IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 7 October 2023, accepted 27 October 2023, date of publication 6 November 2023, date of current version 16 November 2023.

*Digital Object Identifier 10.1109/ACCESS.2023.3330142*

RESEARCH ARTICLE

# Toward Optimal Defect Detection in Assembled Printed Circuit Boards Under Adverse Conditions

**MOHAMMAD NOROOZI, JALAL GHADERMAZI, ANKIT SHAH, (Senior Member, IEEE), AND JOSÉ L. ZAYAS-CASTRO, (Senior Member, IEEE)**
Department of Industrial and Management Systems Engineering, University of South Florida, Tampa, FL 33620, USA
Corresponding author: Ankit Shah (ankitshah@usf.edu)

**ABSTRACT** Defects in the printed circuit board assembly significantly impact product functionality and quality. Automated optical inspection (AOI) systems, employed by manufacturing quality control teams, are designed to accurately detect these defects in a timely manner, thereby reducing the underkill (false negatives) and overkill (false positives) rates. An AOI system requires optimal settings for resolution, brightness, camera angle, and data variety to ensure effective defect detection. However, consistently achieving these ideal conditions in a manufacturing environment presents challenges. Our proposed framework enhances defect detection through data preparation and detection modules, effectively addressing these manufacturing challenges. We developed one- and two-stage object detectors and assessed their performance using precision, recall, and intersection over union metrics. Our framework employs a diverse range of augmentation techniques to effectively train the defect detectors, enabling the expansion of a limited data set. The trained detectors are evaluated using real-world data. We assessed quality control plans across various confidence thresholds. At a 65% confidence threshold, one-stage detector models did not exhibit any false negatives and had minimal false positives. The *You Only Learn One Representation (YOLOR)* model outperformed both one-stage and two-stage detectors, achieving 100% precision and recall, a 96% mIoU, and an impressive inference time of 11 ms, making it an ideal choice for high-production printed circuit board assembly lines.

**INDEX TERMS** Computer vision, faster-R-CNN model, image augmentation techniques, one-stage and two-stage defect detectors, PCBA defect detection framework, printed circuit board defects, YOLO object detection models.

## I. INTRODUCTION

A populated printed circuit board, often referred to as printed circuit board assembly (PCBA), represents a compact yet intricate electronic component. It is manufactured using a circuit board assembly process with several manual and automated steps. Many faults can occur in this process, such as overheating during soldering, wrong component placement, or power supply failure, which can produce defects in the circuit boards. There are two main categories of PCBA defects: functional (e.g., missing component or short circuit) and cosmetic (e.g., component scratch or component skew). A functional defect prevents the circuit board from functioning correctly, whereas a cosmetic defect impacts its quality [1]. Product quality control (QC) plays a critical role in identifying these defects in the PCBAs and thereby helps the manufacturers meet the product quality and functionality requirements of the customers while keeping the manufacturing costs down. The QC team comprises human inspectors and automated tools that aim to reduce the underkill (false negative) and overkill (false positive) rates by accurately identifying both functional and cosmetic defects in the PCBAs.

The typical inspection methods employed by the QC team include manual visual inspection (MVI) and automated optical inspection (AOI). In MVI, a skilled inspector visually inspects the PCBA to identify defects with the help of a magnifier or telescope. However, with an increase in the demand and production of PCBAs and a decrease in trace spacing and components volume, the QC team has to rely on

The associate editor coordinating the review of this manuscript and approving it for publication was Szidonia Lefkovits.

other automated tools to timely and accurately identify their defects. AOI is a machine-based automated visual inspection technique used primarily to check the defects in the electronic boards during the final production stage [2]. An AOI system captures an image of the circuit board, which is then analyzed by processor software. Different analysis techniques are used to differentiate between good and defective products. These techniques include template matching, in which the PCBA image is compared with the ideal PCBA, and pattern matching, in which both the good and defective images are first collected, and then statistical methods are applied to learn their features.

The AOI image capture system relies on the correct lighting source and positioning to ensure that the different areas of the circuit board are well lit and highlight the different types of defects [3], [4], [5], [6]. However, achieving this critical adjustment in the illumination setting (i.e., the angle and intensity of the illumination source) and the camera's viewing angle in the AOI system is challenging [6]. In addition, because of tight scheduling and the high production rate of PCBA products, collecting an extensive data set of PCBA images with a near-equal distribution of all types of defects in them is not feasible [7], [8]. A critical need exists to improve the AOI system by addressing the challenges mentioned above to reduce the underkill and overkill rates of the PCBAs.

In this paper, we propose a PCBA defect detection framework powered by computer vision (CV) algorithms that is robust to changes in the calibration of the AOI capture system. The main contributions of this study include the development of a novel framework with data augmentation techniques and defect detection algorithms to overcome the challenges mentioned above. The other contributions include providing insights for an effective QC strategy with a high detection rate and low false alarm rate (FAR). We conducted our experiments on a real-world data set and performed a comparative analysis of state-of-the-art CV-based defect detection approaches. Beyond the technical advancements, our research offers invaluable insights intended to revolutionize QC strategies. It equips practitioners with actionable insights to enhance their defect detection processes, promising transformative impacts on industry practices and quality assurance standards.

The rest of this article is organized as follows. In section II, we provide a detailed literature review. We explain the details of our proposed defect detection framework in section III. We present the numerical experiment setup in section IV. We provide the results and the analysis of the models in section V, followed by the insights obtained from this research study, and conclusions and future research directions in section VI.

## II. RELATED LITERATURE

There are three categories of automated inspection approaches for PCBs/PCBAs: referential, non-referential, and hybrid approach [1]. We first present the literature review for the referential methods, followed by the more efficient and popular non-referential methods. Thereafter, we highlight the identified limitations in existing literature, followed by the innovations in our approach designed to mitigate these shortcomings.

### A. REFERENTIAL APPROACH

The referential approach utilizes an image template of an ideal product (defect-free) to classify products as defective or non-defective using a difference threshold. An inherent drawback lies in the necessity for precise alignment of test images to template images in dimensions, angles, and contrast [1].

To address challenges in manual inspection, an automated referential method is introduced in [9]. This method compares a defect-free PCB standard image to a potentially defective PCB test image, employing a subtraction algorithm to identify regions of interest (RoIs). Specific image conditions, such as non-uniform illumination and camera angle tilting, are discussed in [10]. These factors led to the underperformance of the reference comparison approach in a study involving a PCB product with four defect types. In [2], addressing extended detection times in PCB referential inspections, a technology for online PCB defect detection is proposed, considering a limited number of defect types (three). The approach involves processing and binarizing color images, deriving a system self-inspection template from a reference image, and using an image aberration detection algorithm to segment threshold values. The method reduces detection algorithm time, optimizing hardware design for rapid PCB defect identification. Another referential method is the pixel subtraction technique. In [11], authors proposed a method for detecting and eliminating PCBA defects using visual subtraction technology, addressing challenges like missing, incorrect, and reverse detection in PCB component assessment. The method efficiently detects defective components in non-stop production lines.

### B. NON-REFERENTIAL APPROACH

In recent years, AOI has gained prominence in the electronics industry for detecting defective products, constituting a non-referential automated inspection approach [12]. AOI has successfully replaced MVI, boosting inspection speed and accuracy [13]. The proliferation of affordable computational power, especially cloud computing services, has integrated deep learning (DL) algorithms into AOI systems [14]. Since the advent of the AlexNet architecture in 2012, a majority of AOI systems have adopted convolutional neural network (CNN) designs [15], [16], [17], [18], [19], [20].

