applied sciences

MDPI

Article

YOLOv4-MN3 for PCB Surface Defect Detection

Xinting Liao, Shengping Lv *, Denghui Li, Yong Luo, Zichun Zhu and Cheng Jiang

College of Engineering, South China Agricultural University, Guangzhou 510642, China; 20193079006@stu.scau.edu.cn (X.L.); lidh@stu.scau.edu.cn (D.L.); taffy@stu.scau.edu.cn (Y.L.); 20202157001@stu.scau.edu.cn (Z.Z.); 20193142013@stu.scau.edu.cn (C.J.)

* Correspondence: lvshengping@scau.edu.cn; Tel.: +86-187-1937-3880

Featured Application: An improved YOLOv4 algorithm for PCB surface defect detection can achieve higher detection accuracy and faster detection speed with lower memory consumption and fewer multiply–accumulate operations compared with the cutting-edge YOLOv4.

Abstract: Surface defect detection for printed circuit board (PCB) is indispensable for managing PCB production quality. However, automatic detection of PCB surface defects is still a challenging task because, even within the same category of surface defect, defects present great differences in morphology and pattern. Although many computer vision-based detectors have been established to handle these problems, current detectors struggle to achieve high detection accuracy, fast detection speed and low memory consumption simultaneously. To address those issues, we propose a cost-effective deep learning (DL)-based detector based on the cutting-edge YOLOv4 to detect PCB surface defect quickly and efficiently. The YOLOv4 is improved upon with respect to its backbone network and the activation function in its neck/prediction network. The improved YOLOv4 is evaluated with a customized dataset, collected from a PCB factory. The experimental results show that the improved detector achieved a high performance, scoring 98.64% on mean average precision (*mAP*) at 56.98 frames per second (*FPS*), outperforming the other compared SOTA detectors. Furthermore, the improved YOLOv4 reduced the parameter space of YOLOv4 from 63.96 M to 39.59 M and the number of multiply-accumulate operations (*Madds*) from 59.75 G to 26.15 G.

Keywords: printed circuit board; surface defect detection; YOLOv4; MobileNetV3

Citation: Liao, X.; Lv, S.; Li, D.; Luo, Y.; Zhu, Z.; Jiang, C. YOLOv4-MN3 for PCB Surface Defect Detection. *Appl. Sci.* **2021**, *11*, 11701. https://doi.org/10.3390/app112411701

Academic Editor: Alfio Dario Grasso

Received: 18 November 2021
Accepted: 6 December 2021
Published: 9 December 2021

1. Introduction

Printed circuit board (PCB) surface defects are local areas of surface that do not meet design or manufacturing requirements. However, PCBs' surface defects not only affect their aesthetics but also their performances and functionalities. Therefore, surface defect detection is indispensable in managing PCB production quality. In modern factories, basic electric tests, such as manual visual inspection, are still commonly used [1]. However, manual detection relies heavily on many experienced inspectors exercising extensive concentration under strong illumination, while electric tests, as a contact-detection technique, may themselves cause defects in the board. Therefore, more attention is being paid to non-contact automated optical inspection (AOI), and practice indicates that it has greatly improved detection accuracy and efficiency [2].

As the core of AOI software, detection algorithms can be divided into traditional detectors and deep learning (DL)-based detectors according to their feature extraction methodology. Traditional detectors utilize traditional machine learning methods, an image-processing approach and prior knowledge to extract low-level defect feature. Wang et al. [3] extracted the center, aperture, roundness and area of holes as features, and compared the feature information between testing images and standard images to detect the defects in PCB holes. Gaidhane et al. [4] utilized companion matrices of testing images and standard images to construct a symmetrical matrix and adopted its rank as a similarity metric to

determine the defects detected on PCB boards. Eun et al. [5] combined the speeded-up robust features and random forest algorithms to extract PCB fault patterns and drew a weighted kernel density estimation map for defect detection based on probability values. Fonseka et al. [6] integrated color transformation, graph-cut based segmentation and k-means color clustering to detect solder bridging, solder voids and excess solder. Liu et al. [7] used the mathematical morphology method to obtain standard images, and then an image aberration detection algorithm was introduced to detect PCB defects. These traditional detectors highly rely on prior knowledge to determine previously seen features, or to store a large number of standard images and then precisely align testing images to them for element matching. Therefore, traditional detectors are not conducive to generalization between different application scenarios.

A DL-based detector utilizes a convolutional neural network (CNN) to extract and defect features and learn inherent patterns of defects, automatically, without standard images or manual design rules [8,9], which greatly improves detection accuracy, efficiency and model generalization. DL-based detectors can be roughly divided into two categories, two-stage detectors and one-stage detectors. Two-stage detectors divide the training process into candidate boxes (region proposals), where extraction and feature classification are based on region proposals over two steps. R-CNN [10], Fast R-CNN [11] and Faster RCNN [12] are classical two-stage detectors. These models can achieve high detection accuracies and precisions of location but are trapped by their detection speeds. One-stage detectors directly regress bounding boxes and probabilities for each object in an input image, simultaneously, without region proposals. YOLO series models (YOLO, YOLOv2, YOLOv3, YOLOv4) are widely used one-stage detectors [13]. These detectors can speed up detection but suffer from a drop in accuracy.

Both one-stage and two-stage DL-based detectors are constituted by backbone, neck and prediction networks (also called a head network), and improvements for the three networks are continuously emerging for better matching within different application scenarios [8]. DL-based detectors, in surface defect detection, have attracted much attention in recent years. Many improved YOLO and R-CNN detectors [14–16] have been developed for surface defect detection with the available labeled surface defect datasets in the industry, such as from steel, metro tunnel and commutator train production. Many studies and practitioners have also introduced this mechanism into PCB defect detection.

Ding et al. [17] developed a tiny defect detection network (TDD-Net) based on fast-RCNN for PCB defect detection and employed an online example of a hard mining technique in their training phase to alleviate the adverse effects of small datasets and sample imbalance. Hu et al. [18] improved the two-stage Faster RCNN for PCB defect detection. First, they replaced the backbone and neck network of Faster RCNN with ResNet50 and feature pyramid networks (FPN) respectively. Second, they introduced a guided anchor region proposal network as a substitute of the original region proposal network for better anchor generation. Additionally, the residual module of ShuffleNetV2 was adopted in their backbone to reduce the model parameter and operation. Dai et al. [1] employed YOLO to locate hundreds of small and dense solder joints automatically in PCB images before defect detection. Zhang et al. [19] combined a dual attention mechanism and Path Aggregation Feature Pyramid Network in MobileNetV2 to build PCB defect detector. The above DL-based models exhibited high detection accuracy but suffered from a large number of parameters and high computational cost. Meanwhile, inter-class diversity of surface defect was not considered in these studies, whereas the same category of detected defect possesses great difference in their morphologies and patterns in practice. Collecting real defect samples from PCB production factory, and establishing detectors with high detection accuracy, fast detection speed, low memory consumption and low multiply-accumulate operations (*Maccs* or *Madds*) is a promising tendency.

To solve the above-mentioned problems, this study proposes a cost-effective DL-based detector called as YOLOv4-MN3 based on cutting-edge YOLOv4 and MobileNetV3 lightweight network. First, we design a PCB image acquisition device. Then, surface defect

images are collected into a dataset with 2008 samples, in which were contained bumpy or broken line, clutter, scratch, line repair damage, hole loss and over oil-filling—the six categories of the most common defects. Second, we utilize the MobileNetV3 lightweight network with small number of parameters and lower *Madds*, replacing CSPDarknet53 as the backbone network. Third, the influence of different activation functions in the neck and prediction networks are tested and compared, for which the Mish activation function is selected. The experimental results show that the proposed YOLOv4-MN3 for PCB surface defect detection achieves a higher detection accuracy, faster detection speed and lower *Madds* compared to SOTA detectors.

The contents of this paper are organized as follows. Section 2 presents the workflow of the proposed detection approach and the architecture of the proposed detector YOLOv4-MN3. Section 3 introduces the building of a customized dataset, which includes a surface defect image acquisition device, defect image collection, data augmentation and labeling. Section 4 gives the training and detection results with comprehensive comparisons. Section 5 contains our conclusions.

2. Methodology

2.1. Framework of the Methodology

The proposed YOLOv4-MN3 for PCB surface defect detection is established based on a cutting edge one-stage detector, YOLOv4, and it consists of dataset building, model training and performance evaluation in three steps, as described in Figure 1. First, all the PCB defect images are collected by a specially designed image acquisition device, after which the augmentation and annotation labeling are conducted. Then, YOLOv4 is modified and trained based on a customized dataset. Finally, the performance of the proposed YOLOv4-MN3 and other SOTA detectors are evaluated and compared.

