes PCB images as input and highlights defects through bounding boxes and labels.

1.1 Purpose

The purpose of this project is to automate PCB quality inspection using a deep learning model capable of identifying multiple defect types. The system ensures fast, accurate, and reliable inspection to reduce human error and improve production efficiency.

1.2 Scope

Included in the system:

I. Training a YOLOv8 model using the PCB Defects Dataset
II. Detecting and classifying PCB defects (open, short, missing hole, mousebite, spur, copper)
III. Preprocessing and splitting dataset into train/validation/test
IV. Producing annotated output images with bounding boxes
V. Visualizing predictions and generating evaluation metrics

Not included in this system (out of scope):

I. Real-time camera integration
II. Web or mobile deployment
III. Multi-layer PCB defect detection
IV. Cloud-based production-level deployment

1.3 Product Perspective

The system is a standalone module for automated image-based defect detection.
It can be integrated into:

I. Industrial PCB production lines
II. Automated optical inspection (AOI) machines
III. Quality assurance software
IV. Smart-manufacturing dashboards

1.4 User Characteristics

User Type	Characteristics
Quality Inspector	Basic understanding of PCB defects; uses system to view results
ML Engineer / Developer	Knowledge of Python, deep learning, and model training
Supervisor / QA Manager	Reviews detection reports and results
Student / Researcher	Uses system for academic learning

1.5 Similar Apps and Systems / Literature Review

Several PCB inspection systems exist, including:

I. **Rule-based AOI systems** — rely on predefined image filters; struggle with unseen defect variations.
II. **Traditional Machine Learning systems** — require hand-crafted features; limited adaptability.
III. **Industrial AOI hardware** — expensive and inflexible for small-scale manufacturers.

Recent research emphasizes YOLO, Faster R-CNN, and ResNet architectures for PCB defect detection. YOLOv8 provides state-of-the-art speed and accuracy, making it ideal for real-time inspection scenarios.

1.6 Proposed Technologies

I. **Programming Language:** Python
II. **Deep Learning Framework:** PyTorch, Ultralytics YOLOv8
III. **Data Processing:** OpenCV, NumPy, Pandas
IV. **Visualization:** Matplotlib, Seaborn
V. **Tools / IDE:** Jupyter Notebook, VS Code
VI. **Dataset Source:** Kaggle PCB Defects Dataset

2. Requirements

The system will preprocess PCB images, train YOLOv8 using labeled defect examples, evaluate model performance, and detect defects on test images. It will produce bounding-box outputs, label predictions, and confidence scores.
The system will also generate evaluation metrics such as mAP, precision, recall, and confusion matrices.

2.1 Functional Requirements

FR001 – Dataset Preparation

Purpose: Prepare and convert PCB data into YOLOv8-compatible format.
User(s): ML Engineer
Input: Raw images, annotation labels
Process:

- Preprocess images
- Convert annotations into YOLO format
- Split data into train/val/test

  Output: Ready-to-train dataset directory

FR002 – Model Training

Purpose: Train YOLOv8 model on PCB defects dataset.
User(s): ML Engineer
Input: Training dataset, hyperparameters
Process:

- Train model
- Validate performance
- Optimize hyperparameters

  Output: Trained model weights

FR003 – Defect Detection

Purpose: Detect PCB defects in unseen images.
User(s): Quality Inspector
Input: New PCB image
Process:

- Run trained YOLO model
- Identify defects
- Draw bounding boxes

  Output: Annotated image with defect labels and confidence scores

FR004 – Evaluation and Reporting

Purpose: Provide performance metrics and visual reports.
User(s): QA Manager, ML Engineer
Input: Validation and test datasets
Process:

- Generate confusion matrix
- Produce mAP, precision, recall, F1-score

  Output: Charts, metrics, and performance reports

2.2 Non-Functional Requirements

Performance Requirements

I. System must detect defects with high accuracy (mAP@50 > 85%).
II. Model inference time should be <100 ms per image for GPU environments.

Reliability

I. The system should perform consistently across diverse image types.
II. Must handle corrupted or low-quality images gracefully.

Usability

I. Interface must clearly display defect bounding boxes.
II. Visual outputs should be easy for non-technical users to interpret.

Scalability

I. Should allow integration with real-time AOI cameras in future.

Portability

I. Must run on standard Python environments.

3. Use Cases and Flow of Processes

Below is the system-level use case diagram description.

3.1 Use Case 1: PCB Defect Detection

ID: UC001
Name: PCB Defect Detection
Description: This use case describes how the system identifies defects in PCB images.
Requirement(s): FR003
Actor(s): Quality Inspector
Precondition: Trained YOLOv8 model must be available
Postcondition: Annotated result image displayed

Basic Flow

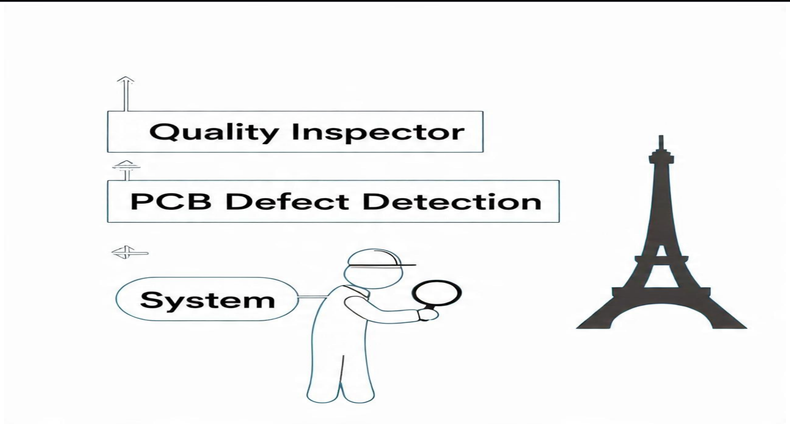
I. User uploads/selects PCB image
II. System loads trained model
III. System processes image
IV. YOLO model detects defects
V. System displays image with bounding boxes and labels

Alternative Flow

I. If model is not trained, system notifies user to train model first
II. User may select multiple images for batch detection

Exceptions

I. Invalid image format
II. Model file missing or corrupted
III. System unable to process due to memory issues



4. References

1. Kaggle PCB Defects Dataset
2. YOLOv8 Documentation
3. Industry 4.0 Research Papers
4. OpenCV Documentation
5. PyTorch Documentation