Image Based Anomaly Detection - PCB Defect Localization and Identification

Introduction

Printable circuit boards (PCB) are fundamental components to many electronic devices where the quality of the device can meaningfully impact performance. With increasing PCB component density and complexity, manual inspection for defects becomes more and more inaccurate. This project focuses on using image processing and machine learning to automatically detect and classify defects in PCBs with the goal of being more accurate than manual inspection.

This project can be split into two core aspects, image pre-processing and localization where PCB images are standardized and split into features for classification, and classification where PCB features are actually identified by a specific defect type.

Related (prior) art

Detection of PCB Defects Via YOLO CNN Architecture
One paper makes use of the YOLO architecture in detecting PCB defects as YOLO can classify more than one object per image, which is critical to this problem as multiple defects can be found on a single PCB. With 11,000 images and an architecture with 24 convolutional layers and 2 fully connected layers, this paper achieves a classification accuracy of 98.82% via YOLO.

This paper makes use of a dataset with data collected from an AOI machine, with labeled data from QA engineers. YOLO works by predicting several bounding boxes with class probabilities using a CNN, which handles both the localization and classification aspect of this problem. (No specific cropping for features or pure image processing is used for localization of defects/features)

Via YOLO based architectures, both localization and classification can be accomplished with a high accuracy. As an expansion, there are many algorithms that generally outperform YOLO, such as fast RCNN as it generally has less localization errors than YOLO.

HRIPCB: a challenging dataset for PCB defects detection and classification

Another paper makes use of a two-stage approach in detecting PCB defects by utilizing extensive preprocessing in conjunction with a morphological operator for defect localization and a CNN for classification. The preprocessing steps behind this detector are registration and binarization. Registration is the process of identifying parts of the image; in this case it would be identifying the components of the PCB board such as the soutering joints and the wires between components. In this implementation, registration was accomplished by the usage of an algorithm called speeded up robust features (SURF) which extracts features from the test image and template image. The features will then be matched up so that a transformation can be determined to rotate and shift the test image to the same orientation as the template image. Registration is then followed by binarization which utilizes an "adaptive threshold segmentation algorithm" that determines the values needed to place a pixel as light or dark based on the luminance of a small region surrounding the pixel. The purpose behind binarization is to decompose the image into a simpler format which would outline the geometry of the PCB board as we are uninterested in the coloring or parts of the image not relevant to the PCB (like the background).

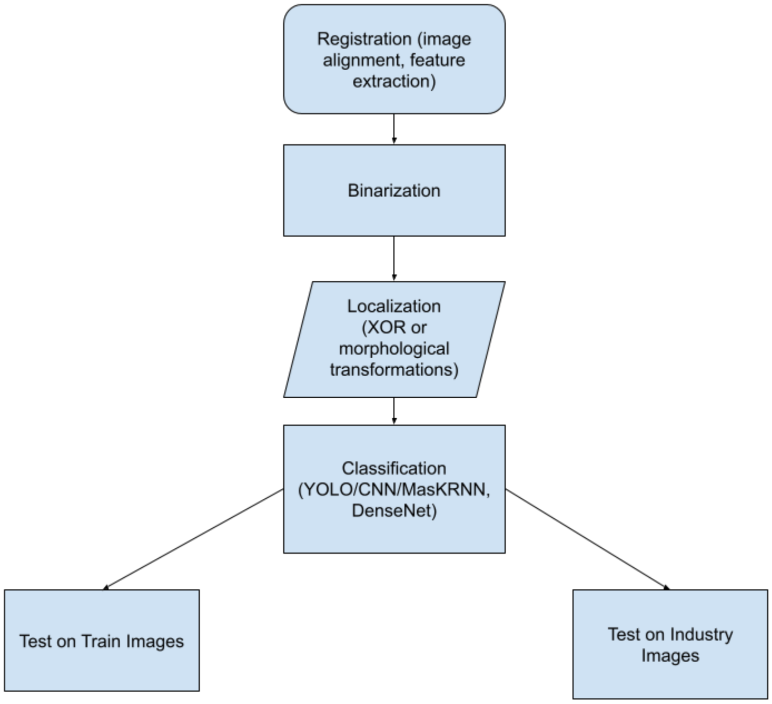
The next step of the architecture is the localization of defects performing an XOR on the test image and template. This step ensures that any features on the test image that are not present in the undefective template are only present. However, since the XOR result is typically noisy, additional median filtering and mathematical morphological processing are utilized.

The actual CNN classification that is then performed utilizes an architecture-style inspired by *Densenet* where a shortcut is present that connects later layers to earlier layers. The purpose of this is to prevent the gradient from approaching from the starting location again as is often characterized in deep neural networks. The architecture has two "blocks" with six convolutional layers that take all previous outputs of the block as an input. In between the two blocks there is a convolution and pooling layer to half the feature map, in addition to there being a convolution and pooling layer before the first block. The output of the last block will then go to a linear layer that classifies the defect region being examined.