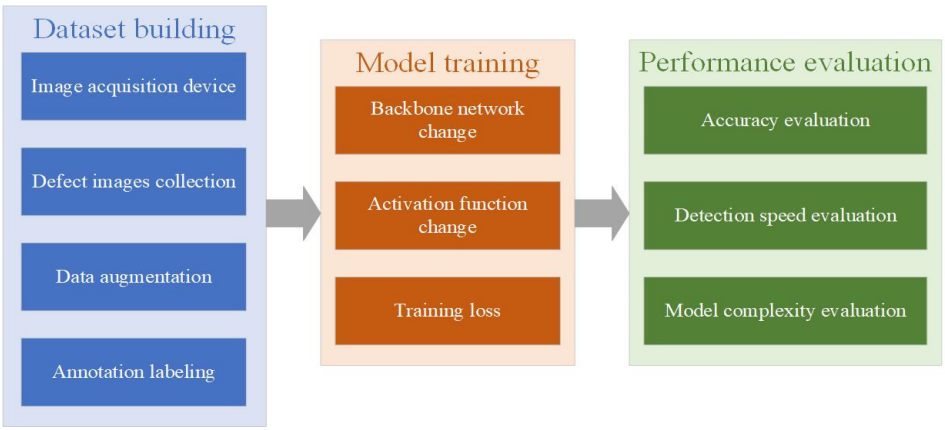


Figure 1. Framework of the proposed methodology.

Dataset building: The PCB surface defect images were collected by an image acquisition device specially designed and manually labeled by the program labelImg to generate training and image-defect testing datasets with corresponding annotation files.

Model training: Bumps, broken lines, clutter, scratched, line repair damage, hole losses and over oil-filling, the six categories of the most common surface defects, were selected for the training. The original backbone network CSPDarknet53 of YOLOv4 was replaced by VGG16, Resnet50, Darknet53, MobileNetV2 and MobileNetV3 for the purposes of selecting an appropriate backbone that could decrease memory consumption and computational cost. To better fit to customized defect dataset, the activation functions of the neck and prediction networks in YOLOv4 were replaced with five different activation functions; thus, six different YOLOv4-MN3 detectors were constructed. Finally, fifteen detectors, including ten backbone or activation function-modified YOLOv4 detectors—original YOLOv4, YOLOv3, Faster RCNN, Retinanet and SDD—were trained with PyTorch.

Performance evaluation: The average precision (*AP*), mean average precision (*mAP*) and F_1 score were adopted as metrics of detection accuracy, and frames per second (*FPS*) as a speed metric. The models' parameters (*Params*) and *Madds* were collected to evaluate model complexity.

2.2. Proposed YOLOv4-MN3

YOLOv4 [20], as one of the cutting-edging one stage DL-based models for object detection, makes many improvements on YOLOv3 [21], including its network architecture, activation function, loss function etc., and integrates many training tricks. The framework of YOLOv4 can also been divided into its backbone, neck and prediction networks. In YOLOv4, a cross-stage partial darknet53 network (CSPDarknet53) is used in the backbone to extract features from the input images. Spatial pyramid pooling (SPP) [22] and path aggregation networks (PANet) [23] were employed as the neck networks to generate a feature pyramid. SPP + PANet, in neck networks, fuse low-level spatial features with accurate location information and high-level semantic features with high semantic information bi-directionally [20]. The prediction network applies anchor boxes to multiscale feature maps of neck network to generate detection boxes.

2.2.1. YOLOv4-MN3 Architecture

YOLOv4 employs the cross-stage partial network (CSPNet) in Darknet53 to construct a CSPDarknet53 backbone. CSPNet partitions the feature map of the input layer into two parts and then merges them through the proposed cross-stage hierarchy for the purpose of enriching gradient combination [24]. However, CSPDarknet53 suffers from high memory consumption, with 29 convolution layers and 27.6 million parameters. Replacing CSPDarknet53 by a lightweight model with fewer parameters while preserving its detection accuracy is a worthy attempt.

For this study, a cost-effective detector YOLOv4-MN3 was developed, in which the CSPDarknet53 in YOLOv4 was replaced by the lightweight network MobileNetV3. MobileNetV3 utilizes depthwise separable convolution to construct feature maps for each layer. The main convolution process is composed of two parts, as given in Figure 2. The first part is depthwise convolution, and it introduces a filter for each input channel and conducts convolutions for each pair of filter and feature map separately. Its second part is pointwise convolution, and it convolutes a 1×1 filter to channels output from the depthwise convolution for the purposes of increasing or decreasing the depth of a feature map. If the convolution input and output are 16 and 32, respectively and the filter size is 3×3, then the parameters of standard convolution and depth separable convolution are $16 \times 32 \times 3 \times 3 = 4608$, $16 \times 3 \times 3 + 32 \times 16 \times 1 \times 1 = 656$ respectively, which indicates that depthwise separable convolution can successfully minimize the number of parameters.

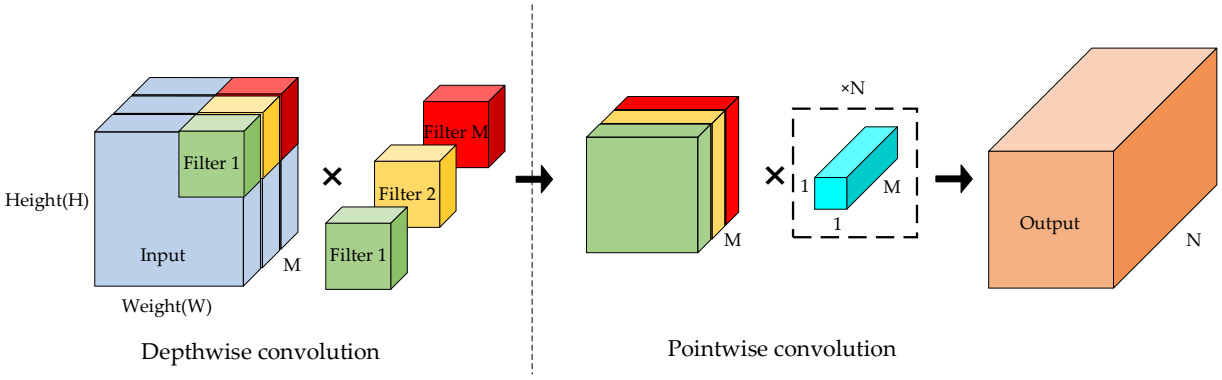


Figure 2. Depthwise separable convolution.

In order to improve the detection accuracy, MobileNetV3 introduces the squeeze and excitation attention module into the bottleneck of MobileNetV2, the basic unit (Bneck) of MobileNetV3, given in Figure 3. Meanwhile, MobileNetV3 modifies the Swish activation function to improve detection accuracy. Experimental results verify the effectiveness and superiority of MobileNetV3 because it can achieve high detection speed and accuracy simultaneously [25]. The MobileNetV3 has MobileNetv3-Small and MobileNetV3-Large—two versions, used according to the depth of its layers. MobileNetV3-Large is employed in YOLOv4-MN3 for a balance of accuracy and speed, based on some initial experimentation. The architecture of YOLOv4-MN3 is given in Figure 4, and only the first 15 Bnecks of MobileNetV3-Large are used to extract three different (52×52, 26×26, 13×13) spatial resolution feature maps, which are directly matched to the input size of the YOLOv4 neck network.

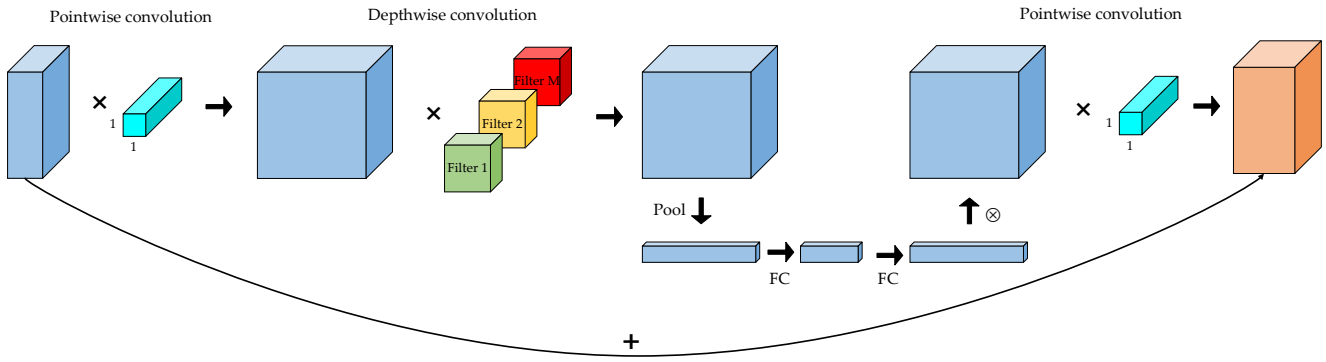


Figure 3. Bneck of MobileNetV3.

