Toronto Metropolitan University

Department of Aerospace Engineering
Faculty of Engineering & Architectural Science

Semester (Term, Year)	Fall, 2025
Course Code	AER 850
Course Section	01
Course Title	Intro to Machine learning
Course Instructor	Dr. Reza Faieghi
Title	Project 3
Submission No.	1
Submission Due Date	2025-12-01
Submission Date	2025-11-27

Submission by (Name):	Student ID (XXXX1234)	Signature
Yuvraj Soni	XXXXX5447	

"By signing the above you attest that you have contributed to this submission and confirm that all work you 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 Academic Integrity Policy 60, which can be found at www.torontomu.ca/senate/policies/"

Table of Contents

List of Figures

List of Tables

Objective

This project's purpose is to design an automated computer vision system for detecting and locating parts on printed circuit boards (PCBs) by utilizing a combination of both conventional computer vision techniques in OpenCV and advanced computer vision techniques with deep learning models. This project has two major tasks. Task one involves separating the PCB from the rest of the board using classical image processing techniques; this includes employing thresholding, edge detection, contour extraction, and morphological operation to create a clean mask of the PCB and eliminate all background clutter and noise. Task two involves developing a deep learning based object detection model using the YOLOv11 architecture and evaluating it. The labelled dataset of PCB components was utilized to train the model over 13 separate classes. The trained model will be evaluated using typical evaluation metrics, including normalized confusion matrix, precision-recall curves, and confidence-based precision analysis. Finally, the trained model will be applied to three previously unseen PCB images (i.e., Arduino Mega, Arduino Uno, and Raspberry Pi) to assess the model's ability to generalize. In total, the overall goal of this project is to demonstrate how the integration of conventional image processing techniques and contemporary deep learning architectures may be used to provide fully automated PCB inspection and part identification.

1.0 Introduction

Automated PCB inspection is essential to ensure that the PCB is reliable and performs well. In addition, because of the complexity and high density of today's PCBs, manual PCB inspection is not an efficient or viable option. Therefore, the application of computer vision and machine learning offers viable alternatives to perform automated detection, classification, and localization of PCB components. This project has two primary objectives.

In the first part, OpenCV was used to perform the necessary image processing to remove the background from the PCB. This required the conversion of the original image to grayscale, followed by the application of a binary threshold to detect edges and to extract the largest contour to obtain an accurate mask of the PCB. The final output was a background removed PCB image that could be analyzed further.

The second part involved using a deep learning-based object detector to detect PCB components. A YOLOv11 large model was trained on a labelled dataset of thirteen component classes, including resistors, capacitors, ICs, switches, pins, and LEDs. Hyperparameters were tuned during training, along with augmentation strategies and performance monitoring. Once training was completed, the model was assessed based on standard performance metrics and visualizations, and then the model was tested on three additional PCB images to determine how well the model generalized.

This report describes the methodology, results, and findings related to each stage of this project. Additionally, this report will demonstrate the applicability of combining traditional computer vision methods with state-of-the-art deep learning techniques to develop fully automated PCB inspection capabilities.

2.0 Results & Discussion

2.1 Open Masking - Open CV

The purpose of Stage 1 was to extract the PCB from the provided motherboard image by employing a sequence of traditional image processing algorithms. The primary goal of this pre-processing phase was to eliminate extraneous information in the background of the image and prepare the image as a clean source for the subsequent detection of individual components. The entire process of the pre-processing phase included a number of steps: image rotation; converting the RGB images to grayscale; applying a binary threshold to separate the PCB from the majority of its background; detecting the edges on the PCB; extracting the contours of the PCB; creating a mask to remove unwanted areas of the image; and applying the mask to the PCB.

The initial motherboard image was loaded into memory via "cv2.imread" after being rotated 90 degrees clockwise to align the PCB properly with the camera's viewpoint. Rotation of the image was required due to inconsistency in alignment of the raw data image with the expected viewing angle.

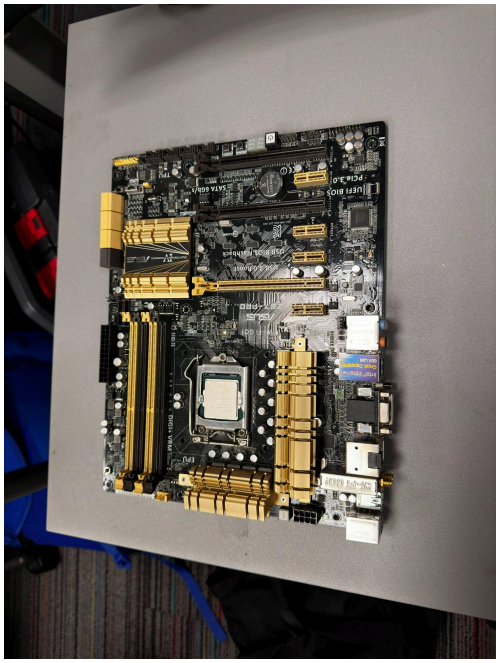

Figure 1: Initial Image of Motherboard

Following rotation of the image, the RGB image was transformed into grayscale in order to simplify the processing stream and enhance the contrast between the PCB and the background. A binary threshold was then applied to the grayscale image by establishing a threshold level of

100 and a maximum intensity value of 255. Any pixel whose value exceeded the threshold was assigned a value of white, and those whose value did not exceed the threshold were assigned a value of black. This effectively separated the PCB from most of the surrounding surface.

