Machine-Assisted Electrical Design Inspection

Isaiah Riley,

Ranai Srivastav,

Varun Jain

Team Number 1

April 27, 2023

Final Project Report for

HCI 575: Computational Perception

Iowa State University

Spring 2023

Table of contents

Abstract

This paper is a project writeup for a computer vision system that will inspect circuit boards in order to identify as much information about their functionality as possible. Key points of inquiry include identifying damage, identifying component types, reading silkscreen and component text, and identifying component connectivity. The major goal of this project will be to notify the user if any components on the board are worth salvaging

1. Introduction

a. Summary

PCBs are an essential part of many daily use items. They are an important part of all electronics, which have become ubiquitous and essential for everything, ranging from cooking food to driving a car and are found everywhere within phones, TVs, and much more. However, with the increase in the use of these components comes a greater generation of waste. The goal of this project is to develop a tool so that users can detect as much information about the circuit board as possible, and reduce waste in doing so.

Before going further, it is necessary to briefly explain the nature and composition of a circuit board. This project will focus on simple circuit boards composed of 1 or 2 layers. Two layer circuit boards are pieces of fiberglass material with copper “traces” running to various parts of the board surface. Components can be placed on either side of the board and are attached using solder, a metal compound with a low melting point. The solder creates a mechanical and electrical connection to the copper traces. In this way, the components are connected to the board and each other. Electrical signals may pass from one side of the board to the other using a via, essentially a small copper tube, through a hole in the board material. Fig 1 illustrates a 2-layer board with vias and traces.

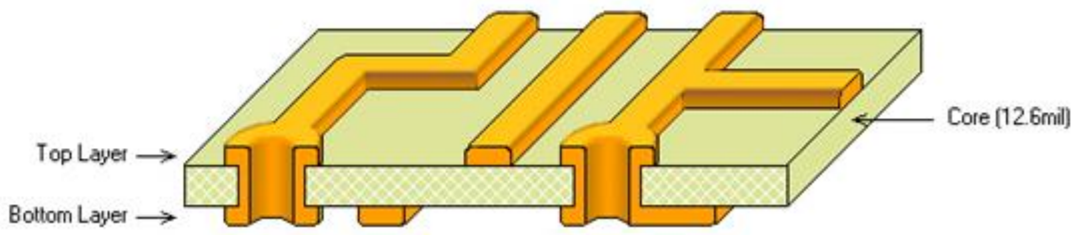


Fig 1. Two Layer PCB Diagram[1]

The operation of the program is simple. The user will take a minimum of 2 photos, one of the top of the board and one of the bottom. These will be used as inputs to the program. From there, the program will process these pictures and do the following:

1. The program will stitch together images of like sides of the board and group them into two categories: Top Side and Bottom Side
2. The program will scan each side of the board for components, identifying damage and component values where possible.
    a. During this scan, the program will attempt to identify damaged components
3. After this scan, IC values will be checked against online marketplaces to determine value, if undamaged.
4. The program will scan each side of the board for traces and vias, identifying which components are connected to each other electrically where possible.
    a. During this scan, the program will attempt to identify damaged elements of the board.
5. The program will provide all of the above information to the user.

This program is intended to run on a computer independent of the imaging system. There is no target time for the length of processing, but it is expected to be fast enough that it is still a benefit to the person using it.

b. Intended Users

The primary user group for this project is electronics salvage companies, which exist to go through electronics salvage in bulk and determine if any of it is suitable for reuse. Typically these companies target rare and expensive components to pull off of boards to sell individually. With an automated inspection process, the work of determining which parts of a board are worth keeping would be greatly reduced, and the skill of the labor necessary could also be reduced. Further, an automated inspection tool may also be able to identify whether or not a board could be re-used in its entirety by inspecting the board for signs of damage.

The additional task of detecting component connectivity would encourage yet more users like repairmen and students since detecting connections means representing those connections in the form of a schematic. Electronic recyclers require fewer features than others, as they have no interest in schematics. The limited requirements and commercial viability through cost savings make the salvage group the ideal target market for this program, but the project will only attempt to expand to other user groups if time allows.

This program is expected to have incidental benefits to 3 parties who are not the intended audience. These groups are hobbyists, repairmen, and reverse engineers. Hobbyists are interested in a tool like this for the purposes of not only repair but also modification, enhancement, education, and entertainment. This tool would allow them to work on circuits that they own, and it would also lower the skill requirement for these hobbyists by making information about their projects more accessible.

Repairing electronics is a costly endeavor. It may take a trained technician hours to examine the root cause of an electronics failure. Alternatively, the repairman may isolate the problem to one board or a cluster of boards and replace them all in an effort to save troubleshooting time. Both options are costly enough that only the most valuable electronics are worth fixing. The price of circuit boards is relatively fixed, and the cost of trained labor is high, but if there were a way to dramatically cut down on troubleshooting time, then the repairmen would be able to lower their price per item, increasing business. It may also make the job easier and allow for downward pressure on labor costs as the barrier to entry is lowered.

Many servicemen are highly specialized. For example, those who specialize in televisions only know how to fix televisions. They have the basic knowledge of electronics to be able to work on anything, but they focus on televisions because they often have similar problems with similar solutions. If a television brand has a reputation for bad HDMI connectors, for instance, a repair tech with this knowledge could dramatically cut troubleshooting time. Delivering sufficient information to cut down troubleshooting time is the stretch goal of this project.

The other group that would benefit is reverse engineers. This group may have noble or questionable intentions behind why they are peeking under the hood of how their electronics work. Developing countries could use a similar tool to copy existing products and manufacture third-party replacements or counterfeits. Because this specific program is designed with 2-layer circuitry in mind and is not expected to work well with more complex board types, it is unlikely that expensive or sensitive boards, such as cell phone circuit boards or military-grade circuit boards that use four or more layers could be copied using this program. Companies and governments that wish to protect their intellectual property from a program like this have the option to coat the board in an opaque material or employ other countermeasures.

2. Related Work

Although no one entity has created what we intend to make through this project, a few of the sub-tasks have been previously attempted or implemented.

Although the application is unique, image stitching is an old technology and is widely used today. The most common experience people have with image stitching tends to be with panoramic photographs. This is widely known enough that it would be impractical to site all of the previous implementations, but it is worth noting that our implementation takes advantage of the Stitcher API, a Python module made specifically for document scans and panoramas. The scan mode in particular was a crucial element that empowered us to pursue the idea to generate input images.

There are several high-quality datasets of components upon which component-detecting machine learning systems can be trained. It is fair then to assume that some have already used these datasets in order to develop component detection machines. In their paper *FICS-PCB: A Multi-Modal Image Dataset For Automated Printed Circuit Board Visual Inspection*[7], Lu, et. al. release the FICS dataset to the public that has high quality DSLR images with annotations of resistors, capacitors, inductors, transistors, and diodes. The main difference between our implementation and theirs, besides the fact that we are independently training our own model, is use cases. Every other implementation of component detection using machine learning has done so with the intention of checking that the components have been correctly placed during the circuit board assembly process. Previous implementations have had interest in the location and type of component, but not in the exact component value as we are.

We use YOLOv5, as published by Jocher, et. al. in 2022 on Instance Segmentation. YOLOv5 is a very popular machine learning framework that designs a backbone for the machine learning algorithm that can then be trained on a custom dataset using Transfer Learning. We chose to use YOLOv5 for ease of use, and efficiency during training and prediction.

Component valuation is accomplished by searching the online marketplace Ebay with the search term of the ID of each chip. The resulting search page is scraped for relevant information about the price and date of sale of recently sold search results. The idea of scraping webpages, including marketplaces is nothing new. There are many tutorials and books available on the topic, such as Web Scraping with Python [1]. Our implementation uses the Python 'requests' library to gather HTML information which is interpreted by the BeautifulSoup library and populates a CSV file with the help of the Pandas library. Our implementation should not be considered unique, as it is highly similar to an existing price-gathering tool made by Youtuber John Watson Rooney[2].

The 1988 paper by S Meyer titled *A Data Structure for Circuit Netlists* talks about using Netlists to store the different connections using a modified graph data structure. The paper talks about a cell primitive to store components and a netlist primitive to store the connections. These form a Directed graph that helps establish the flow of current from one component to another. Although the algorithm does not exactly use this paper's definition of a netlist, it implements a similar concept of a primitive that can be connected to other components using the edges of a graph.

Although many of the sub-tasks of the project have been previously attempted, we believe that our independent implementation and unique combination of functions for a specific purpose set this project apart from previous works. That said, the groundwork has been laid for each of the sub-tasks that we implemented.

3. Experimental Platform

Each of the individual tasks of the project was attempted individually, with a few self-imposed guidelines.

First, all the code for the project was developed in Python 3.10. This is the platform that all team members have the most experience in, and we know that there are a number of good libraries that we can leverage. Second, each of the programs should be created modularly. This would allow for easy testing and easy integration between the components. Finally, the resulting programs should be capable of running on a wide variety of machines.

The environment necessary to develop the project can be found in the requirements.txt. Most of the development could be completed on the team's personal computers, with the exception of training a neural network, which was done on the Iowa State High Performance Computing Cluster.

The inputs to the program are high-quality photos of various circuit boards. The team took steps to create a setup that facilitated the capture of consistent, high-quality camera shots in good lighting conditions. Figure 2 is an image of the setup. Four light sources were spaced equally and had similar intensity level to minimize shadows and maximize clarity. A neutral, contrasting background was chosen for optimal edge detection results against PCBs. Images were taken using the camera of a Pixel 6 Pro cellphone. While this camera is not a DSLR, the resolution is expected to meet the project's needs. We took multiple images of the same portion of the board from different distances so that image stitching could give us a comprehensive picture of the board with more clarity and higher resolution than the original image.

