Toronto Metropolitan University
Department of Aerospace Engineering
Faculty of Engineering & Architectural Science

Course Code: AER 850

Course Title: AER850 - Intro to Machine Learning - F2025

Semester: F2025

Instructor: Dr. Reza Faieghi

Project: 3

Section Number: 02

Submission Date: Dec 1st, 2025

Due Date: Dec 1st, 2025

Name	**Student Number (XXXX99999)**	**Signature**
Ferdos Baikeliaji	501050870	*F.B*

By signing above you attest that you have contributed to this submission and confirm that all work you have contributed to this submission is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a "0" on the work, an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at: www.ryerson.casenate/current/pol60.pdf .

1. Abstract

This project explores a full workflow for detecting components on a printed circuit board. The experiment uses two techniques. The first part applies classical computer vision to separate the PCB from its background. The second part trains a YOLOv11 model to recognize electronic components. Three new images are then used to test how well the model performs.

This experiment presents the process, the reasoning, and the outcomes in a clear and simple way. Each step has a matching figure so the reader can follow the results. Markers are provided to show where each figure should be inserted.

Table of Contents

List of Figures

2. Data processing

The segmentation stage focuses on isolating the PCB before detection. The procedure uses several basic computer vision operations followed a straightforward process that combined classical computer vision with a modern object detection model. The first stage used OpenCV to isolate the printed circuit board from its background. The image was converted to grayscale and smoothed with a Gaussian filter to reduce noise. Canny edge detection was then applied to highlight the outer boundary of the board. A morphological closing step helped connect broken edges so the contour of the board could be detected. The largest contour was taken as the PCB, and a binary mask was created from it. This mask was used to extract the board from the original image. The second stage used a YOLOv11n model. The training dataset followed the standard YOLO folder structure and included images, labels, validation sets, and a configuration file. The model was trained with a resolution of 960, batch size of 4, and 150 epochs. These settings matched the project instructions. The training process produced curves that tracked box loss, classification loss, and overall accuracy. The third stage evaluated the trained model on three PCB images that were not part of the dataset. The best checkpoint from training was loaded and used for prediction. Each evaluation image produced an annotated version with bounding boxes and confidence values. These results showed how well the model could detect components it had never seen before.

The results of trained images:

Figure 1. Original motherboard image [1]

This is the raw PCB image before any processing. It shows the board with all parts still mixed with background.

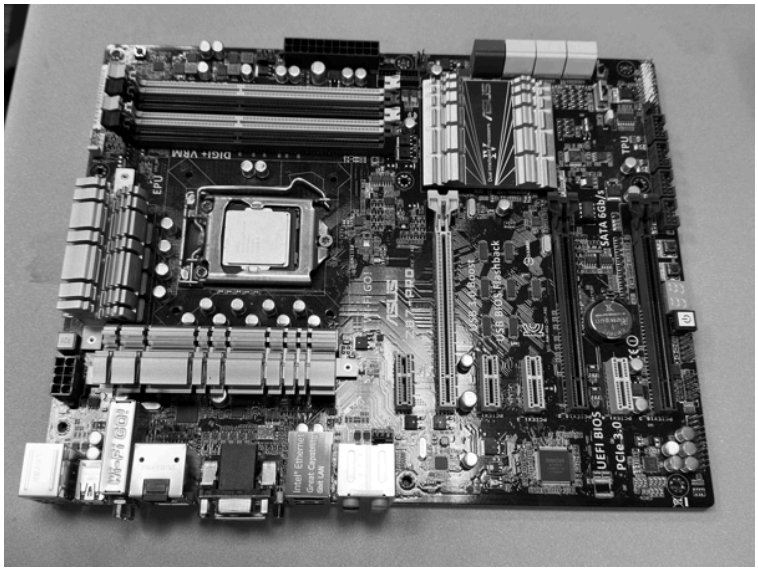

Figure 2. Grayscale

This is the grayscale version of the board. It removes color and keeps only brightness so later steps can focus on structure.