In [15], authors employ a real PCB dataset of 1540 images featuring only two defect types: short circuits and projections. The aim is to reduce the FAR in PCB defect detection. They propose feature pyramid networks (FPNs) combined with faster regions with convolutional neural networks (Faster R-CNNs) to detect PCB defects. Reference [16] addresses

low detection efficiency and high missed detection rates in defect detection methods. This model enhances the you only look once (YOLO) family of models, particularly YOLOv3, for PCB surface defect detection involving six common defect types. In [17], YOLOv3 is further improved by adjusting the output layers and anchor boxes to detect PCB electronic components. Reference [18] uses a deep ensemble self-adaptation method to detect six defect types on PCBs. The method involves a deep ensemble CNN model, achieving high detection rates and utilizing a self-adaptation technique for different production lines and environmental variations.

Following the improvement of the YOLO family of models, [19] presents a DL model based on YOLO for PCB quality inspection. The method employs skilled engineers to record and label defective PCBs, training a YOLO/CNN model to detect defects. Authors in [20] propose a lightweight PCBA-type defect detection model, encompassing four defect types. Their model integrates two sub-models: LD-PCB, for real-time defect detection and accuracy enhancement, and a character recognition model known as CR-PCB, notably improving irregular character recognition accuracy. Reference [21] demonstrates DL implementation in assessing PCB quality via X-ray imaging. Addressing challenges of noisy RoIs and variable imaging sizes, the study introduces and compares two AI-based models. Real-world 3D X-ray experiments substantiate the effectiveness of the proposed methods.

Several studies have employed DL models to address defect detection across diverse applications, aiming to replace human inspectors with automated systems [22], [23], [24], [25], [26]. For instance, Cha et al. [23] introduced a methodology rooted in vision-based DL to identify cracks in concrete and steel surfaces. Another approach, reliant on a Faster R-CNN, has been advanced to identify five distinct defect categories within remote video footage [24]. In the work by the authors in [25], a dual encoder-decoder solution known as the Polyp Segmentation Network (PSNet) was introduced to detect colorectal polyps. The pixel-wise segmentation approaches are mainly utilized in literature when the problem requires precise object boundaries, instance separation, and detailed shape information.

Challenges in DL algorithms for PCB/PCBA defect detection are addressed in [27], [28], [29], [30], and [31]. Reference [27] tackles issues like template design, computational costs, and noise susceptibility by proposing a method for identifying six defect types. To enhance small defect detection, they use a deeper model backbone and replace region proposal network (RPN) with guided anchor RPN (GARPN). In [28], data augmentation using affine transformations mitigates data scarcity and diversity issues, followed by CNN-based model training. Reference [29] addresses low automation, detection rate, and stability problems. Reference [30] introduces as single shot detector (SSD) method achieving 94.69% accuracy, though performance is limited for small defects. Reference [31] recognizes limitations of common object detection techniques for dense PCBA components and proposes an approach exclusively focused on electronic component identification. They compare their approach with region-based CNNs (RCNNs) and SSDs, revealing limitations.

### C. RESEARCH GAPS IN LITERATURE

In our review of the existing literature, we observed the following shortcomings: (i) The literature studies assume that the images in the data sets are acquired using AOI image capture systems, operating under optimal conditions for resolution, brightness, camera angle, and distance. (ii) Many studies primarily explore the detection of a limited number of defects on manufactured boards. These investigations predominantly concentrate on the most common defect types encountered in PCBs. (iii) It is assumed that balanced data sets, crucial for effectively training CV-based defect detection models, are readily available. These data sets are presumed to consist of nearly equal proportions of different defect types. Nonetheless, these assumptions do not align with the complexities of real-world manufacturing settings. Firstly, ensuring consistent optimal calibration of equipment, such as high-definition cameras and lighting systems, for collecting consistently high-quality image data over time is far from guaranteed. Secondly, within the realm of densely populated manufactured PCBAs, the prevalence of diverse defects is a reality. Thirdly, acquiring balanced data sets with a sufficient quantity of image data showcasing proportional representation across all defect types remains an intricate endeavor. To address these challenges, our research introduces a defect detection framework aimed at enhancing the quality control process, ultimately reducing both underkill and overkill rates associated with PCBAs. We employ various data augmentation techniques to expand the PCBA image data set and develop defect detection models for accurate and timely identification of defective PCBAs.

## III. DEFECT DETECTION FRAMEWORK

A schematic of our proposed defect detection framework is shown in Figure 1. The framework comprises two main modules: (i) the data preparation module and (ii) the defect detection module. The data preparation module preprocesses the collected data using various problem-specific techniques and prepares the data set to be utilized by the next module. The defect detection module consists of various computer vision (CV) algorithms for developing the defect detection model using the prepared data set from the former module. Next, we describe these modules in detail.

### A. DATA PREPARATION MODULE

The images collected by the AOI image capture system are processed using the data preparation module. These images are generally not captured in ideal settings with the optimal resolution, brightness, camera angle, and positioning. Hence, in this module, we first preprocess these images by

