Project 3 Report

GitHub Repository: https://github.com/LesterPinlac/Project3.git

Introduction

This project aims to streamline the detection of components on Printed Circuit Boards (PCBs) by combining image masking techniques with YOLOv8, a state-of-the-art object detection model. The process involved two major steps: **object masking** for PCB isolation and **YOLOv8 model training** for component classification. This automation seeks to replace manual inspection methods and enhance manufacturing efficiency.

3.1 Object Masking

Methods Used

To isolate the PCB from the background, OpenCV-based **image thresholding** and **contour detection** techniques were employed:

1. **Thresholding**
   - Used cv2.threshold() to segment the image based on pixel intensity, creating a binary mask.
2. **Edge Detection**
   - Applied the **Canny Edge Detection** method to identify significant edges in the PCB image.
3. **Contour Detection**
   - Detected the largest contour in the binary mask using cv2.findContours() and filtered out smaller regions.
   - The cv2.bitwise_and() operation was applied to extract the PCB region from the original image.

Discussion

The masking was reasonably accurate; however, some edges appeared fragmented due to noise. Improvements could involve:

- Using **Gaussian Blur** before edge detection to reduce noise.
- Experimenting with **adaptive thresholding** for images with uneven lighting.

3.2 YOLOv8 Training & Evaluation

Training Method

The YOLOv8 model was trained on the PCB dataset to detect and classify components such as capacitors, connectors, ICs, and resistors. The training process included:

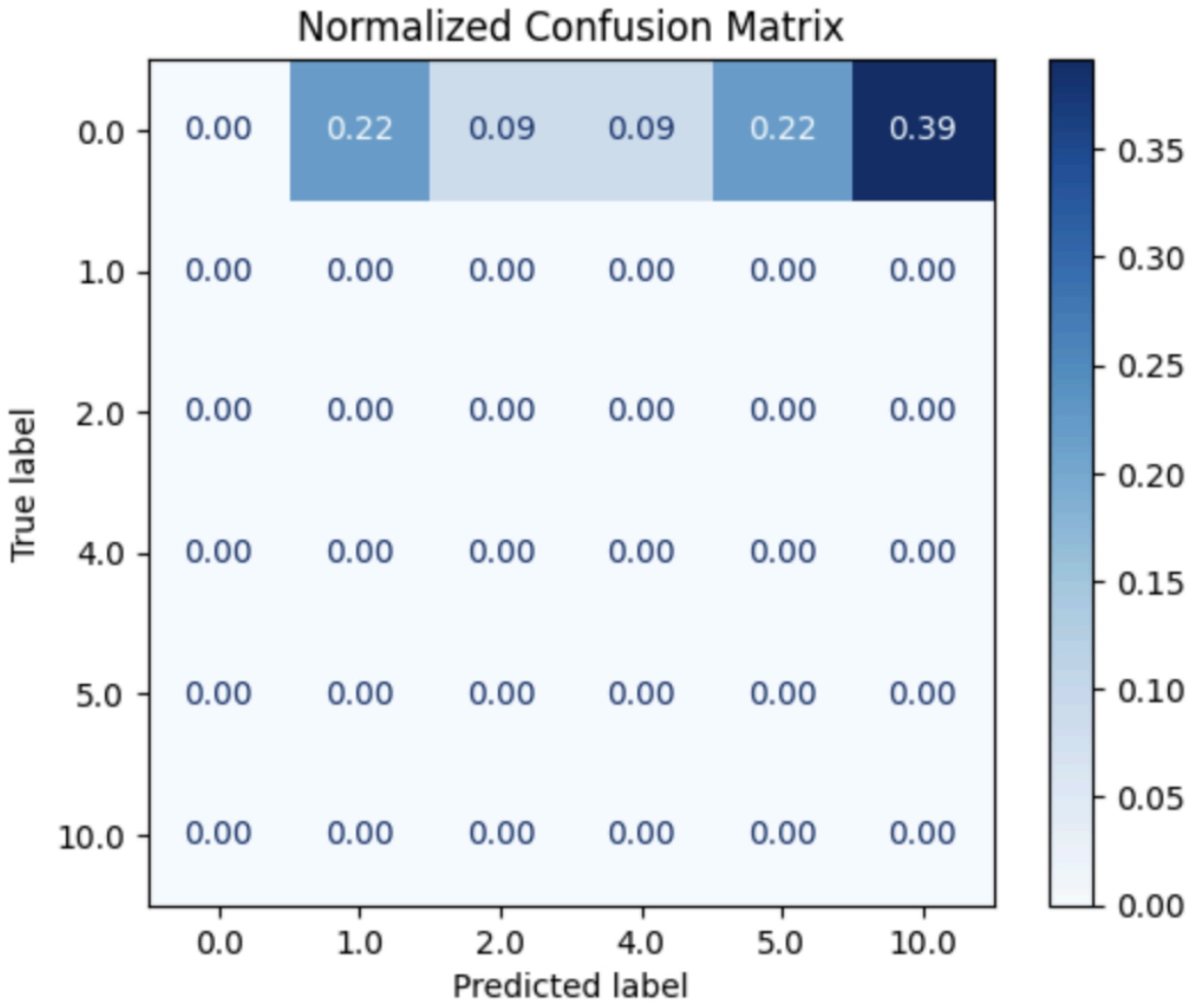
1. **Model:** Pre-trained YOLOv8 model, fine-tuned on the PCB dataset.
2. **Parameters:**
    - Epochs: 100
    - Batch Size: 8
    - image size: 900
3. **Dataset:** Annotated images with bounding boxes for various components.