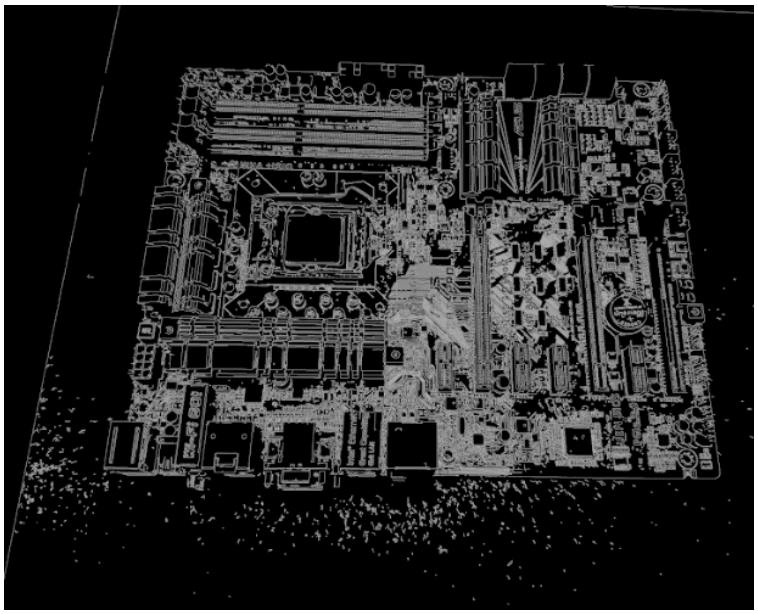

Figure 3. Canny edge result

This shows the edges found in the board. The edges trace the shape of the PCB and its components. It prepares the image for contour detection.

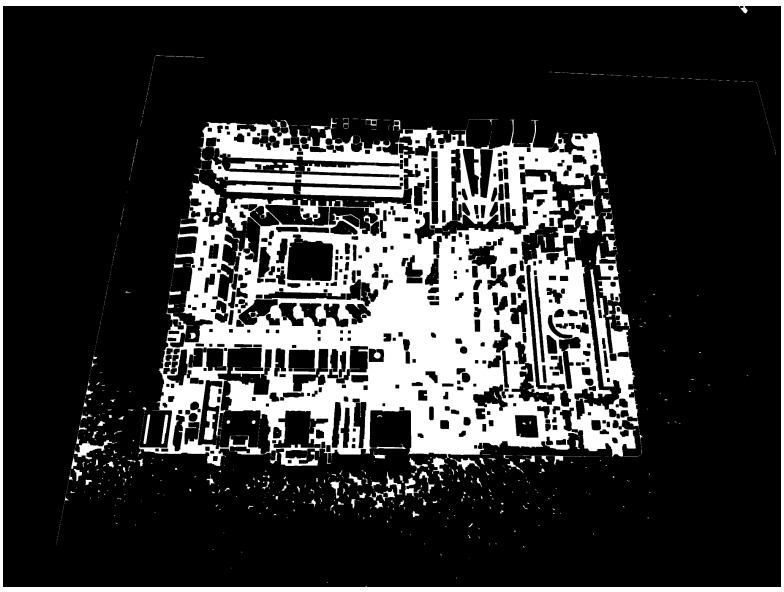

Figure 4. Closed edge result

This image shows the edges after a closing operation. Broken lines are filled so the PCB outline becomes one solid shape.

Figure 5. Binary Mask

This is the mask created from the largest contour. The white area marks the PCB and the dark area is removed

Figure 6. Final extracted result

This pipeline gives a clean view of the PCB and prepares the image for detection in later stages.

3. YOLOv11 Model Training

The second stage trains a YOLOv11n detector using the dataset provided in the project. The configuration follows the assignment requirements. The following are the standard model sizes:

- Base model: yolo11n.pt
- Training epochs: 150
- Image size: 960
- Batch size: 4
- Workers: 0

The dataset is arranged in the YOLO folder format. It includes training images, validation images, and labels. The yolo11n.pt file defines class names and paths. Training outputs, including plots and metrics, appear in the experiment folder created by Ultralytics.