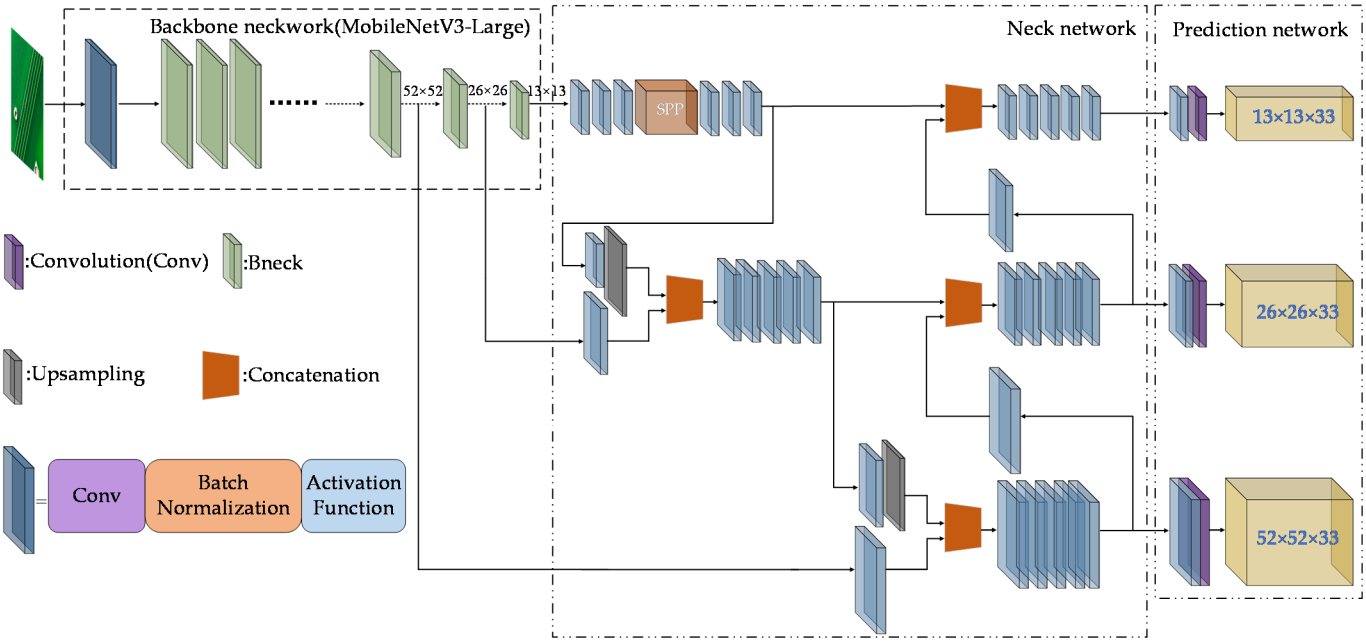


Figure 4. YOLOv4-MN3 architecture.

2.2.2. Activation Functions

It is worth noting that few studies use YOLOv4 to detect PCB surface defect and discuss the influence of different activation functions. Previous research and initial training experiments [26] have shown that activation functions with different properties influence detection performance. Therefore, we analyze the characteristics of different activation functions and conduct comparison experiments between them to select most suitable one

according to their training and detection performances on the customized dataset. Since MobileNetV3 has optimized its activation function, this study only selects the activation function for the neck/prediction network.

Neural network' activation functions greatly influence their customized training convergence procedures because of their derivative, monotonicity properties, among others [27]. The selection of activation functions that perform will in training converging and detection accuracy is an essential step for model establishment [28]. Therefore, the sigmoid, tan hyperbolic (Tanh), rectified linear unit (ReLU), leaky ReLU, Swish, and Mish activation functions were implemented and configured for the training files, based on our customized dataset, to optimize the selection of an activation function.

Figure 5 and Table 1 show the plots and expressions of the six activation functions, respectively. Sigmoid and Tanh, as traditional sigmoid-like units, have dominated neural network practice for several decades. However, they are computationally expensive, and easily lead to gradient vanishing during training. The activation functions of ReLU and Leaky ReLU are widely used in deep CNN. ReLU and Leaky ReLU are not symmetric functions and are unlike the symmetric functions of Tanh and Sigmoid. Thus, they can deal with the problem of gradient vanishing and can update weights continuously during their entire training processes [29,30]. Leaky ReLU introduces an alpha parameter to ensure the gradient of each node would not be zero during the propagation process so that the training loss is easily be trapped into local optima. Swish is a non-monotonic and smooth activation function [31], in which the non-monotonicity property is designed to handle gradient vanishing, while its smoothness is beneficial for model generation and optimization.

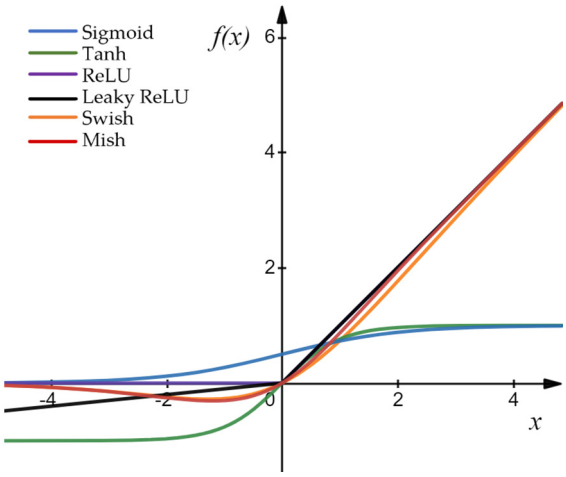


Figure 5. Plots of different activation function.

Table 1. Expression of different activation functions.

Activation Function	Expression	Activation Function	Expression
Sigmoid	$f(x) = 1/(1+1+e^{-x})$	Tanh	$f(x) = (e^{x} - e^{-x})/(e^{x} + e^{-x})$
ReLU	$f(x) = \max(0, x)$	Leaky ReLU	$f(x) = \max(\alpha x, x)$
Swish	$f(x) = x \times 1/(1 + e^{-x})$	Mish	$f(x) = x \times \tanh(\ln(1 + e^{x}))$

Similar to Swish, Mish is a non-monotonic and smooth function with a range of [≈−0.31,∞). Mish outperforms other activation functions in many DL-based detectors across challenging datasets, and we can easily define a Mish activation layer in any standard DL framework for its implementation [26].

2.2.3. Loss Function

The loss function of YOLOv4 includes three parts, confidence, classification, and bounding box regression loss. YOLOv4 employs a novel complete-intersection over union (*IoU*) loss (*CIoU*), replacing the mean–squared–error loss adopted in YOLOv3 with bounding box regression loss [20]. *CIoU* takes the overlap area, center point distance and aspect ratio into consideration simultaneously, improving detection speed and accuracy. *CIoU* introduces a penalty item αv on the basis of distance-*IoU* loss to impose a consistency of aspect ratio for the ground truth bounding box (bb^{gt}) and the prediction bounding box (bb). *CIoU* loss can be defined as in Equation (1)

$$\begin{gathered} Loss_{CIoU} = 1 - (IoU - \frac{\rho^2(b,b^{gt})}{c^2} - \alpha v) \\ \alpha = \frac{v}{1-IoU+v}, v = \frac{4}{\pi^2}(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h})^2 \end{gathered} \tag{1}$$

where b, b^{gt} are the centers of bb, bb^{gt} respectively, $\rho(\cdot)$ denotes Euclidean distance, c represents the diagonal length of the smallest enclosing rectangle covering bb, bb^{gt} and α is a positive trade-off value, v means the consistency of aspect ratio. w,w^{gt} are the widths of bb, bb^{gt} respectively. h, h^{gt} are the heights of bb, bb^{gt}, respectively.

3. Dataset Building

3.1. Image Acquisition Device

A specially designed image acquisition device is depicted in Figure 6, and it consists of an auxiliary module, an illumination module and an image acquisition module. The auxiliary module provides a physical framework for the installation of the illumination and image acquisition modules, and is composed of a dark box (a), motion control parts (b) and a movable loading platform (c). The illumination module (d) provides suitable lights for the camera's image acquisition module to take photographs. The image acquisition module (e) is responsible for collecting PCB images and connects to a computer for image storage and preparation, and consists of a camera support framework, Hikvision MV-CE120-10GC industrial camera with 12 million pixels and an external computer. The camera captures PCB images sequentially under the stable light source provided by the illumination module. The maximum field of view is 120 mm × 90 mm, and multi-point shooting was used for PCB s larger than this size, in which the camera is moved by motion control components to locate its position for each shooting.