Once the images are taken, they can be input into the program using the Command Line Interface (CLI) to interact with the program. The CLI is easy to use and has in-depth documentation that describes how to use the application correctly. The user has to enter two paths, demarcated by the tags –front and —back, each pointing to a folder with the corresponding files

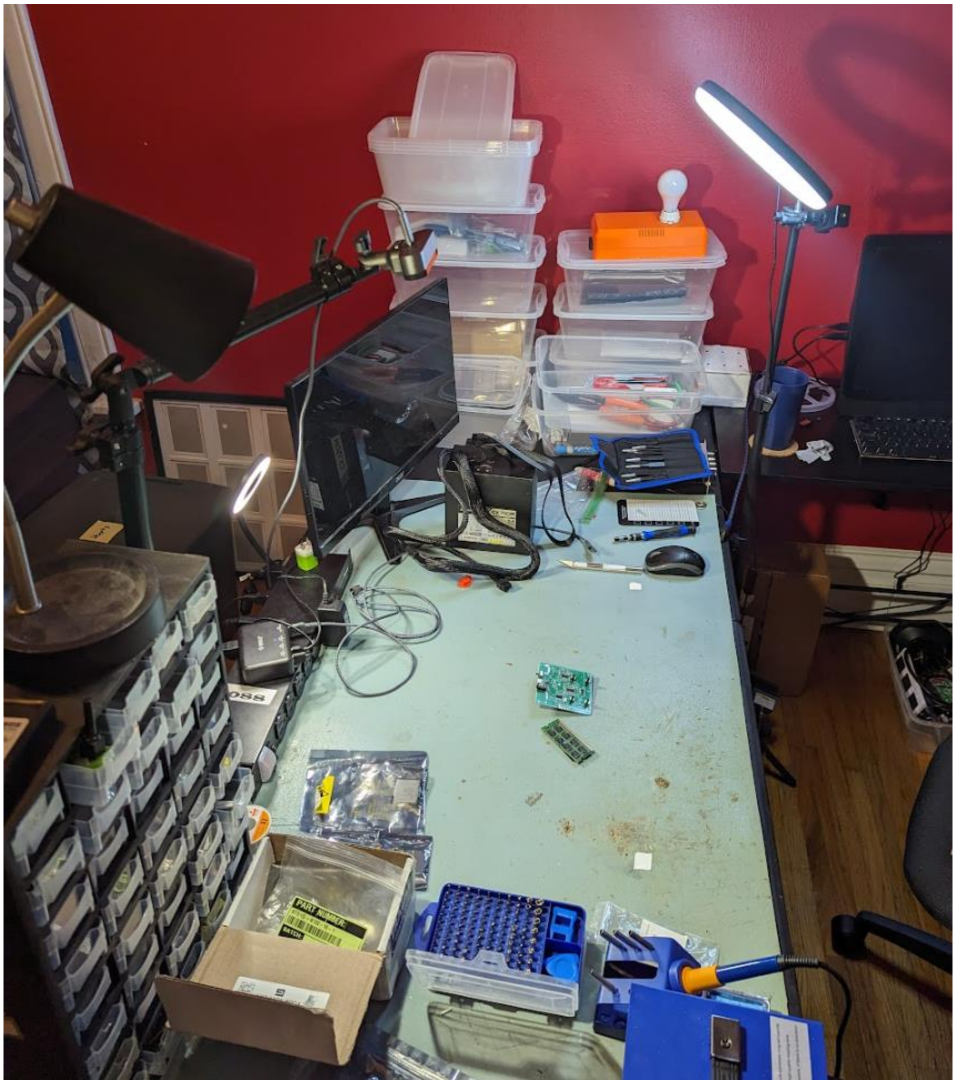


Fig 2. PCB Photo Taking Setup

Despite this effort, publicly available datasets proved even more useful as inputs for the program, as they were often higher quality and more varied than our limited subset of boards. Much of the preliminary development was done on circuit boards as captured with the previously mentioned setup, and many of the figures throughout this paper are captures taken with this setup. That said, those figures and inputs that did not come from this setup were from the annotated PCB dataset *FICS-PCB*. This dataset provided the ideal input criteria for component detection through machine learning, and the set acted as additional testing points for other functions such as trace detection and character recognition.

4. Methodology

As previously stated, the project can be broken up neatly into 6 sub-tasks. These are image stitching, finding damage, reading silkscreen, identifying components, valuing components, and finding traces.

a. Image Stitching

Image stitching is a necessary first step in order to generate images that are both large enough to cover the circuit board, but also high resolution enough that details like component lettering can be made legible. Our program uses an Open CV image stitching library called Stitcher to generate stitched images. Stitcher works by generating keypoints and extracting invariants between images. Once common pixels are identified, a homography matrix is estimated to transform the image from one size and perspective to another. A block diagram that demonstrates the functionality of Stitcher is provided in Figure 3.

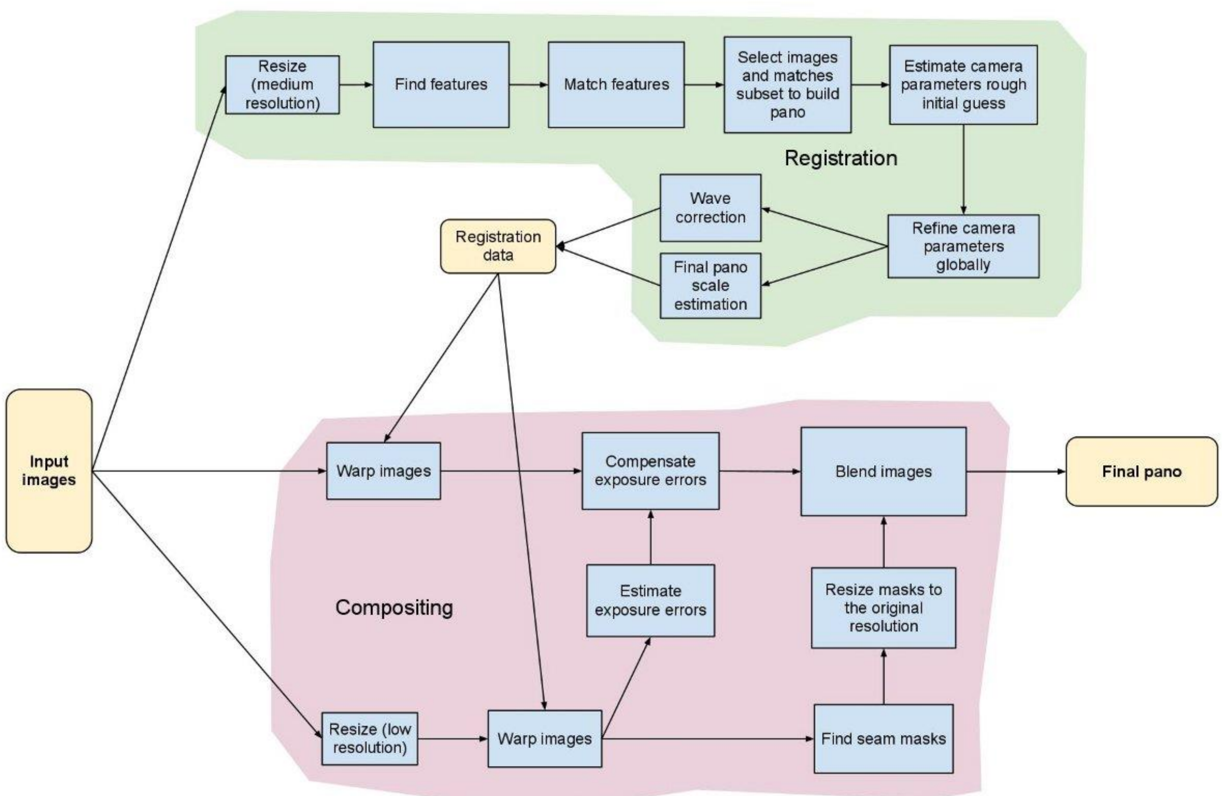


Fig 3. Image Stitching Block Diagram

Figure 4 shows a side-by-side comparison of the same circuit board section. The left image is a zoomed-in portion of a wide-angle shot, and the right image is a zoomed-in portion of a composite photo made by stitching many shots together. This demonstrates the key benefit of image stitching: A single ultra-high-quality image made from several standard-quality images. This is key for character recognition and trace detection.

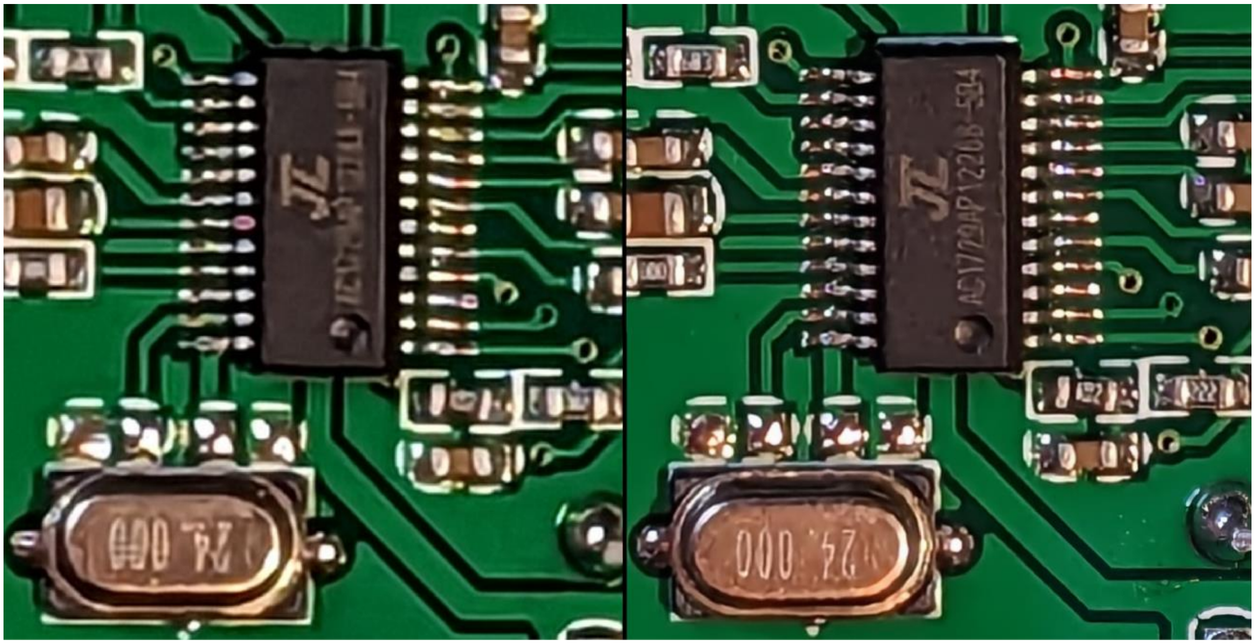


Fig 4. Image Stitching Comparison

b. Finding Damage

Damage to a board comes in a few types. As such, the program needs to be able to spot each of the types. The types that we will attempt to detect are corrosion, scratch, and heat. Note that the program does not need to distinguish between these types, only identify that damage exists.

