Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 20(7): 12341–12359.
DOI: 10.3934/mbe.2023549
Received: 21 March 2023
Revised: 27 April 2023
Accepted: 04 May 2023
Published: 22 May 2023

Research article

Bearing surface defect detection based on improved convolutional neural network

Xian Fu[1], Xiao Yang[1,*], Ningning Zhang[1], RuoGu Zhang[1], Zhuzhu Zhang[1], Aoqun Jin[1], Ruiwen Ye[1] and Huiling Zhang[2]

[1] Department of Computer and Information Engineering, Hubei Normal University, Huangshi 435000, China

[2] Huangshi Bangke Technology Co., Ltd., Huangshi 435000, China

* **Correspondence:** Email: 2497865883@qq.com.

Abstract: This paper addresses the issue of artificial visual inspection being overly reliant on subjective experience and the difficulty for the human eye to accurately identify dense and non-significant defects. To solve this problem, we have implemented an automatic object detection algorithm based on an improved version of YOLOv5.First, we use the K-means++ clustering algorithm to automatically calculate the Anchor of the model to reduce the effect of the close location of the initial clustering centers on the clustering of the sample data.Second, we add the Coordinate Attention (CA) attention mechanism to the model to allow the model to better capture and understand important features in the images. Then, we add a new detection layer with a downsampling multiplier of 4 to the Neck network to improve the precision of the model. Finally, we use the lightweight network MobileNetV3 instead of YOLOv5's backbone network to reduce the model detection time overhead.Our model achieves 85.87% mAP, which is 6.44% better than the YOLOv5 network, and the detection time for a single image is only 54ms, which is 50% faster than the YOLOv5 network. After testing, we have proven that our proposed algorithm can quickly and accurately detect the condition of bearing appearance defects, improving detection efficiency and reducing costs.

Keywords: target detection; defect detection; attention mechanism; lightweight network

1. Introduction

Bearings are essential components of freight trains during operation, and their working environment demands them to have three characteristics: high-speed rotation, high pressure, and low fault tolerance. Defects can appear on the bearing surface during production due to improper assembly, poor lubrication, and improper storage. The condition of the bearings determines the safety of the train, so

maintenance personnel must regularly overhaul the train's bearings and carry out corresponding maintenance treatments according to the types of defects. Currently, industrial production primarily relies on manual visual inspections to detect bearing surface defects, which places too much emphasis on the inspector's experience. The variety of defects and their manifestations, along with non-significant defects, increases the difficulty of inspectors' detection, and the inspection workshop's environment is not conducive to extended periods of inspector work.

Several researchers have proposed methods for detecting defects on bearing surfaces. L. Eren and A. Karahoca [1] improved the bearing defect detection procedure by wavelet transform and Radial Basis Function (RBF) neural network. Kankar and Sharma [2] used Artificial Neural Network (ANN) and Support Vector Machine(SVM) to detect defects on the bearing surface from the perspective of bearing vibration signals. Tastimur and Karakose [3] used a deep learning framework in the visual task of bearing defect detection to complete automatic detection of four types of bearing defects. Senanayaka and Khang [4] proposed a fault diagnosis method based on convolutional neural network pattern recognition, which can effectively detect not only single faults but also multiple faults simultaneously. Sobie and Freitas [5] applied proven statistical feature-based methods to convolutional neural networks to improve the accuracy of mechanical fault detection. Sadoughi and Hu [6] used bearings and their physical knowledge of bearings and their fault features as input to a deep neural network to propose a convolutional neural network (PCNN) based on physical characteristics for simultaneous detection of multiple bearings. Kim and Lee [7] applied deep learning models to detect ball bearing faults under complex conditions and obtained very high accuracy. Bapir and Aydin [8] used variational modal decomposition and convolutional neural network to complete the feature extraction and classification of bearing surface defects. Kone and Yatsugi [9] proposed an adaptively tuned convolutional neural network for the detection of multiple scratches defects on bearing surfaces. Chen and Yu [10] improved the Faster R-CNN model for fabric defect detection by embedding Gabor kernels and using a two-stage training method based on Genetic Algorithm and back-propagation. Luo and Yang [11] proposed a decoupled two-stage object detection framework for FPCB surface defect detection that achieved state-of-the-art accuracy. Zhang and Ma [12] proposed and evaluated a sparse regular diagnosis algorithm for feature enhancement in planetary gearbox fault diagnosis.

The detection model needs to implement the classification and localization of the target defects to obtain statistical information about the class and location of the bearing defects. YOLO is one of the most important target detection models, offering advantages in terms of accuracy and speed. In this paper, we propose an improved YOLOv5 model for bearing surface target defect detection. Due to the curvature and texture of the bearing surface, detection presents certain challenges. In addition, we found that the target defect area only accounts for 0.03% of the bearing surface area, yet our model still achieved an mAP of 85.87%. To support our research, we formed a private dataset called "SKF-KS2022". In practical applications, our model performs well. The detection time for a single image is only 54 ms, which can meet the requirements of real-time detection in industrial applications.

2. Dataset

The dataset is composed of two parts. One part is a unified image collection of bearing defects that was created using industrial cameras with the help of engineers from the factory inspection center, and is from freight train service station. The other part of the data is publicly available from the "Severstal:

Steel Defect Detection" competition on the Kaggle website, and is the steel surface defects dataset. In total, there are 1406 images with a resolution of 2048 × 2048. Due to the limited data collected, only 50 images of the remaining 17 defect types were accumulated. Therefore, the main types of defects studied were identified as scratch, corrosion, peeling, and rolling skin. Figure 1 shows the four types of targets.

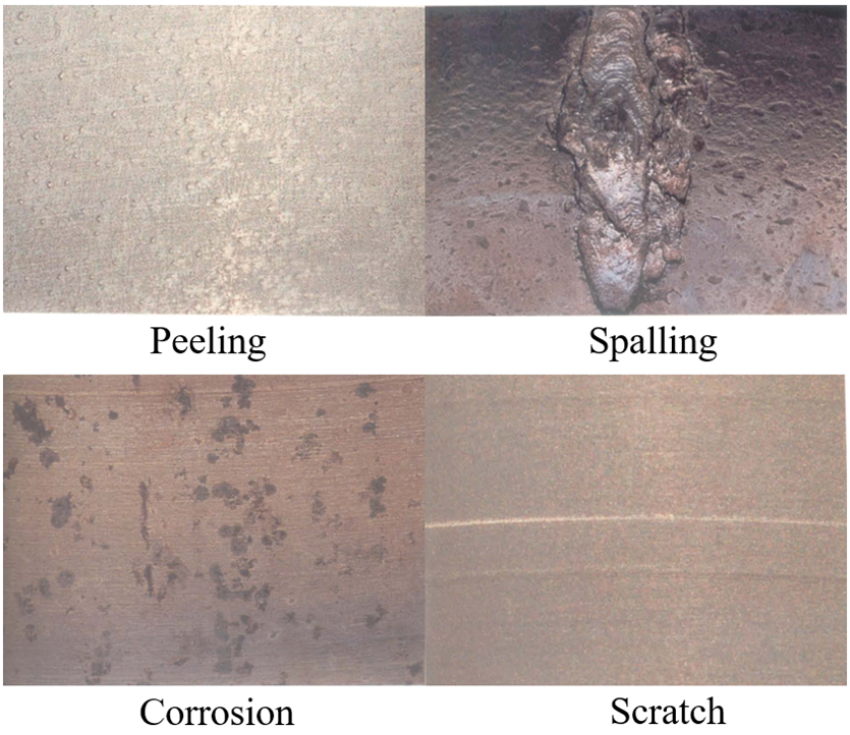


Figure 1. 4 types of defective targets.