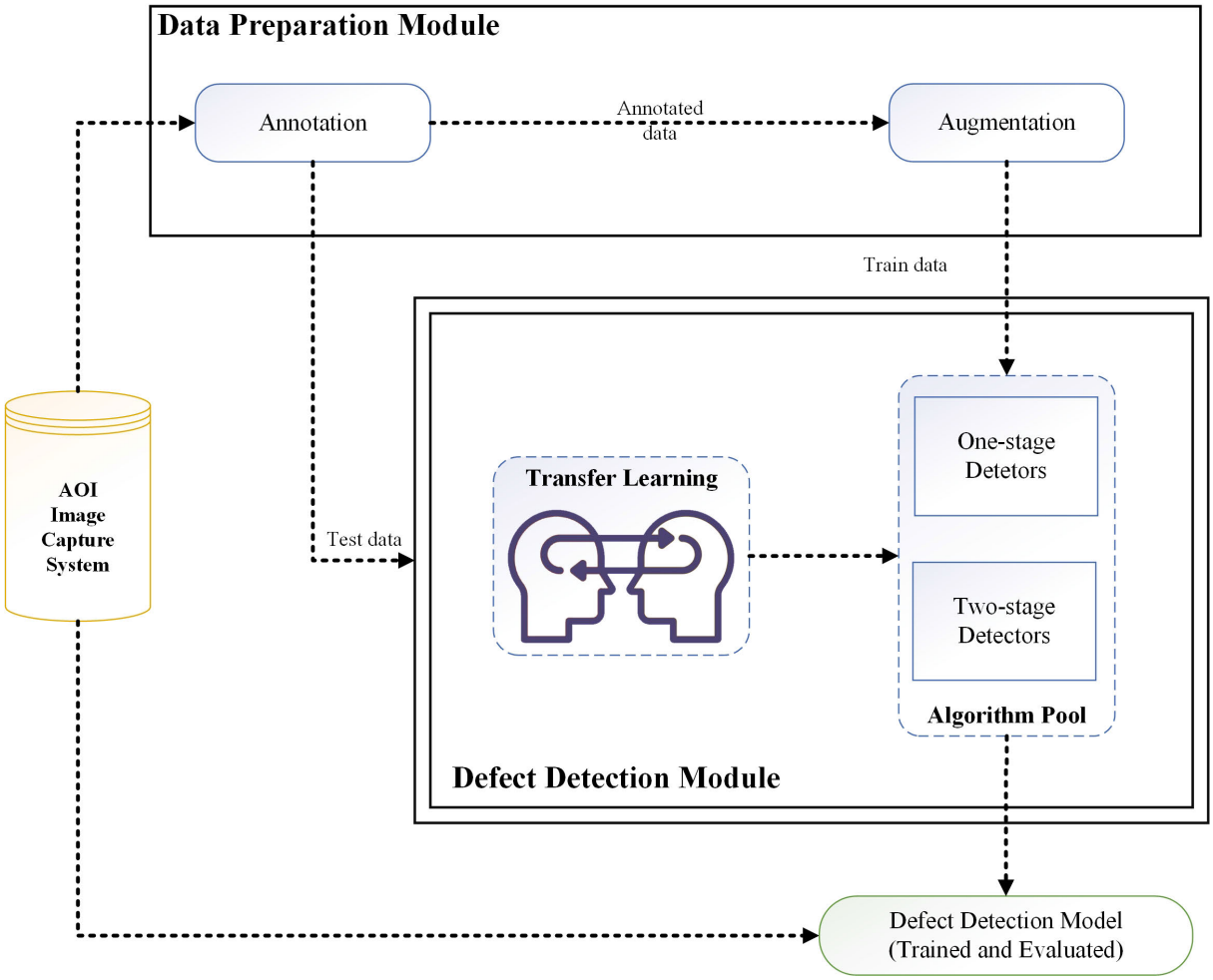



**FIGURE 1.** Defect detection framework.

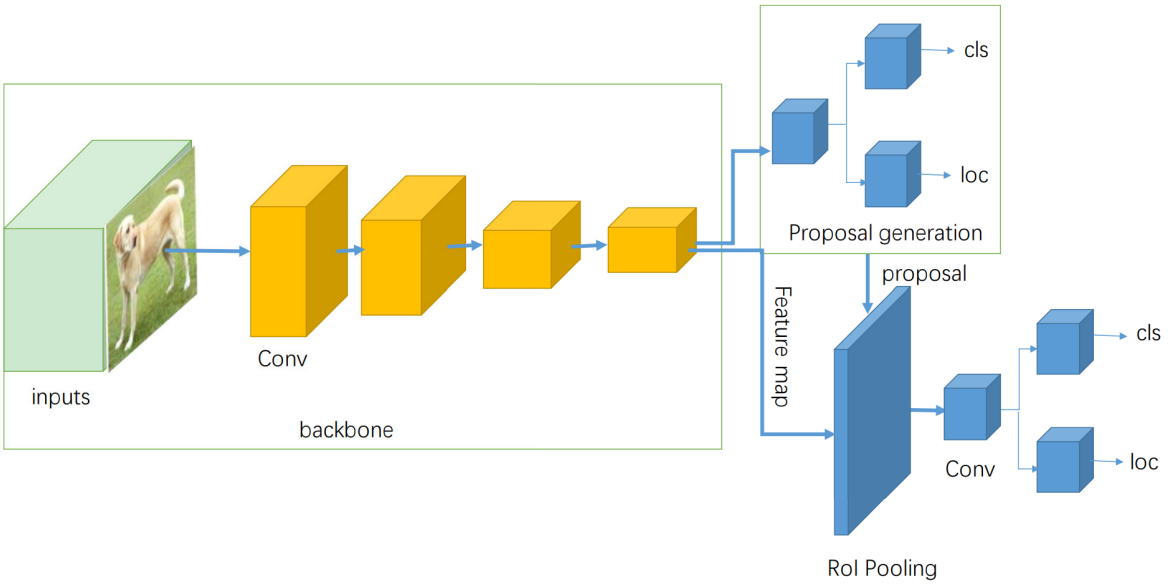
identifying the types and locations of the defects in them with the help of expert QC team members. We then split the labeled (annotated) image data into training and testing data sets. Next, we augment the training data set with artificial images to help make the defect detection models (in the next module) more robust in detecting defects in real-world manufacturing environments. The data preparation module consists of the following components:

- **Annotation component:** Image annotation is a primary preprocessing step for developing CV-based detection models. Annotation allows computers to gain a high-level understanding of digital images. Annotation files identify the types of defects in the images and specify their respective coordinates.
- **Augmentation component:** Data augmentation techniques help expand a training data set by creating modified versions of the images [32]. These techniques rely on altering image pixels to create synthetic images, which are then used to train the CV-based models to make them more robust to variations in defects. The augmentation techniques are problem and data set specific. For this research problem, some of the augmentation techniques that will be explored are described as follows [32].

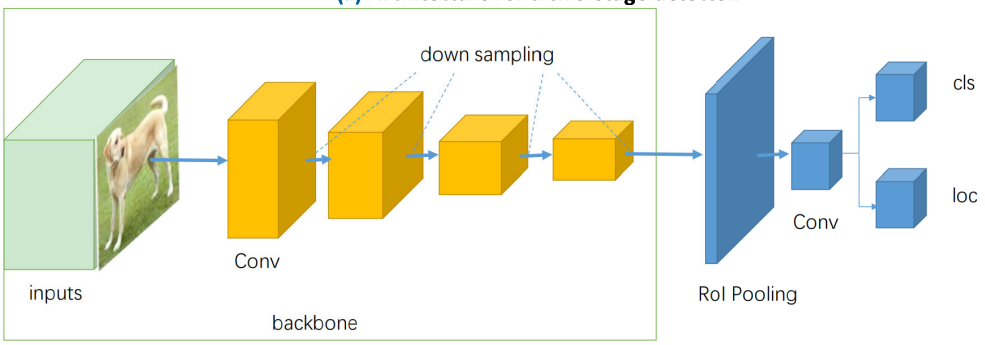
**Color space transformation**: The variations in the intensity of light are due to changes in the illumination and angle of the light source. To make the models less sensitive to variations in light and color, we will explore the color space transformation. This will be achieved by altering the RGB and HSV values in the raw images.

**Rotation**: The images may have been taken from different camera angles as well as the products may not be fixed in the same position. This would result in different orientations of the components. With the rotation transformation, an image is rotated along an axis or the center of the image. We will explore this augmentation technique to make the models less sensitive to identifying objects (defects) with their specific orientations.

**Kernel filters**: Due to the tilting in the AOI image capture system, there may be some noise appearing in the product images. By exploring kernel filters to




**(a) Architecture for a two-stage detector.**




**(b) Architecture for a one-stage detector.**

**FIGURE 2.** Architectures of one-stage and two-stage object detection models [33].

blur or sharpen the images, such sensitivities can be decreased.

The augmented training data set containing annotated images is then passed on to the defect detection module.

### B. DEFECT DETECTION MODULE

We consider both two-stage and one-stage object detection models in this study. Two-stage detectors have high localization and object recognition accuracy, whereas the one-stage detectors achieve high inference speed. The two stages of two-stage detectors can be divided by an RoI pooling layer. For instance, in Faster R-CNN, a RPN, proposes candidate object bounding boxes in the first stage. In the second stage, features are extracted by RoI pooling operation from each candidate box for the classification and bounding-box regression tasks [33]. On the other hand, one-stage detectors skip the region proposal stage of two-stage models and run detection directly over a dense sampling of locations. Figure 2a shows the basic architecture of two-stage detectors, and Figure 2b exhibits the basic architecture of one-stage detectors. The main difference between them is that in the two-stage object detector, the models employ region proposal network to feed region proposal into classifier and regressor for object recognition, whereas, in the one-stage object detectors, the models predict bounding boxes from input images directly [33]. Next, we describe the popular and top-performing object detection models from both the categories, which will be explored in the algorithm pool.

1) **YOLOv4:** You Only Look Once (YOLO) object detector family of models are one-stage object detectors known for their high speed and accuracy [34]. YOLOv4 is one of the popular models known for achieving state-of-the-art performance on the publicly available COCO image data set [35]. The object detection task in this model is divided into two parts. The first part uses regression to identify object positioning via bounding boxes and the second part uses classification to determine the object's class. The implementation of YOLOv4 uses the Darknet framework [36].

