ISBN: 978-981-18-3959-7
2022 the 12th International Workshop on Computer Science and Engineering (WCSE 2022)
doi: 10.18178/wcse.2022.06.033

A Fast Workpiece Surface Defect Detection Approach Based on Multiple Hierarchical Features

Yuxuan Han [1], Jun Guo [2] and Youguang Chen [1] +

[1] School of Data Science Engineering, East China Normal University, Shanghai, China
[2] Department of Vision Algorithm, Changzhou Micro-Intelligence Co. Ltd., Changzhou, China

Abstract. Workpiece surface defect detection is of great importance to the product quality. In this paper, a lightweight convolutional neural network is improved. Based on the multiple hierarchical features extracted by the network, the images are divided into girds with different sizes, which are responsible for the detection of different defects. Furthermore, a new post-processing method is introduced to merge these grids. The experimental results show that our approach achieves good detection results and has a great advantage in inference speed and computing power demand.

Keywords: defect detection, CNN, anchor selection, workpiece surface defects

1. Introduction

Defect detection is crucial to the improvement of qualified workpiece rate. In the past, defect detection was mostly done manually, which was inefficient and expensive. With the development of computer vision technology, automatic defect detection using machines has gradually replaced manual detection. In the beginning, traditional image processing technology dominated the field of defect detection [1-3]. With the development of deep learning, convolutional neural network has become the tool for image feature extraction. [4] combined convolutional neural networks with the sliding window to detect defects and achieved better results than traditional methods. [5-6] used big two-stage object detection model for defect detection.

Although the deep learning methods mentioned above have been applied to different defect detection problems, they ignored the requirement of inference speed and the limitation of computing power in real industrial environment. In the workpiece surface defect detection problem, the goal is to judge whether the workpiece is defective and get the approximate position of defects. Our topic not only achieves above goals, but also makes a good balance between speed and accuracy. First, the lightweight network mobilenetv3 [7] is improved to get more multiple hierarchical features. The object detection frameworks like [8-9] use the anchor to judge whether there is an object in a specific area. We introduce a new method for anchor selection which is simple and effective. In the process of feature extraction by convolutional neural networks, every point in the feature maps corresponds to specific receptive field on the image. We use these square receptive fields as anchors and then every point in the feature maps is responsible for the detection of its related anchor. For each anchor, if its intersection area with a defect is over the threshold, it will be labelled as foreground anchor; otherwise, it will be labelled as background anchor. At last, we design a new post-processing method. By merging foreground anchors from the same and different feature maps, we can get the more accurate location of defects. Experimental results show that our model performs well in the detection effect and has a great advantage in inference speed with low demand for computing power.

+ Corresponding author. Tel.: + 138 1627 7246.
E-mail address: ygchen@cc.ecnu.edu.cn

2. Method

2.1. Anchor Selection

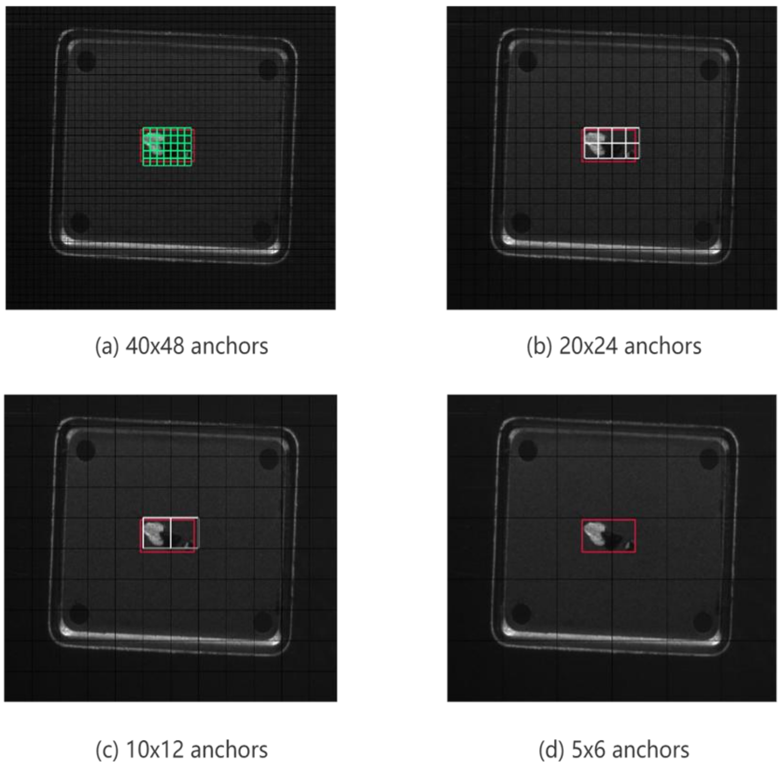

Fig. 1: (a), (b), (c) and (d) correspond to P4, P5, P6, P7 features respectively. The input image is divided into anchors according to the size of feature map. The black anchors are labelled as background anchors. The white anchors are labelled as foreground anchors. The green anchors need to be ignored in training. The red rectangle is the ground truth bounding box of the defect.