3. Method

3.1. YOLOv5 model

The YOLO series [13] is a regression-based target detection algorithm that creatively combines the Region of Interest (RoI) module and detection phases into one to improve the detection speed. YOLOv5 model mainly consists of Backbone, Neck and Head. The structure of YOLOv5 is shown in Figure 2.

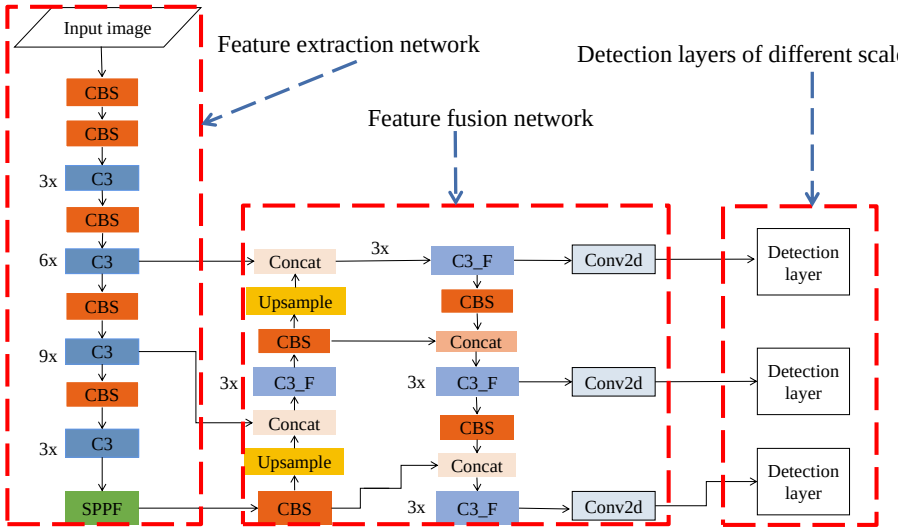


Figure 2. The structure of YOLOv5.

3.2. Prior box

The prior box is a box of different sizes and aspect ratios preset on the image in advance to help the model learn the location and size of the target more easily, and the reasonable setting of the Anchor greatly affects the performance of the final model. The K-means++ algorithm [14] uses an "additive" strategy to select the initial clustering centers, maximizing the distance between them. This strategy ensures that the model achieves the highest prior frame and improves detection accuracy.

The steps of the K-means++ algorithm are as follows [15]:

Algorithm 1: K-means++ clustering algorithm

Input: Dataset X=$\{x_1, x_2, ..., x_n\}$, n is the number of data

Output: Cluster center points $\{c_1, c_2, ..., c_k\}$, k is the number of center points

Algorithm steps:

1) Randomly select one point from the dataset as the initial clustering center point c_1;

2) Calculate the minimum distance D(x) between each sample and the currently existing cluster center;

3)Take one new center c_i, choosing $x \in X$ with maximum probability P(x) = $\frac{D(x)^2}{\sum_{x \in X} D(x)^2}$;

4) Repeat step 2). Until k cluster centers are selected;

5) Clustering is done according to the classical K-means algorithm, until convergence.

3.3. Adding the attention mechanism module

Since the same type of defect manifests itself in various ways and the data contains both densely detected targets and non-significantly detected targets. For this reason, we introduce the Coordinate Attention(CA) [16]. It allows the network to extract regions of interest, resist the interference of confusing information and focus on the key information of valid targets.

CA is a type of attention mechanism that can be used to enhance the feature representation capabilities of mobile networks. It takes intermediate features as input and outputs enhanced features of the same size. CA focuses on both channel and spatial attention attention attention mechanisms. It first aggregates feature maps along the vertical and horizontal directions, respectively, into two separate feature maps with directionality. This transformation allows the attention module to capture long-term dependencies along one spatial direction and to preserve precise location information along the other spatial direction. The CA structure [16] is shown in Figure 3.

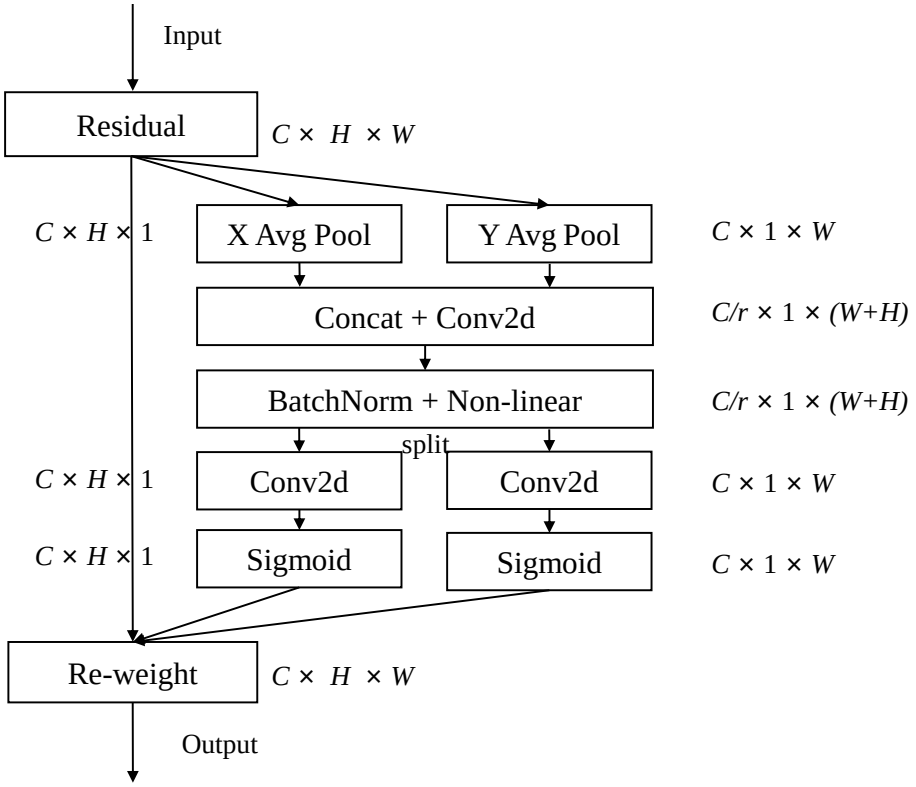


Figure 3. CA structure.