The program performs a test for each of the types of damage. Corrosion damage is identifiable by a shift in the color of exposed copper areas. The color may turn brown with rust or white/green from battery acid exposure. Scratch damage can be identified as a long thin line that breaks a trace or component. Heat damage will manifest as brown or black scorched areas of the components or the board. The program makes no attempt to assess the severity of any damage, only to notify the user of its presence. Detecting damage is implemented by morphological operations and or discoloration detection. Figure 5 shows a side by side comparison between a PCB and an identical PCB with heat damage..

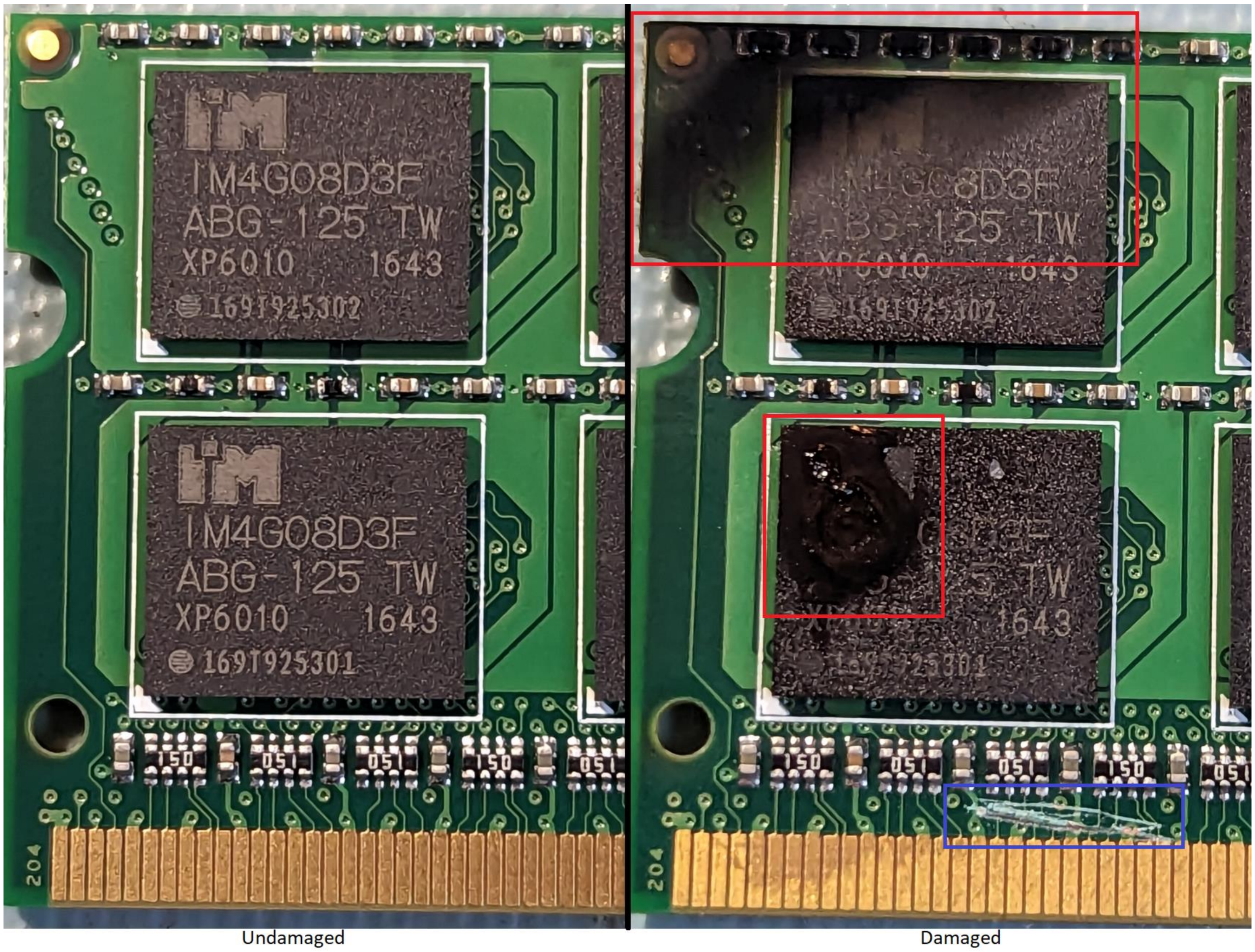


Fig 5. Board Damage Comparison

c. Identifying Components

For components, there are a few major types to identify. These are resistors, capacitors, inductors, diodes, transducers, and ICs. Other types of components are not particularly common, and most other components are not commonly supported by identification algorithms. The program that handles component identification takes advantage of machine learning to locate and distinguish between components.

We are using a neural network trained using YOLOv5 to identify components. Since our possible inputs to the program can be under a wide variety of lighting conditions and angles and from devices with different resolutions, we needed a wide dataset that could account for this range of inputs. Thus we used a subset of the FICS-PCB dataset that gives us annotations for the same component from different magnification levels. This would give our model the ability to deal with images from different distances from the board. The dataset also consists of images of boards in poor lighting conditions and ones where the backlight is too strong. We included these since we cannot anticipate the quality of images the user will give us.

Using a dataset of this size also requires a lot of resources to train. However, this overhead can be negated using Transfer learning. Transfer learning reduces required resources by utilizing the weights from a pre-trained machine learning model which was trained to perform similar tasks and reusing those weights to do slightly different tasks. We compared the performance with YOLOv5-nano, YOLOv5-small, YOLOv5-medium and yolov5-large. With this less resource-intensive training method, we still had to rely on Iowa State University High Performance Compute Cluster, Pronto to train our algorithm effectively.

The FICS-PCB dataset is divided into 32 different folders, where each folder consists of details of the front and back of one PCB. Each folder also contains images from different magnification levels. Thus, each folder has a DSLR subdirectory with annotations and an image of the entire board along with a "Microscope" directory that contains images with different magnifications. Overall we have 900 images and their annotations with roughly 10 annotations per file. We chose to go with 80% of the dataset for training, 10% for validation and 10% for testing. Figure 6 shows the distribution of classes in our dataset.

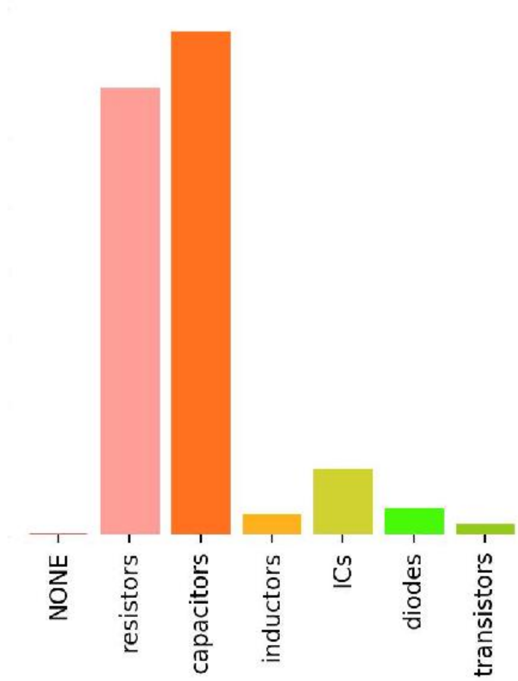


Fig 6. Distribution of classes in our subset of dataset

YOLOv5 accepts input in terms of an image and a text file, with the same name where the text file denotes the annotations in <class><x><y><width><height> where x, y, width, and height are normalized with respect the resolution of the image in their respective axes. However, the FICS dataset provides input in the form of CSV files where each line contains a JSON object of the class and location of each annotation on a particular image. Thus to prepare the data to train the YOLO model, we made a crawler that would go through and find the relevant annotation file and the relevant image folder and generate a YOLOv5 compatible text annotation file.

d. Reading Text

The target population is interested in ICs, particularly ICs of moderate to high value. Once an IC is identified, it is flagged for character recognition on its surface. IC manufacturers will brand each model with an indicator of its model number (Fig 6). This text is passed into the component valuation program which searches the web to find for indicators of value and notifies the user.