Next, edge detection was accomplished using the Canny operator with threshold levels established at 10 and 150. These threshold levels were chosen to identify both major and minor edges present throughout the board surface. In addition to identifying edges, morphological dilation was employed with a 9x9 rectangular kernel using two iterations to amplify the identified edges and fill in small gaps between them. The combination of these two methods resulted in a clean and unbroken outline of the PCB boundaries and thus was well suited for contour identification.

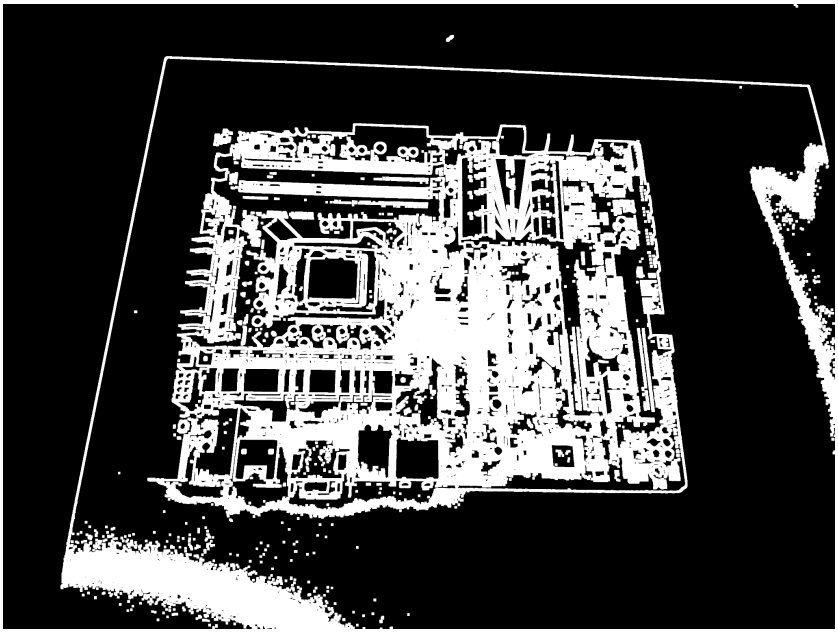

Figure 2: Edge Detection of Motherboard

Contours of the PCB were next determined using "cv2.findContours", specifying "cv2.RETR_EXTERNAL". Using "cv2.RETR_EXTERNAL", only the outermost contour was considered when evaluating the contours of the PCB. The contour that had the greatest area among the detected contours was assumed to represent the outermost boundary of the PCB and therefore was used as the basis for generating a binary mask of the PCB. The contour was filled using cv2.drawContours to create the binary mask.

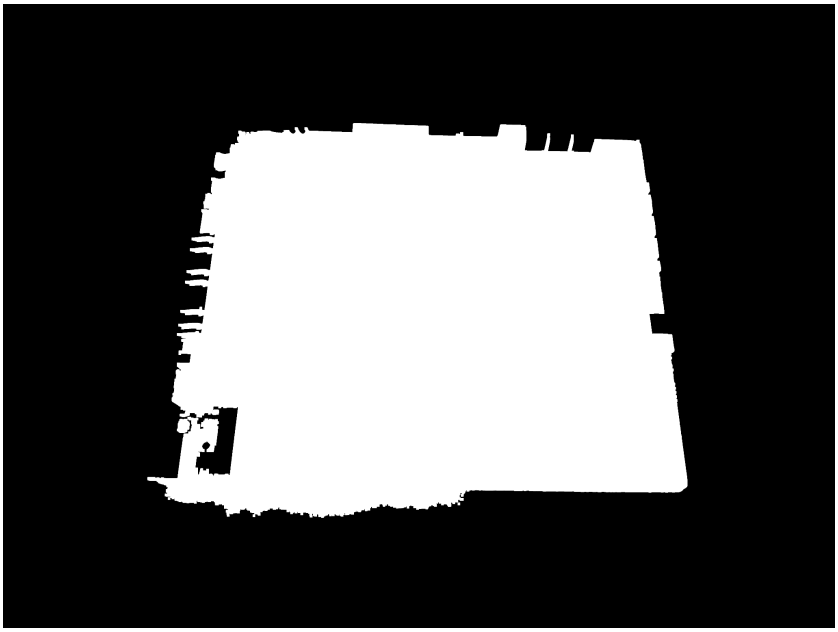

Figure 3: Mask Image of Motherboard

Finally, the binary mask created from the outermost contour of the PCB was superimposed over the rotated image of the motherboard using "cv2.bitwise_and", resulting in the removal of all non-PCB elements of the image and the isolation of the PCB. A Gaussian blur using a 5x5 kernel was applied to the result to soften it, thereby reducing the amount of noise present along the contours of the PCB.