In many object detection models, the design of anchors is complex. Faster R-CNN uses 9 kinds of anchors with different aspect ratios and sizes [9]. In yolov3, K-means is used to cluster ground truth bounding boxes of the training data to determine the design of anchors [8]. These methods cost extra time for manual design and data processing.

In the process of image feature extraction, the convolution operation associates the point in the feature map with a rectangular region on the input image. The rectangular region is named as receptive field. Inspired by this correlation, we divide the image into grids according to the sizes of feature maps. Each point in the feature map will be assigned to a corresponding grid which is its receptive field. We use these grids as anchors. Our network extracts 4 different levels of features. Their sizes are $40 \times 48 \times c1$, $20 \times 24 \times c2$, $10 \times 12 \times c3$, and $5 \times 6 \times c4$. We denote them as P4, P5, P6 and P7 features. The size of the input image is 640×768 .The length and width of $P(k)$ feature are $\frac{1}{2^k}$ of those of the input image. For example, for P4 feature, the input image is divided into 40×48 anchors. Each point in the feature map is responsible for the detection of defects in its related anchor. The points in the high-level small-size feature map correspond to the large anchors, which are responsible for detecting large defects. Anchors with different sizes are used to detect defects with different sizes. If the intersection area of an anchor and the bounding box of a defect exceeds $H \times \text{anchor\_area}$, then the anchor is labelled as the foreground anchor. We set H to 0.5 here.

In Fig. 1(c), 2 white anchors are labelled as foreground anchors. In Fig. 1(a), green anchors are labelled as ignored anchors although their intersection area with the defect is over the threshold. In order to avoid using small anchors to detect large defects, if the intersection area exceeds the threshold and the length of the long side of the defect exceeds F times of the side length of anchors, these anchors will not be used in training and will be labelled as ignored anchors. We set F to 5 here. In Fig. 1(d), the anchors with size of 128×128 are too large for this defect. No anchor has intersection area over the threshold. All the anchors are labelled as background anchors. Through the above method, anchors cover the whole image without overlaps. Anchors with different sizes are responsible for detecting defects with different sizes.

2.2. Network Architecture

Fig. 2 shows the architecture of our network. The size of input is $n \times 640 \times 768 \times 3$ and n is the batch size. The backbone of our network contains 15 bottleneck blocks. These 15 bottleneck blocks have the same structure design as mobilenetv3 [7]. We denote the output feature maps of bottleneck block 12 and bottleneck block 15 as C4 feature and C5 feature. They have strides of 16 and 32 pixels with respect to the input image. Inspired by FPN [10], we merge the spatially coarser and semantically stronger C5 feature and low-level C4 feature by upsampling and element-wise addition to get P4 feature. For the detection of large defects, we use 3×3 conv with $\text{stride} = 2$ to get P6 feature and P7 feature. The heights and widths of P4, P5, P6, P7 features are 1/16,1/32,1/64,1/128 of those of the input image. We set the number of channels of all P(k) features to 256. Every point in these features is a vector. The length of the vector equals to the number of channels. The vector is used in the classification of anchors.