2) **YOLOR:** You Only Learn One Representation (YOLOR) is a novel object detection model [37], which proposes a unified network to encode implicit and explicit knowledge. It achieves its explicit knowledge based on features obtained from the shallow layers in the neural network and implicit knowledge from the features obtained from the deep layers. It uses Scaled YOLOv4 CSP for the explicit model [37].
3) **Faster R-CNN:** R-CNN object detection family comprises the most popular two-stage object detection methods. The R-CNN family of models includes R-CNN, Fast R-CNN, and Faster R-CNN. First, Girshick et al. [38] proposed R-CNN method which utilizes selective search in order to extract around 2000 regions from an image. These regions were referred to as ''region proposals'' by the authors. The Fast R-CNN method was later proposed by Girshick [39], which addressed some of the drawbacks of the R-CNN model. The Fast R-CNN method generates a convolutional feature map by feeding an input image to the CNN and based on the convolutional feature map, region proposals are identified. Nevertheless, both R-CNN and Fast R-CNN were time-consuming due to selective search. Consequently, Shaoqing Ren et al. [40] developed the Faster R-CNN object detection algorithm, in which a CNN network learns the appropriate region proposals, thus eliminating the need for the selective search algorithm. Faster R-CNN uses three neural networks: feature network, RPN, and detection network. The feature network aims to find the main characteristics of objects. RPN aims to generate several bounding boxes called RoIs that are highly likely to contain any object. Finally, in the detection network (also, referred to as the RCNN), the class and bounding boxes are created based on the inputs of both the feature network and RPN [40].

Transfer learning is the ability to transfer the knowledge obtained from solving a different but related problem to an existing problem. In the context of the proposed study, transfer learning is expressed as the retraining of the CNN model, which is already trained on another related data set and utilizing its trained network weights to quick start the training process for the problem proposed in this study. We will use models pre-trained on a large-scale image data set, COCO [41], including 80 classes of objects in approximately 330,000 images. Next, we describe the metrics for performance evaluation of the defect detection models with respect to the research objective of reducing the underkill and overkill rates of the PCBAs.

### C. PERFORMANCE EVALUATION METRICS

The true positive (TP) value represents the number of defects accurately detected, the false positive (FP) value represents the number of defects inaccurately predicted, and the false negative (FN) value represents the number of actual defects that the object detection model misses. The metrics used by the proposed object detection framework are as follows.

1) The precision metric measures the rate at which the model correctly predicts the defects. The precision is calculated as

$$\text{Precision} = \frac{TP}{TP + FP} \tag{1}$$

2) The recall metric measures how well the model finds the defects. The recall is calculated as

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2}$$

3) The false acceptance rate (FAR) metric [42] measures the fraction of false defect detection when there are no defects on the PCBA. The FAR is calculated as

$$\text{FAR} = \frac{FP}{TP + FP} \tag{3}$$

4) The intersection over union (IoU) metric evaluates the accuracy of the predicted bounding boxes when compared to the ground truth in an image.

$$\text{IoU} = \frac{B_p \cap B_g}{B_p \cup B_g}, \tag{4}$$

where $B_p$ denotes the prediction area and $B_g$ represents the area corresponding to the ground truth of the object. The average metric value across all classes is the mean IoU, also denoted as mIoU.

## IV. EXPERIMENTS

In this section, we first describe the data set used for the experiments, followed by the data preparation steps and model setup.

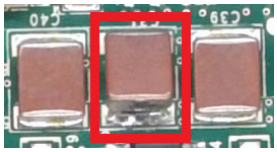
### A. DATA SET

We conducted our experiments using a real-world PCBA image data. These product images were captured with different settings of the optical equipment in the AOI capture system. The characteristics of the captured image data are as follows. The images were captured from varying distances. The camera angle was varied resulting in different perspectives of the image data. There was also a variation in the brightness of the images. The products were not fixed in the same position and direction. There were some images that did not have the complete PCBA and there was a variation in the background as well.

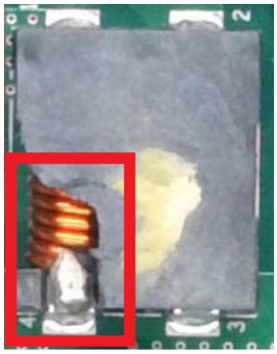
Table 1 shows the details of the data set. The image data had a total of 309 images, of which 270 belonged to the defective classes and 39 of them were non-defective. There were 19 different defect types (classes) associated with the defective product images. These contained both functional and cosmetic defects. The defective product images had either single or multiple defect types. Out of the 270 defective images, 15 of them had two defect types while the others had only one defect type present. There were 15 images of each defect type in this data set. We selected this data set

**TABLE 1.** Details of the image data set.

| Total images | Non-defective products images | Defective product images | Defect types (classes) | Images per class | Total number of defects | Image size (in pixels) | |
|---|---|---|---|---|---|---|---|
| | | | | | | Non-defective products | Defective products |
| 309 | 39 | 270 | 19 | 15 | 285 | 3840*2160 | 2048*1536 |



**(a) L26 Damaged**



**(b) C31 Raised**

**FIGURE 3.** Instances of defect types (red rectangular box indicates the defect position).

for our experiments as it had all the characteristics that could be found in a real-world manufacturing environment, which include non-ideal settings for resolution, brightness, camera angle, distance, and very few samples of a large number of defects.

Table 2 shows the types of defects and their categories. The defect type title contains the component and the description of the inconsistency. Each component is identified by a reference designator. For example, "C31 Raised" means that the component is raised from the designated position. Similarly, "Missing" denotes the absence of a component in its designated spot, "Damaged" refers to a deformed component, and "Displaced" indicates that the component is not precisely placed in its intended location. Out of the 19 defect types in the data set, two existed together, namely, R20 Damaged and Screw/Cables Solder. Figure 3 shows samples of two defect types along with the locations of the defective components.

### B. DATA PREPARATION

Data preparation is one of the main modules in the proposed framework. There are various steps taken to prepare the data so it can be used to develop the CV-based defect detection models in the next module. The data preparation process is described as follows.

**TABLE 2.** Defect types and categories.

| Defect | Defect class | |
|---|---|---|
| | Cosmetic | Functional |
| C31 Raised | ✓ | |
| D20 Missing | | ✓ |
| D3 Displaced | | ✓ |
| C39-Displaced | | ✓ |
| R20 Damaged | | ✓ |
| U13 Solder Balls | | ✓ |
| C55 Extra Component | | ✓ |
| J6 Damaged | | ✓ |
| U14 Short Circuit | | ✓ |
| C76 Missing | | ✓ |
| L8 Damaged | | ✓ |
| U18 Foreign Component | ✓ | |
| C119 Damaged | | ✓ |
| L26 Damaged | | ✓ |
| U26 Short Circuit | | ✓ |
| C139 Missing | | ✓ |
| L26 Damaged 2 | | ✓ |
| U32 Raised | ✓ | |
| Screw/Cables Solder | | ✓ |

The data set contained different sizes of the captured raw images. Before performing the annotation task, all the images (defective and non-defective) were resized to the same dimension. The annotation of all the images comprising non-defective and defective products was performed manually by our team with the help of the manufacturing quality control team. We annotated them into 19 classes of defect types. We peculiarly annotated the image data. Due to the similarities among most of the components on the PCBA and the complexity in the dense PCBA, we expanded beyond the defected area. We enlarged the RoI by encompassing area nearby adjacent components to train the models more efficiently. However, it is to be noted that the added area in the annotation box did not contain other defective components. The LabelImg application, a graphical image annotation tool written in Python, was used for generating the annotations for

the images in the data set [43]. The annotation output files of this graphical user interface application were converted from text to JSON files to meet the requirement of the faster R-CNN model training (explained in the next section). Next, the data was divided into training and testing sets. We selected three images of each defect type out of the 15 images available in the original data set for the testing set. Eight images of non-defective products were added to the test set. Finally, our test set included 62 images with their corresponding annotations. This test set was not modified and kept aside for the experiments.