Figure 4: Final Extracted Motherboard

In summary, the overall procedure for creating the mask was successful in extracting the PCB from the original image of the motherboard. Although some minor defects were evident in regions where the illumination of the board was inconsistent or where there was overlap of various components, especially in the corners of the board, they could have been eliminated by using adaptive thresholding or adjusting the parameters of the Canny algorithm. Nevertheless, the final extracted PCB image was sufficient for use as the base for the YOLO-based component detection in Stage 2.

2.2 Component Detection - YOLOv11 Training & Evaluation

During the second phase of the project, an object detection model based on deep learning was trained to detect individual components on a printed circuit board (PCB), using a YOLOv11 Large (YOLOv11l) model; the model was chosen because it had a good trade-off between precision and processing power. Prior to training, a labelled data set of 13 different types of components on a PCB, including resistors, capacitors, integrated circuits (ICs), connectors, light-emitting diodes (LEDs), pads, inductors, diodes, and switches, was created. Training was conducted using the Ultralytics library within Google Colab, utilizing an existing dataset formatted to match the YOLO model requirements, i.e., images and corresponding label files for both the training set and validation set.

Below are the critical hyperparameters utilized during training:

Table 1: Parameters used for training

Model	YOLOv11l
Epochs	190
Batch Size	2
Image Size	1280
Optimizer	AdamW
Initial Learning Rate	5×10^{-4}
Learning Rate Schedule	Cosine Decay
Weight Decay	0.0005
Augmentations	HSV adjustments, translation, scaling, and mosaic
Early Stopping Patience	60 Epochs

These hyperparameters were determined to optimize the performance of the model on the available GPU resources while maintaining training stability. At the completion of the 190-epoch training cycle, the model demonstrated excellent performance on the validation set. Specifically, the model attained:

Table 2: Model Performance

Precision	0.93
Recall	0.887
mAP@50	0.928
mAP@50-95	0.753

As indicated by these metrics, the model was very successful in minimizing false positives (high precision) while successfully identifying approximately 89% of all true components (strong recall). The model's performance was consistent with typical behavior for mid-size YOLO models trained on moderately complex datasets.

A normalized confusion matrix revealed additional details about how well each type of component was detected. Specifically, the model identified buttons, LEDs, ICs, and electrolytic capacitors extremely effectively. Conversely, the model was less successful in identifying capacitors, resistors, and pads due to their relatively small size, close proximity to neighbouring components, and similarities in appearance among those components. This behavior is consistent with typical difficulties found in PCB inspection tasks, where smaller components tend to be much harder to isolate and classify than larger ones.