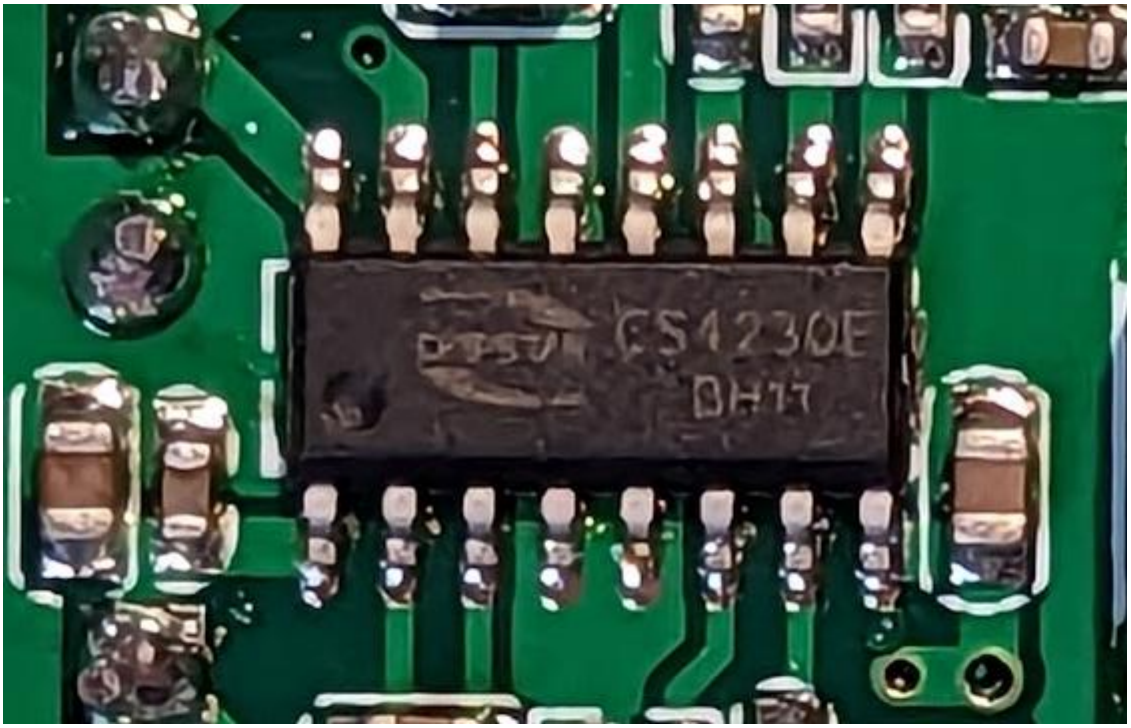


Fig 7. Example of an IC Surrounded by Capacitors

Resistors of sufficient size will also commonly support on-component text describing the value of the resistor (Fig 7). Typically, these numbers are a set of 3, the first 2 describing the 1st and second digit of the components value and the 3rd number describing the power factor of the first two numbers. For example, a resistor marked “103” would be a 10K resistor (10 x 10^3 ohms). This program reads and interprets these values, associating each value with its parent component for future reference.

Fig 8. A Combination of Resistors and a Capacitor

The program that reads the silkscreen and components uses Optical Character Recognition (OCR) to translate image text into strings that can be interpreted. There are 2 text sources expected: Silkscreen text and component text. Silkscreen text comes from the circuit board itself, whereas component text is printed or pressed on the component packages. It is important to distinguish between these types because the data contained within each needs to be handled differently. The text reading program relies on the component detection program to know what text came from a component, and what text came from the board.

Component text is used to verify the value or identifier of a component. Using the component identification program as an input, the text recognition program reads the text from within the component bounding box and associates that text with the component. This text can then be passed to the component valuation program.

Silkscreen text is used to verify the findings of the component identification program. For example, it would be safe to assume that if R201 is the highest number associated with the R letter printed, then no more than 201 resistors are present on the board. A noncomprehensive list of common reference letters is available in Appendix A. Because the component identification primarily targets resistors, capacitors, inductors, diodes, and ICs, the text recognition program focuses on these identifiers for verification purposes.

e. Valuing Components

The component valuation program is one of the most essential functions for the end user. The program receives text from the text reading program and uses a technique called web scraping to search eBay for similar, recently sold products. The incoming text is modified to fit eBay URL standards, and a request is sent to the appropriate URL that returns HTML data. This data is parsed (or scraped) for items of interest. This program focuses on gathering listing titles, sold prices, and sold dates. Critical information is stored in a CSV file, which may be up to 50 lines long depending on the results of the search. This is done by searching the HTML file for specific keywords that indicate where data of various types are stored on the webpage. Figure 8 is a visual representation of how the program gathers the data.

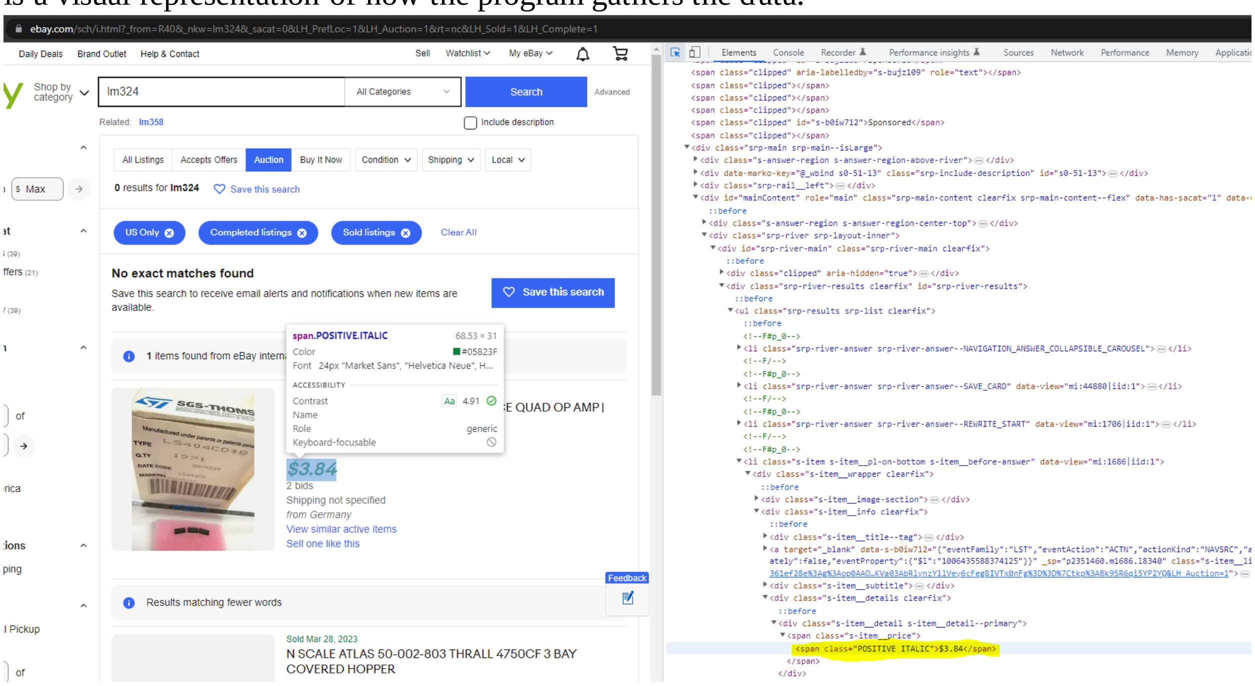


Fig 9. eBay Search Results

The data is filtered to only return results from exact results and results from international sellers in order to ensure quality in the returned results. This is done by scraping the page to find how many exact or international results were returned. The program then only returns as many results as are legitimate. Figure 9 shows the output of an example search that has 1 international result and several illegitimate results. The illegitimate results are correctly filtered out, leaving only the international result.