Augmentation is critical in adding synthetic image data to expand the training data set for developing robust CV-based defect detection models. Also, creating annotation files for the synthetic samples to train the object detection models is tedious. Hence, to counter the time-consuming and expensive process requiring manual reproduction of the annotation files, we utilized different color space transformations and image rotation techniques to generate a data set of synthetic analogues of the original image data. It should be mentioned that the YOLO object detection models employ data augmentation techniques in its bag of freebies framework. It uses several data augmentation techniques to generate images in different conditions. Photometric distortions including brightness, contrast, hue, saturation, and noise, and geometric distortions including random scaling, cropping, flipping, and rotation are the augmentation techniques that are used in the YOLO detectors. After applying the augmentation techniques, we obtained a data set of 2000 images per defect type.

The two-stage object detection model (Faster-RCNN) does not accept images without annotation in the training phase. Hence, the training data set only contained the annotated images for the defective products (i.e., 2000 images * 19 defect types = 38,000 images). However, for the one-stage detection (YOLO) models, we included an additional 2500 images of non-defective products to facilitate efficient learning [36]. The augmentation process encompassed rotation, color space transformation, and kernal filters. For each image, a rotation of approximately 2.16 degrees was applied, generating 167 images with varying angles for each class. This rotation filter resulted in augmenting each image to a total of 2000 images. Subsequently, the generated images were classified into three distinct brightness levels: ''dim,'' ''dark,'' and ''bright.'' These terms correspond to specific levels of luminosity or light intensity, wherein ''dim'' signifies a lower brightness level, ''dark'' conveys an even lower intensity, and ''bright'' denotes a higher luminosity. The adjustments in light and noise introduced by the color space transformation and kernel filters were calibrated to exhibit approximately 33 percent variation from one category to another. Table 3 shows the total number of samples in the training and testing data sets for the different categories of object detection models.

K-fold cross-validation is a technique used to reduce the risk of overfitting by assessing the model across multiple data subsets. It involves training and evaluating the model on k different subsets, exposing it to diverse variations in the data. The final performance metric is often the average across all k folds, mitigating the impact of overfitting to specific subsets. By evaluating the model on various test sets, k-fold cross-validation provides a realistic estimate of its generalization to new, unseen data distinguishing between learned training patterns and underlying patterns likely to generalize. We utilized a 5-fold cross-validation technique. In this method, we consistently followed the procedure of selecting three out of the total 15 images per defect type for evaluation, while applying augmentation techniques to the remaining images. This meticulous process was employed to systematically assess the model's performance, with the primary aim of mitigating the risk of overfitting.

**TABLE 3.** Training and testing data sets.

| Model | | Number of test data | Number of training data |
|---|---|---|---|
| One-stage | Non-defective | 8 | 2500 |
| | Defective | 54 | 38000 |
| | Total | 62 | 40500 |
| Two-stage | Non-defective | 8 | NA |
| | Defective | 54 | 38000 |
| | Total | 62 | 38000 |

### C. MODEL SETUP

We trained the various defect detection models using the Azure cloud platform. Specifically, the models were trained with Standard NC6s v3 (6 vcpus, 112 GB memory) Azure Virtual Machine (VM) running on Ubuntu 18.04. Table 4 shows the training settings of the different object detector models. We developed the YOLOv4, the YOLOR-P6, and the Faster R-CNN (using ResNeXt-101-FPN 3x) models in the object detection module. The backbone for the Faster R-CNN model is selected based on its demonstrated performance on the COCO data set (see Figure 4 [44]). Hyper parameter tuning process was conducted to select the best parameter values. Table 5 shows the different network configurations and transfer learning weights (pre-trained models) used in the experiments. It is to be noted that by increasing the input image size and batch size in CNN-based object detection models, the learning rate may improve; however, there is a trade-off with the computational costs. We found 640*640 to be an ideal image size for the models. This standardized image size was maintained throughout the inference period when assessing the processing speed of the models. Next, we present the performance comparison of these models.

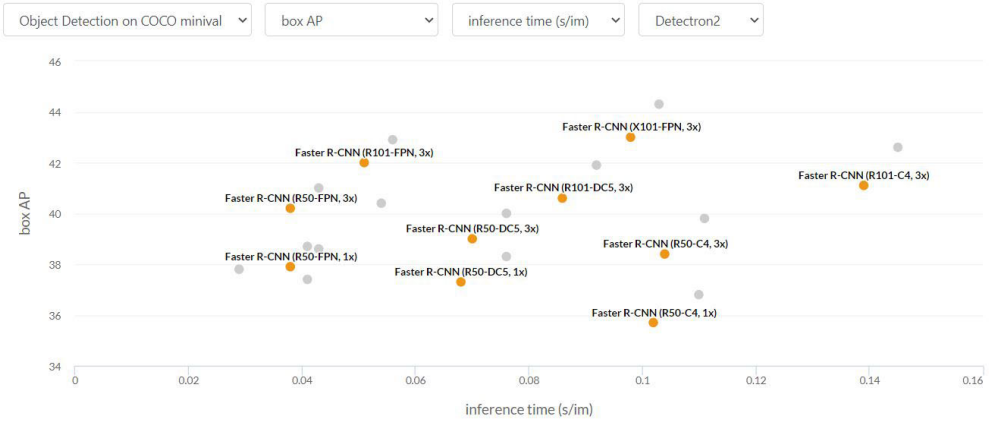



**FIGURE 4.** Performance comparison among the various backbones for Faster R-CNN models [44].

**TABLE 4.** Training settings of different defect detector models.

| Settings | Model | | |
|---|---|---|---|
| | YOLOv4 | YOLOR | Faster-RCNN |
| Training Library | Darknet | Pytorch | Detectron2 |
| Image size (in pixels) | 640*640 | 640*640 | 640*640 |
| Batch size | 8 | 8 | 4 |
| Number of epochs | 50 | 50 | 50 |
| Annotation format | .txt | .txt | .json |
| Learning rate | 0.001 | 0.00261 | 0.001 |
| Momentum | 0.949 | 0.949 | 0.9 |
| Decay | 0.0005 | 0.0005 | 0.0001 |

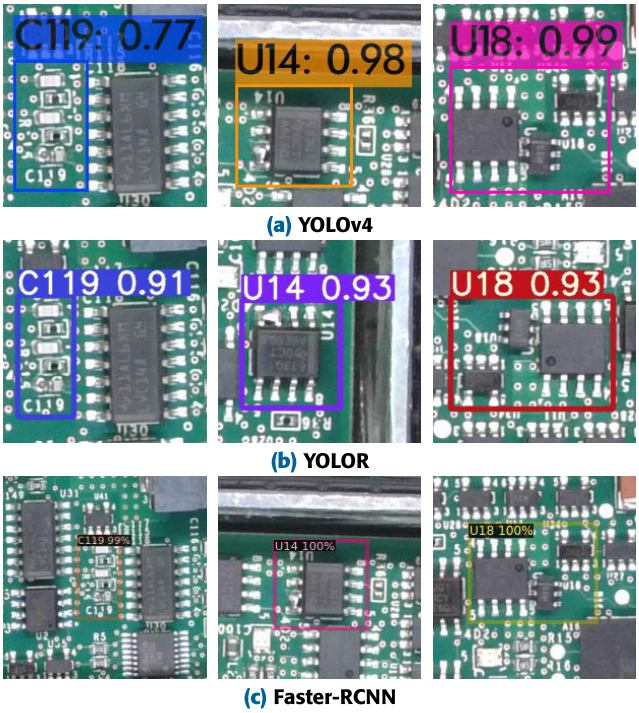



**FIGURE 5.** Model output examples for three different defect types (from left to right: C119 Damaged, U14 Short Circuit, U18 Foreign Component).