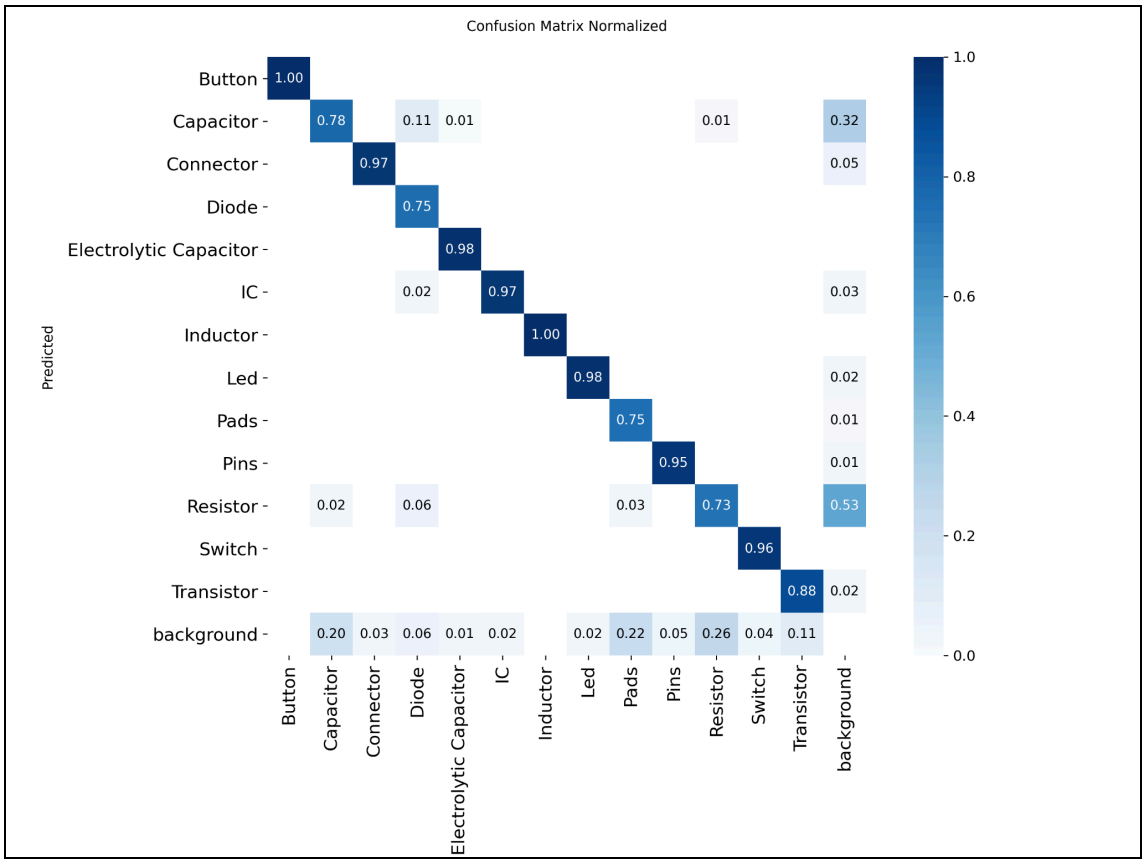

Figure 5: Normalized Confusion Matrix

Precision Recall (PR) Curves further validated the model's behavior across varying levels of confidence. Many of the component classes exhibited high precision over a large portion of the recall range, validating the model's ability to localize objects accurately. Additionally, the button class exhibited a nearly ideal PR curve, whereas classes such as resistors and pads exhibited steep decreases in precision as the recall increased. The LED class showed high recall but noticeably lower precision, consistent with its PR curve behavior.

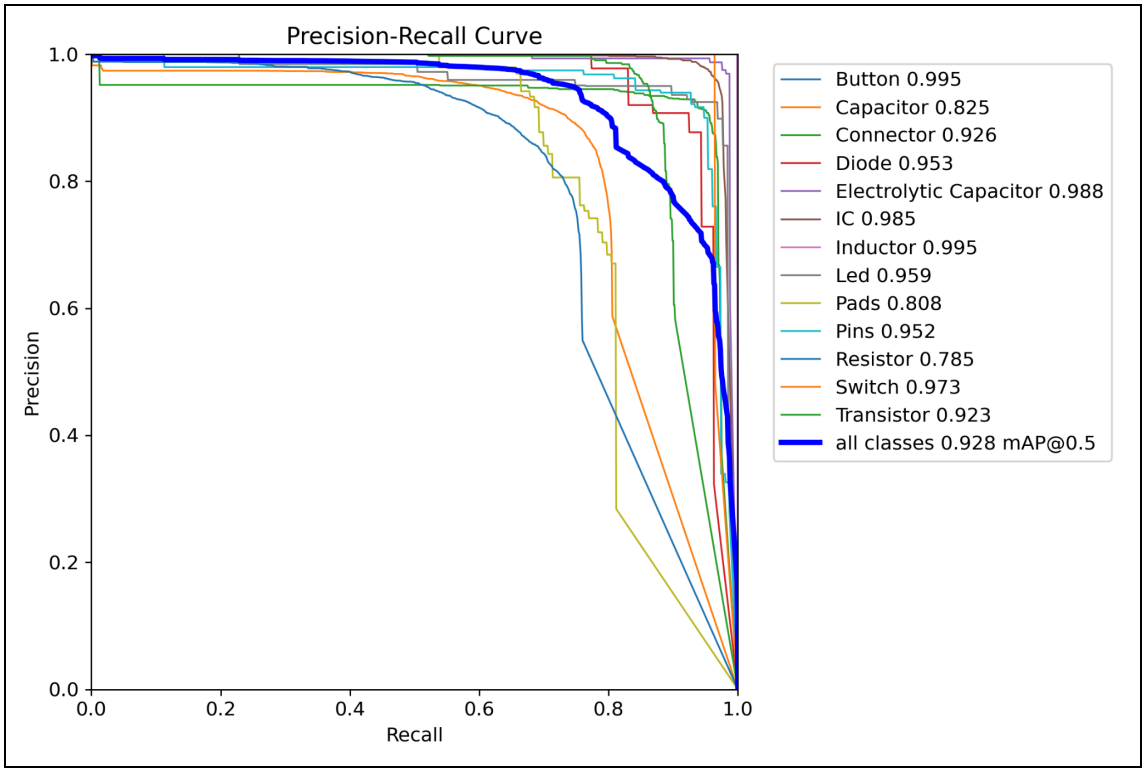

Figure 6: Precision Recall Curve

Additionally, the precision-confidence curves illustrated the relationship between the model's confidence scores and precision. As shown in Figure 7.0, most of the component classes were able to achieve high precision for confidence values greater than 0.6 and then experienced a sharp decrease in precision when confidence values fell below 0.6. Therefore, predictions made by the model with confidence values under 0.6 are more likely to represent noise or ambiguous visual representations.