The result of this architecture demonstrates it is highly effective on the synthetic PCB defect dataset it was trained and tested on. On the test data, the localisation accuracy was 0% while the classification accuracy was 97.74%. Additionally, the time required for the algorithm to analyze a single PCB image was 0.9899s with registration by far being the slowest step in the algorithm.

Project scope and major ideas/functionalities

The purpose being this project is to create an algorithm for the localization and classification of PCB defects. Unlike previous years, the primary objective behind this project is to build an algorithm that works accurately with industry-pictures of a PCB instead of synthetically altered images. The difference is that the synthetic PCB images are two dimensional and do not vary much in lighting and have a high picture quality. The industry-pictures of a PCB board face several challenges not present in these synthetic images: the three dimensional aspect introduces complexities such as lighting and reflection within the images, the low picture quality introduces a greater level of noise within the data, and less amount of data for the algorithm to be trained on compared to the synthetic PCB dataset. The algorithm for this year's project will utilize the ideas of preprocessing, localization, and classification that were present in previous projects but must modify these ideas to make them compatible with the challenges present when utilizing industrial images of a PCB. The goal of this project is to ensure the same accuracy and efficiency that was present in previous projects for synthetic PCB images but applied to actual industrial PCB images.


Registration (image alignment, feature extraction)
Binarization
Localization (XOR or morphological transformations)
Classification (YOLO/CNN/MasKRNN, DenseNet)
Test on Train Images
Test on Industry Images

Main approaches (2-3 alternatives) and data sources etc.

Approach 1: Semantic Segmentation for Localization + Classification via YOLO
With this approach, we use an algorithm capable of drawing bounding boxes with a classification (with classification accuracy) on defect features. Approaches would be using algorithms such as YOLO or Mask RCNN for segmentation and classification. This approach does not depend on a template to extract differences, but relies on pure train data where defect properties are learned, not the differences between a test and its respective template. This would also imply that test data must be in the format of multiple bounding boxes with classifications per defect in an image, not just a cropped picture of a defect with a single classification.

While this approach is simpler from an image processing perspective since the model itself handles the localization, the localization accuracy can potentially be quite poor without ample preprocessing. (especially since there are a lot of features that would need to be inspected with a PCB)

Data Sources:

1) PCB Synthetic dataset with 1386 images with 6 types of defects
https://www.kaggle.com/datasets/akhatova/pcb-defects

Approach 2: Defect Localization using XOR + Classification via CNN
The XOR operator can be used in this situation where we have an image of what the PCB should look like and an image
of the actual PCB. Taking the XOR of these images yields a resulting image that highlights the differences between the two. Ideally, this would mean the defects are more pronounced in the final image where we can conduct further analysis to localize the defect. The classification portion would be handled using a simple CNN. The number of layers and types of layers would be adjusted based on the need for classification. We could evaluate the accuracy of different neural network architectures and determine which one produces the best accuracy.

Data Sources:

1) This paper outlines the use cases for XOR in regards to image processing
https://www.iosrjournals.org/iosr-jce/papers/Vol17-issue2/Version-5/B017250715.pdf
2) Deep PCB with 1,500 image pairs with templates and aligned test images – useful if a template approach is taken where the differences are inputted into the model
https://github.com/tangsanli5201/DeepPCB

Main tasks, milestones, roles of each team member, and schedule

1. Image pre-processing **(4/1/23)**
    a. Define area of PCB

    b. Binary Mask to standardize all images
2. Image Enhancement **(4/18/23)**
    a. Sharpen PCB images to emphasize possible errors
    b. Filter out noise that could hinder predictions
3. Use one of the main approaches to determine location and type of defect **(4/29/23)**
4. Analysis of Results **(5/9/23)**

These four steps are the main milestones we need to accomplish in order to be successful. We have allocated the most amount of time to image pre-processing as we believe that this will be the most complicated and time consuming task. Additionally, the thoroughness of our image pre-processing will have a direct correlation to the success of our final algorithm to detect PCB defects. It is also important to note that the dates provided are tentative and are subject to change based on project needs.

Specific roles of each team member will be added when the project tasks become more apparent.

References

Adibhatla, V.A.; Chih, H.-C.; Hsu, C.-C.; Cheng, J.; Abbod, M.F.; Shieh, J.-S. Defect Detection in Printed Circuit Boards Using You-Only-Look-Once Convolutional Neural Networks. Electronics 2020, 9, 1547. https://doi.org/10.3390/electronics9091547

Huang, W., Wei, P., Zhang, M. and Liu, H. (2020), HRIPCB: a challenging dataset for PCB defects detection and classification. The Journal of Engineering, 2020: 303-309. https://doi.org/10.1049/joe.2019.1183

https://www.iosrjournals.org/iosr-jce/papers/Vol17-issue2/Version-5/B017250715.pdf

Open Questions

- Do we have access to templates for a template matching approach?