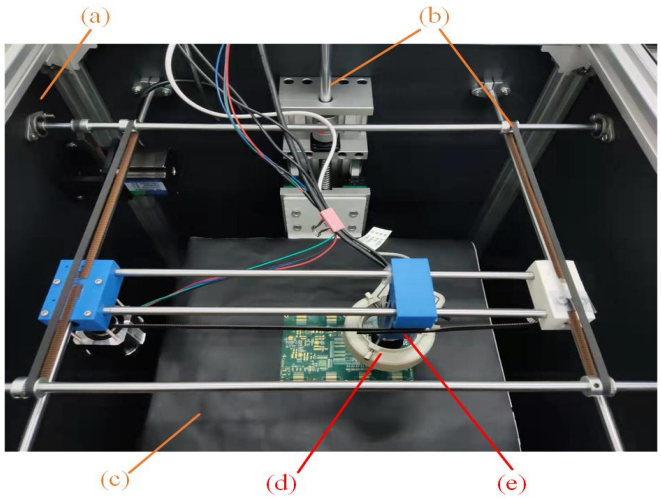
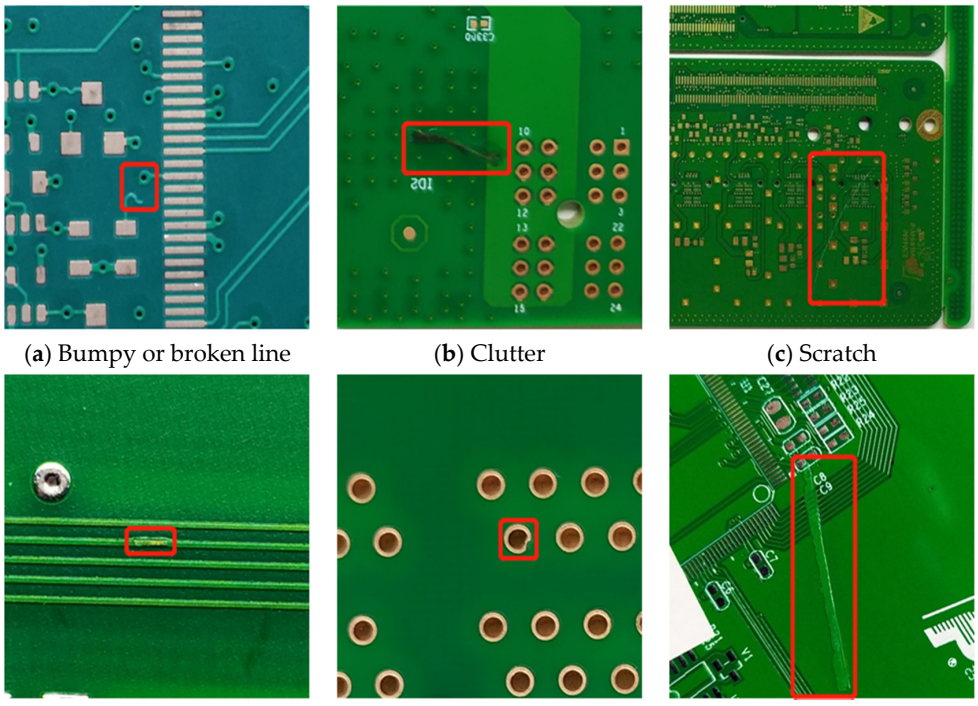


Figure 6. PCB image acquisition device. (a) dark box, (b) motion control parts, (c) platform, (d) illumination module and (e) image acquisition module.

3.2. Defect Images Collection

The original 2008 surface defect images, with one surface defect in each, were collected from a PCB production factory in Guangzhou, China, with the device given in Figure 6, and the size of each defect image was 4024 × 3036 pixels. Six categories of defects, including bumpy or broken line, clutter, scratch, line repair damage, hole loss and over oil-filling, were selected, as these six categories of defect account for more than 80% of surface defects in PCB factories, according to historical statistical data. The instances of each category are given in Figure 7. Each type of surface defect possesses several different morphologies and patterns. As shown in Figure 8, there are five typical morphologies of bumpy or broken lines.



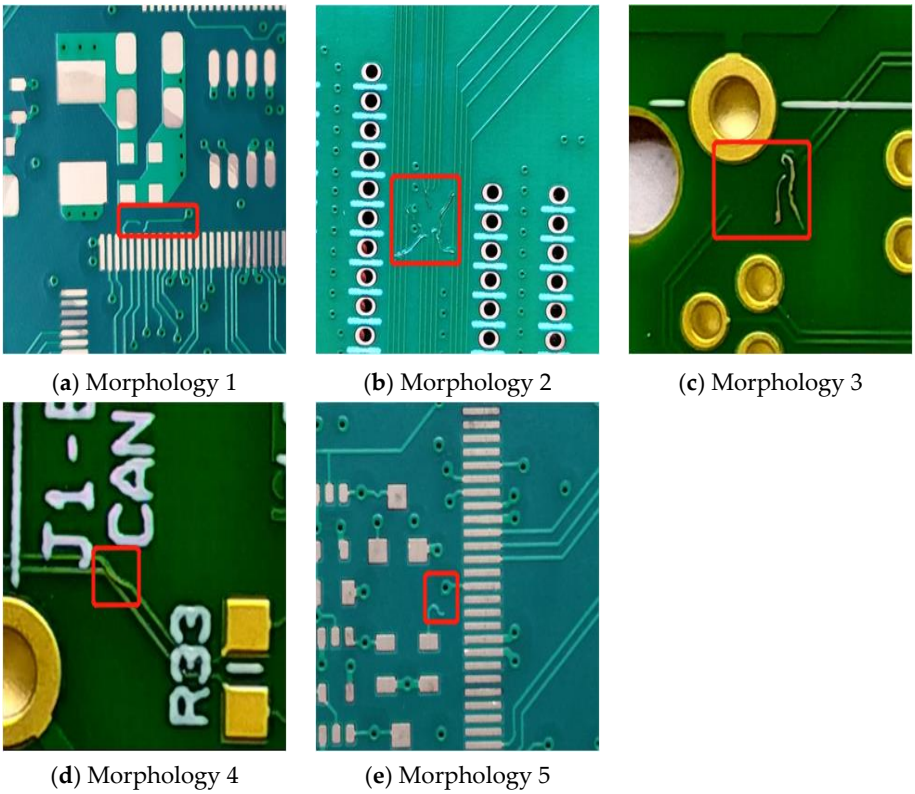
(**a**) Bumpy or broken line (**b**) Clutter (**c**) Scratch

(**d**) Line repair damage (**e**) Hole loss (**f**) Over oil-filling

Figure 7. PCB surface defect instances.

3.3. Data Augmentation and Labeling

Data augmentation can improve the sample diversity and enhance model generalization. It has been widely used in DL-based model development, especially for industrial applications for which it is difficult to obtain large, labeled samples [27]. Random rotation, cropping, translation, horizontal and vertical flipping, luminance balance, etc. are the commonly used data augmentation techniques. Random clipping, image rotation and luminance changes were conducted to augment the original images in this study. Random clipping crops the image randomly, with a mouse click positioned as the operation's center, to get different sizes of image. Rotation augmentation rotates the image at 90°, 180° and 270° angles, such that four different angles of the same defect can be obtained. Luminance changes specify brightness values, adjusting the brightness of the original image. Rotation and randomly clipping images aid detection performance and the robustness of improvement. Luminance changes simulate the deviating brightness of different environmental lighting and improves models' adaptability to different lighting [32]. Some instances of these augmentations are given in Figure 9.

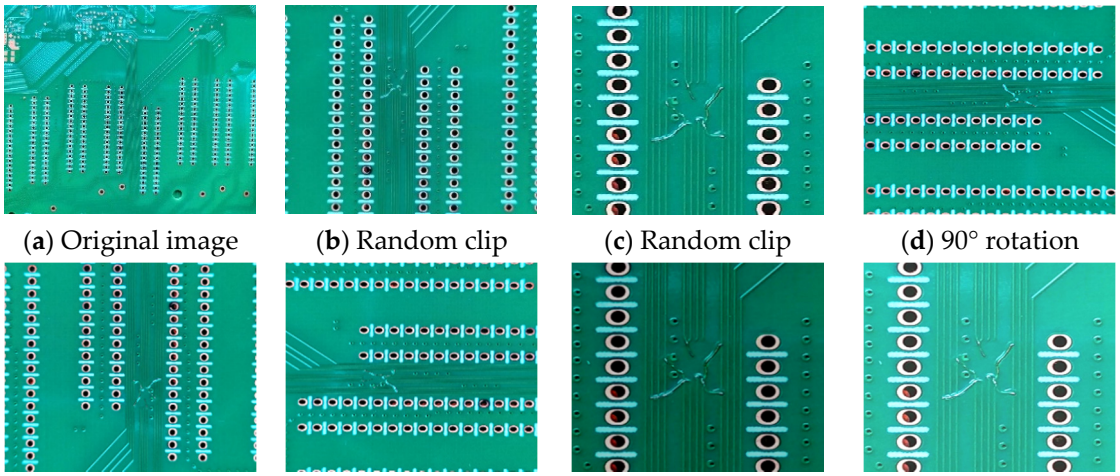


(**a**) Morphology 1 (**b**) Morphology 2 (**c**) Morphology 3

(**d**) Morphology 4 (**e**) Morphology 5

Figure 8. Five different morphologies of bumpy or broken line.