In Figure 4, we can see the model's structure with the newly added CA attention mechanism. This feature is incorporated after the SPPF module to improve the model's semantic perception capabilities.

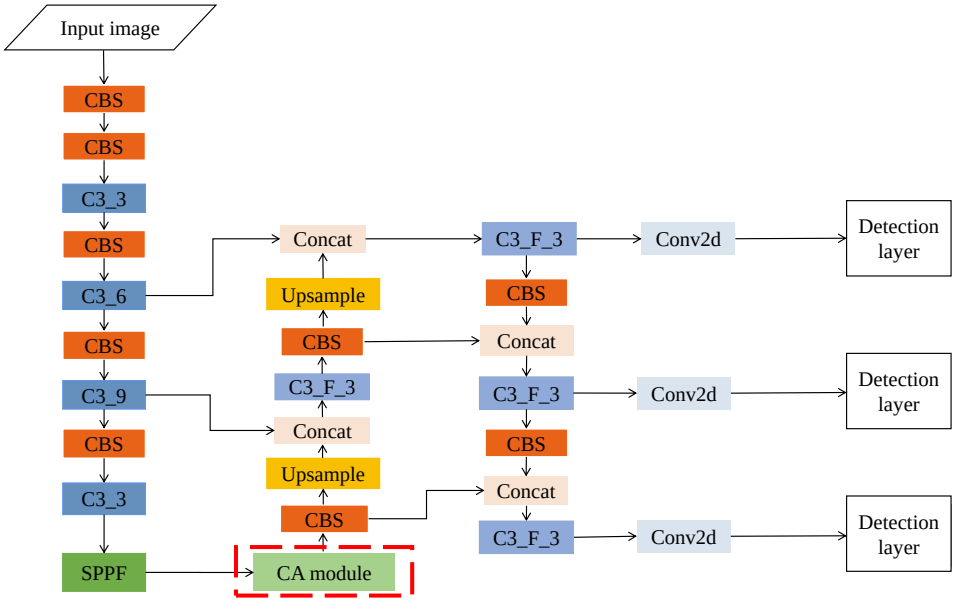


Figure 4. Network structure after adding CA module.

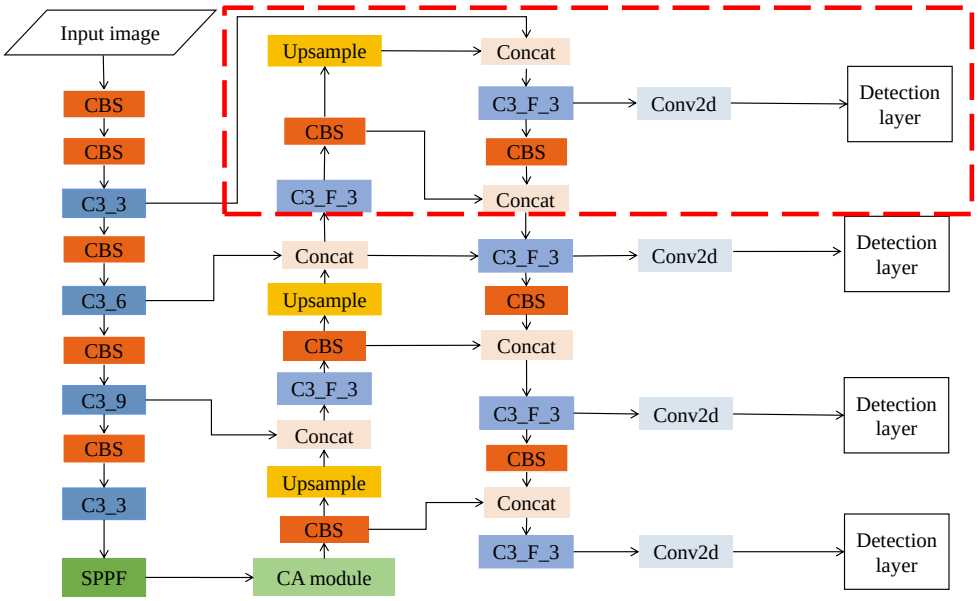


Figure 5. Network structure after adding new detection layer.

3.4. Adding new detection module

To improve the model's detection accuracy and reduce the miss detection rate for small defects, we added a detection head for small target detection based on increased attention. The detection layer

responsible for detecting small targets in the YOLOv5 model is obtained by downsampling the original image by a factor of 8. However, excessive downsampling results in an excessive area of pixel points on the output feature map, making it difficult to retain feature information for smaller target defects. To address this, we construct a detection layer for small targets by splicing the feature map of the shallow network with the feature map of the deep network. The size of the new detection layer will be 4 times the size of the input image for downsampling, and the size of the feature map of the detection layer will be expanded to 512 × 512. See Figure 5 for the improved structure, where the red box indicates the newly added detection module calculation process is illustrated in Figure 6.

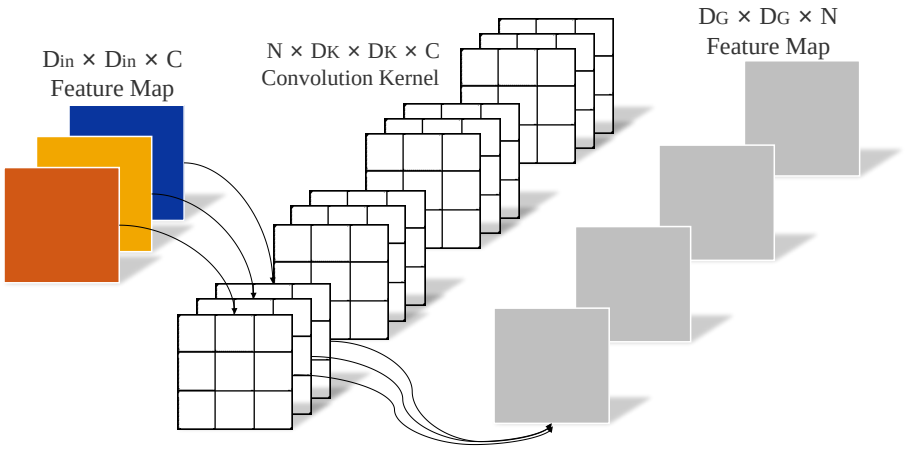


Figure 6. Standard convolution process.

3.5. Lightweight network

In real-world scenarios, the model needs to perform defect inference and analysis within a specific timeframe. YOLOv5's backbone network employs numerous C3 modules and standard convolutional modules for feature extraction. The feature extraction of the standard convolutional layer is completed in the same step as the feature combination, and the standard convolutional.

The number of parameters generated by the standard convolution process is Eq (3.1), where D_K represents the kernel size in the convolution operation, D_G represents the size of the output feature map, C represents the number of channels in the input feature map, and N represents the number of convolution kernels.

$$D_G \times D_G \times D_k \times D_k \times C \times N \tag{3.1}$$

MobileNetV3 [17] uses depth-separable convolution, which combines channel-by-channel convolution for feature extraction with point-by-point 1 × 1 convolution kernels for feature map updimensioning. This is illustrated in Figure 7.