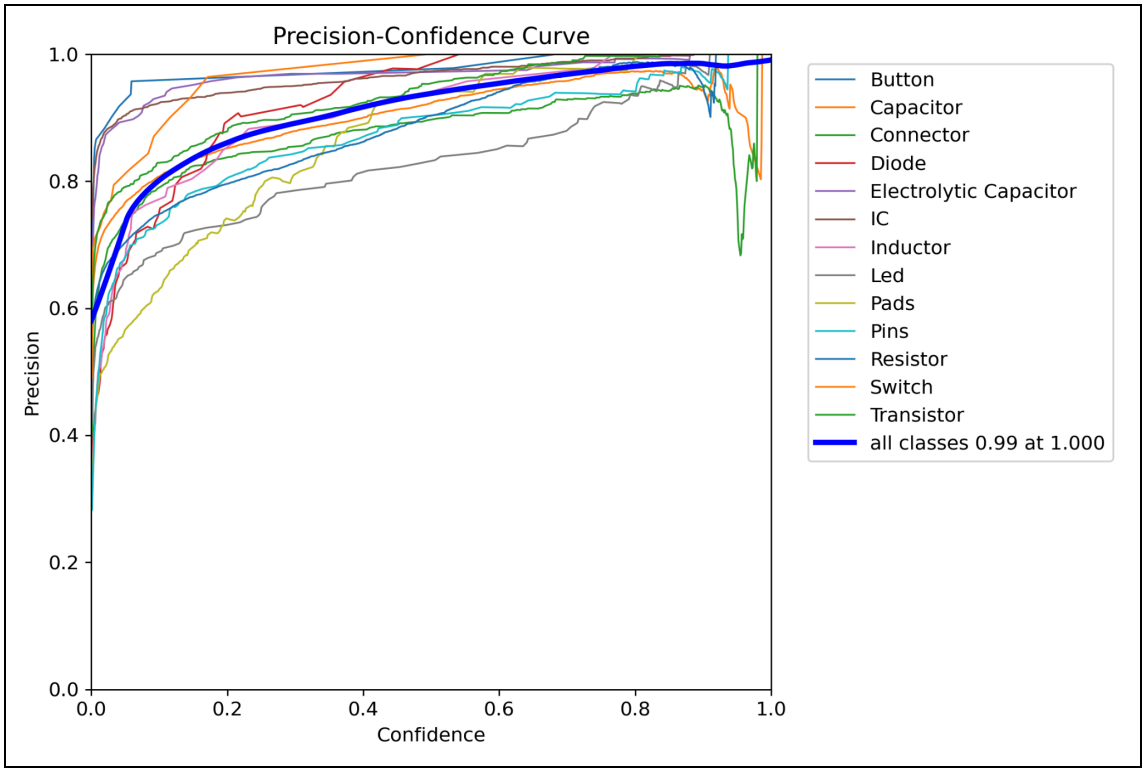

Figure 7: Precision-Confidence Curve

In summary, the YOLOv11l model exhibited high-quality performance in detecting the majority of the component types on a PCB but encountered issues with fine detail or visually confusing components. Thus, the use of YOLO-based detection for PCB inspection is supported by this work and provides insights into potential areas for future improvements to the model through the addition of new data samples to the dataset and/or further exploration of optimal hyperparameter combinations.

2.3 Evaluation on External PCB Boards

Three new and previously unseen images of PCBs (an Arduino Mega, an Arduino Uno, and a Raspberry Pi) were used to evaluate the ability of the trained YOLOv11l model to generalize to new data. The three PCBs are very different in terms of layout, density of components, and size, which provides a reasonable real-world test of how far out-of-distribution from the original training data the model can function. All images were evaluated using a resolution of 928 × 928 and a confidence threshold of 0.1. The output was written and manually examined for correctness and consistency of component localization. On the Arduino Mega, the model identified all major types of components present on this board, including the button, capacitors, connectors, electrolytic capacitors, ICs, and resistors. Although the Arduino Mega

has many components and is physically much larger than the other two boards; the detector functioned well, particularly with respect to the identification of larger and visually distinct components like ICs and connectors. Minor inconsistencies did occur in areas of low contrast and/or high component density; however, the detected localizations for most components were correct.