Taking the original high-resolution images as input is conducive to improving detection accuracy. However, large input sizes greatly increase model burden and computing resources. Therefore, all images of the 3018 × 4096-pixels dataset were resized to 416 × 416 pixels in this study. The data augmentation was performed in Python and 19,029 images were obtained, and the number of images belonging to each category of defect are given in Table 2.



(**a**) Original image (**b**) Random clip (**c**) Random clip (**d**) 90° rotation

(**e**) 180° rotation (**f**) 270° rotation (**g**) Luminance change (**h**) Luminance change

Figure 9. Data augmentation instances.

Table 2. PCB surface defect dataset.

Defect Category	Original Images	Augmented Images
bumpy or broken line	345	3090
clutter	332	3458
scratch	443	3463
line repair damage	298	2816
Hole loss	263	3132
over oil-filling	327	3070

Each of the 19,029 defect images were manually marked with a rectangle and labelled with their category. The annotated images mark the baseline truth for each defect, and they can be utilized to evaluate the training loss of *IoU* when combined with the predicted bounding box. The surface defect in each image was labeled by the program LabelImg and stored in PASCAL VOC format. Finally, we randomly split the dataset into training and testing sets, which included 90% and 10% of the images, respectively.

4. Experiment

We conduct three experiments based on the customized PCB dataset to validate the proposed YOLOv4-MN3 in this section. First, the accuracy, parameters, operators and detection-speed performance of different backbone networks are compared for the selection of MobileNetV3. Second, the accuracy performances of different activation functions in the neck/prediction network are compared to facilitate Mish selection. Third, the performance of YOLOv4-MN3 is compared with Faster R-CNN, RetinaNet, SSD, YOLOv3 and YOLOv4 to verify the superiority of the proposed approach. We use the deep learning framework PyTorch1.7 to implement YOLOv4-MN3 and all the compared SOTA models. The experimental environment was ubuntu18.04, CUDA11.0, CUDNN8.0.5 and an NVIDIA GeForce RTX 3080.

4.1. Evaluation Metrics

AP, *mAP*, and F_1 score are taken to evaluate the detection accuracy, and they can be defined as follows:

$$AP = \int_0^1 P(R)dR \tag{2}$$

$$mAP = \frac{\sum_{i=1}^{C} AP_i}{C} \tag{3}$$

$$F_1 = 2 \times \frac{P \times R}{P + R} \tag{4}$$

where $P(R)$ is the precision of a class when recall is R and C is the number of all categories in the image dataset. Recall and precision can be defined as follows:

$$R = \frac{TP}{TP + FN} \tag{5}$$

$$P = \frac{TP}{TP + FP} \tag{6}$$

where *TP*, *FN* and *FP* refer to true positive, false negative and false positive respectively. The prediction anchor box will be categorized into true positive (*TP*) if the IoU is greater than or equal to a pre-setting threshold T, and it is set as 0.5 in this study.

The number of model parameters (*Params*), *Madds* are used to quantitatively evaluate the computational burden of model [25]. And *FPS* is used to evaluate the detection speed. The higher *mAP*, F_1 score and *FPS* but lower *Params* and *Madds*, the better the detector.

4.2. Training Details for YOLOv4-MN3

Pre-training on large, natural-image datasets is a common strategy for DL-based detector training. Therefore, we pre-trained the proposed YOLOv4-MN3 on VOC2007 first. Then, the model training is split into two stages. In the first stage, pre-trained parameters are taken as the initial weights and the weights of the backbone network are frozen, then the parameters of the neck and head networks are trained and optimized. In the second stage, all the parameters are trained based on the first stage network weights. An Adam optimizer is employed to update the parameters with a weight decay of 5×10^{-4}. A total of 100 epochs are performed in the model training, and both first and second stage last 50 epochs. The memory occupancy of the first stage is smaller than the second stage because of the pre-training. Accordingly, the batch sizes were set to eight and two for the two training stages, respectively, and their learning rates were set to 0.001 and 0.0001, respectively.

Optimizing for anchor size that can match well with the size of detected defect facilitates the model's achieving better detection performance. The default anchor size of YOLOv4 is generated based on natural image objects that were not optimized for PCB surface defect detection. Therefore, K-means is employed to cluster and reset the anchor size based on the labeled data to better match the size of defect. Nine anchor boxes of sizes (26,23), (38,38), (52,48), (60,69), (94,83), (116,131), (140,41), (157,284), (239,149) were obtained after clustering. Different three-scale prediction output layers are employed to detect defects of different scales in YOLOv4-MN3. The three smallest anchor boxes were allocated to the 52×52 prediction output layers, the three middle-sized anchor boxes were assigned to 26×26 prediction output layers, and the 13×13 prediction output layers used the three largest anchor boxes.

4.3. Impacts of Different Backbone Networks

The CSPDarknet53 backbone network in the original YOLOv4 suffers from a large number of parameters and a huge computational burden. We tried replacing it with lightweight networks; however, not without a loss of detection performance. Six backbone networks—VGG16, Resnet50, Darknet53 (backbone network of YOLOv3), CSPDarknet53 (backbone network of YOLOv4), MobileNetV2 and MobileNetV3—were selected and comparison experiments were conducted to verify their impact on YOLOv4. VGG16 and Resnet50 are commonly used backbone networks, while Darknet53 and CSPDarknet53 are the SOTA backbone networks for one-stage detectors. The low-power and low-latency of parameterized MobileNet is commonly used for small models [33]. The comparison results are given in Table 3 according to the evaluation metrics given in Section 4.1.

Table 3. Performance of different backbone network.

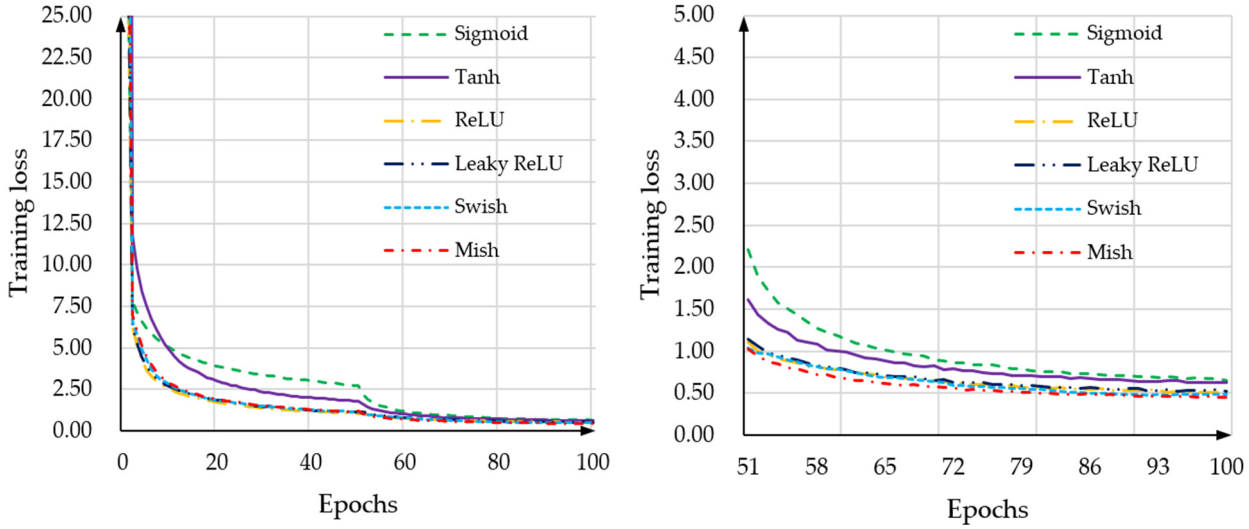
Backbone Network	*mAP* (%)	F_1 (%)	*Params* (M)	*Madds* (G)	*FPS*
VGG16	96.00	93.17	51.80	206.03	55.58
Resnet50	95.10	90.67	61.54	53.98	56.74
Darknet53	95.61	93.00	77.93	74.24	54.93
CSPDarknet53	96.49	95.17	63.96	59.75	51.64
MobileNetV2	94.95	91.33	38.66	26.71	56.99
MobileNetV3	**97.26**	**95.83**	**39.59**	**26.15**	**57.12**