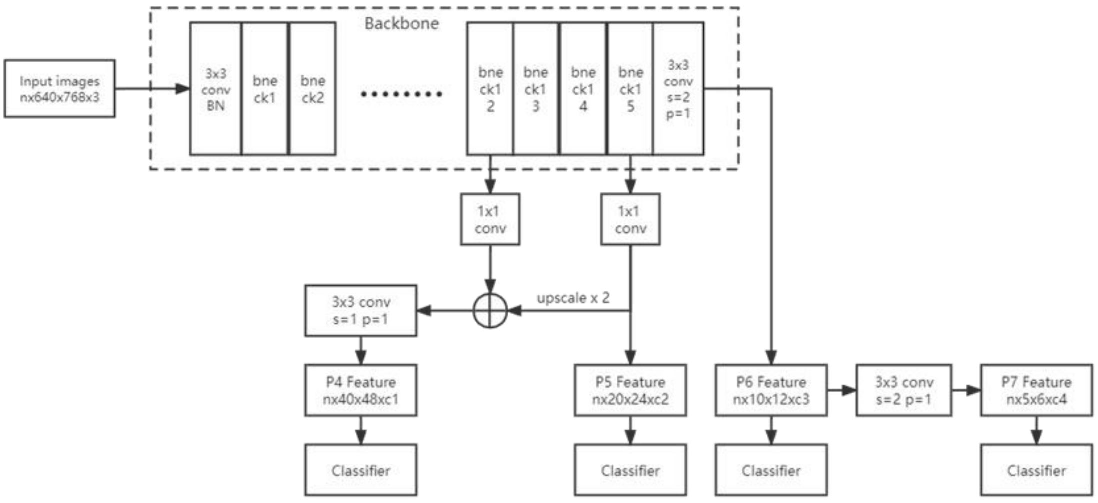

Fig. 2 : The network architecture

2.3. Defect Detection

As introduced in 2.1, all anchors are labelled as foreground anchors, background anchors and ignored anchors. The classifiers contain the fully convolutional layer and sigmoid layer. After the P4, P5, P6 and P7 features are fed into the classifiers, the score p of each anchor is obtained. The score p represents the predicted possibility of foreground anchor. In training, the loss function of each branch is shown in (1).

$$loss^{(P(k))} = -\frac{1}{N^{(P(k))}} \sum_{n=1}^{N^{(P(k))}} [p_n' \log(p_n) + (1-p_n') \log(1-p_n)] \tag{1}$$

In (1), P(k) is defined in 2.1. $N^{(P(k))}$ is the total number of foreground and background anchors in P(k) branch. The ignored anchors are not used in training. p_n is the predicted possibility that the n-th anchor is the foreground anchor. $p_n{}'$ is the true label of the n-th anchor. $p_n{}'$ of foreground anchor is 1 and $p_n{}'$ of background anchor is 0. Total loss of the model is shown in (2).

$$loss_{\text{total}} = loss^{(P4)} + loss^{(P5)} + loss^{(P6)} + loss^{(P7)} \tag{2}$$

2.4. Post-processing

In testing, the network outputs the classification results of all anchors from 4 branches. For the large defect, the small foreground anchors cannot well reflect the position of the defect. Therefore, we design a method to merge foreground anchors.

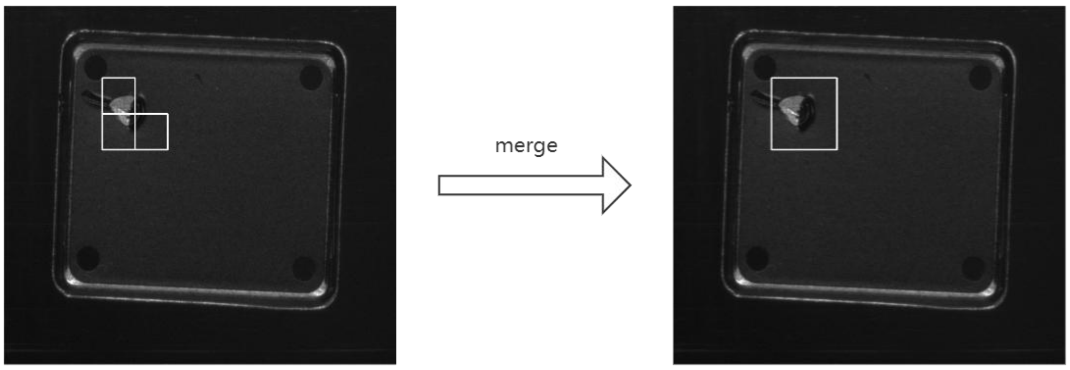

Fig. 3: Merge foreground anchors from P6 branch

First, merge foreground anchors generated by the same branch. For foreground anchors from the same branch, if they are no more than T anchors apart, find the coordinates of the top left point and the bottom right point. Merge these anchors into a new bounding box with the same top left point and bottom right point. For small anchors from P4 and P5 branches, we set T to 2. For anchors from P6 and P7 branches, we set T to 1. In Fig. 3, 3 foreground anchors from P6 branch are merged. In this way, anchors with the same size are merged to get the new bounding boxes.