## V. RESULTS AND ANALYSIS

In this section, we first provide the results of the experiments for the defect detection models. Then, we analyze the results and present the key findings from this analysis.

We used different confidence thresholds indicating the prediction confidence of the defect detection models to help the quality control (QC) team in their decision-making process. Table 6 shows the results obtained from the experiments. There is a trade-off among the performance metrics at each threshold. As the confidence threshold increases from 25% to 95%, the precision score increases, while the recall score decreases. We show three different confidence thresholds, 25%, 65%, and 95%, in Table 6. Both TP and FP values decrease as the confidence threshold value is increased. A larger threshold value reduces the number of positive detections. Conversely, the FN value increases as the confidence threshold value increases. A larger threshold value increases the number of missed detections.

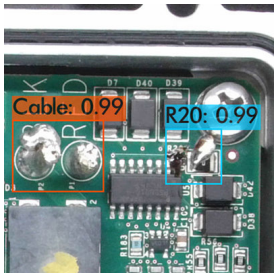
In the case of a high confidence threshold (95%), the QC plan will revolve around identifying the products only if there is a high certainty of the defects. Although this model behavior will result in a minimum number of FP products, it will fail to identify some defective products correctly (i.e., the FN rate will be higher). Table 6 shows the high FN values associated with each model at this threshold. In such a threshold setting, the manufacturer will risk shipping defective products; hence, this confidence threshold may not be an ideal choice in the QC plan. In the case of a low confidence threshold (25%), the QC plan will revolve around inspecting many non-defective products that the model may have identified with defects with a low probability. Table 6 shows the FN rate to be 0 and a high FAR rate associated with each model at this threshold. With this threshold setting, the QC plan will need a human evaluation of these products and

**TABLE 5.** Model configuration and pre-trained weights for TL [36], [37], [45], [46].

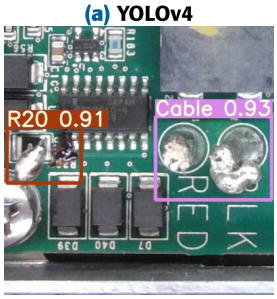
| Model | Backbone | Pretrained weights on COCO data set [41] |
|---|---|---|
| **YOLOv4** | CSPDarkNet-53 | yolov4.conv.137.weights |
| **YOLOR-P6** | CSPDarkNet-53 | yolor_p6.pt |
| **Faster-RCNN** | ResNeXt-101-FPN 3x | model_final_68b088.pkl |

**TABLE 6.** Performance summary of different defect detection models on PCBA data set.

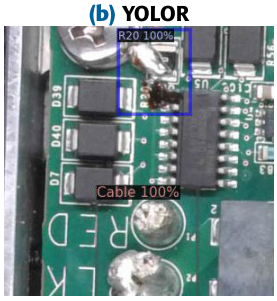
| | | Confidence threshold | | |
|---|---|---|---|---|
| Model | Metric | 25% | 65% | 95% |
| YOLOv4 | TP (No.) | 57 | 57 | 51 |
| | FP (No.) | 13 | 7 | 4 |
| | FN (No.) | 0 | 0 | 6 |
| | Precision (%) | 81 | 89 | 93 |
| | Recall (%) | 100 | 100 | 89 |
| | FAR (%) | 19 | 11 | 7 |
| | mIoU (%) | 64 | 84 | 76 |
| | Inference Time (ms) | | 16 | |
| YOLOR-P6 | TP (No.) | 57 | 57 | 38 |
| | FP (No.) | 10 | 0 | 0 |
| | FN (No.) | 0 | 0 | 19 |
| | Precision (%) | 85 | 100 | 100 |
| | Recall (%) | 100 | 100 | 67 |
| | FAR (%) | 15 | 0 | 0 |
| | mIoU (%) | 84 | 96 | 73 |
| | Inference Time (ms) | | 11 | |
| Faster-RCNN | TP (No.) | 57 | 56 | 43 |
| | FP (No.) | 17 | 12 | 8 |
| | FN (No.) | 0 | 1 | 14 |
| | Precision (%) | 77 | 82 | 84 |
| | Recall (%) | 100 | 98 | 75 |
| | FAR (%) | 23 | 18 | 16 |
| | mIoU (%) | 57 | 66 | 53 |
| | Inference Time (ms) | | 92 | |




(a) YOLOv4




(b) YOLOR




(c) Faster-RCNN

**FIGURE 6.** Successful multiple defect type detection of test image data.

will increase the team's workload and the costs. Based on our discussions with the manufacturing QC team, we identified that it is critical to keep the FN rate as close to 0 as that would prevent the defective products from shipping out to the customers. In addition, it is important to keep a very low FP rate to minimize the workload of the human operators (and reduce the associated costs). Based on the results in Table 6, we found the favorable confidence threshold to be 65% among all the threshold values and across all the different models that would keep both the underkill and overkill rates to a minimum (closer to 0).

It is to be noted that the production rate of printed circuit board assembly lines may also influence the QC plans. If the rate of PCBA production is high, then a QC plan may need to include a model with a fast inference time for real-time defect detection. This will help in maintaining the high production rate and decrease the cost of delays in the inspection process. In such a case, both the YOLOv4 and the YOLOR-P6 are top modeling choices. Table 6 shows the model inference times.

Next, for the confidence threshold of 65%, we looked at the models' performances for each defect type. Table 7 shows the TP and FP values for all 19 different defect types obtained using the YOLOv4, the YOLOR-P6, and the Faster-RCNN

**TABLE 7.** TP and FP values per defect type for different models.

| Defect Type | YOLOv4 | | | YOLOR | | | Faster-RCNN | | |
|---|---|---|---|---|---|---|---|---|---|
| | TP | FP | IoU (%) | TP | FP | IoU (%) | TP | FP | IoU (%) |
| **C31 Raised** | 3 | **3** | 56 | 3 | 0 | 92 | 3 | **7** | 21 |
| D20 Missing | 3 | 0 | 99 | 3 | 0 | 98 | 3 | 0 | 70 |
| D3 Displaced | 3 | 0 | 74 | 3 | 0 | 100 | 3 | 0 | 71 |
| C39-Displaced | 3 | 0 | 79 | 3 | 0 | 99 | 3 | 0 | 57 |
| R20 Damaged | 3 | 0 | 91 | 3 | 0 | 98 | 3 | 0 | 82 |
| U13 Solder Balls | 3 | 0 | 76 | 3 | 0 | 93 | 3 | 0 | 73 |
| C55 Extra Component | 3 | 1 | 63 | 3 | 0 | 93 | 3 | 1 | 54 |
| J6 Damaged | 3 | 0 | 89 | 3 | 0 | 94 | 3 | 0 | 79 |
| U14 Short Circuit | 3 | 0 | 94 | 3 | 0 | 97 | 3 | 0 | 58 |
| C76 Missing | 3 | 0 | 95 | 3 | 0 | 100 | 3 | 0 | 72 |
| L8 Damaged | 3 | 0 | 88 | 3 | 0 | 100 | 2 | 0 | 51 |
| U18 Foreign Component | 3 | 0 | 95 | 3 | 0 | 97 | 3 | 0 | 72 |
| C119 Damaged | 3 | 0 | 77 | 3 | 0 | 95 | 3 | 1 | 62 |
| L26 Damaged | 3 | 0 | 98 | 3 | 0 | 90 | 3 | 0 | 73 |
| U26 Short Circuit | 3 | 0 | 91 | 3 | 0 | 100 | 3 | 0 | 80 |
| C139 Missing | 3 | 0 | 79 | 3 | 0 | 100 | 3 | 0 | 91 |
| L26 Damaged 2 | 3 | 0 | 94 | 3 | 0 | 92 | 3 | 0 | 80 |
| **U32 Raised** | 3 | **3** | 59 | 3 | 0 | 94 | 3 | **4** | 36 |
| Screw/Cables Solder | 3 | 0 | 98 | 3 | 0 | 94 | 3 | 0 | 78 |