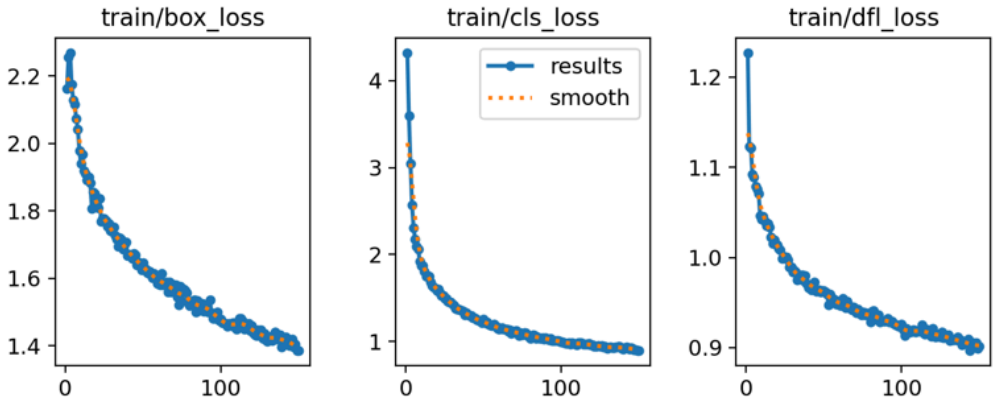

Figure 8. Training loss curves for box/cls/dfl

These curves show how the losses drop during training. They show that the model learns steadily and becomes more accurate over time.

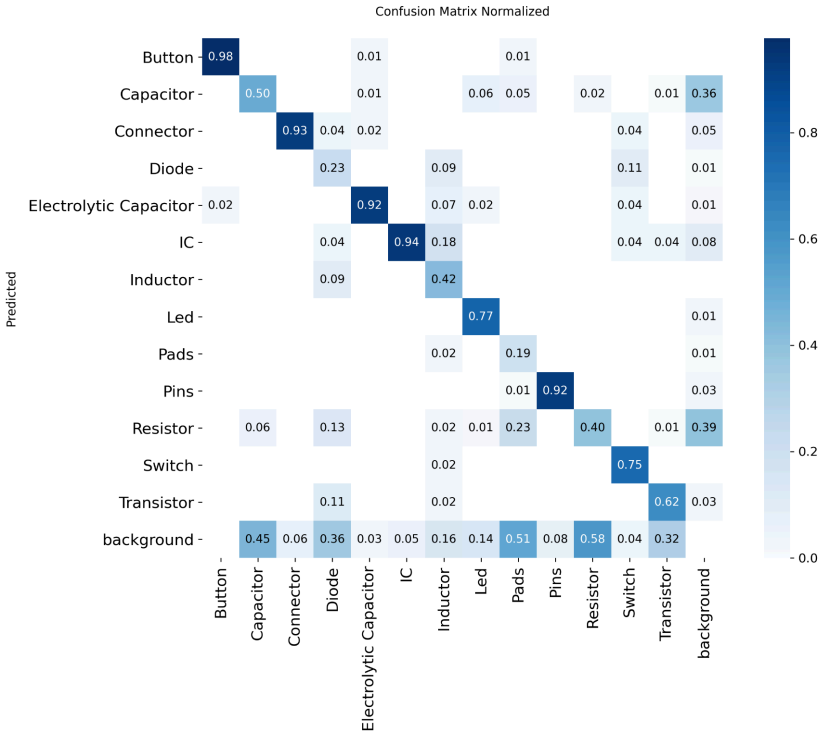

Figure 9. Normalized confusion matrix

This plot shows how well the model tells each class apart. Bright diagonal boxes mean the model predicts that class well. Off-diagonal blocks show confusion between classes.