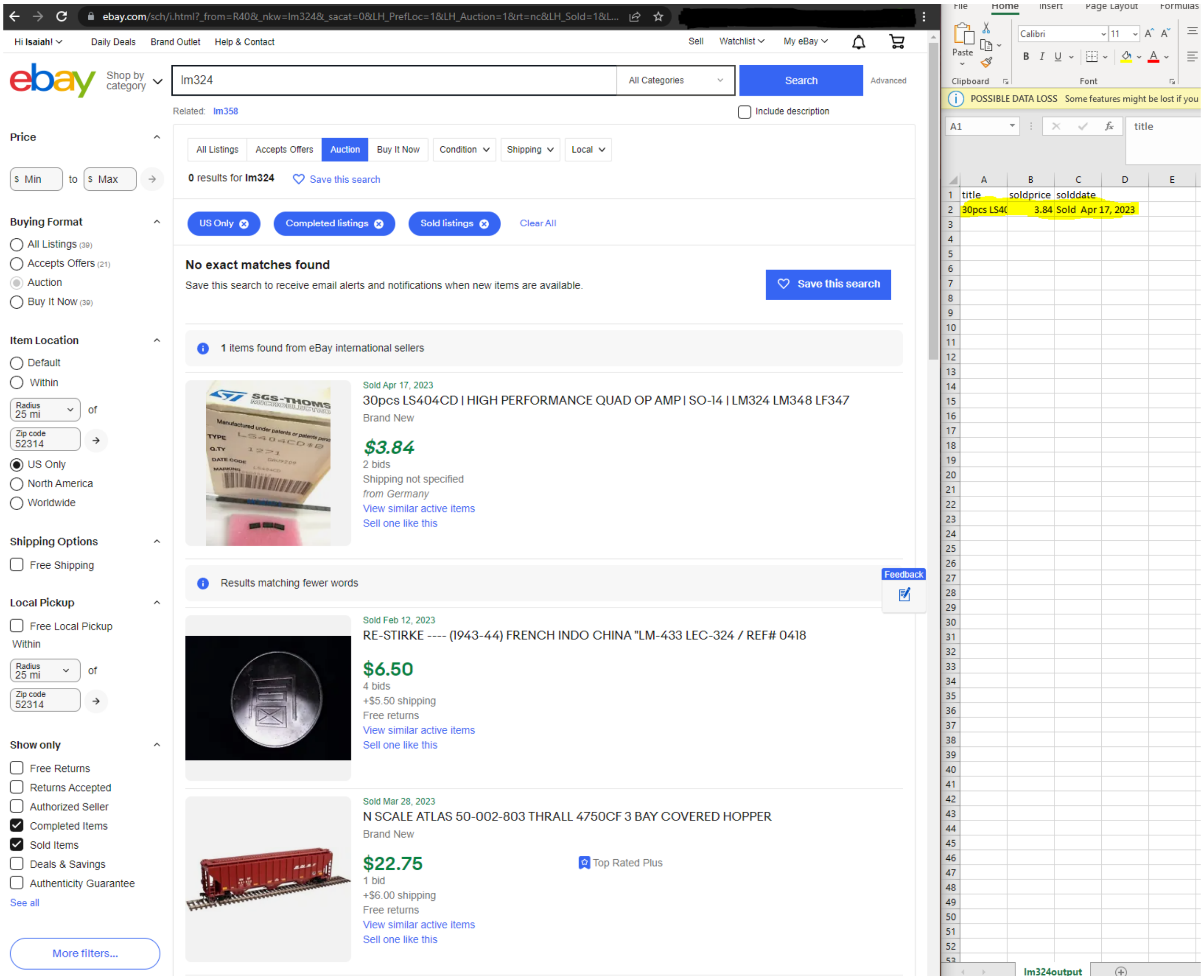


Fig 10. Component Evaluation Results

In the case of multiple legitimate results, the prices can be interpreted as a set of high, low, and mean values. The high values are of minimal value to the user. This is because often on eBay, results may contain items that have sold for excessive amounts without cause, or the program may return a search result in which multiple items are listed in a single listing. As a safeguard against outlier values, it is recommended that the user consider averaging out the results. In particular, the mode result may be of the most value to the user, as it provides the most likely price point at which their item will sell.

f. Identifying traces

Physically, a trace is a section of the board that contains copper plating under the surface, and the presence of the copper is visible through the thin top surface finish. Circuit boards may be made in many colors, but the most common color is green, so the program is tuned to work with green boards specifically. The program detects the edge of the board and takes a sample of the color close to the edge. This process takes advantage of the fact that circuit boards are rarely manufactured with copper traces too close to the edge due to process variation. The sampled color is then detected across the entirety of the board. Variations in the colors of the board are detected using an HSV color mask and interpreted as the presence or absence of traces. This combined with edge detection and maksing helps us detect traces.

5. Results

The results of each sub-task will be examined separately before discussing the overall results.

a. Image Stitching

The results of image stitching are overall very promising, but results can be mixed. Figure 11 shows an example of image stitching that has gone mostly well, but has a few major defects. This type of artifacting has a detrimental impact to trace detection, and is a natural part of the image stitching process. These artifacts are not common. In the above example, this imperfection affects only 2 traces in a composite image of an estimated 60 traces. This represents a failure rate of <5%. While this artifacting is not trivial, it is well within our pre-defined metric for success as long as subsequent processes are equally accurate.

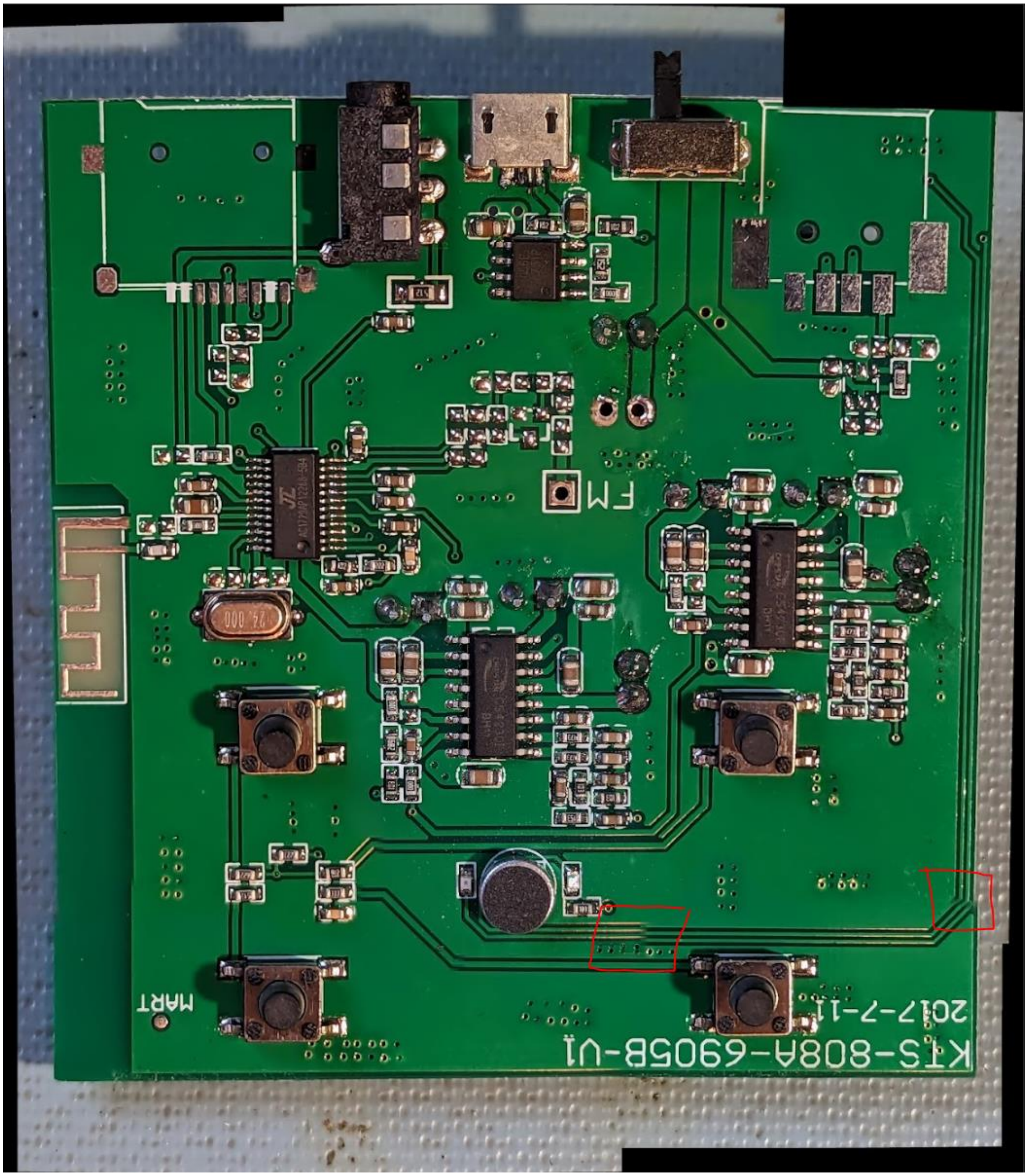


Fig 11. Stitched Image of a PCB. Error Areas are Bounded in Red

b. Finding Damage

We were not able to obtain any results in this section due to time constraints. Given our goals for the project, we would have expected a result similar to Figure 5 in section 4b. At a minimum, we would expect burn detection from the program and we would expect an accuracy of greater than 70%. Fortunately for the success of the project, the accuracy of this part of the program will not be measured. Unfortunately, there are no results to evaluate.

c. Identifying Components

Despite training on the Iowa State HPC Pronto, we had a difficult time being allocated the memory needed to train the neural network. This restricted memory limited the batch size, and resulted in poor neural network accuracy. Initially, we aimed to train each of YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x on the dataset. However, some of the models were incompatible with the image sizes that FICS-PCB provided. We trained 2 models on this dataset, YOLOv5m and YOLOv5l.

The YOLOv5m had a maximum achievable recall of 0.95 for 500 epochs with a batch size of 8. In order to meet accuracy goals, it was necessary to train the model for the maximum possible number of epochs. Attempts to training for 500 epochs on the cluster resulted in overshooting the maximum available time per session. Due to this complication, the training of the model was extremely limited. Team members tried training the model using GPU-parallelization and reshaping input images in an attempt to improve performance. However, each of these resulted in technical difficulties that we were unable to resolve given time constraints.

The model with the most training was trained for 160 epochs out of the desired 500 before timing out and had a component detection accuracy of 2%. This clearly falls well below our goal for component detection accuracy of 70%. The results of this level of accuracy can be seen in Figure 12. This figure illustrates the unacceptability of a 2% accuracy level. Many object types are detected where objects are not present, and many objects are undetected. In this figure, the colors of the bounding box correspond to those of Figure 6 for each object type detected.

