The Journal of Engineering

The 3rd Asian Conference on Artificial Intelligence Technology (ACAIT 2019)

IET Journals
The Institution of Engineering and Technology

Single-shot detector with enriched semantics for PCB tiny defect detection

eISSN 2051-3305
Received on 11th October 2019
Accepted on 19th November 2019
doi: 10.1049/joe.2019.1180
www.ietdl.org

Wei Shi[1], Zhisheng Lu[1] ✉, Wei Wu[2], Hong Liu[1]

[1]*Key Laboratory of Machine Perception, Peking University, Shenzhen Graduate School, People's Republic of China*
[2]*Shenzhen Skyworth-RGB Electronic, Co., Ltd., Shenzhen, People's Republic of China*
✉ *E-mail: zhisheng_lu@pku.edu.cn*

Abstract: Printed circuit board (PCB) defect detection is one of the primary problems in quality control of the most electronic products. Usually, the industrial PCB imagery has high resolution, but defects take up a small proportion (often only ∼10 pixels in size), which makes it difficult to use traditional machine vision methods. To this end, a novel single shot object detector (SSDT) is proposed for tiny defect detection in PCBs in this study. Specifically, a semantic ascending module, which propagates the semantic property of deep layers to shallow layers, is presented by fusing features of different levels. An attention mechanism is utilised to learn the relationship of the features to be fused across channels and a shuffle module is used to eliminate the aliasing effect after fusion. Moreover, the improved non-maximum suppression is proposed to extenuate the overlap effect for testing the whole PCB image. The proposed detector can rapidly detect tiny defects and the results of SSD and SSDT are further compared not only in PCB defect dataset but also the object detection public dataset PASCAL VOC2007 where SSDT achieves 81.3% mAP, better than SSD (79.5%). In final, the proposed detector is validated to be robust to rotation and blur.

1 Introduction

In industrial production, a printed circuit board (PCB) may exist some tiny defects, such as mouse-bite, missing-hole and spur, which would affect industrial use. Generally, on production lines, tiny defects need to be manually picked up, which is extremely time-consuming, inefficient and labor intensive. Therefore, there are many machine vision methods for defect inspection. Lu *et al.* [1] tried the support vector machine with feature fusion to detect the PCB defects. Ozturk and Akdemir [2] used fuzzy *c*-means algorithm in unsupervised learning to detect pad defects in the PCB and get a better performance than some existing methods. Zhang *et al.* [3] proposed a three-layer convolutional neural network to classify defects. Compared with manual inspection, these methods have many advantages, such as persistence, high speed and low cost, but they cannot locate the defects precisely. For this reason, object detection techniques are used to locate and recognise tiny defects in PCB imagery taken by the industrial camera in this paper.

Computer vision techniques have made great strides in the past few years since the introduction of convolutional neural networks [4–7] in the ImageNet [8] competition. In terms of object detection, there are three main frameworks: Faster R-CNN [9–11], SSD [12], and YOLO [13]. Faster R-CNN typically ingests 1000×600 pixel images, and YOLO runs on either 416×416 or 544×544 pixel images, whereas SSD uses the input images with 300×300 or 512×512 pixels. Although the performance of all these frameworks is commendable, none can handle the typical PCB images with an input size of $\sim 3000 \times 3000$. Among these three frameworks, SSD has the greatest inference speed and highest accuracy on the PASCAL VOC dataset. Besides, SSD can be flexibly adapted to the different detection tasks. Due to the speed, accuracy, and flexibility of SSD, we use it as the backbone and then make a series of improvements based on it.

Compared with objects of general size, SSD has the limitation that small objects cannot be detected well. This is not the problem only for SSD but the problem for most object detection algorithms. Also, SSD does not consider the relationship between the different scales because each layer predicts the various scale boxes for one object, respectively. The low-level features in the shallow layer which detects small objects have little semantic information, and the high-level features in the deep layer have little spatial information. Hence, the low-level features are helpful for object location regression sub-task and the high-level features are discriminative for classification. The original SSD has poor ability for small object detection, because of the lack of semantics in low-level features. To this end, we propose a novel single shot detector called single shot object detector (SSDT). Specially, our SSDT propagates the features from deep layers after up-sampling to shallow layers to enrich the semantics by a semantic ascending module. After fusion, a shuffle module is proposed to eliminate the aliasing effect. To guarantee the effectiveness of fusion features, the squeeze-and-excitation module is used to inhibit the useless channels. All these help solve the problem that low layer has less semantic information to increase the detection performance for small objects.

It is not easy to utilise deep learning method to traditional machine vision detection pipelines because the unique aspects of PCB imagery necessitate algorithmic contributions to address challenges related to the small spatial extent of tiny defects, rotation variance and a large scale search space. The proposed algorithms must adjust for following aspects:

Small spatial extent: In PCB imagery, defects are often very small as Fig. 1 shows rather than the large and prominent subjects typical in ImageNet data [8]. The area of a PCB usually is $10\ \text{cm} \times 10\ \text{cm}$ and a tiny defect only has $3\ \text{mm} \times 3\ \text{mm}$, which means that tiny defects will be only ∼10 pixels in extent even at really high resolution. If the entire image is input into the detector, it is difficult to extract features of defects.