It can be seen that MobileNetV3 obtained the highest *mAP* among all backbone networks. Compared with VGG16, Resnet50, Darknet53, CSPDarknet53, and MobileNetV2, the value of *mAP* obtained by MobileNetV3 improves by 1.26%, 2.16%, 1.65%, 0.77% and 2.31% respectively. Meanwhile, MobileNetV3 achieved the highest F_1 score among the six backbone networks with 2.66%, 5.16%, 2.83%, 0.66% and 4.50% greater improvement over VGG16, Resnet50, Darknet53, CSPDarknet53, and MobileNetV2 respectively. The results of the *mAP* and F_1 score presented in Table 3 indicate that MobileNetV3 outperforms other compared backbone networks in prediction accuracy. Comparing the results from

Params and *Madds* shows that MobileNetV3 greatly simplified the backbone network, requiring only 76.43%, 64.33%, 50.80% and 61.90% of the model parameters used by VGG16, Resnet50, Darknet53 and CSPDarknet53, respectively, and it achieved the lowest *Madds* among these networks as well. The results also show that MobileNetV3 exhibited no significant difference between either MobileNetV2 or MobileNetV3 with respect to *Params* and *Madds*. The *FPS* comparison indicated that MobileNetV3 achieved the highest detection speed, with 57.12 *FPS*. Notably, the *Madds* of MobileNetV3 reduced by 33.60 G, whereas its *FPS* increased by 5.48 as compared with CSPDarknet53 in the original YOLOv4. In summary, MobileNetV3 achieved the highest detection accuracy, low model parameters, lowest *Madds* and fastest detection speed, making it distinctly advantageous for detection.

4.4. Impacts of Different Activation Functions

Popular activation functions in neck and prediction networks, including Sigmoid, Tanh, ReLU, Leaky ReLU, Swish, and Mish were implemented and experimentally compared based on the customized dataset. The detection performance of the six activation functions is shown in Figure 10 and Table 4. In Figure 10a, the training losses of Sigmoid and Tanh are higher than other four activation functions at the first 50 epochs, and they tend to stop converging at the end of the first training stage. However, the training losses of all activation functions decreased again, at the 51st epoch, when all the parameters were unfrozen at the second training stage. Since the training losses were close to each other in the second stage, we selected the training loss from epoch 51–100, given in Figure 10b, to amplify the loss difference between different activation functions clearly. The detailed comparison result given in Figure 10b shows that Mish obtained the lowest training loss, which indicates it had the best training result, generally. It can be roughly concluded that Mish outperformed the other activation functions.



(**a**) Training loss of 100 epochs (**b**) Training loss from epoch 51 to 100

Figure 10. Training loss of different activation functions.

As shown in Table 4, the detection performance varies for different activation functions. Mish's activation function obtained the best *mAP* value, of 98.64%, a 2.95%, 2.02%, 1.62%, 1.38% and 1.36% increase over Sigmoid, Tanh, ReLU, Leaky ReLU and Swish, respectively. Meanwhile, Mish achieved the highest F_1 score, with a 1.5–3.3% increase as compared with the other five activation functions. The *AP* value obtained by Mish outperformed the compared activation functions in the surface defect categories bumpy or broken line, scratch and line repair damage, and showed only slight inferiority to the best of the five activation functions for the defects of clutter and hole loss. One interesting finding is that the detection accuracy for scratches was worse than other defects for all

activation functions. The reason may be that the color of the scratches was similar to their backgrounds, and they are characteristically tiny lines occupying a small number of pixels relative to the overall board, which is not conducive to distinguishing them from their backgrounds. Meanwhile, the diversity of scratch patterns and scales may have hindered the detection performance. However, Mish still achieved 94.20% *AP* and exhibited obvious superiority over the compared activation functions, which further shows that it is suitable for improving detection accuracy.

Table 4. Performance of different activation functions on YOLOv4-MN3.

Activation Function	*AP* (%)						*mAP* (%)	F_1 (%)
	Bumpy or Broken Line	Clutter	Scratch	Line Repair Damage	Hole Loss	Over Oil-Filling		
Sigmoid	98.12	96.80	87.18	97.44	95.64	98.95	95.69	94.50
Tanh	98.56	95.94	88.33	98.43	99.15	99.32	96.62	95.67
ReLU	98.89	99.41	88.53	96.35	98.95	99.99	97.02	95.75
Leaky ReLU	99.10	98.98	90.92	95.73	99.51	99.32	97.26	95.83
Swish	99.66	98.41	86.80	98.87	99.91	100.00	97.28	96.33
Mish	**100.00**	**98.69**	**94.20**	**99.36**	**99.61**	**100.00**	**98.64**	**97.83**

4.5. Comparison of Different Detectors

The detection accuracy, speed and model complexity of YOLOv4-MN3 were tested and compared with the other five SOTA detectors, Faster RCNN, RetinaNet, SSD, YOLOv3, and original YOLOv4. The experimental results of the different indicators are illustrated in Figure 11. It can be seen that the proposed YOLOv4-MN3 achieve the highest detection accuracy, with a 98.64% *mAP* value, which is 5.69%, 4.58%, 3.28%, 1.14% and 2.15% superior to Faster-RCNN, RetinaNet, SSD, YOLOv3 and YOLOv4, respectively. Meanwhile, YOLOv4-MN3 outperformed the other five compared models in F_1, at 20.66%, 14.00%, 10.66%, 2.33% and 2.66% higher than Faster-RCNN, RetinaNet, SSD, YOLOv3 and YOLOv4 respectively. In terms of model complexity, YOLOv4-MN3 greatly reduced its parameters and *Madds*. The number of *Params* needed by YOLOv4-MN3 reduced by 28.94%, 64.32% and 61.90% those of Faster RCNN, YOLOv3 and YOLOv4, respectively, and the *Madds* of YOLOv4-MN3 was 90.45 G, 39.29 G and 33.60 G lower than of Faster RCNN, YOLOv3 and YOLOv4, respectively. The reason for this is that MobileNetV3 uses depthwise separable convolution, replacing the standard convolution, allowing the convolution weights and operations to be reduced greatly. In addition, RetinaNet and SSD had small numbers of parameters but large *Madds*. The reason for this is that the number of parameters indicates the space complexity or storage consumption, and *Madds* is the metric of time complexity. Detectors with small *Params* cannot ensure lower *Madds* or higher detection speed, and vice versa. For example, the activation function and pooling operations increased *Madds* while *Params* remained unchanged. Meanwhile, we can see that the *Madds* of YOLOv4-MN3 is 43.72 G and 90.26 G lower than RetinaNet and SSD, although with a slight increase in parameters. YOLOv4-MN3 also had significant advantage in detection speed, and its *FPS* value was greater by 10.60, 3.01, 1.98, 11.49 and 5.34 than those of Faster RCNN, RetinaNet, SSD, YOLOv3 and YOLOv4. Based on the comparison of SOTA detectors, the YOLOv4-MN3 model delivered significant advantage in terms of detection accuracy, model parameters, operations and detection speed.