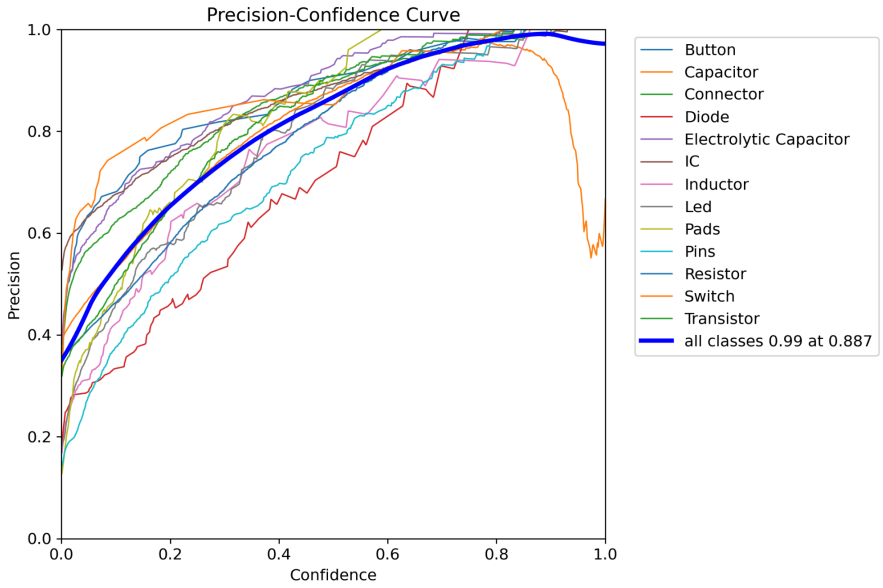

Figure 10. Precision-Confidence Curve

This figure shows how precision changes as the confidence threshold moves. Higher curves mean stronger predictions.

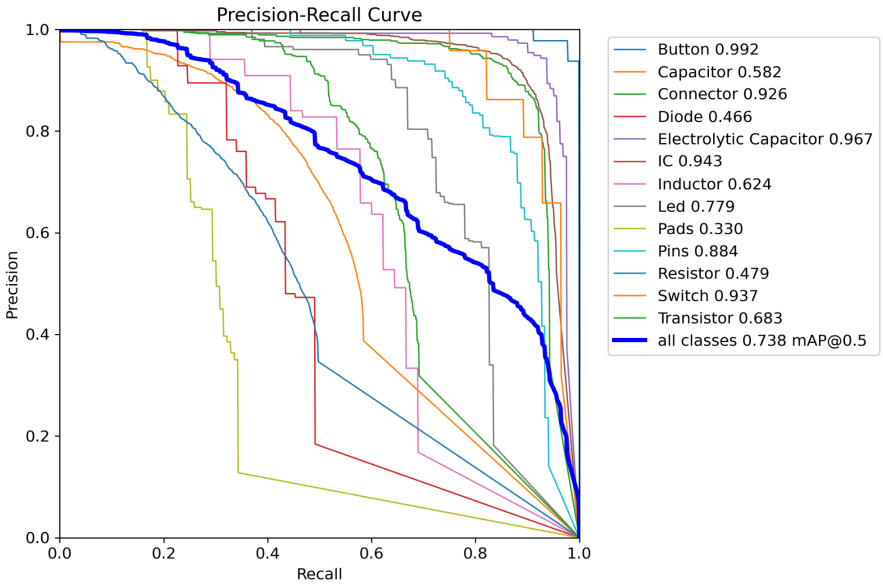

Figure 11. Precision-Recall Curve

This plot compares precision and recall. It shows how well the model balances correct detections and missed detections.

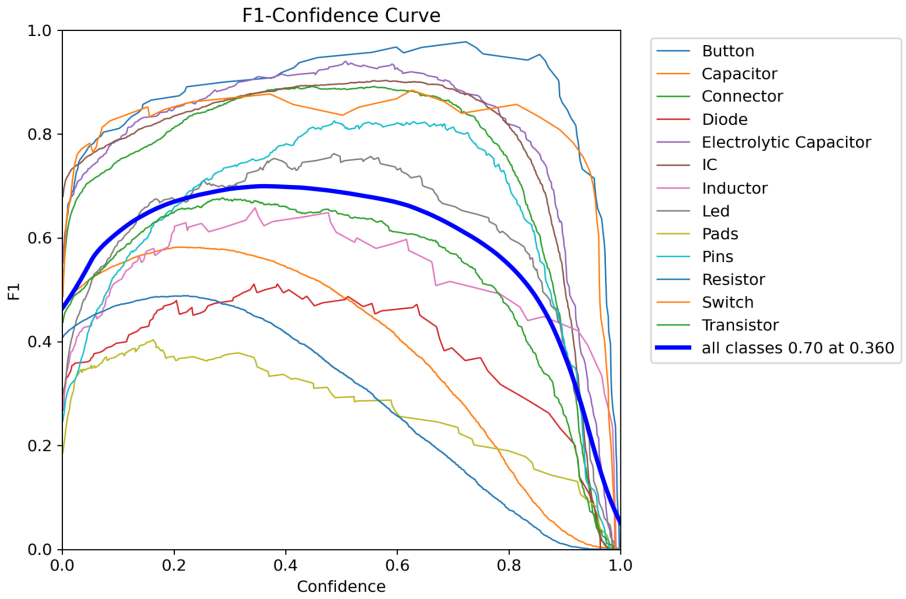

Figure 12. F1-Confidence curve

This plot shows the F1 score for each class. Higher scores mean the model finds the right objects and avoids false detections.