Rotation variance: In actual production lines, camera may shake and shift, so the captured images may be askew. Tiny defects viewed from above can have any orientation.

Lack of enough training data: Today, there are many datasets for common objects in actual scenarios. However, industrial datasets are really rare. The main reason is that factories and companies almost do not disclose their data. There is a relative dearth of training data.

Ultra high resolution: In general, industrial PCB imagery is large scale and the input size of the detector is generally small. If we resize it directly, tiny defects may be hard to be discovered and even disappear. So simply down-sampling to the input size required by most algorithms is not an option.

The contributions of this work can be summarised as follows:

(i) A novel single-shot detector called SSDT is proposed for tiny defects detection. It consists of three modules: a semantic ascending module for the enhancement of semantics in low-level features, an attention mechanism for learning the relationship of features across channels and a shuffle module for easing the aliasing effect.
(ii) An improved non-maximum suppression method is proposed to reduce the error rate for testing the whole PCB images by sliding window.
(iii) Extensive experiments on PCB defect dataset and VOC2007 dataset are carried out to evaluate the effectiveness of our method.n PCB production process.

2 Single shot detector for tiny defects

In SSD, although multi-stages are used for localisation and classification, there is no connection between stages, as shown in Fig. 2*d*. The SSD uses the shallow layer's feature map to detect small objects, but lower layers have less semantic information. So it is not robust enough for SSD to detect small objects. To address this challenge in PCB tiny defect detection, we propose a novel object detection framework for PCB image: single shot detector for tiny defect detection (SSDT).

The architecture is shown in Fig. 3. Like SSD, VGG16 is used as the backbone. In order to make the defects as large as possible, the 512×512 size is used as the input size. Two independent semantic ascending modules are built to deliver high-level features to shallow layers to enrich its semantic information. Followed by them there are two shuffle modules to eliminate the aliasing effect. Some training strategies are designed to meet our requirements for PCB tiny defect detection. Meanwhile, we improve the non-maximum suppression (NMS) method due to the truncation of defects for testing the whole PCB images.

2.1 Semantic ascending module

As shown in Fig. 2, there have been a lot of algorithms trying to effectively utilise the pyramidal features. Fig. 2*a* shows that images with different scales compute features independently. Sometimes only the top features are used to detect objects, which is used in some two-stage detectors such as Faster R-CNN [11] and R-FCN [14], as shown in Fig. 2*b*. In Fig. 2*c*, feature are fused from top to bottom layer by layer which is adopted by feature pyramid networks (FPN) [15]. Fig. 2*d* uses each feature layer generated from a ConvNet to predict like SSD [12]. Our proposed feature fusion method is shown in Fig. 2*e*. Features from high layers are added to the bottom layer to enrich the semantics specifically for small objects. The most common method is like Fig. 2*c*. This type of feature fusion is used in FPN [15] and DSSD [16], and is verified to improve the detector's performance a lot. However, this design needs multiple feature merging processes and consumes a lot of time since multiple features are processed by element-wise summation. Hence, we propose a lightweight and efficient semantic ascending module for small objects and tiny defects in PCB images. Our motivation is to let high-level rich semantic information pass to the shallow features. Assuming that X_i, $i \in C$ are the feature maps that we want to fuse, and X_f, $i \in F$ are the shallow feature maps, the feature fusion module can be described as follows:

$$X_{\mathrm{f}} = \zeta_{\mathrm{f}\in F}\{T_i(X_i)\} \quad i \in C, \tag{1}$$

$$l_{\mathrm{loc,class}} = \zeta_{\mathrm{c,l}}\left(\{X_{\mathrm{f}}\} \cup^{\{X_i\}}\right) \quad f \in F, i \in C, \tag{2}$$

where T_i is the transform function of each feature map before feature fusion. ζ_{f} is the feature fusion function. $\zeta_{\mathrm{c,l}}$ is the loss function of localisation and classification. There are several factors we should consider:

C and F: In the conventional SSD500 based on VGG16, the author chooses conv4_3, fc_7 (we change it to conv_7) of the VGG16 and newly added layer conv6_2, conv7_2, conv8_2, conv9_2, conv10_2 to generate features to perform object detection. The corresponding feature size is 64×64, 32×32, 16×16, 8×8, 4×4, 2×2, 1×1. We add the feature map of layer conv6_2, conv7_2, conv8_2, conv9_2, conv10_2 to the layer conv_7 and add the feature map of layer conv_7, conv6_2, conv7_2, conv8_2, conv9_2, conv10_2 to the layer conv4_3. So F is the set of layer conv4_3 and conv_7 and it means there are two feature fusion pathways independently.