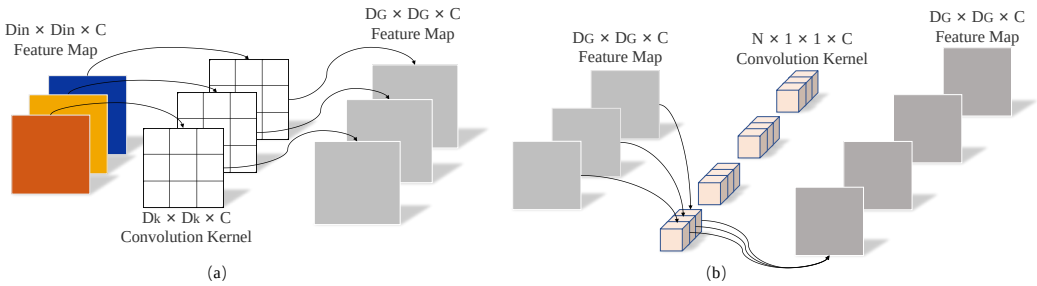


Figure 7. Deeply separable convolution process.

The number of covariates generated by the degree-separable convolution process is Eq (3.2). The comparison of the number of covariates between the two convolution methods is Eq (3.3). If the convolution kernel size of the backbone network is 3×3, the computational effort using the depth-separable convolution is about 1/8 to 1/9 of that of the standard convolution. We use MobileNetV3 lightweight network to replace the backbone network of YOLOv5, which can reduce the computation and the model inference time.

$$D_G \times D_G \times D_k \times D_k \times C + D_G \times D_G \times 1 \times 1 \times N \times C \tag{3.2}$$

$$\frac{D_G \times D_G \times D_k \times D_k \times C + D_G \times D_G \times 1 \times 1 \times N \times C}{D_G \times D_G \times D_k \times D_k \times C \times N} = \frac{1}{N} + \frac{1}{{D_k}^2} \tag{3.3}$$

Figure 8 shows the overall structure of the model after replacing the backbone network.

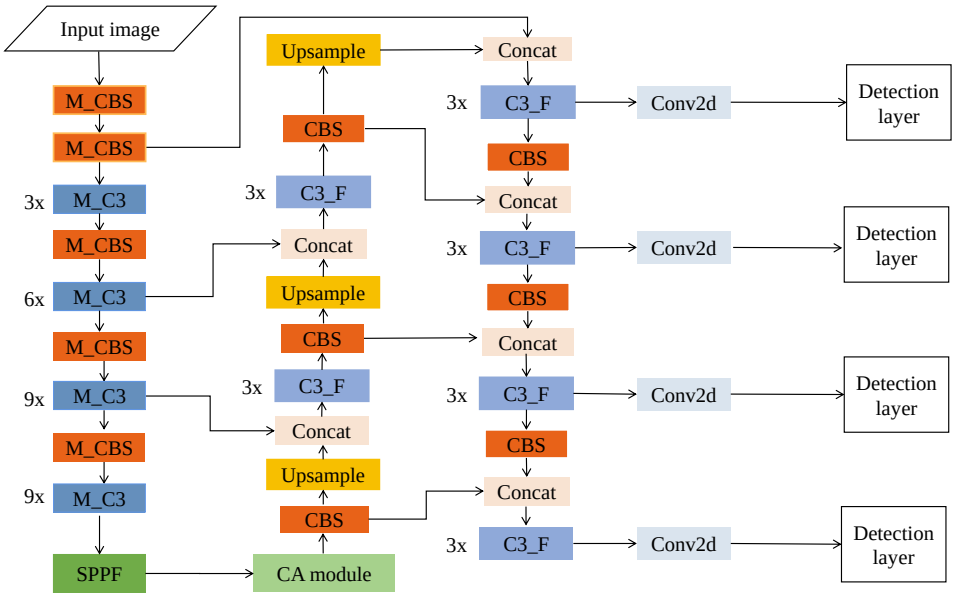


Figure 8. Schematic diagram of our improved model.

4. Experiments

4.1. Data augmentation

Due to the insufficient number of samples, morphological processes such as contrast enhancement, left-right flip, and random image Gaussian blur were performed on the collected data, with the effect shown in Figure 9, to produce similar but different sample data and expand the size of the dataset.

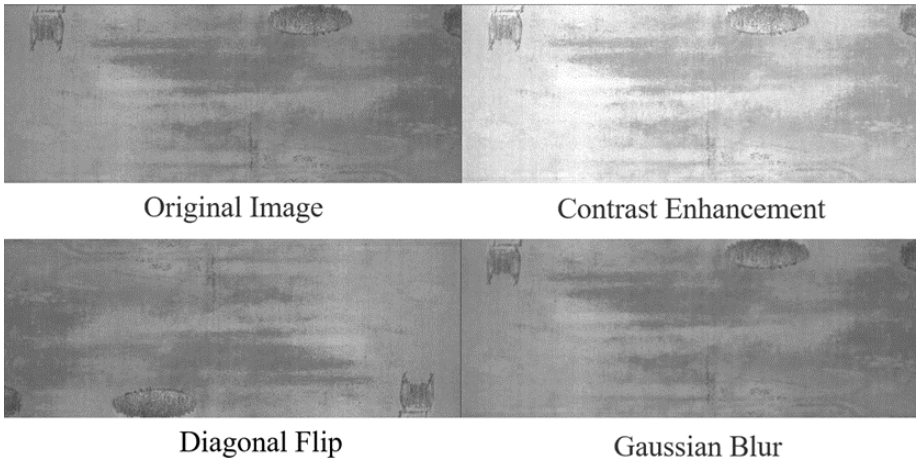


Figure 9. Morphological treatment effect.

The original image contains 350 images of rolled skin defects, 357 images of peeling defects, 348 images of corrosion defects, and 351 images of abrasion defects, for a total of 1406 images. The number of images increased to 4900 after data enhancement, and Figure 10 shows the change of the number of each target.

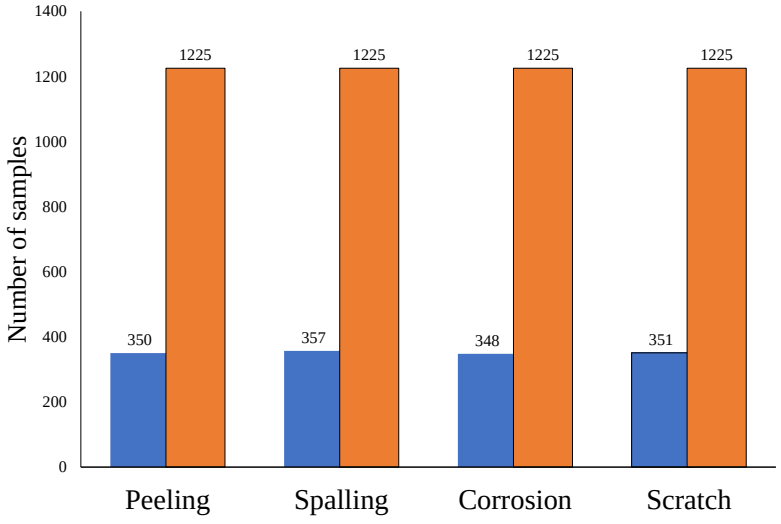


Figure 10. Number of samples in each category after data augmentation.