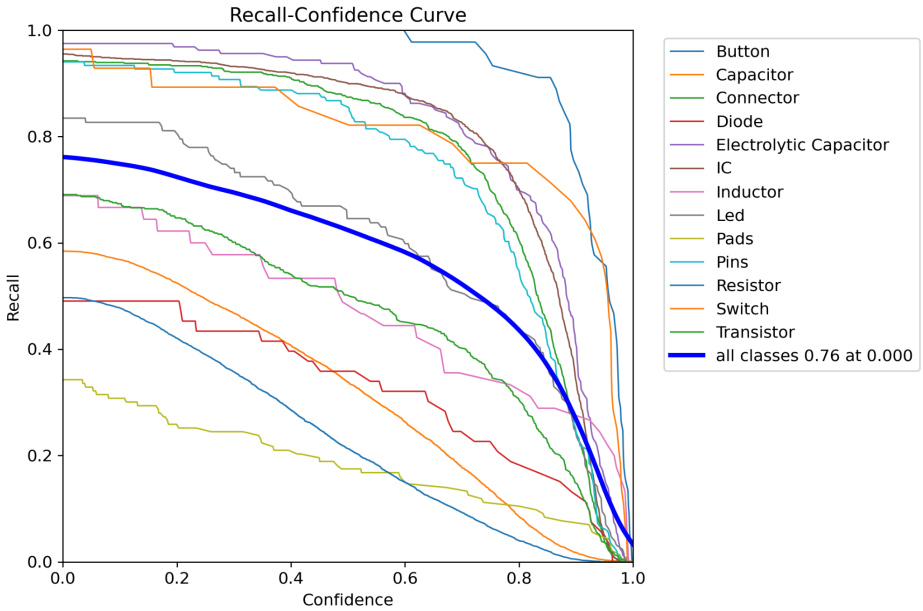

Figure 13. Recall-Confidence Curve

This shows how recall changes with confidence. It helps show how many objects the model finds at different confidence levels.

These plots show how the model improves during training and how well it recognizes each component category.

4. Evaluation

The trained detector is tested on three provided images：ardmega.jpg，arduno.jpg，rasppi.jpg. The model loads the best checkpoint, “best.pt”, which was produced at the end of training. Each image goes through inference and produces a version with bounding boxes and confidence values. The results of trained images:

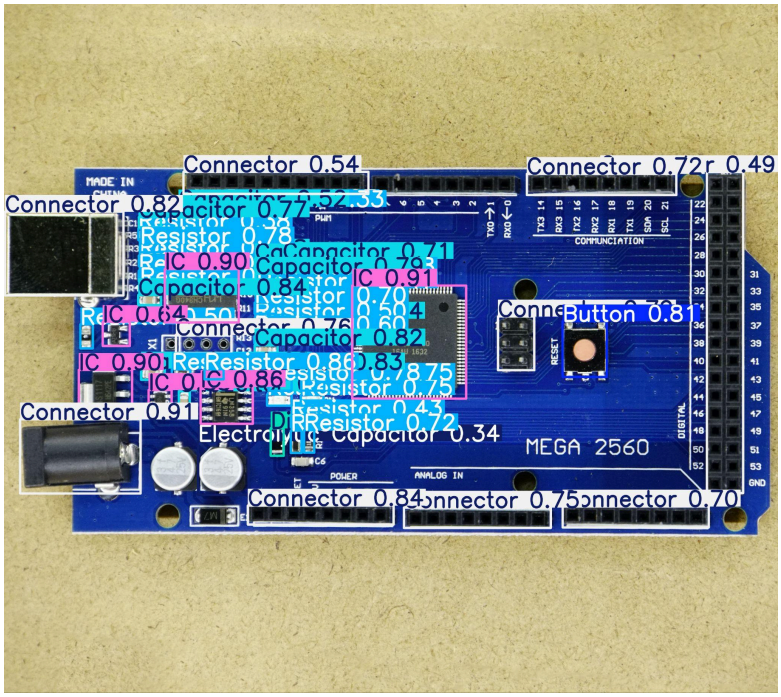

Figure 14. Prediction result on ardmega.jpg

This is the model’s detection on a new board. It shows bounding boxes and confidence scores for each component the model recognizes.