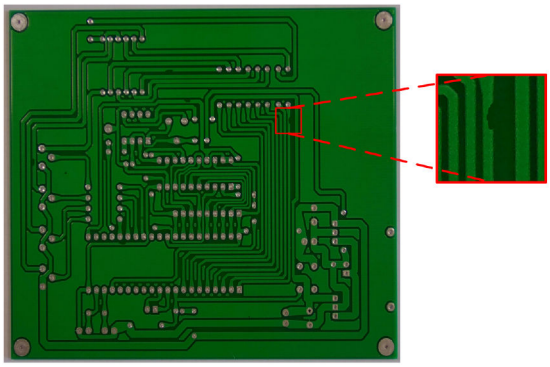

Fig. 1 *Example of PCB image (2838 × 2316 pixels). One mouse-bite defect (5 × 12 pixels) is shown in red*

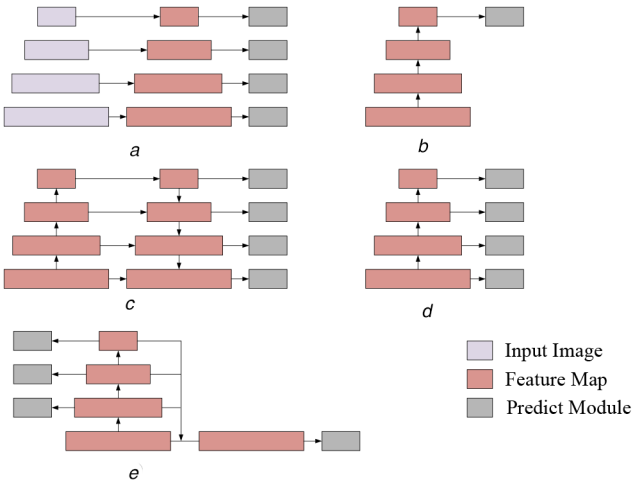

Fig. 2 *Five kinds of feature pyramid methods*

ζ_f: Usually, there are two ways to fuse different feature maps together: concatenation and element-wise summation. Concatenation does not need feature maps with the same channels but it will increase the number of channels. If using concatenation in SSDT, the channels of conv4_3 will be up to 2816. So we prefer to use the element-wise summation. The result in Section 3.5.3 that element-wise summation performs better than concatenation also proved our opinion.

T_i: To match the channels and the size between each feature map with different scales, the following strategy is adopted. First, Conv 1×1 is applied to each of the feature layer $\in C$ to unite the number of channels. Then feature maps fused to the layer conv4_3 are up-sampled to 64×64 and feature maps fused to the layer conv_7 are up-sampled to 32×32 by bilinear interpolation. By this way, all the features have the same size on spatial dimension and the same number of channels.

2.2 Attention mechanism

If simply passing the deep features to the shallow layers, the features of most channels are beneficial, but it is also possible that features of some channels are useless or even will decrease the performance. Hence, an attention mechanism is requisite to filter out features of useless channels. In SSDT, as Fig. 4*c* shows, the 'squeeze-and-excitation' block [17] is utilised to model interdependencies between channels. It mainly consists of two fully

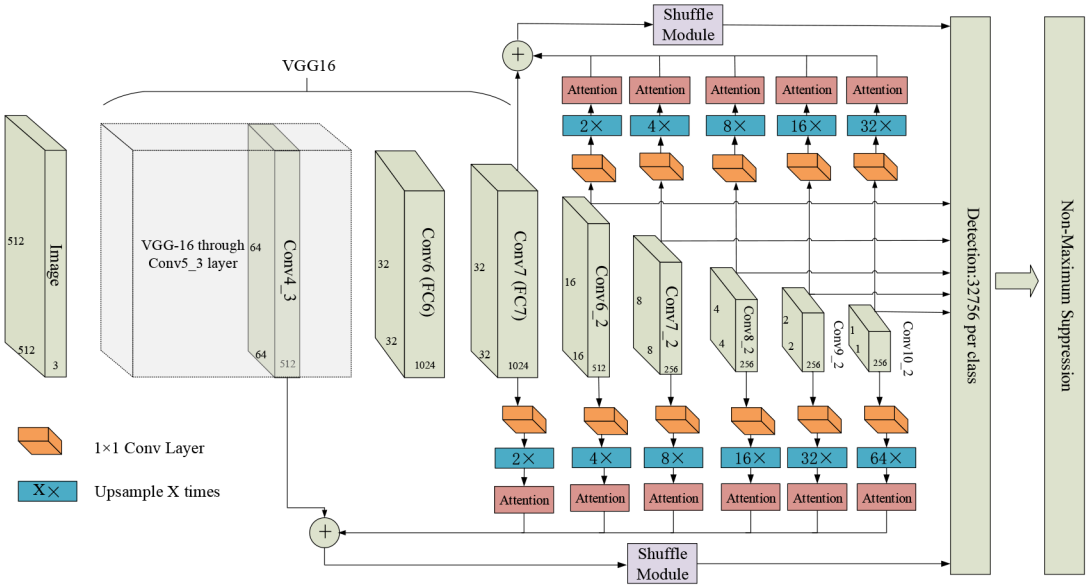

Fig. 3 *Architecture of SSDT. The shallow features are fused with deep features after upsampling and attention mechanism. Then the fusion features are used for classification and localization following the shuffle module*