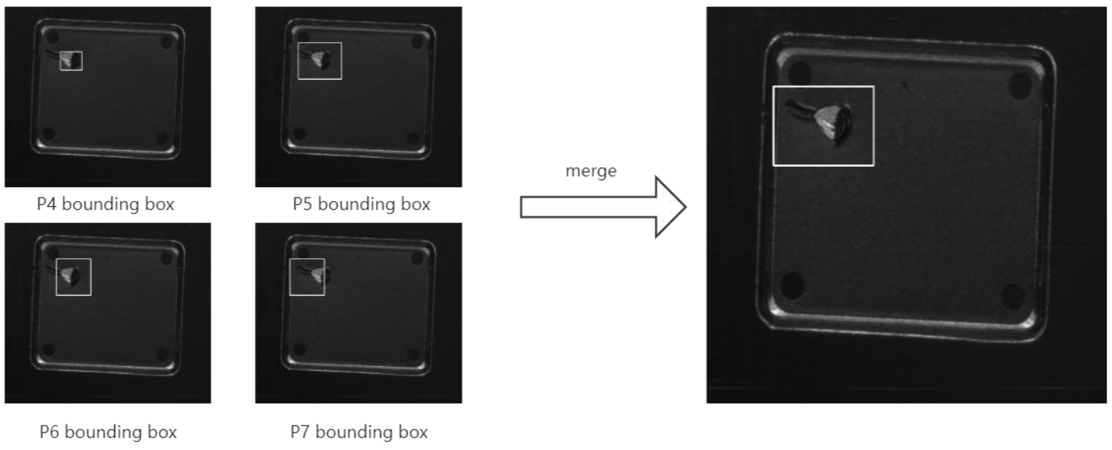

Fig. 4: Merge bounding boxes from 4 branches

Then, merge bounding boxes from different branches. For bounding boxes from different branches, if IoU is over 0.3, they are merged into a new bounding box. In Fig. 4, 4 bounding boxes from all the branches can be merged into a new bounding box, which is the final location of the defect. The merging is implemented in the same way as the first step. The final bounding box better locates the defect. With above approach, small foreground anchors are merged into large bounding boxes to better represent the location of defects.

3. Experiments

3.1. Dataset

To evaluate our model, we collect images of workpiece surface defects to make a dataset, which is named as workpiece surface defects dataset (WSDD). All images in WSDD are taken from real factory assembly lines using high-definition cameras. WSDD contains 1929 images. The size of the images is 640×768. All the bounding boxes of defects are labelled manually. We randomly split the dataset into the training set and the test set in a ratio of 7:3. The examples of the dataset are shown in Fig. 5.

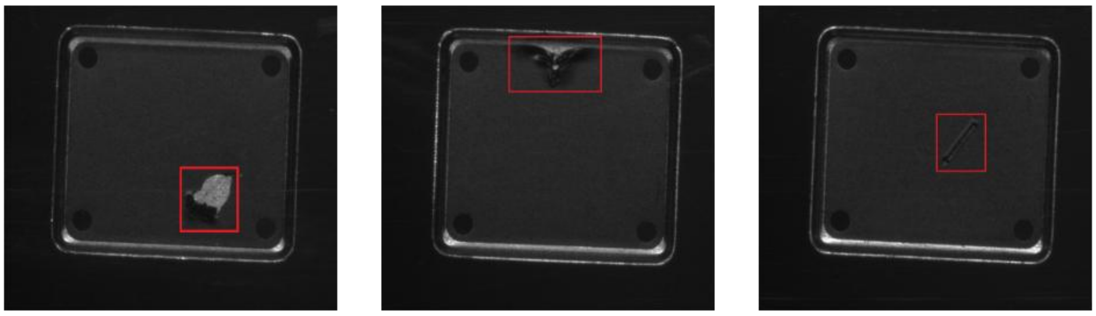

Fig. 5: The examples of workpiece surface defects

3.2. Metrics

In workpiece surface defect detection, we need to judge whether a workpiece is defective and get the approximate location of the defect. A workpiece is defective if the detection result shows that it has at least one defect. If too many workpieces are defective, a warning will be issued and these workpieces need to be checked manually. Therefore, if the model detects many qualified workpieces as defective, it will cost a lot of extra labour. The workpieces that the model identifies as defective but qualified are defined as false defective workpieces. We use workpiece error detection rate (WEDR) to evaluate the performance of our model in the above problem. $C_{\text{false defective}}$ is the number of false defective workpieces. $C_{\text{qualified}}$ is the total number of qualified workpieces. WEDR is a picture-level metric that indicates the proportion of false defective workpieces to total qualified workpieces. Lower WEDR means better detection effect.