Figure 15. Prediction result on arduno.jpg

This figure shows detections on another board. It tests how well the model works on a different layout.

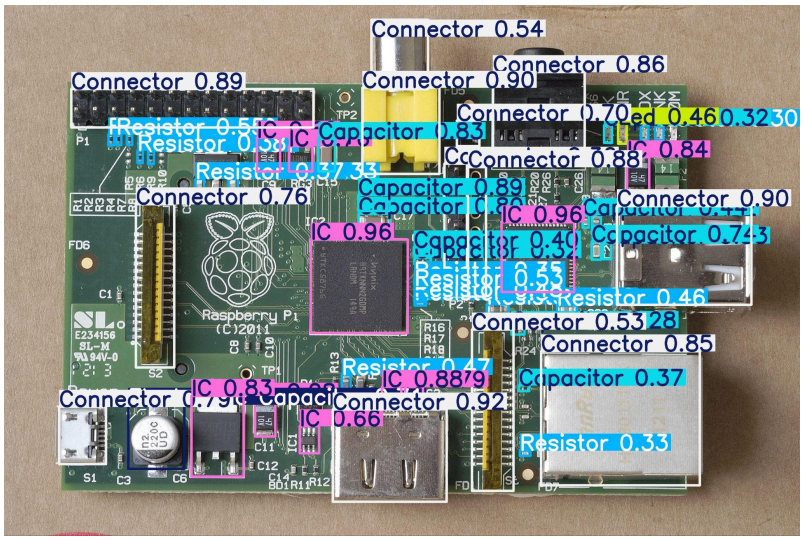

Figure 16. Prediction result on rasppi.jpg

This is the third test image. It shows that the model can detect parts even when the board layout changes again.

These results show how well the model generalizes to new boards.

5. Results

The YOLOv11n model reaches useful and consistent performance.
The training shows a smooth drop in loss and stable metric curves.
The normalized confusion matrix indicates that most classes are recognized correctly.
Some classes show moderate confusion which is expected for a smaller model and a dataset with fine-detail components.

The evaluation confirms this pattern. The model detects many of the large and medium components with good confidence. Small features like thin resistors and small connectors sometimes appear faint or missed. Some prediction errors come from lighting and reflections on the board.