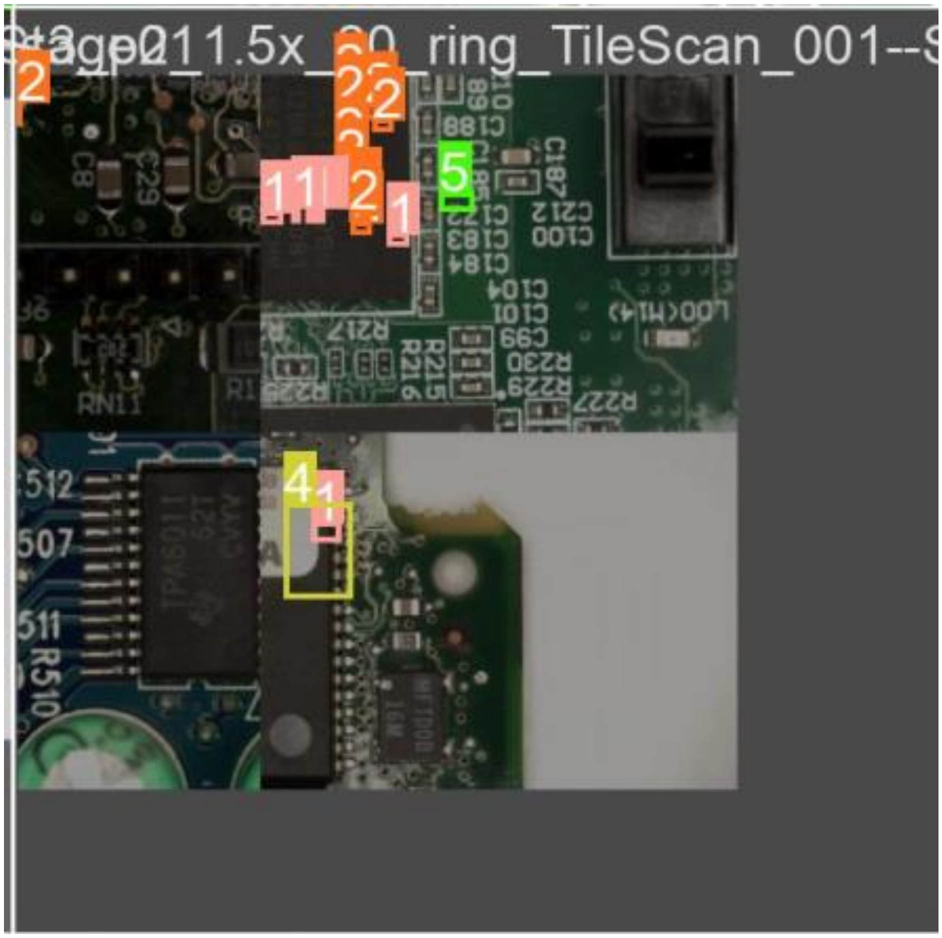


Fig 12: Result of running inference on the model trained using YOLOv5m

The model with the highest accuracy was a YOLOv5l model trained for 100 epochs. It had a maximum achievable recall of 0.99. The confusion matrix for this model is shown in Figure 13. A confusion matrix is used to pictorally depict the what percentage of predictions match the ground truth for each of the classes and what classes are misclassified. A confusion matrix can thus be used to calculate the precision and recall for a model. Precision can be interpreted as the ratio of objects that truly belong to a class divided by the number of true and false detections in each class. Recall can be interpreted as the ratio of items that belong to a class divided by the how many should have actually belonged to the class and the detections by the mode for the same. In Figure 14, FN stands for False Negatives, FP stands for False positives, TP stands for True positives, and TN stands for True Negative[8]. Figure 14 denotes the Precision-Recall vs Epoch curve for the YOLOv5 Large model.

$$\text{Precision} = \frac{tp}{tp + fp} \qquad \text{Recall} = \frac{tp}{tp + fn}$$

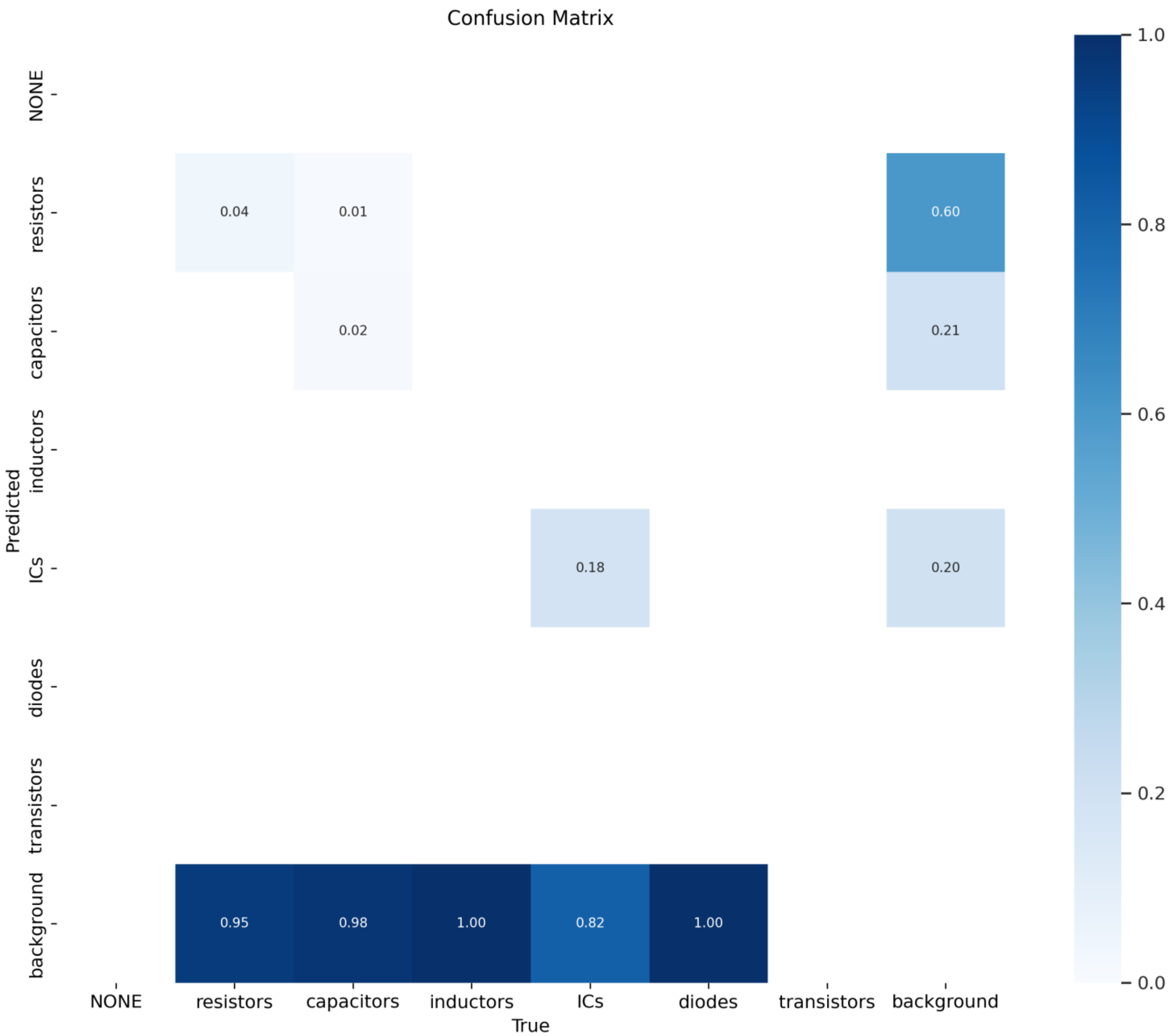


Fig 13: Confusion matrix for YOLOv5-large, 100 epochs

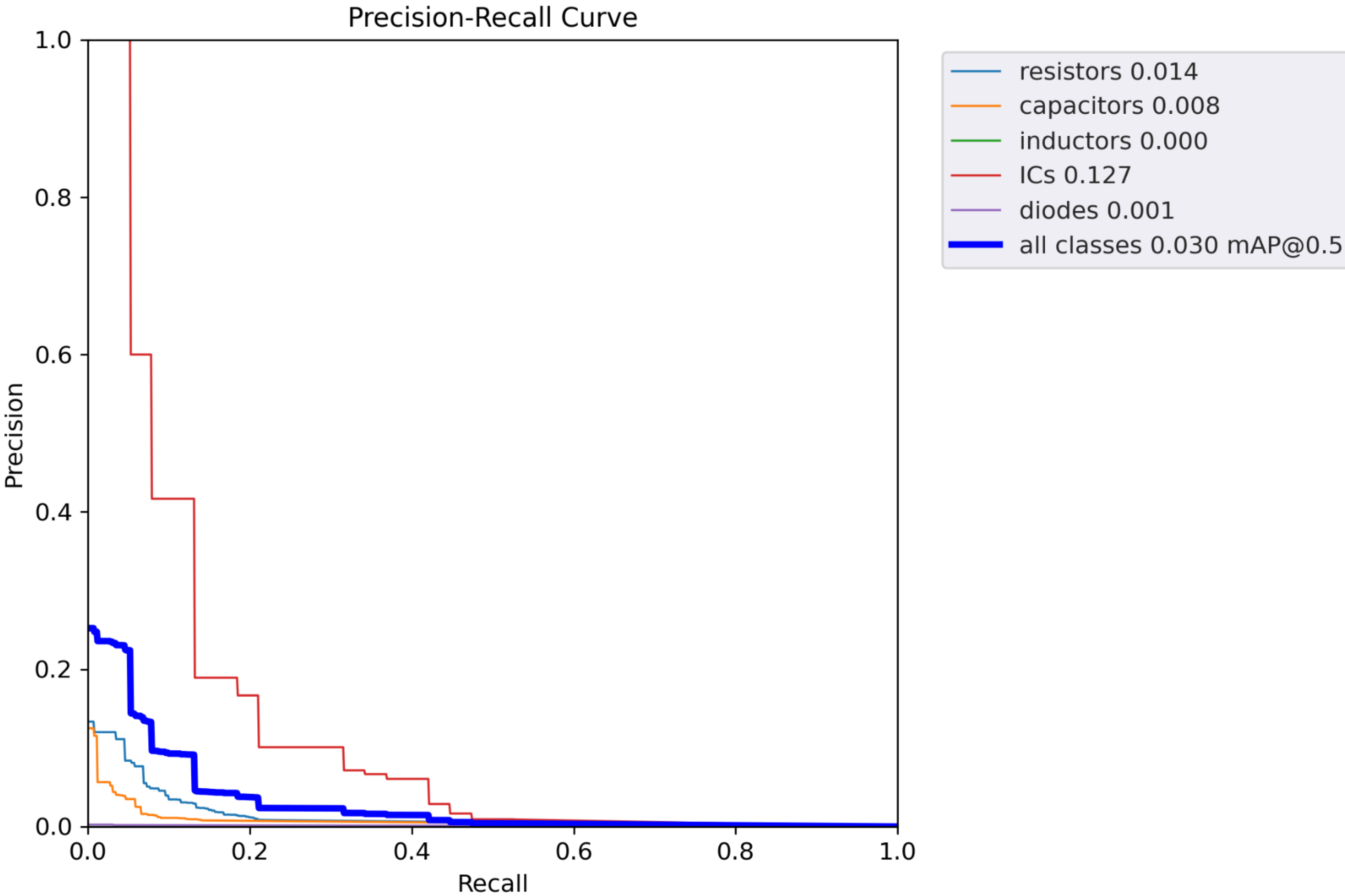


Fig 14: Precision-Recall curve for model trained using YOLOv5-large

The precision-recall curve shows the tradeoff between precision and recall for different thresholds, where high precision relates to a low false positive rate, and high recall relates to a low false negative rate. A model with high recall but low precision returns many results, but most of its predicted labels are incorrect when compared to the training labels. A modelwith high precision but low recall is just the opposite, returning very few results, but most of its predicted labels are correct when compared to the groudn truth. Thus, we have the best chance of acuurately detecting ICs and the worst chance of detecting inductors.

d. Reading Text

We tested multiple OCR libraries to accomplish this goal. All OCR libraries used one or two trained deep learning models on their backend to detect text and recognize it. We have reached partial success at reading text using the 'easyocr' library, with uses two deep learning models. Figure 16 is an example of the 'easyocr' library reading and outputting text from Figure 15.