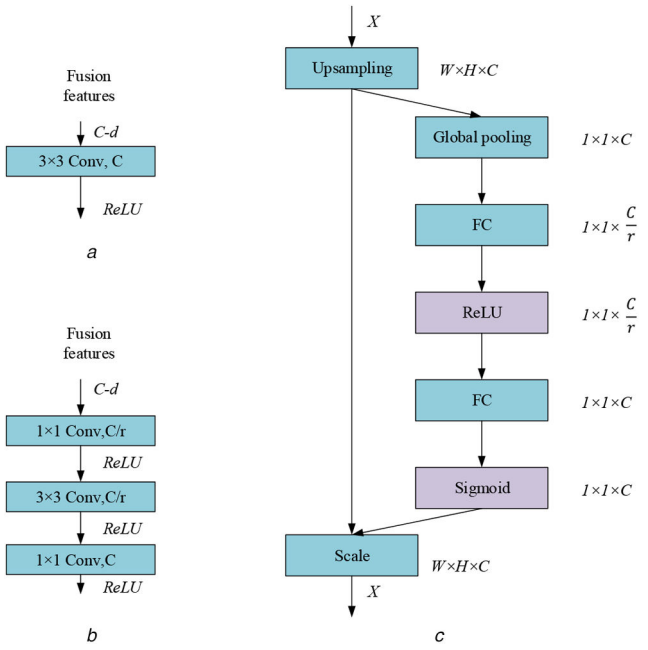

Fig. 4 *Detailed architecture*

(a) The conventional 3 × 3 convolution, ***(b)*** The shuffle module we used, ***(c)*** Attention mechanism, C is the channels of feature map *X*, *r* is a real number to reduce dimension

connected layers and a sigmoid layer. Two fully connected layers squeeze features along the spatial dimension, turning each two-dimensional feature channel into a real number. This number is sent into the sigmoid layer to generate weights of each feature channel. These weights will be weighted to previous features by element-wise multiplication as the importance of each feature channel.

In SSDT, global average pooling is first used to get the spatial information. Assuming $F \in R^{C\times H\times W}$ are feature maps to be fused, the global squeeze can be described as

$$Z_i = \frac{1}{H\times W}\sum_{h,w} F_{i,H\times W}, \quad i \in C, \tag{3}$$

where $Z_i \in R^{C\times 1\times 1}$. After squeeze operation, a simple attention mechanism is adopted with a sigmoid activation:

$$S = \text{Sigmoid}(W_2 \cdot \text{ReLU}(W_1 Z)), \quad S \in R^{C\times 1\times 1}, \tag{4}$$

Table 1 Comparison of parameters

Shuffle module	No. of parameters
none	117 M
our method	122 M
conventional convolution	161 M

where $W_1 \in R^{(C/r)\times C}$ and $W_2 \in R^{C\times(C/r)}$. Those two are produced by two fully connected layers around the non-linearity and r is a reduction ratio to reduce computation. At last, the input F are rescaled by S:

$$F_{\text{out}} = \delta(F_i, S_i), \quad i \in C, \tag{5}$$

where $F_{\text{out}} \in R^{C\times H\times W}$ is the output of this attention mechanism and δ is the multiplying operation. By all this, useless features across channels are restrained.

Batch normalisation (BN) is used in the layer conv4_3 after element-wise summation but not in the layer conv_7. It can achieve better performance compared to using BN in both two layers.

2.3 Shuffle module

As mentioned in FPN [15], the up-sampling has aliasing effect on fusion features. The conventional method appends a 3 × 3 convolution after feature fusion. However, in SSDT, the layer conv4_3 has 512 channels and conv_7 has 1024 channels. If we directly use the architecture like Fig. 4*a*, the parameters will increase a lot.

Inspired by Bottleneck [18], our method is to use 1 × 1 kernel to reduce the dimension as Fig. 4*b* shows. Suppose that the original feature map has C channels, 1 × 1 convolution kernel is first used to reduce to (C/r) channels. Then the 3 × 3 convolution and 1 × 1 convolution kernel are utilised to increase the number of channels to the original number. Our shuffle module can effectively ease the aliasing effect and wouldn't add too much computation. As Table 1 shows, our shuffle module only increases 5M parameters while conventional convolution increases 44M parameters.

2.4 Training strategy

We follow almost the same training policy as SSD. First, a set of default boxes are matched to target ground truth boxes. For each ground box, it is matched with the best overlapped default box and any default box whose Jaccard overlap is larger than a threshold (e.g. 0.5). Among the non-matched default boxes, certain boxes are selected as negative samples based on the confidence loss so that

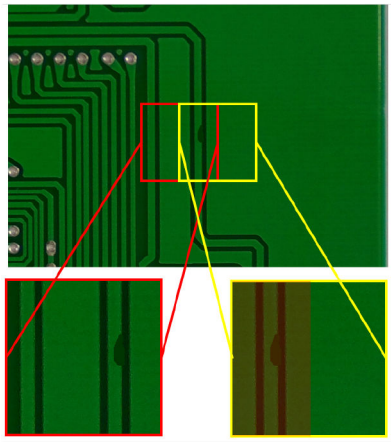

Fig. 5 *Partition of a PCB image (only show a part) into cutouts of 256 × 256 pixels with overlap 0.5 from left to right*

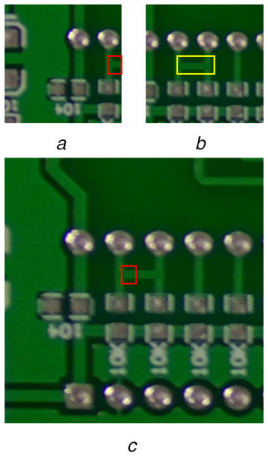

Fig. 6 *Schematic diagram of defect truncation*
(a) A cutout that cuts off a defect, ***(b)*** A cutout that contains the whole defect, ***(c)*** The detection result if the red bounding box has higher score

the ratio with the matched ones is 3:1. Then the joint localisation loss (e.g. Smooth L1 loss) and confidence loss (e.g. Softmax loss) are minimised. Extensive data augmentation is done by randomly cropping the original image plus random photometric distortion and randomly flipping of the cropped patch. Particularly, a random expansion augmentation trick is proved to be extremely helpful for detecting small objects [16] and it is also adopted in our SSDT framework.