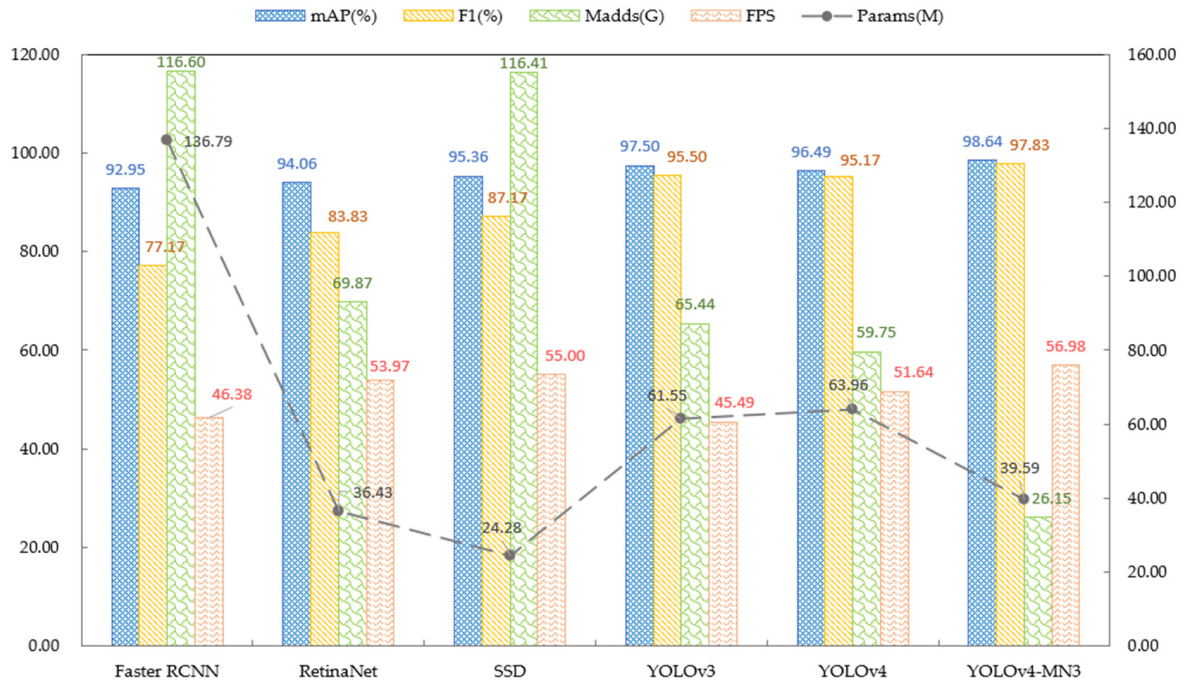
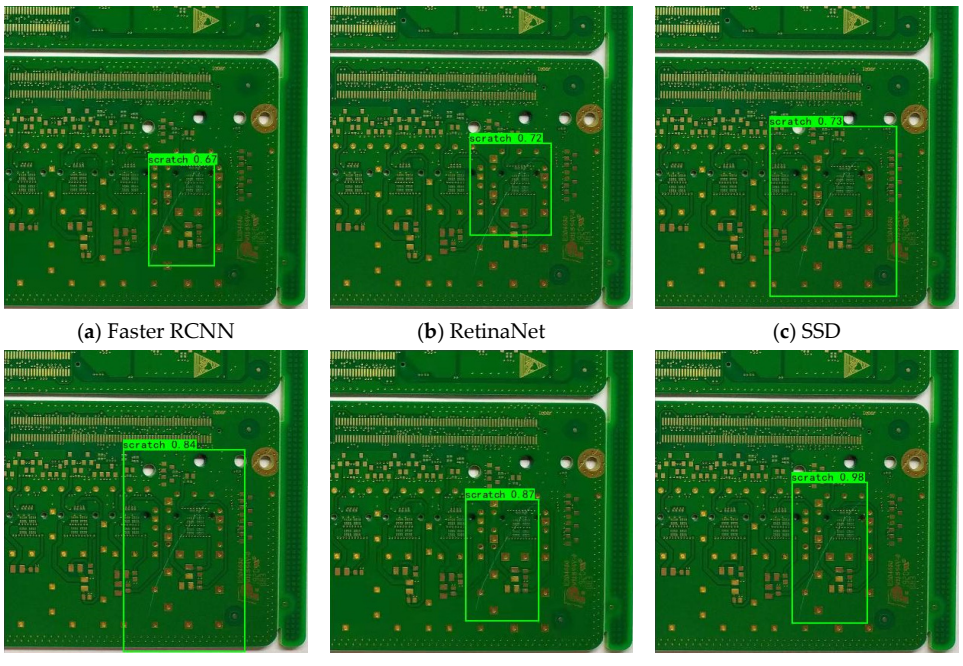


Figure 11. Performance of different SOTA models.

Figure 12 gives some detection instances obtained by YOLOv4-MN3 and the compared detectors. The detection instances given in Figure 12a,b show that Faster RCNN and RetinaNet only detected a part of the defect with low confidence. SSD and YOLOv3 defect the whole defect but also with low confidence, as shown in Figure 12c,d. Figure 12e,f indicates that both YOLOv4 and YOLOv4-MN3 could detect the whole defect. However, YOLOv4-MN3 achieved the highest confidence 0.98, while the confidence of YOLOv4 was only 0.87, in this instance.

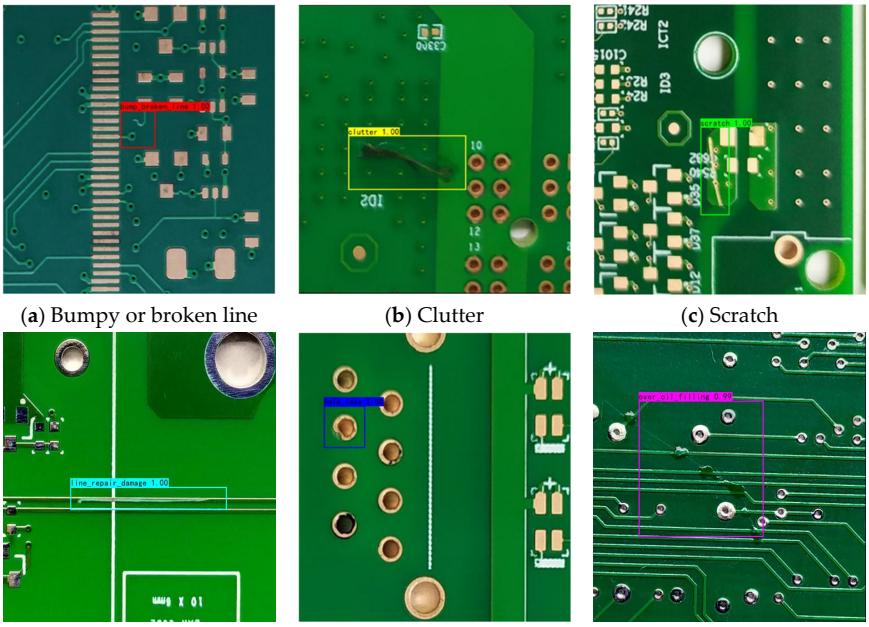


(**a**) Faster RCNN (**b**) RetinaNet (**c**) SSD

(**d**) YOLOv3 (**e**) YOLOv4 (**f**) YOLOv4-MN3

Figure 12. Detection instance obtained by different detectors.

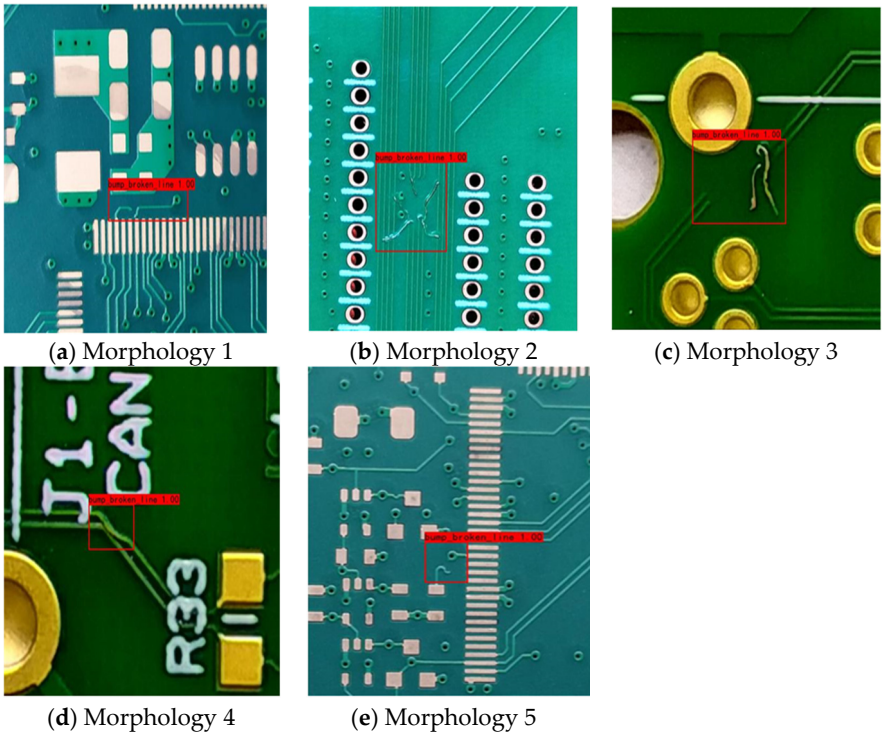
Figure 13 shows some detection instances for the aforementioned six categories of defects. Figure 14 gives an instance of detecting "bumpy or broken line" defect category, with its five different morphologies. The instances detected with high confidence, given in Figures 13 and 14, also indicate that the proposed YOLOv4-MN3 can adapt to different categories of surface defect, and it can handle the difficult problem of a diversity of defect morphologies.



(**a**) Bumpy or broken line (**b**) Clutter (**c**) Scratch

(**d**) Line repair damage (**e**) Hole loss (**f**) Over oil-filling

Figure 13. Detection instance of different defects obtained by YOLOv4-MN3.