models. The results show that the YOLOR-P6 model has the best performance for detecting actual defects among all three models. It is also able to keep the FP rate at a minimum among all models for all defect types. The YOLOv4 and the Faster-RCNN models do not have a desirable FAR value for a couple of defect types. Both models struggled with the raised components, including C31 Raised and U32 Raised. These two defect types were misclassified for a total count of six and 11 times in YOLOv4 and Faster-RCNN model outputs, respectively. It indicates that these models have poor performance, mainly with the defects in which the component is raised. Components with these defect types have similar visionary characteristics as the non-defective components.

Figure 5 shows some examples of outputs obtained from each of the models. The magnifying glass highlights the defective component and displays the confidence score associated with detecting the defect. Figure 6 displays the outputs of the models with images containing multiple defects. The three test PCBA images contain R20 damage and screw/cables solder defects, which are detected with over 90% confidence in all three models. The Faster-RCNN model detects both defects with 100% accuracy. In fact, the Faster-RCNN model outperforms the other two models in terms of the level of certainty in detecting the defects in all these images in Figures 5 and 6. However, the Faster-RCNN model has an abysmal performance on the non-defective products image data (as demonstrated by comparatively larger FP values in Table 7). We evaluated our framework on various PCBA products and obtained similar results.

## VI. CONCLUSION AND FUTURE RESEARCH DIRECTIONS

In this paper, we investigated the problem of automated inspection for the PCBA manufacturing process under adverse conditions. There are various conditions in real-world manufacturing systems that impact the performance of the AOI systems. Our defect detection framework comprises problem-specific data preparation and defect detection modules to overcome these conditions. We evaluated our framework on a real-world manufacturing data set. The selected PCBA product was complex and dense, and the difference between the defective and the non-defective areas was proportionally a tiny region. We employed various augmentation techniques to counter the issue of images taken in non-ideal settings for the resolution, brightness, camera angle, and distance. We demonstrated that our framework is able to expand a small data set containing PCBA images with a large number of defects and use this new balanced data set to train the object detection models. We developed

one- and two-stage object detectors and evaluated their performance using precision and recall metrics. We compared different quality control plans based on the models and their performance under various confidence thresholds. We found a favorable confidence threshold of 65%, at which there were no false negatives in one-stage detector models, and the false positives were also minimal. The YOLOR model outperformed the other one-stage and two-stage detector models in the precision and recall metrics. We also found that the one-stage detector models, the YOLOR and the YOLOv4, have a significantly faster inference time and they would be an ideal choice in high production printed circuit board assembly lines. In summary, our object detection framework accurately and timely identifies the various functional and cosmetic defects in PCBA and thereby helps in increasing manufacturing productivity while lowering the underkill and overkill rates.

As a direction for future research, delving into generative techniques like generative adversarial networks holds promise. These methods could assist in generating expansive real labeled PCBA data sets with reduced human intervention. Subsequently, evaluating the object detection models trained on these augmented synthetic data sets in comparison to models trained solely on real data could provide valuable insights. Another promising research direction is the exploration of pixel-wise segmentation methods for this problem. While we employed state-of-the-art object detection models to tackle the research problem, evaluating the performance of novel segmentation techniques for pixel-wise defect identification and precise defect location remains untapped.