The input size of SSDT is 300 or 512. If the PCB images of high resolution directly resize (e.g. 2400 × 3000) into the corresponding scale, tiny defects will shrink or even vanish. At the same time, global information in PCB image is similar to the local information. Hence, the PCB images of arbitrary size are first resized into 2048 × 2048. This scale ensures that the defects cannot change too much. Then the images are partitioned into cutouts of 256 × 256 size with overlap. The overlap rate (overlap area divided by cutout area) is 0.5. However, partitioning cannot change the size of tiny defects. In the PCB images, some tiny defects only account for 10 pixels. The object with the size of 10 pixels is very difficult to detect for SSDT. The minimum size of conv4_3 is 35.84, and the aspect ratios are 2 and 3. So effective minimum size of object is 20 (35.84 divided by the square root of 3). In order to make the defects occupy more pixels, the size of cutouts is doubled by bilinear up sampling so that the pixel number of defects also double. It is worth noting that when partitioning the images, cutouts in the dataset would not truncate the defects, and cutouts would retain the entire defects (see Fig. 5).

For each cutout the bounding box position predictions returned from the classifier are adjusted according to the row and column values of that cutout. This provides the global position of each bounding box prediction in the original input image. The 50% overlap ensures all regions will be analysed, but also results in overlapping detections on the cutout boundaries. So improved non-maximum suppression is applied to the global matrix of bounding box predictions to alleviate such overlapping detections.

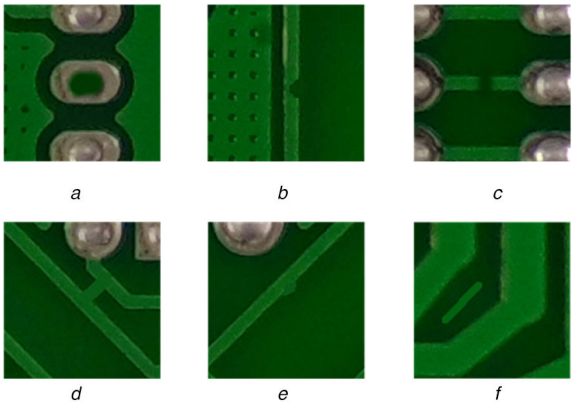

Fig. 7 *Six types of PCB defects*

2.5 Improved non-maximum suppression

For testing a whole PCB image, the sliding window method is used with stride 128. The size of window is the same as the cutouts in the dataset. For the block in the window, we also use bilinear up-sampling method. However, when sliding, the window may cut off the defect like Fig. 6, and the next window will contain the whole defect. If the defect is detected in both two blocks, a special situation may appear as shown in Fig. 6*c*.

In traditional NMS, when the detection frame M of the maximum score is selected, any detection frame that overlaps with the detection frame M by more than the overlap threshold will also be removed. If the smaller bounding box in the figure has higher score, NMS would filter the bigger one. Nevertheless in fact, the bigger one is better. So in our framework, we propose an improved NMS as

$$B = \arg\max_{i \in I} \left(C_i + \lambda \times \frac{S_i}{S_{\max}} \right), \tag{6}$$

where I is the set of indexes of bounding boxes that IoU with each other more than the threshold, λ is the adaptive parameter, C_i is the score of the ith bounding box, S_i is the area of the ith bounding box. The improved NMS considers the area factor and the problem of selecting between the small cutouts with partial defect and the cutouts with integral defect.

3 Experiments and discussions

3.1 Datasets

The design drawings of the PCB used in the dataset are designed by ourselves and processed by the factory. The PCB images are taken by a 16 megapixel industrial camera equipped with a CMOS sensor that can be adjusted by manual, remote control or computer software. In order to maintain the proper proportion of all PCB boards in the image without distortion if the height of the camera is not adjusted, the camera is also equipped with an undistorted zoom industrial lens, and the focal length can be adjusted between 6 and 12 mm.

There are six kinds of defects in our PCB images: missing-hole, mouse-bite, open circuit, short, spur and spurious-copper, as Fig. 7 shows. Defects are synthesised according to common defects standards in the factory and meet the condition of 'tiny'. In training set, there are 5379 images including 902 missing-holes, 978 mouse-bite, 993 open circuits, 765 short, 887 spur and 854 spurious-copper. The testing set has 1270 images including 244 missing-holes, 230 mouse-bite, 240 open circuits, 168 short, 196 spur and 192 spurious-copper. It is worth noting that the images of training set and testing set are taken from different PCB to ensure the effectiveness of evaluation.