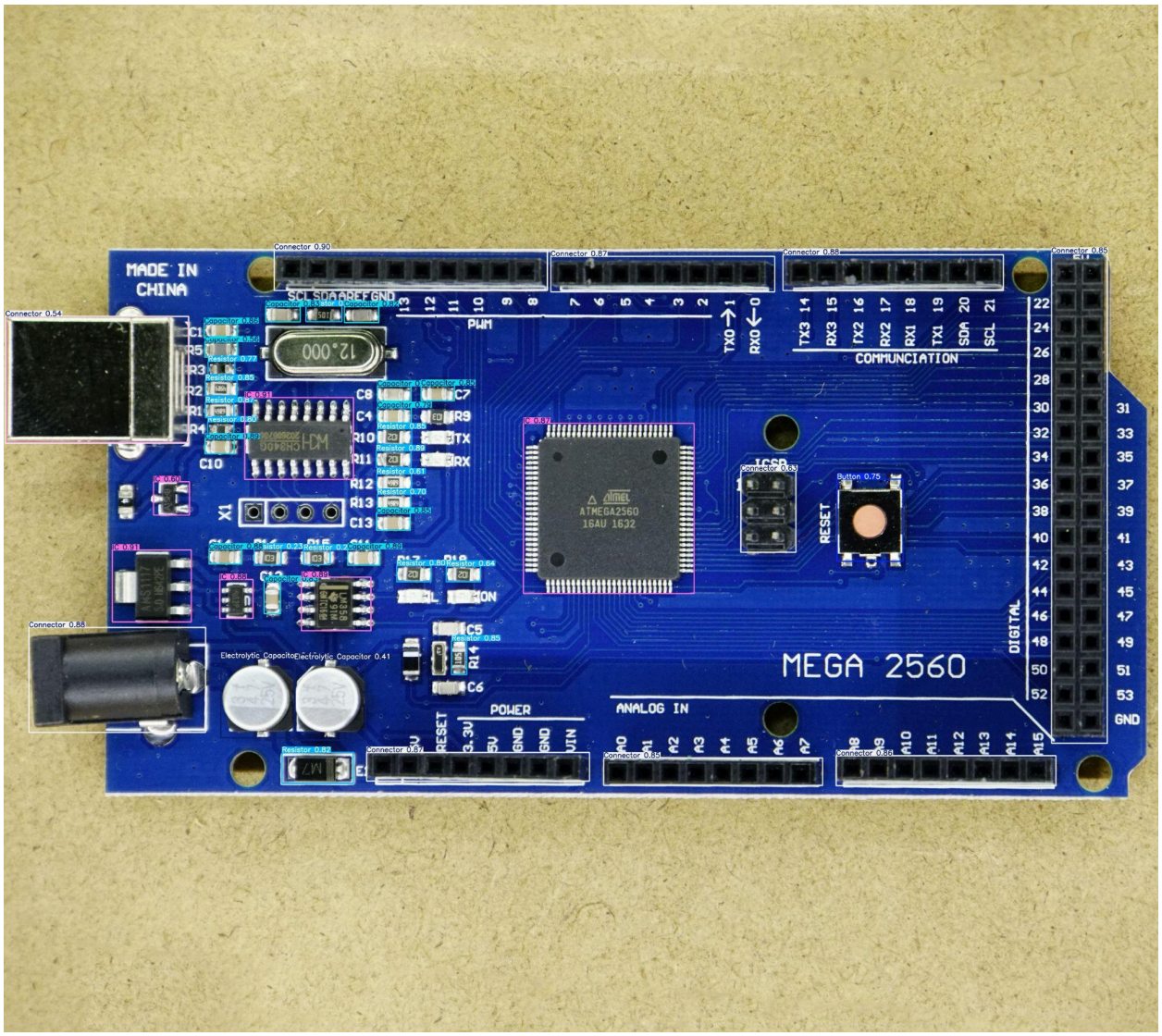

Figure 8: Arduino Mega Evaluation Image

In comparison to the Raspberry Pi, the Arduino Uno was an easier test model to evaluate based on size and number of components in the printed circuit board (PCB). Major components such as ICs, electrolytic capacitors, connectors, and LEDs were accurately identified by the model. Due to the small size of the components, detecting smaller components, including some of the resistors and pads, sometimes caused either the detection of these components to be

missed, or slight misclassifications. In spite of this, the performance of the model was still good, and the results of the predictions compared well with the actual layout of the PCB.