The training set, validation set, and test set are divided according to the ratio of 6:2:2, as shown in Table 1.

Table 1. Statistics of image dataset.

Train	Validation	Test	Total
2940	980	980	4900

4.2. Experiment configuration and evaluation index

We use Ubuntu 18.04 as the operating system, a GTX TITAN V 8G as the GPU, and PyTorch 1.13.0 as the deep learning framework. During training, the batch size was 4, the iterations epoch was 100, and the learning rate was 10^{-3}.

The evaluation metrics used in the experiments were precision, recall, average precision (AP), mAP, speed and leakage rate. The precision (P) and recall (R) are as follows.

$$P = \frac{TP}{TP + FP}, \tag{4.1}$$

$$R = \frac{TP}{TP + FN}, \tag{4.2}$$

where TP is the number of samples that were positive and also correctly classified as positive. FP is the number of samples that were negative but incorrectly classified as positive. FN is the number of samples that were positive but classified as negative.

After obtaining P and R for each category, the precision-recall (P-R) curve can be displayed. AP is represented by the P-R curve and the area surrounded by coordinates, and mAP is the average of the AP values for all categories. AP and mAP are calculated as follows.

$$AP = \int_0^1 PR \, \mathrm{d}R, \tag{4.3}$$

$$mAP = \frac{1}{N} \sum_{k=1}^{N} AP(k), \tag{4.4}$$

where N represents the total number of categories, and represents the AP of the category K.

The model detection speed is the average time for each image tested by the model. First, all the time consumed by each model to predict all the test set images was counted. Then the average time required for each image prediction was calculated based on the number of test set images. It is important to note that the time we calculated includes the time consumed by the pre-processing, inference, and post-processing processes for each image.

Leakage rate of the model is the ratio of the number of undetected targets to the total number of actual targets.

4.3. Experimental results

4.3.1. YOLOv5

Table 2 shows the results of the network model comparison experiments. Faster R-CNN has the highest accuracy, but it requires the most time overhead. The detection accuracy of YOLOv5 and

YOLOv7 is comparable, but YOLOv7 consumes more memory than YOLOv5 during detection. This drawback is particularly significant in resource-constrained environments. Considering both the accuracy of the model and the detection speed, we selected YOLOv5 as the original model for this study for subsequent experiments.

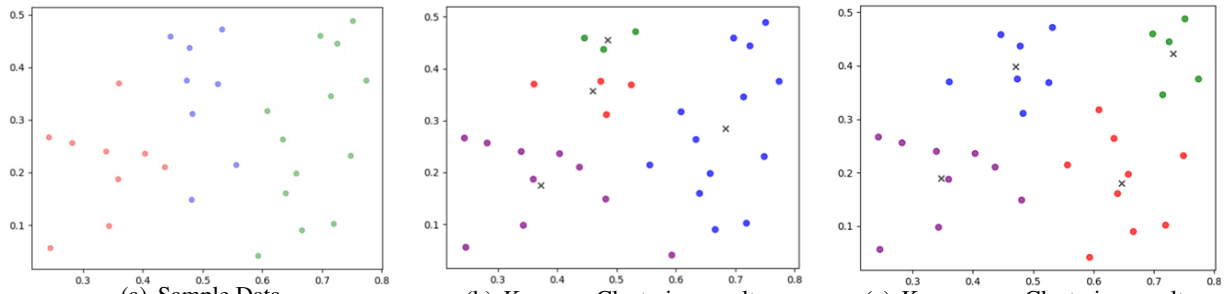
Table 2. Comparison experiments.

Model	Accuracy %	Speed (ms)
Faster R-CNN [18]	83.73	264
SSD [19]	80.71	242
RetinaNet [20]	82.92	216
YOLOv3 [21]	78.85	103
YOLOv5	79.03	71
YOLOv6 [22]	76.14	78
YOLOv7 [23]	78.94	81
YOLOv8 [24]	81.15	96

4.3.2. K-means++

The default clustering algorithm of YOLOv5 is the K-means algorithm, and Figure 11 shows the results of the K-means and K-means++ clustering algorithms for clustering the same defective sample data.

The results from Figure 11(b) show that the K-means algorithm is sensitive to the initialization of the central cluster, and Figure 11(c) shows that the K-means++ algorithm can better complete the clustering of the sample data when the initial cluster centers are close together.



(a) Sample Data (b) K-means Clustering results (c) K-means++ Clustering results

Figure 11. Comparison of clustering results.

Figure 12 shows the comparison of the loss function curves of the YOLOv5 model taking the two clustering methods. Taking the K-means algorithm has been in a large oscillation state without showing a convergence trend, while taking the K-means++ clustering algorithm starts to converge slowly and stabilizes at 58 epochs.

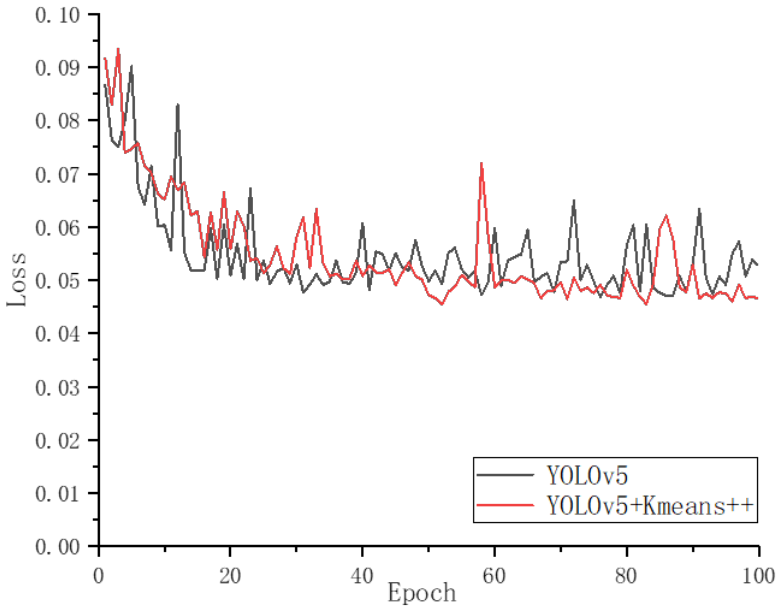


Figure 12. Loss curve comparison.

Comparison of Table 3 shows that the mAP value of the model improved by 0.4% after using the K-means++ algorithm.

Table 3. Comparison of models adopting clustering algorithms.