Table 2 Detection results on PCB defect dataset

Method	mAP, %	Missing-hole	Mouse-bite	Open circuit	Short	Spur	Spurious copper	FPS
SSD [12]	0.9277	0.9017	0.9933	0.9050	0.8908	0.9033	0.9720	125
SSTD*	0.9489	0.8995	**0.9987**	0.9969	0.8861	0.9921	0.9168	110
SSTD$^+$	0.9563	0.8996	0.9973	0.8845	**0.9869**	0.9849	0.9846	84
SSTD	**0.9785**	**0.9907**	0.9982	**0.9975**	0.9028	**0.9968**	**0.9847**	67

SSTD* does not fuse features into conv_7 and excludes attention mechanism. SSTD$^+$ excludes attention mechanism.

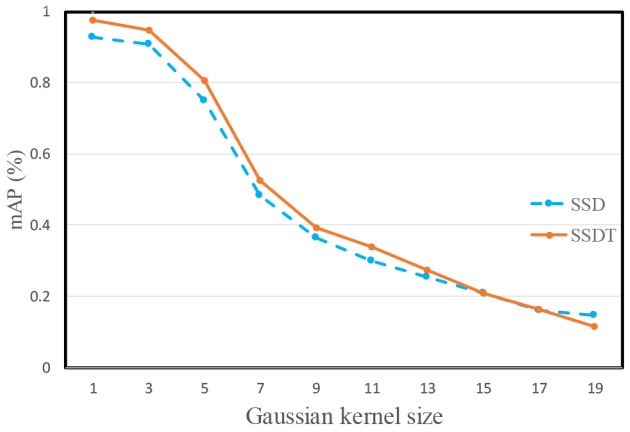

Fig. 8 *Blur test result of SSD and SSDT*

3.2 PCB defect detection

Stochastic gradient descent is used with a momentum of 0.9 and a weight decay of 0.0005 when training. We first train the model with 2×10^{-4} learning rate for 20K iterations, and then continue training for 10K iterations with 2×10^{-5} and 10K iterations with 2×10^{-6}. The batch size is 16 and training and testing are on the NVIDIA 1080Ti GPUs. As Table 2 shows, our SSDT performs better than SSD on the PCB tiny defect detection by nearly 4.1% mAP although the speed is the half of SSD. The reason is that the two extra feature fusion pathways enrich the semantics in shallow layers and the gate mechanism enhances the representation ability of features. However, such processing speed can still meet the requirements of the factory. The effectiveness of attention mechanism and feature fusion module in SSDT is also evaluated on PCB defect dataset. Table 2 shows that each module increases the performance of our method for PCB tiny defect detection.

3.3 Generalisation capacity

In actual industrial production, PCBs may have offsets and PCB images captured by cameras may blur. For these two problems, two experiments are done to validate the robustness of our detector.

Rotation consistence: In the actual production line, the camera may have a slight rotation, causing the rotation of the captured PCB images. If using traditional image processing method to align the image, a certain amount of time will be wasted. So the detector is supposed to have the characteristic of rotation consistence.

In the test set, we randomly select sixty images and rotate them at different angles (randomly rotate $-10^{\circ} - 10°$). Then these PCB images are tested using the SSDT512 trained with PCB defect dataset. The result is 0.8854 mAP which can be acceptable.

Low-resolution test: Sometimes, the pixel of camera may be not very high, so the defects in the image are not clear. Hence, the detector needs to be robust to slight blur. To study the effects of resolution on defect detection, the raw 256×256 cutouts are convolved with different size Gaussian kernel. The larger the size of the Gaussian kernel, the more blurred the image will be.

The SSDT512 is tested with different size of the Gaussian kernel blurring the images in the test set. Fig. 8 demonstrates that both SSD and SSDT are poor to inference in high degrees of blur. However, SSDT still performs better than SSD in almost all levels and the performance of it on low-level blur is acceptable.

3.4 Results on PASCAL VOC

We use VOC2007 `trainval` and VOC2012 `trainval` to train SSDT following SSD [12]. The SSDT300 is trained on four Nvidia 1080Ti GPUs with batch size 32 for 150k iterations. The initial learning rate is 4×10^{-3} and then divided by 10 at 80k, 100k, 120k iterations. For VOC2007, Table 3 shows that our low resolution SSDT300 model is already more accurate than some classic two-stage detection methods, surpassing Fast R-CNN [10] by 9.1%, Faster-RCNN [11] with VGG16 backbone by 5.9% and Faster-RCNN [19] with ResNet-101 backbone by 2.7%. The accuracy of SSDT300 is higher than some one-stage detectors such as SSD300 [12], SSD321 [12] and DSSD321 [16].

Table 3 demonstrates that although DSSD513 with backbone ResNet101 has higher average precision than SSDT512, SSDT512 is more accurate on small object detection as well as SSDT300, such as bottle and potted plant. It means that the several modules that we proposed are useful for small object detection.

3.5 Ablation experiments

3.5.1 Attention mechanism: The semantic ascending module does not learn the relationship of features across channels. Actually, passing deep features to shallow features will enrich the semantic information, but it will also bring some useless information. If the fusion features are directly used for classification and regression, it may reduce the detection performance. Our attention mechanism suppresses unwanted channels by applying weights to each channel of the feature map. Its benefit is noted by comparing the SSDT w or w/o it (mAP increases 0.8%) in Table 4.

3.5.2 Shuffle module: Table 4 summarises the performance of shuffle module. We can see that the aliasing effect caused by element-wise summation has a great influence on object detection. The shuffle module we add can eliminate this effect to a certain extent. Also as mentioned in Section 3.2, the shuffle module would not increase computational complexity a lot.