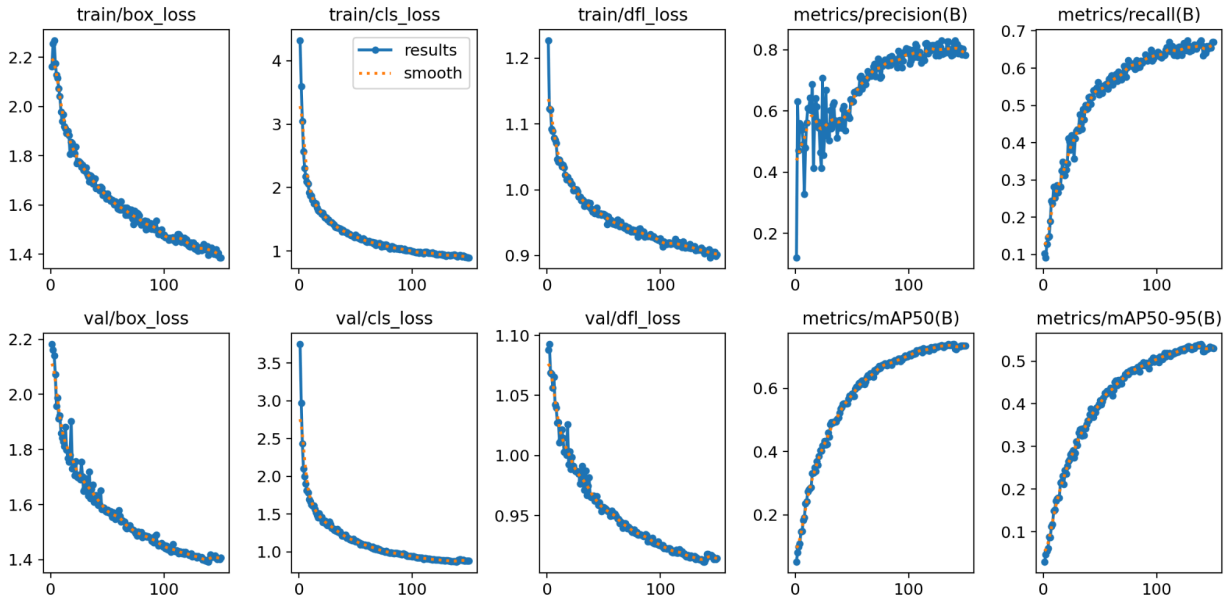

Figure 17. Results

This shows summary results from training. It includes metrics that describe the model's overall performance.

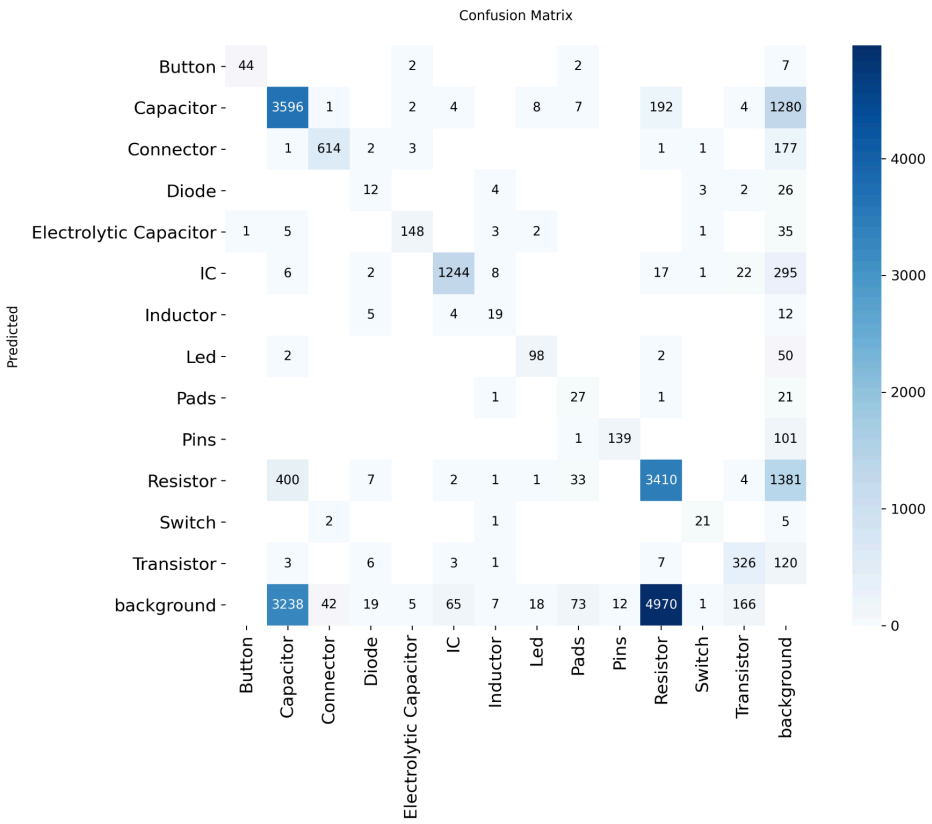

Figure 18. Confusion Matrix

This confusion matrix shows how often each class was predicted correctly or incorrectly. It helps identify which classes were easy or difficult for the model.

6. Discussion

The complete pipeline works well. The OpenCV segmentation step creates clean inputs and reduces background noise. The YOLOv11n model performs reliably with the training settings chosen. Larger models or more epochs could improve accuracy further, but these were kept within project limits.

There are clear paths for improvement. More images for training could help the model distinguish similar components. Higher resolution input images could also capture fine PCB details. More augmentation may help reduce overfitting and make the detector more robust.

Overall, the system meets the goals of the assignment and successfully demonstrates both classical and modern computer vision methods applied to PCB analysis.

7. Reference

[1] Toronto Metropolitan University, *Course Content – AER 715 AVIO*, Brightspace by D2L, accessed Nov. 27, 2025.
https://courses.torontomu.ca/d2l/le/content/1073184/viewContent/6696449/View

8. Repository Link:

https://github.com/FerdosB/AER-850-Project-files/blob/main/Project_3_Ferdos.ipynb