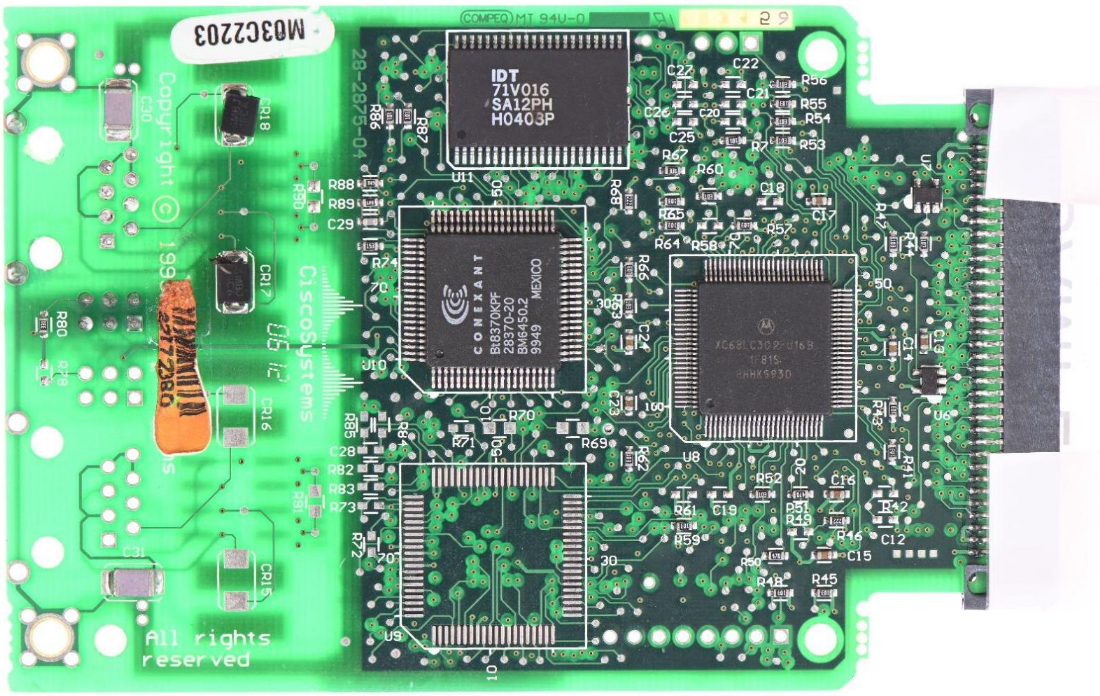
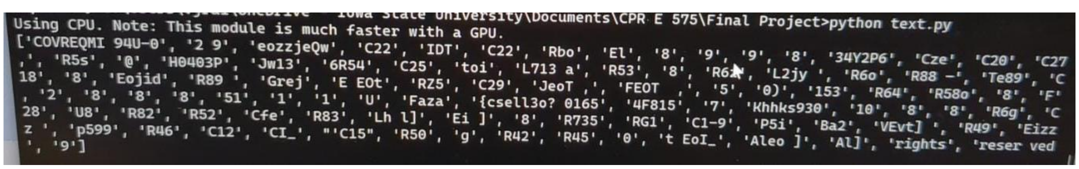


Fig 15: Input to EasyOCR



```
Using CPU. Note: This module is much faster with a GPU.
['COVREQMI 94U-0', '2 9', 'eozzjeQw', 'C22', 'IDT', 'C22', 'Rbo', 'El', '8', '9', '9', '8', '34Y2P6', 'Cze', 'C20', 'C27
', 'R5s', '@', 'H0403P', 'Jw13', '6R54', 'C25', 'toi', 'L713 a', 'R53', '8', 'R6', 'L2jy ', 'R6o', 'R88 -', 'Te89', 'C
18', '8', 'Eojid', 'R89 ', 'Grej', 'E EOt', 'RZ5', 'C29', 'JeoT ,', 'FEOT  ,', '5', '0)', '153', 'R64', 'R58o', '8', 'F'
, '2', '8', '8', '8', '51', '1', '1', 'U', 'Faza', '{csell3o? 0165', '4F815', '7', 'Khhks930', '10', '8', '8', 'R6g', 'C
28', 'U8', 'R82', 'R52', 'Cfe', 'R83', 'Lh l]', 'Ei ]', '8', 'R735', 'RG1', 'C1-9', 'P5i', 'Ba2', 'VEvt] ', 'R49', 'Eizz
z ', 'p599', 'R46', 'C12', 'CI_', "'C15", 'R50', 'g', 'R42', 'R45', '0', 't EoI_', 'Aleo ]', 'Al]', 'rights', 'reser ved
', '9']
```

Fig 16: OCR output

Table 1 below showcases some of the OCR libraries tested, their success and shortcomings.

Method	Description	Result	Faults
easyocr	OCR module that uses detection and recognition pre-trained models on its back-end	Partial Success	Depends on orientation of image. Wrongly recognizes certain characters
PyTesseract	Use wrapper module for Google Tesseract OCR that uses a pre-trained model in its back-end	Partial to Low Success	Too difficult to properly set up. Skips text when setup is correct. Wrongly recognizes text
keras-ocr	OCR module using keras with tensorflow as the back-end with pre-trained models	No Success	Heavy and time taking installation process. Internal libraries are buggy causing program to not run
simple-opencv-ocr	OCR module using opencv and KNNs	No Success	Sparse documentation. Needs to be retrained every time

Table 1: Partial list of OCR libraries tested

e. Valuing Components

The component valuation program provides acceptable results. Figure 17 is an example of a search result for “LM324”, a common variety of IC

	A	B	C
1	title	soldprice	solddate
2	30pcs LS404CD | HIGH PERFORMANCE QUAD OP AMP | SO-14 | LM324 LM348 LF347	3.84	Sold Apr 17, 2023
3			

Fig 17: Output of component evaluation

This result contains all of the necessary information and is true to the same search when performed manually. Some searches fail because no results are available, but the search program is true to manual search methods. The reliability of this function does not affect overall program performance based on our previously defined success criteria.

f. Identifying traces

The trace identification sub-task proved to be one of the more difficult to implement. Although the initial results looked promising, it was difficult to produce results that highlighted only trace locations. Figure 18 shows the results of 2 approaches to an original image. The left photo is the original capture. The middle photo is a detection of copper area (traces) with hardcoded color values. This gave the best result for trace detection and actually works very well, but it is not easily repeatable, and hardcoding values is not a best practice. The image on the right is an image generated by applying an adaptive threshold to the original image. The threshold does a good job of identifying areas of relative contrast and could be a key component in edge detection of traces in the final implementation of this module, but it identifies areas of the entire image, many of which need to be masked out.