3.5.3 Feature fusion: In Table 4, using element-wise summation to fuse features can achieve 79.1% mAP while concatenation can only achieve 77.3% mAP. The result shows that element-wise summation can better fuse high-level semantics into low-level features. On the other hand, if the concatenation operation is adopted, the number of channels of feature map after fusion will increase, which will cause the addition of computation complexity.

3.5.4 Improved non-maximum suppression: Due to the overlap effect for testing the whole PCB image, the improved non-maximum suppression method is proposed. The effectiveness of our method is validated in 60 PCB images with 293 defects in them. In Fig. 9, the different λ values are tested and we can see that when $\lambda \geq 1$ the performance achieves the best. Two main factors are responsible for this: (i) sliding windows may truncate defects for testing the whole PCB image. (ii) our improved non-maximum suppression method considers the area and the confidence score of the candidate bounding box simultaneously.

4 Conclusions

In this paper, we present a novel single-shot detector for PCB tiny defect detection, called SSDT. This framework contains a semantic ascending module to convert the semantics of high-level features to

Table 3 PASCAL VOC 2007 `test` detection results. The first section contains some representative baselines [10, 11, 19–21], the second section contains low resolution SSD [12], DSSD [16] and SSDT, and the last section contains high resolution SSD, DSSD and SSDT. The mAP of all categories is shown below

Method[a]	backbone	mAP	Aero	Bike	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	mbike	Person	Plant	Sheep	Sofa	Train	TV
fast [10]	VGG16	70.0	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	74.8	80.4	70.4
faster RCNN [11]	VGG16	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
ION [20]	VGG16	75.6	79.2	83.1	77.6	65.6	54.9	85.4	85.1	87.0	54.4	80.6	73.8	85.3	82.2	82.2	74.4	47.1	75.8	72.7	84.2	80.4
faster RCNN [19]	ResNet101	76.4	79.8	80.7	76.2	68.3	55.9	85.1	85.3	**89.8**	56.7	87.8	69.4	88.3	88.9	80.9	78.4	41.7	78.6	79.8	85.3	72.0
RON384++ [21]	VGG16	77.6	86.0	82.5	76.9	69.1	59.2	86.2	85.5	87.2	59.9	81.4	73.3	85.9	86.8	82.2	79.6	52.4	78.2	76.0	86.2	78.0
SSD300 [12]	VGG16	77.5	79.5	83.9	76.0	69.6	50.5	87.0	85.7	88.1	60.3	81.5	77.0	86.1	87.5	84.0	79.4	52.3	77.9	79.5	87.6	76.8
SSD321 [12]	ResNet101	77.1	76.3	84.6	79.3	64.6	47.2	85.4	84.0	88.8	60.1	82.6	76.9	86.7	87.2	85.4	79.1	50.8	77.2	**82.6**	87.3	76.6
DSSD321 [16]	ResNet101	78.6	81.9	84.9	80.5	68.4	53.9	85.6	86.2	88.9	61.1	83.5	78.7	86.7	88.7	86.7	79.7	51.7	78.0	80.9	87.2	79.4
SSDT300	VGG16	79.1	82.7	86.7	77.6	**75.1**	55.6	87.5	87.1	87.6	62.2	84.8	77.7	85.4	88.8	86.8	80.1	51.6	79.2	78.6	87.4	78.8
SSD512 [12]	VGG16	79.5	84.8	85.1	81.5	73.0	57.8	87.8	88.3	87.4	63.5	85.4	73.2	86.2	86.7	83.9	82.5	55.6	81.7	79.0	86.6	80.0
SSD513 [12]	ResNet101	80.6	84.3	87.6	**82.6**	71.6	59.0	88.2	88.1	89.3	64.4	85.6	76.2	**88.5**	88.9	**87.5**	83.0	53.6	83.9	82.2	87.2	81.3
DSSD513 [16]	ResNet101	**81.5**	86.6	86.2	82.6	74.9	62.5	**89.0**	88.7	88.8	65.2	87.0	**78.7**	88.2	**89.0**	87.5	83.7	51.1	**86.3**	81.6	85.7	**83.7**
SSDT512	VGG16	81.3	**87.6**	**87.8**	82.5	74.8	**64.5**	88.9	**88.8**	86.0	**65.2**	**89.0**	76.0	86.3	88.6	87.3	**83.9**	**57.8**	84.5	79.4	**87.7**	80.3

[a]All models are trained on VOC2007 `trainval` and VOC2012 `trainval` and tested on VOC2007 `test`.

Table 4 Ablation experiments of SSDT300 on VOC2007

Shuttle module	Attention mechanism	Feature fusion	mAP, %
×	×	ele-sum	76.5
×	✓	ele-sum	77.3
✓	×	ele-sum	78.8
✓	✓	ele-sum	**79.1**
✓	✓	concat	78.6

the low-level features to enhance the discrimination ability of shallow prediction layers which mainly detect small objects. Moreover, a lightweight shuffle module is proposed to eliminate the aliasing effect caused by feature fusion and an attention mechanism is utilised to learn the relationship of fusion features between channels. An improved non-maximum suppression is proposed for testing the whole PCB images, which is shown to be available. Our SSDT is proven to be effective not only on PCB defect dataset but also on PASCAL VOC dataset. Our framework also shows robustness to rotation and blur of PCBs in the industrial production process.