$$WEDR = \frac{C_{\text{false\_defective}}}{C_{\text{qualified}}} \tag{3}$$

For defective workpieces, we use the defect detection rate (DDR) to evaluate the performance of the model. If the IoU of the bounding box generated by the model and the ground truth bounding box is greater than 0.3, the defect is successfully detected. C_{detected} is the number of successfully detected defects. C_{total} is the total number of defects. DDR is a defect-level metric that indicates the proportion of successfully detected defects to all defects.

$$DDR = \frac{C_{\text{detected}}}{C_{\text{total}}} \tag{4}$$

3.3. Experimental Results

In order to evaluate the performance of the model, several object detection methods are selected for comparison. We test out method on a Tesla T4 GPU. Table 1 shows the detection performance comparison between our model and other object detection methods. The WEDR of our model is much lower than that of other methods. The DDR of our model is higher than most other methods and is close to the two-stage models with high computing power demand such as Faster R-CNN.

Table 2 shows the inference speed and the complexity of different models. Our model requires far less computing power than other methods. At the same time, our method has a great advantage in inference speed and can meet the requirement of real-time inference better.

Table 1: The WEDR and DDR of different methods

Method	Backbone	WEDR	DDR
Faster R-CNN [9]	ResNet-50	0.1736	0.86
YOLOv3 [8]	DarkNet-53	0.1149	0.79
CenterNet [11]	ResNet-18	0.2469	0.745
Ours	Ours	0.0513	0.815

Table 2: Comparison of FPS and other metrics

Method	Backbone	FPS	GFLOPs	Params(M)
Faster R-CNN [9]	ResNet-50	16.94	106.42	41.12
YOLOv3 [8]	DarkNet-53	22.9	93.05	61.52
CenterNet [11]	ResNet-18	53.4	24.49	14.21
Ours	Ours	55.86	1.9	6.64

4. Conclusion

In this paper, we propose a fast workpiece surface defect detection approach. Inspired by correlation between the feature map and the receptive field, we design an anchor selection method based on multiple hierarchical features. Images can be divided into anchors according to sizes of feature maps. We improve a lightweight network to meet the demand for high inference speed and low computing power. Furthermore, a new post-processing method is introduced to help locate the defects. Experimental results show that our approach performs well on WSDD and achieves a balance between speed and accuracy.

5. Acknowledgements

The author would like to thank Professor Jun Guo and Youguang Chen for their guidance and help in providing the GPU used in the experiments.

6. References

[1] Zhang, H., Jin, X., Wu, Q. J., Wang, Y., He, Z., & Yang, Y. (2018). Automatic visual detection system of railway surface defects with curvature filter and improved Gaussian mixture model. *IEEE Transactions on Instrumentation and Measurement*, 67(7), 1593-1608.

[2] Li, D., Liang, L. Q., & Zhang, W. J. (2014). Defect inspection and extraction of the mobile phone cover glass based on the principal components analysis. *The International Journal of Advanced Manufacturing Technology*, 73(9), 1605-1614.

[3] Zhang, H., Guo, Z., Qi, Z., & Wang, J. (2012, August). Research of glass defects detection based on DFT and optimal threshold method. *In 2012 International Conference on Computer Science and Information Processing (CSIP)* (pp. 1044-1047). IEEE.

[4] Wang, T., Chen, Y., Qiao, M., & Snoussi, H. (2018). A fast and robust convolutional neural network-based defect detection model in product quality control. *The International Journal of Advanced Manufacturing Technology*, 94(9), 3465-3471.

[5] Fang, J., Tan, X., & Wang, Y. (2021, January). ACRM: Attention Cascade R-CNN with Mix-NMS for Metallic Surface Defect Detection. *In 2020 25th International Conference on Pattern Recognition (ICPR)* (pp. 423-430). IEEE.

[6] Luo, J., Yang, Z., Li, S., & Wu, Y. (2021). FPCB surface defect detection: A decoupled two-stage object detection framework. *IEEE Transactions on Instrumentation and Measurement*, 70, 1-11.

[7] Howard, A., Sandler, M., Chu, G., Chen, L. C., Chen, B., Tan, M., ... & Adam, H. (2019). Searching for mobilenetv3. *In Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 1314-1324).

[8] Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv*:1804.02767.

[9] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

[10] Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. *In Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2117-2125).

[11] Zhou, X., Wang, D., & Krähenbühl, P. (2019). Objects as points. *arXiv preprint arXiv*:1904.07850.