Method	mAP@0.5/%
YOLOv5	79.03
YOLOv5+K-means++	79.43

Table 4. Comparison of models with added attention mechanisms.

Method	mAP@0.5/%
YOLOv5+K-means++	79.43
YOLOv5+K-means++ +SE	81.74
YOLOv5+K-means++ +CA	84.42

4.3.3. Adding the attention mechanism module

To enhance the model's performance, the Squeeze-and-Excitation (SE) module and the CA module are incorporated. A comparison of the loss function curves of the model with the addition of these two attention mechanisms is illustrated in Figure 13. The network loss function is minimized by adding CA module.

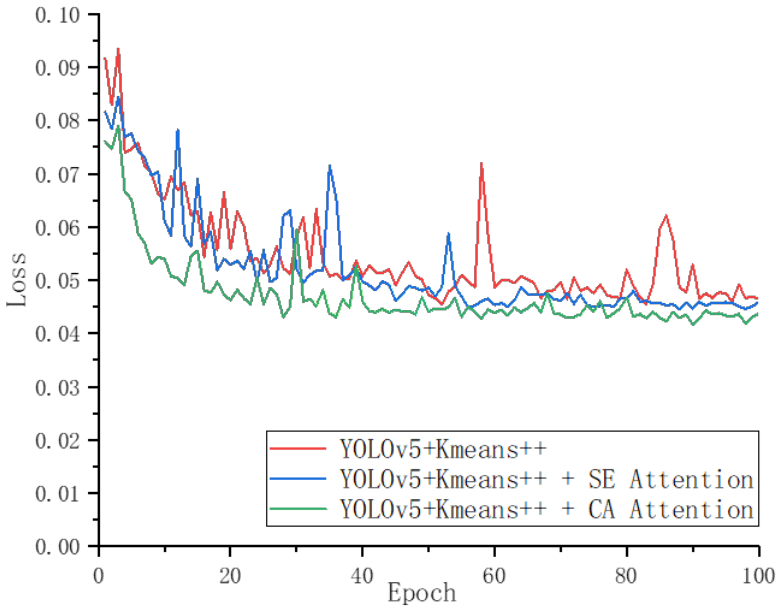


Figure 13. Loss curve comparison.

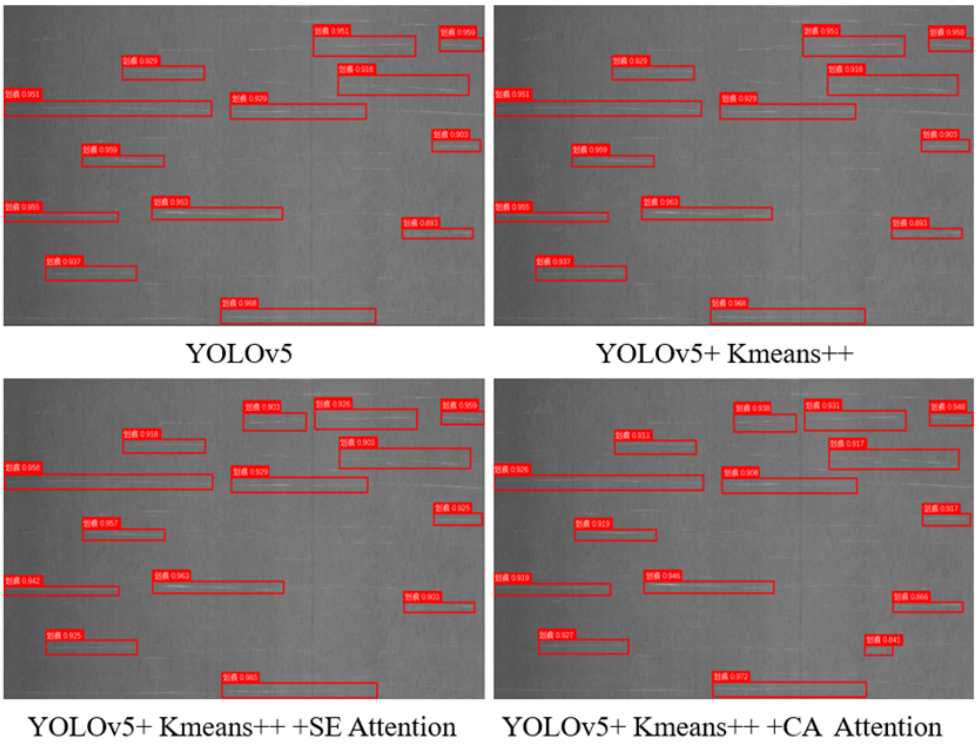


Figure 14. Detection effect of small targets.

Table 4 shows the comparative results of the models with the addition of the two attention mech-

anisms. Adding the attention mechanism can improve the detection precision of the network model, and the highest mAP is obtained after adding CA attention.

Figure 14 shows the detection effect of the model on target defects after adding the attention mechanism. For the same data, the model with the original model and the model after using the K-means++ clustering algorithm were consistent in detecting the target defects, and 13 target defects were detected. After adding SE attention, the model detected 14 target defects. After adding CA attention, the model detected 15 target defects. Compared with that without adding the attention mechanism, the detection effect of the model is improved, but the model still misses non-significant target defects and dense target defects seriously.

Although adding the CA attention module is as unsatisfactory as adding the SE attention module for small target detection, the CA loss function is the smallest and the mAP value is the highest; therefore, the CA attention module is chosen to be added to the improved model.

4.3.4. Adding detection module

A comparison of the loss function curves of the model before and after adding the new detection layer is shown in Figure 15. The improved model converges slower, but the oscillation frequency is relatively smaller and the loss values are lower.

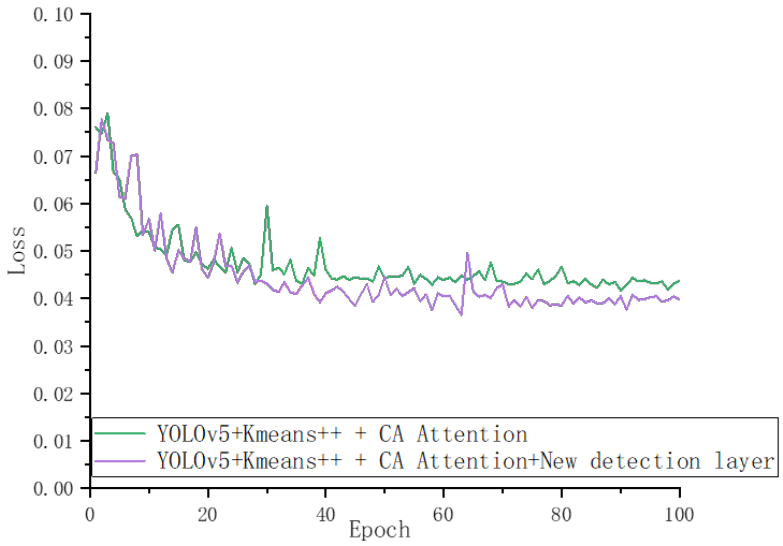


Figure 15. Loss curve comparison.