(**a**) Morphology 1 (**b**) Morphology 2 (**c**) Morphology 3

(**d**) Morphology 4 (**e**) Morphology 5

Figure 14. Detection instance of bumpy or broken line with different morphologies obtained by YOLOv4-MN3.

5. Conclusions

In order to improve the performance of PCB surface defect detection, a DL-based detector, YOLOv4-MN3, was proposed. YOLOv4-MN3 replaces the CSPDarknet53 backbone network of YOLOv4 with a lightweight one, MobileNetV3, and the original activation function in the neck/prediction network was optimized. To verify the efficiency and the effectiveness the proposed detector, a customized dataset was built using a specially designed image-acquisition device. Experimental results on the customized dataset showed that YOLOv4-MN3 achieved the highest detection accuracy, with 98.64% *mAP* and 97.83% F_1, the fastest detection speed, at 56.98 *FPS*, and the lowest *Madds* as compared with the SOTA models. Generalization experiments of YOLOv4-MN3 indicated that it can adapt to different categories of surface defect and handle the difficult problem of defect morphology diversity, which has promising application prospects.

Although the detector proposed in this paper achieved good results, there are some problems to explore. Firstly, PCB surfaces' backgrounds, defect categories and morphology diversity, in real industry, are complex and diverse. Thus, it is necessary to update PCB surface defect samples continuously and facilitate the training of practicability-oriented DL-based models. Secondly, YOLOv4-MN3, as a supervised learning method, requires a large amount of manually labeled samples, and exploring semi-supervised or unsupervised mechanism for this problem is a worthy attempt.

Author Contributions: Conceptualization, X.L. and S.L.; methodology, X.L. and S.L.; software, X.L. and D.L.; validation, X.L. and Y.L.; formal analysis, X.L. and Z.Z.; investigation, X.L. and D.L.; re-sources, X.L. and D.L.; data curation, C.J. and Y.L.; writing—original draft preparation, X.L. and S.L.; writing—review and editing, X.L. and S.L.; visualization, X.L.; supervision, S.L.; project administration, S.L.; funding acquisition, S.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research is funded by Natural Science Foundation of Guangdong, China with grant number 2021A1515012395.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Acknowledgments: The authors wish to thank the editor and reviewers for their suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dai, W.; Mujeeb, A.; Erdt, M.; Sourin, A. Soldering defect detection in automatic optical inspection. *Adv. Eng. Inform.* **2020**, *43*, 101004.
2. Ebayyeh, A.; Mousavi, A. A review and analysis of automatic optical inspection and quality monitoring methods in electronics industry. *IEEE Access.* **2020**, *8*, 183192–183271. [CrossRef]
3. Wang, W.C.; Chen, L.B.; Chang, W.J.; Chen, S.L.; Li, S.M. A machine vision based automatic optical inspection system for measuring drilling quality of printed circuit boards. *IEEE Access.* **2017**, *5*, 10817–10833. [CrossRef]
4. Gaidhane, V.H.; Hote, Y.V.; Singh, V. An efficient similarity measure approach for PCB surface defect detection. *Pattern Anal. Appl.* **2018**, *21*, 277–289. [CrossRef]
5. Yuk, E.H.; Park, S.H.; Park, C.-S.; Baek, J.-G. Feature-Learning-Based Printed Circuit Board Inspection via Speeded-Up Robust Features and Random Forest. *Appl. Sci.* **2018**, *8*, 932. [CrossRef]
6. Fonseka, C.; Jayasinghe, J. Implementation of an automatic optical inspection system for solder quality classification of THT solder joints. *IEEE Trans. Compon. Packag. Manufact. Tech.* **2018**, *9*, 353–366. [CrossRef]
7. Liu, Z.; Qu, B. Machine vision based online detection of PCB defect. *Microprocess. Microsyst.* **2021**, *82*, 103807. [CrossRef]
8. Zou, Z.; Shi, Z.; Guo, Y.; Ye, J. Object detection in 20 years: A survey. *arXiv preprint* **2019**, arXiv:1905.05055. Available online: https://arxiv.org/abs/1905.05055 (accessed on 16 May 2019).
9. Rida, I.; Al-Maadeed, N.; Al-Maadeed, S.; Bakshi, S. A comprehensive overview of feature representation for biometric recognition. *Multimed. Tools Appl.* **2020**, *79*, 4867–4890. [CrossRef]

10. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
11. Girshick, R. Fast R-CNN. *arXiv preprints* **2015**, arXiv:1504.08083. Available online: https://arxiv.org/abs/1504.08083 (accessed on 30 April 2015).
12. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
13. Sharma, V.; Mir, R.N. A comprehensive and systematic look up into deep learning based object detection techniques: A review. *Comput. Sci. Rev.* **2020**, *38*, 100301. [CrossRef]
14. Kou, X.; Liu, S.; Cheng, K.; Qian, Y. Development of a YOLO-V3-based model for detecting defects on steel strip surface. *Measurement* **2021**, *182*, 109454. [CrossRef]
15. Li, D.; Xie, Q.; Gong, X.; Yu, Z.; Xu, J.; Sun, Y.; Wang, J. Automatic defect detection of metro tunnel surfaces using a vision-based inspection system. *Adv. Eng. Inform.* **2021**, *47*, 101206.
16. Shu, Y.F.; Li, B.; Li, X.; Xiong, C.; Cao, S.; Wen, X.Y. Deep learning-based fast recognition of commutator surface defects. *Measurement* **2021**, *178*, 109324.
17. Ding, R.; Dai, L.; Li, G.; Liu, H. TDD-net: A tiny defect detection network for printed circuit boards. *CAAI Trans. Intell. Technol.* **2019**, *4*, 110–116. [CrossRef]
18. Hu, B.; Wang, J. Detection of PCB Surface Defects with Improved Faster-RCNN and Feature Pyramid Network. *IEEE Access* **2020**, *8*, 108335–108345. [CrossRef]
19. Zhang, Y.; Xie, F.; Huang, L.; Shi, J.; Yang, J.; Li, Z. A Lightweight One-Stage Defect Detection Network for Small Object Based on Dual Attention Mechanism and PAFPN. *Front. Physics.* **2021**, *9*, 491. [CrossRef]
20. Bochkovskiy, A.; Wang, C.Y.; Liao, H. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint* **2020**, arXiv:2004.10934. Available online: https://arxiv.org/abs/2004.10934 (accessed on 23 April 2020).
21. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv preprint* **2018**, arXiv:1804.02767. Available online: https://arxiv.org/abs/1804.02767 (accessed on 8 April 2018).
22. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans Pattern Anal. Mach. Intell.* **2014**, *37*, 1904–1916. [CrossRef]
23. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path Aggregation Network for Instance Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768. Available online: https://ieeexplore.ieee.org/document/8579011 (accessed on 17 December 2018).
24. Wang, C.Y.; Liao, H.; Wu, Y.H.; Chen, P.Y.; Yeh, I.H. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 1571–1580.
25. Howard, A.; Sandler, M.; Chen, B. Searching for mobileNetV3. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 1314–1324.
26. Misra, D. Mish: A self regularized non-monotonic neural activation function. *arXiv preprint* **2019**, arXiv:1908.08681. Available online: https://arxiv.org/abs/1908.08681 (accessed on 23 August 2019).
27. Guo, F.; Qian, Y.; Shi, Y. Real-time railroad track components inspection based on the improved yolov4 framework. *Autom. Constr.* **2021**, *125*, 103596. [CrossRef]
28. Ramachandran, P.; Zoph, B.; Le, Q.V. Searching for activation functions. *arXiv preprint* **2017**, arXiv:1710.05941. Available online: https://arxiv.org/abs/1710.05941 (accessed on 16 October 2017).
29. Glorot, X.; Bordes, A.; Bengio, Y. Deep sparse rectifier neural networks. *J. Mach. Learn. Res.* **2011**, *15*, 315–323. Available online: https://www.researchgate.net/publication/215616967 (accessed on 14 January 2010).
30. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *Proc. IEEE Int. Conf. Comput. Vision* **2015**, *1*, 1026–1034.
31. Eger, S.; Youssef, P.; Gurevych, I. Is it Time to Swish? Comparing Deep Learning Activation Functions across NLP tasks. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 4415–4424.
32. Zhao, J.; Zhang, X.; Yan, J.; Qiu, X.; Yao, X.; Tian, Y.; Zhu, Y.; Cao, W. A Wheat Spike Detection Method in UAV Images Based on Improved YOLOv5. *Remote Sens.* **2021**, *13*, 3095. [CrossRef]
33. Kulkarni, U.; Meena, S.M.; Gurlahosur, S.V.; Bhogar, G. Quantization friendly mobilenet (qf-mobilenet) architecture for vision based applications on embedded platforms. *Neural Netw.* **2021**, *136*, 28–39. [CrossRef]