Evaluation and Results

Normalized Confusion Matrix

The confusion matrix highlights the performance of the model across detected classes.

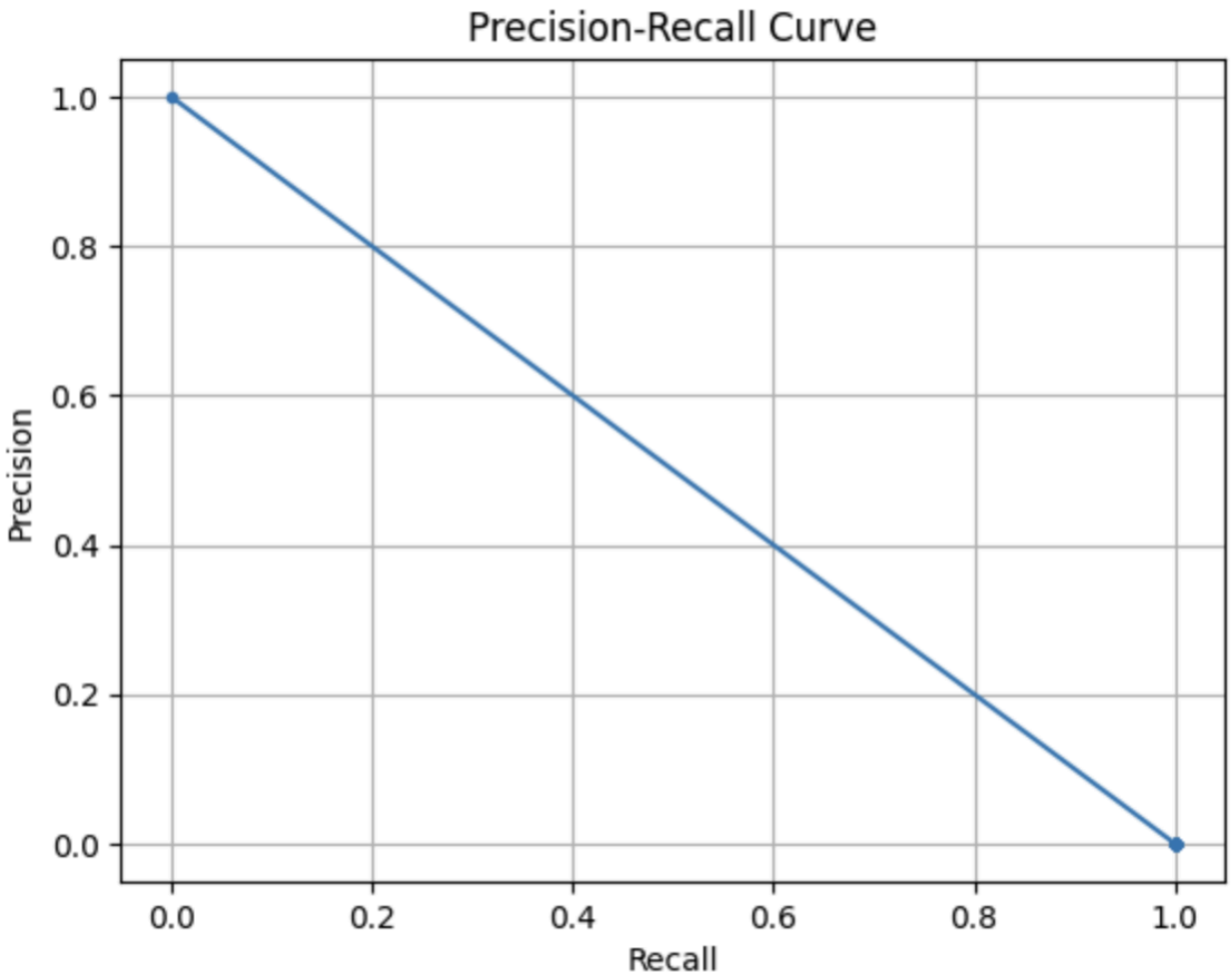
- Observations:
    - The model achieved correct predictions for some components but misclassified others.
    - Class 0 (resistors) had the highest count of misclassifications.
    - Missing true labels for certain components led to zero predictions in rows (e.g., Classes 1, 2, and 4).



Precision-Recall Curve

The Precision-Recall Curve indicates the tradeoff between precision and recall for the classification results.

- Observations:
    - The model struggles to balance precision and recall, reflected by the steep linear curve.
    - Precision drops significantly at higher recall values, suggesting inconsistent confidence scores.



Test Image Analysis

1. Test Image: ardmega.jpg

- Components detected: **1 Button, 8 Capacitors, 7 Connectors, 6 ICs, 7 Resistors**.
- Observations: The model successfully detected most components but missed a few smaller ones (e.g., LEDs).

2. Test Image: arduno.jpg

- Components detected: **5 Capacitors, 3 Connectors, 2 Electrolytic Capacitors, 5 ICs, 9 Resistors**.
- Observations: Slight under-detection of connectors and electrolytic capacitors.

3. Test Image: rasppi.jpg

- Components detected: **4 Capacitors, 6 Connectors, 1 Electrolytic Capacitor, 5 ICs**.

- Observations: Overall good detection, but minor misclassifications for some connectors.

GitHub Repository
https://github.com/LesterPinlac/Project3.git