As seen in Table 5, the mAP value of the our network with the addition of the new detection layer increased by 2.29%, and also deepened the network depth, leading to an increase in the computation of the improved model and an increase in the detection time of a single image from 117 ms to 231 ms. Although this scheme can increase the detection accuracy of small targets, the complexity of the network leads to a decrease in the inference speed to the extent that it is impossible to real-time detection problem. Follow-up experiments were conducted to lighten the model to improve the inference speed of the model.

Table 5. Comparison of models with the addition of new detection layers.

Method	mAP@0.5/%	Speed (ms)
YOLOv5+K-means++ +CA	84.42	149
YOLOv5+K-means++ +CA+New detection layer	86.71	231

4.3.5. Lightweight network

Figure 16 shows the change of the loss function of each model during the training process. The loss value of the model after replacing the backbone network is higher than before the improvement, which is due to the fact that the MobileNetV3 module drastically reduces the number of parameters during the operation, resulting in a decrease in the ability of the model to represent the features.

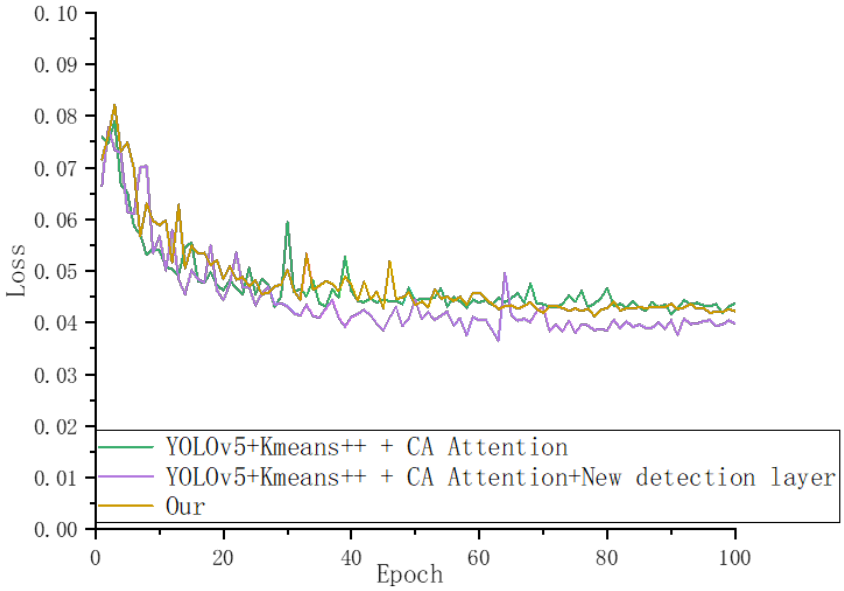


Figure 16. Loss curve comparison.

According to Table 6, the mAP value of our final model is reduced by 0.84% compared to the model before the improvement, and the detection precision is reduced. However, the detection speed of a single image is reduced from 231 ms to 54 ms.

Table 6. Comparison of model performance using different methods.

Method	mAP@0.5/%	Speed (ms)
YOLOv5+K-means++ +CA	84.42	149
YOLOv5+K-means++ +CA+New detection layer	86.71	231
Ours	85.87	54

Figure 17 shows the detection effect of the model for target defect detection after the introduction of the lightweight network. The left shows the model before the introduction of the lightweight network,

which detected 20 target defects. On the right is our model with 18 target defects detected. When four different improvement methods are gradually added, our model can substantially improve the detection efficiency of the model within a reasonable sacrifice of detection accuracy. This detection effect is consistent with the results in Table 6.

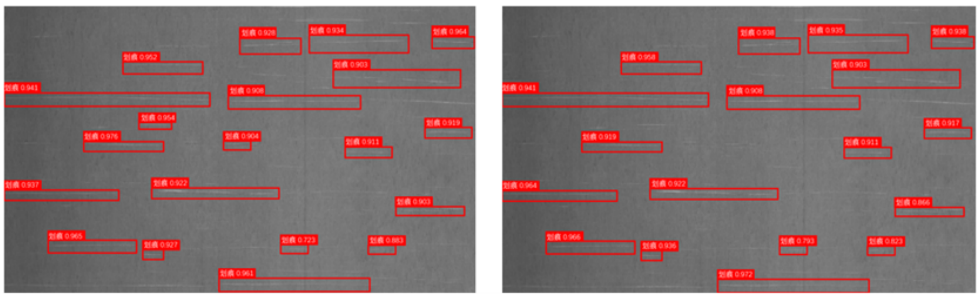


Figure 17. Detection effect of small targets.

4.4. Training effect analysis

Figure 18 shows the comparison between our model and the original network YOLOv5 to detect the target defects. Our model can detect the defects missed by the original model and does not mistakenly detect the stitching traces of the images as scratch-like defects with higher detection accuracy.

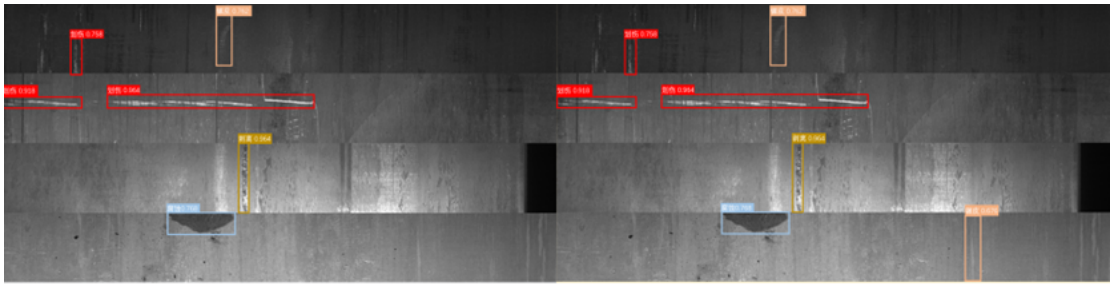


Figure 18. Overall detection effect comparison.

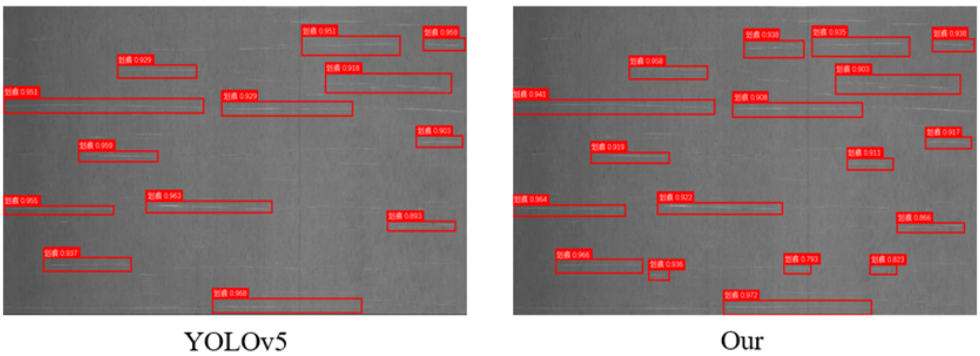


Figure 19. Non-significant defect detection effect comparison.