5 Acknowledgments

This work is supported by National key R&D program of China (2018YFB1308600, 2018YFB1308602), Specialized Research Fund for Strategic and Prospective Industrial Development of Shenzhen City (No. ZLZBCXLJZI20160729020003).

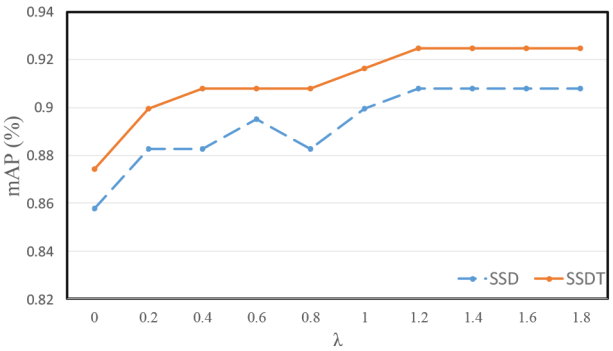

Fig. 9 *Performance of different λ in improved non-maximum suppression*

6 References

[1] Lu, Z., He, Q., Xiang, X., *et al.*: 'Defect detection of PCB based on Bayes feature fusion', *J. Eng.*, 2018, **16**, pp. 1741–1745

[2] Ozturk, S., Akdemir, B.: 'Detection of PCB soldering defects using template based image processing method', *Int. J. Intell. Syst. Appl. Eng.*, 2017, **5**, pp. 269–273

[3] Zhang, L., Jin, Y., Yang, X., *et al.*: 'Convolutional neural network-based multi-label classification of PCB defects', *J. Eng.*, 2018, **16**, pp. 1612–1616

[4] Krizhevsky, A., Sutskever, I., Hinton, G.E.: 'ImageNet classification with deep convolutional neural networks'. Advances in Neural Information Processing Systems, Lake Tahoe, 2012, pp. 1097–1105

[5] Simonyan, K., Zisserman, A.: 'Very deep convolutional networks for large-scale image recognition', arXiv preprint arXiv: 1409.1556, 2014

[6] Huang, G., Liu, Z., Van Der Maaten, L., *et al.*: 'Densely connected convolutional networks'. Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 2017, pp. 4700–4708

[7] Yang, Y., Zhong, Z., Shen, T., *et al.*: 'Convolutional neural networks with alternately updated clique'. Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition, Venice, Italy, 2018, pp. 2413–2422

[8] Deng, J., Dong, W., Socher, R., *et al.*: 'ImageNet: a large-scale hierarchical image database'. Computer Vision and Pattern Recognition, Miami, FL, USA, 2009, pp. 248–255

[9] Girshick, R., Donahue, J., Darrell, T., *et al.*: 'Rich feature hierarchies for accurate object detection and semantic segmentation'. Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition, Columbus, Ohio, 2014, pp. 580–587

[10] Girshick, R.: 'Fast R-CNN'. Proc. of the IEEE Int. Conf. on Computer Vision, Santiago, Chile, 2015, pp. 1440–1448

[11] Ren, S., He, K., Girshick, R., *et al.*: 'Faster R-CNN: towards real-time object detection with region proposal networks'. Advances in Neural Information Processing Systems, Montreal, Canada, 2015, pp. 91–99
[12] Liu, W., Anguelov, D., Erhan, D., *et al.*: 'SSD: single shot multibox detector'. European Conf. on Computer Vision, Amsterdam, Netherlands, 2016, pp. 21–37
[13] Redmon, J., Divvala, S., Girshick, R., *et al.*: 'You only look once: unified, real-time object detection'. Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition, Las Vegas, Nevada, 2016, pp. 779–788
[14] Dai, J., Li, Y., He, K., *et al.*: 'R-FCN: object detection via region-based fully convolutional networks'. Advances in Neural Information Processing Systems, Barcelona, Spain, 2016, pp. 379–387
[15] Lin, T.-Y., Dollár, P., Girshick, R., *et al.*: 'Feature pyramid networks for object detection'. Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, p. 4
[16] Fu, C.Y., Liu, W., Ranga, A., *et al.*: 'DSSD: deconvolutional single shot detector', arXiv preprint arXiv:1701.06659, 2017
[17] Hu, J., Shen, L., Sun, G.: 'Squeeze-and-excitation networks'. Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition, Salt Lake City, Utah, 2018, pp. 7132–7141
[18] He, K., Zhang, X., Ren, S., *et al.*: 'Deep residual learning for image recognition'. Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition, Las Vegas, Nevada, 2016, pp. 770–778
[19] Bell, S., Lawrence Zitnick, C., Bala, K., *et al.*: 'Inside-outside net: detecting objects in context with skip pooling and recurrent neural networks'. Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition, Las Vegas, Nevada, 2016, pp. 2874–2883
[20] Everingham, M., Van Gool, L., Williams, C.K.I., *et al.*: 'The PASCAL visual object classes (VOC) challenge', *Int. J. Comput. Vis.*, 2010, **88**, pp. 303–338
[21] Kong, T., Sun, F., Yao, A., *et al.*: 'Ron: reverse connection with objectness prior networks for object detection'. IEEE Conf. on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 2017, p. 2