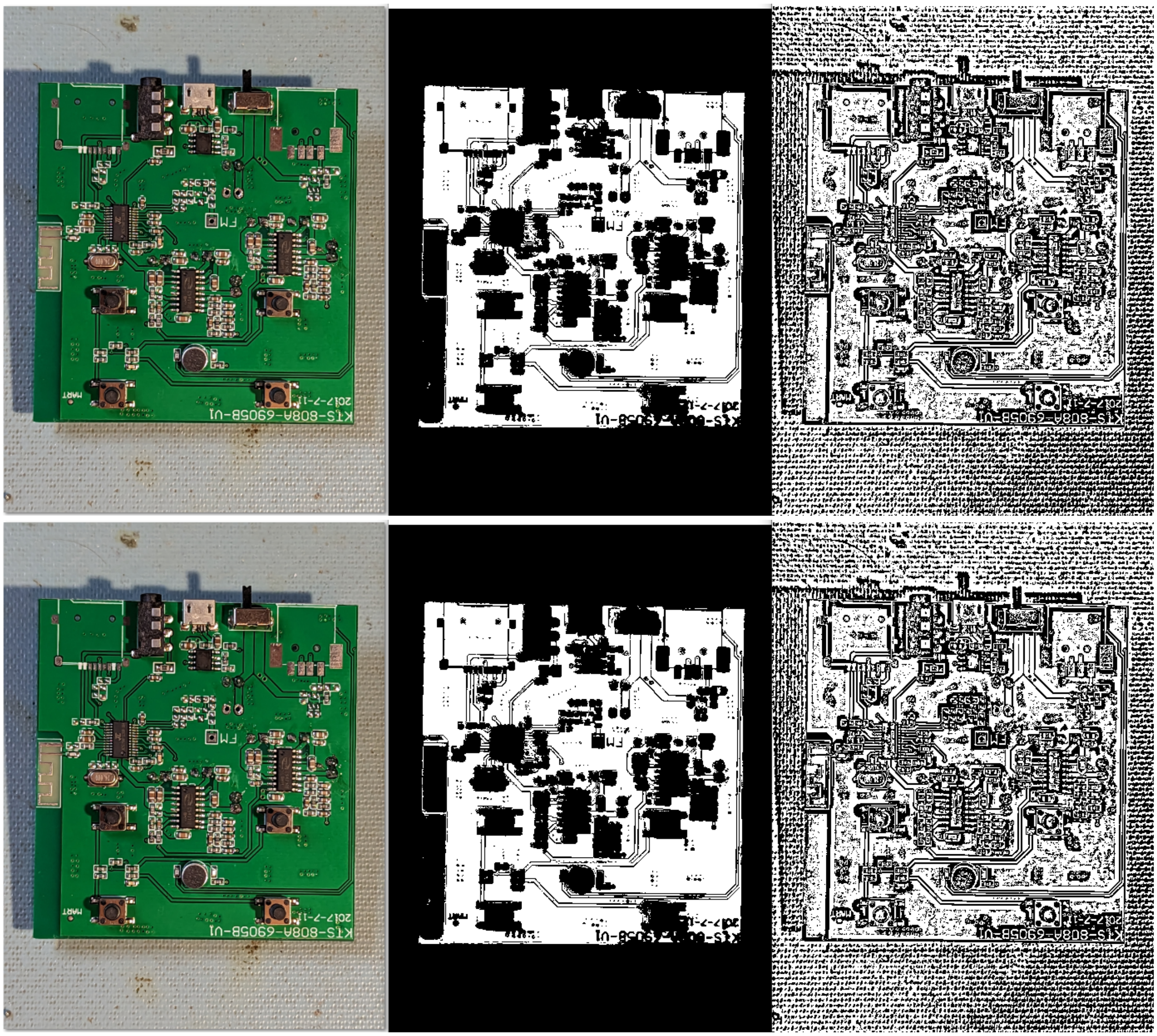


Fig 18: Original input and approaches to Trace Detection

6. Discussion

One of the major goals of the project was the detection of damage. Because this had to be cut for time, it will impact our determination of the success of the project negatively. While the success of the project does not hinge solely on the completion of this task specifically, it must be considered in the final evaluation.

Further, there are a few modules that produced results that we did not set targets for. These results are from the sections on component evaluation and image stitching. Image stitching was a key component in every following activity, and its accuracy is reflected in other accuracy measurements as a result. For example, trace detection relies heavily on the absence of artifacting within the image stitching process. In this way, the measurement of the accuracy of the trace detection is also a measurement of image stitching accuracy. For component valuation, This data is less dependent on other programs. The reason that this program did not have an accuracy target is that the concept for the program was limited in scope at the time of the project proposal. The results of component valuation can be questionable at times due to the nature of the data it gathers. eBay is not always the most reliable source for information, and some listings are inaccurate or invalid. That said, steps can be taken to mitigate the risk of mispricing components. Safeguards such as using average or minimum prices to judge component valuation can remove outliers and anomalies. Making conservative estimates of component value is the best safeguard against bad pricing information.

In the project proposal, the team targeted an accuracy of 70-75% in three categories: Correctly identified characters, correctly identified traces, and correctly identified components. This number was chosen to give a wide margin for error intentionally due to factors outside of the control of the team influencing the quality of the dataset being used. After testing, the accuracy for trace detection was around 92%. This value is in part due to the consistency in lighting conditions and manufacturing processes in PCB designing which helps reduce variations in color, etc. However, it’s really hard to account for errors that creep in due to the temperature of the light being used, camera parameters, and artifacts from image stitching. The experimental accuracy of component detection is 3%. The low accuracy is in part due to a lack of knowledge and skill we had with training neural networks and the poor structure and image quality in parts of FICS-PCB in the latter sections. The FICS-PCB dataset had images which ranged from 5260x2650 pixels to 100x50 pixels, which further complicated the training process since inputs to the network need to be of a fixed size (640x480 for YOLOv5). In regards to OCR, an accuracy of around 80% has been registered. This accuracy takes orientation into account. The errors are due to backlighting of the board as well as the similarity between characters in different fonts. Backlighting causes the silkscreen to become invisible to the program as they don’t contrast in the area. Characters like ‘l’ are similar to characters like ‘]’ to the program and thus it has a hard time distinguishing them. Based on our previous definitions and the amount of work completed, the project should be considered a partial success despite the challenges the project encountered.

7. Future Work

Each sub-task has opportunities for improvement.

The image stitching task could be improved by adding image stacking. Image stacking is a technique commonly used in astrophotography where images of various resolution and zoom levels need to be combined into a single image with the maximum possible resolution. This would be an excellent failsafe for our image stitching implementation, as currently if a wide shot of the board is provided, the program will ignore the zoomed-in shots and all the detail they contain.

The damage identification program was originally intended to detect damage from battery corrosion as well as scratch and burn damage, but limited access to boards with the corrosive type of damage made testing this capability impossible. An improvement could be made in this section by testing for the corrosion case and making adjustments as necessary.

The program that identifies components can be improved by increasing the size and variability of the dataset. According to the YOLOv5 documentation, the ideal dataset has over 1500 images per class with over 10,000 annotations for each of the classes. The documentation also mentions that 5% of images in the dataset should be just background to help reduce false positives. Currently, we do not have any background-only images. The FICS-PCB dataset is not well structured and more than half of the subfolders in the dataset are empty. Thus, a more comprehensive training sample will include data from multiple datasets and different settings so that the component detection is as accurate as can be.

The text reading program can be improved in different ways. One improvement possible is increasing the contrast between the text and the board. This would help the OCR is detecting and interpreting the text. Another possible improvement would be training a model on several images. This would allow for a more standard model with a singular font, as the Font is the same across most PCBs.

The program that values components could be improved in a couple of ways. The first improvement potential is to add a cached memory component to the program. This could be implemented by writing to another file with the search term and filtered price and date information. This file could be used as a lookup before the web search is performed to save time, or it could be used after a failed web search as a backup. It would also have the added benefit of allowing the user to set values for particular ICs that they care about for reasons other than monetary value.

The second possible upgrade for the valuation program is the addition of other websites. EBay was chosen for its ubiquity and ease of implementation, but other marketplaces such as Digikey and Mouser have more comprehensive information on component prices. Adding alternate marketplaces could give the user the option to search with a particular site, or it could balance information between results from each.

The trace identification could also be improved by allowing the program to work with additional board colors. Early on in the project, a strategic decision was made to focus on green boards as they are by far the most common. That said, off-color boards are becoming more common, and the ability to interpret the traces of these boards would further broaden the usefulness of the project.

Lastly, the user interface of the program could be improved by providing the user with a graphical interface as opposed to a command line interface. This would expand the user base to people who are not comfortable with using a CLI to interact with the application. Such an application will need to have separate sections where users could add images from the front of the board and the back of the board.

8. Conclusion

Machine-assisted Electrical Design Inspection aims to use neural networks, computation, computer vision techniques, and information commonly available on the internet to inform the user of the value of a PCBs components and the possibility of damage on the board. The application, written in Python, aims to reduce the time and cost of recycling electronic components and the learning curve for beginners. The project succeeded in its initial goals of functionality and accuracy, and many paths have been identified to make progress on the next revision of the program. Overall, the team is proud of the work they have done and what they were able to accomplish with the amount of time available.

Appendix A

C	Capacitor
D	Diode
F	Fuse
FB	Ferrite bead
J	Jack (least-movable connector of a connector pair)
JP	Jumper (treat as trace)
L	Inductor or coil
P	Plug (most-movable connector of a connector pair)
Q	Transistor
R	Resistor
S	Switch (all types, including buttons)
T	Transformer
TP	Test point
U	Integrated circuit (IC)
VR	Variable resistor
Y	Crystal or oscillator

References

1. Mitchell, R. (2018). *Web scraping with python: Collecting data from the modern web*. O'Reilly.
2. Jhnwr. (n.d.). *JHNWR/ebay-prices*. GitHub. Retrieved May 3, 2023, from https://github.com/jhnwr/ebay-prices
3. “The advantages of two layers PCB,” *What Makes Two Layers PCB Popular - ALLPCB.com - ALLPCB.com*. [Online]. Available: https://www.allpcb.com/two_layer_pcb.html. [Accessed: 08-Mar-2023].
4. S. Meyer, "A data structure for circuit net lists," 25th ACM/IEEE, Design Automation Conference.Proceedings 1988., Anaheim, CA, USA, 1988, pp. 613-616, doi: 10.1109/DAC.1988.14827.
5. Rosebrock, Adrian. “Image Stitching with Opencv and Python.” *PyImageSearch*, 17 Apr. 2021, pyimagesearch.com/2018/12/17/image-stitching-with-opencv-and-python/.
6. Glenn Jocher, Alex Stoken, Jirka Borovec, NanoCode012, ChristopherSTAN, Liu Changyu, Laughing, tkianai, Adam Hogan, lorenzomammana, yxNONG, AlexWang1900, Laurentiu Diaconu, Marc, wanghaoyang0106, ml5ah, Doug, Francisco Ingham, Frederik, … Prashant Rai. (2020). ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements (v3.1). Zenodo. https://doi.org/10.5281/zenodo.4154370
7. Lu, Hangwei & Mehta, Dhwani & Paradis, Olivia & Asadizanjani, Navid & Tehranipoor, Mark & Woodard, Damon. (2020). FICS-PCB: A Multi-Modal Image Dataset for Automated Printed Circuit Board Visual Inspection.

8. “Classification: Precision and recall | machine learning | google developers,” *Google*. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall. [Accessed: 04-May-2023].