Figure 19 shows the comparison of detection results on non-significant defects. Our model has higher detection accuracy and more precise localization for small-sized target defects.

The comparison in Table 7 shows that the mAP value of our model increased from 79.43% to 85.87%, an increase of 6.44%. The leakage rate for 4 types of defects detection is reduced by 1.34%. The detection time of a single image is shortened from 71 ms to 54 ms, a reduction of 17 ms. The single image detection speed of our model is improved by 30%.

Table 7. Comparison of detection of improved models.

Method	Leakage rate%	mAP@0.5/%	Speed (ms)
YOLOv5+K-means++	8.12	79.43	71
Ours	6.78	85.87	54

5. Conclusions

The experimental results of our model for bearing surface defect detection show that it can automatically extract the target features in complex images, significantly improve the detection of non-significant defect targets, and the detection speed and accuracy of detection can meet the requirements of actual production, which can quickly and accurately detect the condition of bearing appearance defects, improve detection efficiency and reduce costs.

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

1. L. Eren, A. Karahoca, M. J. Devaney, Neural network based motor bearing fault detection, in *Proceedings of the 21st IEEE Instrumentation and Measurement Technology Conference (IEEE Cat. No. 04CH37510)*, **3** (2004), 1657–1660. https://doi.org/10.1109/IMTC.2004.1351399
2. P. K. Kankar, S. C. Sharma, S. P. Harsha, Fault diagnosis of ball bearings using machine learning methods, *Expert Syst. Appl.*, **38** (2011), 1876–1886. https://doi.org/10.1016/j.eswa.2010.07.119
3. C. Tastimur, M. Karakose, I. Aydın, E. Akin, Defect diagnosis of rolling element bearing using deep learning, in *2018 International Conference on Artificial Intelligence and Data Processing(IDAP)*, (2018), 1–5. https://doi.org/10.1109/IDAP.2018.8620743
4. J. S. Senanayaka, H. V. Khang, K. G. Robbersmyr, Multiple fault diagnosis of electric powertrains under variable speeds using convolutional neural networks, in *2018 XIII International Conference on Electrical Machines(ICEM)*, (2018), 1900–1905. https://doi.org/10.1109/ICELMACH.2018.8507096
5. C. Sobie, C. Freitas, M. Nicolai, Simulation-driven machine learning: Bearing fault classification, *Mech. Syst. Signal Process.*, **99** (2018), 403–419. https://doi.org/10.1016/j.ymssp.2017.06.025
6. M. Sadoughi, C. Hu, Physics-based convolutional neural network for fault diagnosis of rolling element bearings, *IEEE Sens. J.*, **19** (2019), 4181–4192. https://doi.org/10.1109/JSEN.2019.2898634

7. D. W. Kim, E. S. Lee, W. K. Jang, B. H. Kim, Y. H. Seo, Effect of data preprocessing methods and hyperparameters on accuracy of ball bearing fault detection based on deep learning, *Adv. Mech. Eng.*, **14** (2022), 1900–1905. https://doi.org/10.1177/16878132221078494
8. A. Bapir, İ. Aydin, A comparative analysis of 1D convolutional neural networks for bearing fault diagnosis, in *2022 International Conference on Decision Aid Sciences and Applications (DASA)*, (2022), 1406–1411. https://doi.org/10.1109/DASA54658.2022.9765229
9. S. E. M. P. Kone, K. Yatsugi, Y. Mizuno, H. Nakamura, Application of convolutional neural network for fault diagnosis of bearing scratch of an induction motor, *Appl. Sci.*, **12** (2022), 1406–1411. https://doi.org/10.3390/app12115513
10. M. Chen, L. Yu, C. Zhi, Improved faster R-CNN for fabric defect detection based on Gabor filter with genetic algorithm optimization, *Comput. Ind.*, **134** (2022), 103551. https://doi.org/10.1016/j.compind.2021.103551
11. J. Luo, Z. Yang, S. Li, Y. Wu, FPCB surface defect detection: A decoupled two-stage object detection framework, *IEEE Trans. Instrum. Meas.*, **70** (2021), 1–11. https://doi.org/10.1109/TIM.2021.3092510
12. X. Zhang, R. Ma, M. Li, Feature enhancement based on regular sparse model for planetary Gearbox fault diagnosis, *IEEE Trans. Instrum. Meas.*, **71** (2022), 1–16. https://doi.org/10.1109/TIM.2022.3176244
13. J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, (2016), 779–788. https://doi.org/10.48550/arXiv.1506.02640
14. B. Bahmani, B. Moseley, A. Vattani, Scalable k-means++, preprint, arXiv:1203.6402. https://doi.org/10.48550/arXiv.1203.6402
15. G. Yuan, J. Liu, H. Liu, Detection of cigarette appearance defects based on improved YOLOv4, *Electron. Res. Arch.*, **31** (2023), 1344–1346. https://doi.org/10.3934/era.2023069
16. Q. Hou, D. Zhou, J. Feng, Coordinate attention for efficient mobile network design, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (2021), 13708–13717. https://doi.org/10.48550/arXiv.2103.02907
17. A. Howard, M. Sandler, G. Chu, Searching for mobilenetv3, in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, (2019), 1314–1324. https://doi.org/10.1109/ICCV.2019.00140
18. S. Ren, K. He, R. Girshick, Faster r-cnn: Towards real-time object detection with region proposal networks, *Adv. Neural Inf. Process. Syst.*, **28** (2015). https://doi.org/10.48550/arXiv.1506.01497
19. W. Liu, D. Anguelov, D. Erhan, SSD: Single shot multiBox detector, in *Computer Vision–ECCV 2016. ECCV 2016. Lecture Notes in Computer Science*, (2016), 21–37. https://doi.org/10.1007/978-3-319-46448-0_2
20. T. Y. Lin, P. Goyal, R. Girshick, Focal loss for dense object detection, in *Proceedings of the IEEE International Conference on Computer Vision*, (2017), 2980–2988. https://doi.org/10.48550/arXiv.1708.02002

21. J. Redmon, A. Farhadi, Yolov3: An incremental improvement, preprint, arXiv:1804.02767. https://doi.org/10.48550/arXiv.1804.02767

22. C. Li, L. Li, H. Jiang, YOLOv6: A single-stage object detection framework for industrial applications, preprint, arXiv:2209.02976. https://doi.org/10.48550/arXiv.2209.02976

23. C. Y. Wang, A. Bochkovskiy, H. Y. M. Liao, YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, preprint, arXiv:2207.02696. https://doi.org/10.48550/arXiv.2207.02696

24. J. Terven, D. Cordova-Esparza, A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond, preprint, arXiv:2304.00501. https://doi.org/10.48550/arXiv.2304.00501

AIMS Press