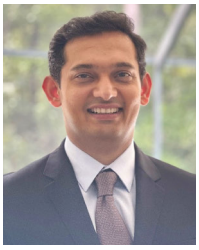
## REFERENCES

[1] M. Moganti and F. Ercal, "Automatic PCB inspection systems," *IEEE Potentials*, vol. 14, no. 3, pp. 6–10, 1995.
[2] Z. Liu and B. Qu, "Machine vision based online detection of PCB defect," *Microprocessors Microsyst.*, vol. 82, Apr. 2021, Art. no. 103807.
[3] C.-T. Liao, W.-H. Lee, and S.-H. Lai, "A flexible PCB inspection system based on statistical learning," *J. Signal Process. Syst.*, vol. 67, no. 3, pp. 279–290, Jun. 2012.
[4] T. Wang, Y. Chen, M. Qiao, and H. Snoussi, "A fast and robust convolutional neural network-based defect detection model in product quality control," *Int. J. Adv. Manuf. Technol.*, vol. 94, nos. 9–12, pp. 3465–3471, Feb. 2018.
[5] A. Kulkarni and C. Xu, "A deep learning approach in optical inspection to detect hidden hardware trojans and secure cybersecurity in electronics manufacturing supply chains," *Frontiers Mech. Eng.*, vol. 7, p. 70, Jul. 2021.
[6] A. A. R. M. A. Ebayyeh and A. Mousavi, "A review and analysis of automatic optical inspection and quality monitoring methods in electronics industry," *IEEE Access*, vol. 8, pp. 183192–183271, 2020.
[7] L. Xie, R. Huang, N. Gu, and Z. Cao, "A novel defect detection and identification method in optical inspection," *Neural Comput. Appl.*, vol. 24, nos. 7–8, pp. 1953–1962, Jun. 2014.
[8] I. Volkau, A. Mujeeb, W. Dai, M. Erdt, and A. Sourin, "The impact of a number of samples on unsupervised feature extraction, based on deep learning for detection defects in printed circuit boards," *Future Internet*, vol. 14, no. 1, p. 8, Dec. 2021.
[9] A. Raj and A. Sajeena, "Defects detection in PCB using image processing for industrial applications," in *Proc. 2nd Int. Conf. Inventive Commun. Comput. Technol. (ICICCT)*, Apr. 2018, pp. 1077–1079.
[10] Z. Lu, Q. He, X. Xiang, and H. Liu, "Defect detection of PCB based on Bayes feature fusion," *J. Eng.*, vol. 2018, no. 16, pp. 1741–1745, Nov. 2018.
[11] R. Li, B. Xue, K. Zhao, H. Chu, and B. Jiao, "PCB defect recognition and elimination based on secondary error and statistical histogram," in *Proc. Int. Wireless Commun. Mobile Comput. (IWCMC)*, Jun. 2020, pp. 1981–1984.
[12] A.-A.-I. M. Hassanin, F. E. Abd El-Samie, and G. M. El Banby, "A real-time approach for automatic defect detection from PCBs based on SURF features and morphological operations," *Multimedia Tools Appl.*, vol. 78, no. 24, pp. 34437–34457, Dec. 2019.
[13] R. Ye, M. Chang, C.-S. Pan, C. A. Chiang, and J. L. Gabayno, "High-resolution optical inspection system for fast detection and classification of surface defects," *Int. J. Optomechatronics*, vol. 12, no. 1, pp. 1–10, Jan. 2018.
[14] S. S. Zakaria, A. Amir, N. Yaakob, and S. Nazemi, "Automated detection of printed circuit boards (PCB) defects by using machine learning in electronic manufacturing: Current approaches," *IOP Conf. Ser., Mater. Sci. Eng.*, vol. 767, no. 1, Feb. 2020, Art. no. 012064.
[15] D. Li, S. Fu, Q. Zhang, Y. Mo, L. Liu, and C. Xu, "An improved PCB defect detector based on feature pyramid networks," in *Proc. 4th Int. Conf. Comput. Sci. Artif. Intell.*, Dec. 2020, pp. 233–239.
[16] Z. Lan, Y. Hong, and Y. Li, "An improved YOLOv3 method for PCB surface defect detection," in *Proc. IEEE Int. Conf. Power Electron., Comput. Appl. (ICPECA)*, Jan. 2021, pp. 1009–1015.
[17] J. Li, J. Gu, Z. Huang, and J. Wen, "Application research of improved YOLO V3 algorithm in PCB electronic component detection," *Appl. Sci.*, vol. 9, no. 18, p. 3750, Sep. 2019.
[18] Y.-T. Li, P. Kuo, and J.-I. Guo, "Automatic industry PCB board DIP process defect detection system based on deep ensemble self-adaption method," *IEEE Trans. Compon., Packag., Manuf. Technol.*, vol. 11, no. 2, pp. 312–323, Feb. 2021.
[19] V. A. Adibhatla, H.-C. Chih, C.-C. Hsu, J. Cheng, M. F. Abbod, and J.-S. Shieh, "Defect detection in printed circuit boards using you-only-look-once convolutional neural networks," *Electronics*, vol. 9, no. 9, p. 1547, Sep. 2020.
[20] J. Shen, N. Liu, and H. Sun, "Defect detection of printed circuit board based on lightweight deep convolution network," *IET Image Process.*, vol. 14, no. 15, pp. 3932–3940, Dec. 2020.
[21] Q. Zhang, M. Zhang, C. Gamanayake, C. Yuen, Z. Geng, H. Jayasekaraand, X. Zhang, C.-W. Woo, J. Low, and X. Liu, "Deep learning based defect detection for solder joints on industrial X-ray circuit board images," in *Proc. IEEE 18th Int. Conf. Ind. Informat. (INDIN)*, vol. 1, Jul. 2020, pp. 74–79.
[22] W. Choi and Y.-J. Cha, "SDDNet: Real-time crack segmentation," *IEEE Trans. Ind. Electron.*, vol. 67, no. 9, pp. 8016–8025, Sep. 2020.
[23] Y.-J. Cha, W. Choi, and O. Büyüköztürk, "Deep learning-based crack damage detection using convolutional neural networks," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 32, no. 5, pp. 361–378, May 2017.
[24] Y. Cha, W. Choi, G. Suh, S. Mahmoudkhani, and O. Büyüköztürk, "Autonomous structural visual inspection using region-Based deep learning for detecting multiple damage types," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 33, no. 9, pp. 731–747, Sep. 2018.
[25] J. Lewis, Y.-J. Cha, and J. Kim, "Dual encoder–decoder-based deep polyp segmentation network for colonoscopy images," *Sci. Rep.*, vol. 13, no. 1, p. 1183, Jan. 2023.
[26] D. H. Kang and Y.-J. Cha, "Efficient attention-based deep encoder and decoder for automatic crack segmentation," *Struct. Health Monitor.*, vol. 21, no. 5, pp. 2190–2205, Sep. 2022.
[27] B. Hu and J. Wang, "Detection of PCB surface defects with improved faster-RCNN and feature pyramid network," *IEEE Access*, vol. 8, pp. 108335–108345, 2020.
[28] C. Zhang, W. Shi, X. Li, H. Zhang, and H. Liu, "Improved bare PCB defect detection approach based on deep feature learning," *J. Eng.*, vol. 2018, no. 16, pp. 1415–1420, Nov. 2018.
[29] H. Xin, Z. Chen, and B. Wang, "PCB electronic component defect detection method based on improved YOLOv4 algorithm," *J. Phys., Conf. Ser.*, vol. 1827, no. 1, Mar. 2021, Art. no. 012167.
[30] G. Ran, X. Lei, D. Li, and Z. Guo, "Research on PCB defect detection using deep convolutional nerual network," in *Proc. 5th Int. Conf. Mech., Control Comput. Eng. (ICMCCE)*, Dec. 2020, pp. 1310–1314.
[31] M. A. M. Sathiaseelan, O. P. Paradis, S. Taheri, and N. Asadizanjani, "Why is deep learning challenging for printed circuit board (PCB) component recognition and how can we address it?" *Cryptography*, vol. 5, no. 1, p. 9, Mar. 2021.

[32] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, no. 1, p. 60, Dec. 2019.
[33] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection," *IEEE Access*, vol. 7, pp. 128837–128868, 2019.
[34] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.
[35] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto, and E. A. B. da Silva, "A comparative analysis of object detection metrics with a companion open-source toolkit," *Electronics*, vol. 10, no. 3, p. 279, Jan. 2021.
[36] A. Bochkovskiy. (2020). *Darknet*. Accessed: Feb. 24, 2022. [Online]. Available: https://github.com/AlexeyAB/darknet
[37] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "You only learn one representation: Unified network for multiple tasks," 2021, *arXiv:2105.04206*.
[38] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.
[39] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.
[40] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 1–9.
[41] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 740–755.
[42] L. R. Barnes, E. C. Gruntfest, M. H. Hayden, D. M. Schultz, and C. Benight, "False alarms and close calls: A conceptual model of warning accuracy," *Weather Forecasting*, vol. 22, no. 5, pp. 1140–1147, Oct. 2007.
[43] T. Lin. (2015). *LabelImg*. Accessed: Feb. 24, 2022. [Online]. Available: https://github.com/tzutalin/labelImg
[44] Meta. (2022). *Real-Time Object Detection on COCO*. Accessed: Feb. 24, 2022. [Online]. Available: https://paperswithcode.com/lib/detectron2/faster-r-cnn
[45] Y. Wu, A. Kirillov, F. Massa, W. Y. Lo, and R. Girshick. (2019). *Detectron2*. [Online]. Available: https://github.com/facebookresearch/detectron2
[46] W. Kin-Yiu. *GitHub*. Accessed: Feb. 24, 2022. [Online]. Available: https://github.com/WongKinYiu/yolor

**MOHAMMAD NOROOZI** is currently pursuing the Ph.D. degree with the Industrial and Management Systems Engineering Department, University of South Florida. He is passionately involved in advancing complex systems' efficiency. His research interests include AI, machine learning, optimization methods, deep learning, and computer vision, shaping a future where these methodologies empower transformative solutions.

**JALAL GHADERMAZI** is currently pursuing the Ph.D. degree with the Industrial and Management Systems Engineering Department, University of South Florida. His research interests include machine learning, artificial intelligence, computer vision, and real-time analytics. He integrates principles from computer science, operations research, data science, and information technology to develop innovative solutions for complex problems.



**ANKIT SHAH** (Senior Member, IEEE) is currently an Assistant Professor of industrial and management systems engineering with the University of South Florida. His research interests include the intersection of computer science, operations research, data science, and information technology, with a strong focus on artificial intelligence for defense applications. His current research work is in the area of deep reinforcement learning and machine learning for secure and efficient systems.

**JOSÉ L. ZAYAS-CASTRO** (Senior Member, IEEE) is currently a Professor of industrial and management systems engineering with the College of Engineering, University of South Florida. In addition, he works at the intersection of engineering entrepreneurship and innovation, the integration of research and engineering education, and diversity and inclusion. His work has been funded by federal and state agencies and private industry. His research interests include health care systems, including modeling, analysis, data analysis, economic analysis, and process improvement.