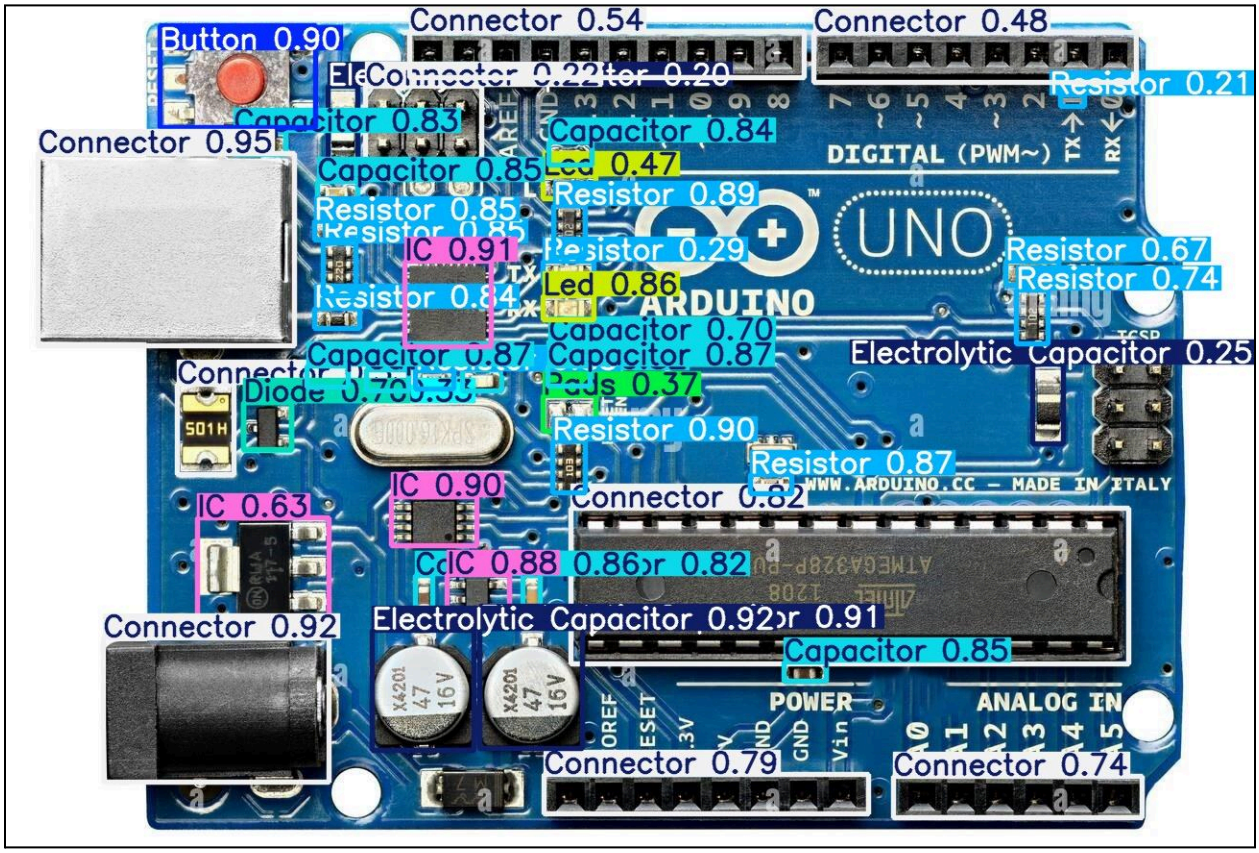

Figure 9: Arduino UNO Evaluation Image

The image of the Raspberry Pi showed the largest challenge due to the many small parts (and therefore high density) and large amount of visual complexity on the PCB. The model performed well at detecting ICs, connectors, inductors, and LEDs. It was able to identify a significant portion of the capacitors and resistors as well; however, the increased density of the smaller components significantly increased the probability for both false positives and occasionally missed detection. Nonetheless, the detector still generated a meaningful and informative list of predictions across the entire board and therefore indicates a high degree of adaptability to a wide variety of more complex PCB layouts.

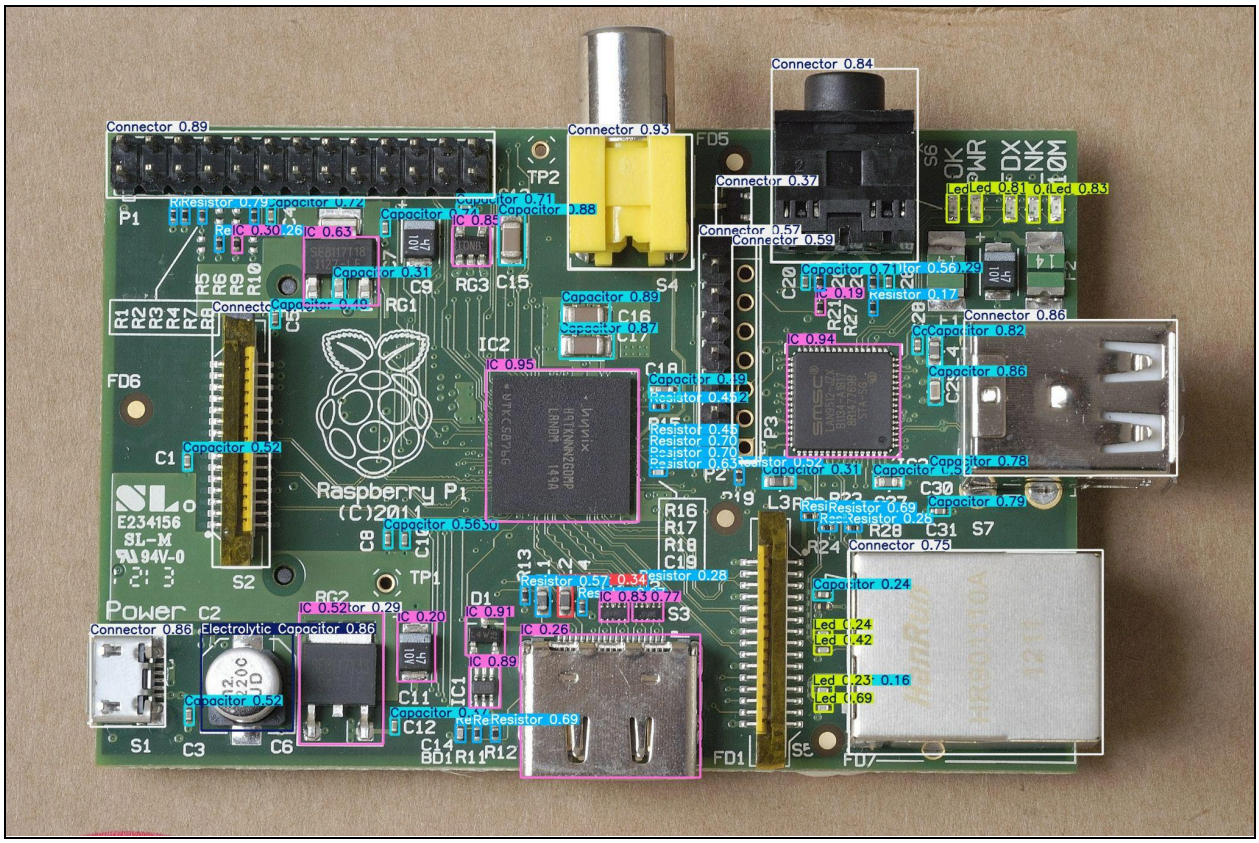

Figure 10: Raspberry Pi Evaluation Image

This model's average inference speeds were about 6.1 milliseconds for preprocessing, 8.3 milliseconds for inference, and 1.1 milliseconds for post-processing per image. These results show near real-time performance and may be useful for automated inspection systems. In addition, across each of the three boards used, the detector showed excellent ability to generalize (especially when it comes to larger and more distinct component types). However, smaller items such as resistors, pads, and some capacitors were more variable in terms of whether or not they were detected due to their relatively small size and visual similarity to other parts of the board. The overall results from testing the YOLOv11l model are consistent with the expectation that the model will perform well regardless of the variety of different printed circuit board design styles, providing an effective starting point for developing automated analysis tools for printed circuit boards.

Conclusion

This research has shown the potential of combining classical approaches to computer vision with new techniques based on deep learning to automatically inspect printed circuit boards (PCBs). The initial phase of this study used OpenCV to remove the PCB from its surrounding environment by using a series of image processing steps (thresholding, edge detection, dilation, and contour detection) to isolate the PCB from the background. The resultant masked image removed all irrelevant information from the background and created a clean base to analyze each of the individual components.

The second phase involved training a YOLOv11l model on a labelled dataset of PCB images, with 13 classes of individual components. After completing the 190-epoch training process with a set of optimized hyperparameters, the model performed well over the entire range of the data, achieving a precision of 93.0%, a recall of 88.7%, and an mAP@50 of 92.8%. When testing the model with a variety of external images of PCBs, it was able to effectively identify the main components on three different types of PCBs (Arduino Mega, Arduino Uno, and Raspberry Pi), including ICs, connectors, electrolytic capacitors, LEDs, and buttons. However, smaller and visually similar components (resistors and pads) were found to be difficult for the model to detect as expected due to the nature of PCB inspection.

Overall, these findings demonstrate the ability of object detection models (YOLO-based) to provide a reliable method for identifying individual components on printed circuit boards. Additional enhancements to the current design could include adding additional examples to the dataset, increasing the resolution of the images being analyzed, or modifying how the images are augmented to improve the reliability of component detection, especially for smaller or closely grouped components. Nevertheless, the system developed for this project demonstrates a viable example of the capabilities of both classical and deep-learning-based approaches for use in PCB-related computer vision applications.

Appendix

[1] GitHub Repository Link

[2] Google Drive: AER850 Project 3 Files