YOLO V5s-based Deep Learning Approach for Concrete Cracks Detection

Zhen Yu*

School of Software, Jiangxi Normal University, Nanchang 330027, China

ABSTRACT: Complex environmental conditions can lead to a variety of cracks in concrete engineering structures, and if these cracks are not promptly investigated and repaired, it is likely to lead to serious engineering accidents. Most of the traditional crack detection method is by manual exclusion, which overly relies on the knowledge and experience of inspectors, and different inspectors have different definitions of crack detection standards, so it lacks a certain objectivity in quantitative analysis, and the efficiency of manual detection will gradually decrease as the workload rises. In recent years, deep learning networks have made many developments in the field of computer vision by their strong feature extraction ability and autonomous learning capability. The main objective of this paper is to detect crack information in crack images using YOLO V5s-based deep learning algorithm. Considering the complexity of the crack image background, the author adopted the threshold segmentation method based on the Otsu maximum inter-class variance to achieve the purpose of removing the background noise from crack images by constructing a connected domain for grayscale change points so as to fuse the noise points with the background. After that, the author used the YOLO V5s model to train and test the 3500 manually labeled crack images, and adopted the K-Means method to calculate the optimal initial anchor box size and pass it to the model for training, so as to improve the model's detection of cracks. The evaluation index of the model after these two optimization methods was 84.37% for average precision (AP), 76.01% for average recall (AR), and 79.97% for average F1-score.

1. INTRODUCTION

Concrete engineering structures, such as highways, bridges and buildings, are subject to elastic deformation, hydraulic contraction, thermal contraction or expansion due to environmental factors such as air oxidation, rain corrosion and sunlight exposure, which will eventually lead to the cause of cracks1. The presence of cracks can pose a significant safety risk to engineering buildings, so it is necessary to regularly detect cracks, which can infer some potential causes of cracking such as inherent damage and deterioration in buildings and infrastructure based on their morphology and location2.

At present, most of the investigation of cracks in engineering structures relies on the traditional manual identification, marking and recording, which is not only inefficient and requires a lot of human resources, but also relies excessively on manual experience, making it difficult to objectively assess the crack information quantitatively3. In addition, there are also infrared thermal imaging detection, ultrasonic detection, laser detection methods, etc.4, but they all have disadvantages such as complex processes and high prices, and cannot be used as an easy, fast and low-cost method in the field of crack detection.

In the research process to achieve automated crack detection, it can be broadly divided into two directions of research, one based on digital image processing and the other on deep learning networks. Zou et al.5 proposed a fully automated crack detection method called CrackTree, which first developed a geodesic shadow-removal algorithm to eliminate pavement shadows while preserving cracks. Prasanna et al.6 proposed a STRUM (spatially tuned robust multifeature) classifier, which combines a robust line segment detector, a spatially tuned multifeature computation, and a machine learning classifier. The classifier uses straight line fitting to find cracked segments and performs well even there are noise and clutter.

Nowadays, deep learning is widely used as a complementary branch of machine learning, and deep learning neural networks have strong feature extraction and autonomous learning capabilities7, making them efficient and accurate for practical needs.

Zhang et al.8 first proposed a Deep Convolutional Neural Network (DCNN) algorithm for crack detection, which uses ConvNets to learn discriminative features directly from square image blocks (patches) in a given original image, for the classification of cracked and uncracked blocks. Zhang et al.9 developed a CNN-based deep learning network, CrackNet, to achieve crack detection at the pixel level. Compared to traditional CNN networks, CrackNet does not have pooling layers. The reason for this design is that the pooling layers lead to a

*Corresponding author. Email: yzhen_zane@163.com

reduction in the dimensionality of the feature mapping 10, which affects the accuracy of the network for crack detection at the pixel level. However, because CrackNet uses more than one million parameters, this makes the network training very difficult. Therefore, Yang et al.2 achieved detection and measurement of cracks at the pixel level by using a fully convolutional network (FCN), and although the accuracy of this model is not as high as CrackNet, the training time of FCN is less than 1% of CrackNet. The FCN consists of two parts, down-sampling layers and up-sampling layers. This special network structure allows FCN to detect targets at multiple scales.

The You Only Look Once (YOLO) network series11121314 unifies the classification and localization problems of target detection into a regression problem, and the YOLO networks integrate the prediction of bounding box coordinates and the calculation of class probabilities into a single neural network model, so it greatly accelerates the training of the network, while achieves a high accuracy.

The overall goal of this paper is to detect and localize cracks by using the YOLO V5s model. In this paper, the author used a threshold segmentation method based on the Otsu maximum inter-class variance to exclude the interference of background noises. In addition, the K-Means method is also used in this paper to obtain the optimal initial anchor box size of manually labelled crack dataset, which accelerated the convergence of the network and improved the accuracy of the model for crack detection at the same time.

2. MATERIALS AND MODELS

2.1. Image date set

The concrete crack image dataset15 used in this study was published open source by Özgenel, Ç.F., Gönenç Sorguç, and is shown in Figure 1. The dataset was generated on the basis of 458 high-resolution images using the method proposed by Zhang et al. 8. The dataset was collected from METU campus buildings and divided into negative and positive crack images for image classification, with 227 pixels × 227 pixels RGB channels. The dataset covers almost all common forms of concrete cracks, including horizontal, vertical and diagonal cracks. The dataset also contains a variety of natural light conditions, including dark and bright light. Besides, the concrete where the cracks are located include both smooth and rough conditions.

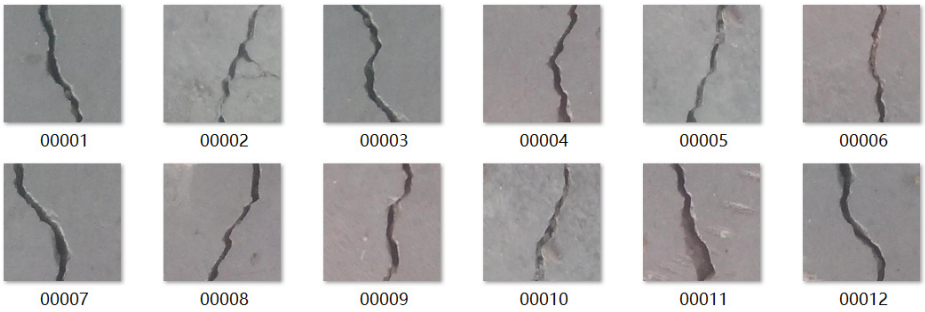

Figure 1 Crack dataset

In this study, a total of 4000 concrete crack images were selected without any random rotation or overturning data enhancement. After the manual labeling process, 3500 pictures under different noise and illumination conditions are used as the training set, and the other 500 pictures are used as the testing set.

2.2. Labelling

In this study, the author used LabelImg, an image labelling tool commonly used in the field of deep learning computer vision, as shown in Figure 2. This tool makes it easy to quickly label the location and class of target objects in an image. The process of using LabelImg to save the information of target objects is as follows.

(1) Step 1: Importing the image dataset folder.

(2) Step 2: Locating the crack object with a rectangular box (groungtruth box).

(3) Step 3: Labeling the rectangular box.

(4) Step 4: Selecting the appropriate file format to be saved (txt format is required by the YOLO V5 Model) to save the crack information.

In the crack information files, there are five numbers in each line, representing the information of a target object in the image. The first number represents the label of the target object, the second and third numbers are the coordinates of the centre of the rectangular labeling box obtained by taking the upper left corner of the image as the origin coordinates, and the fourth and fifth numbers represent the relative width and height of the groundturth box respectively.

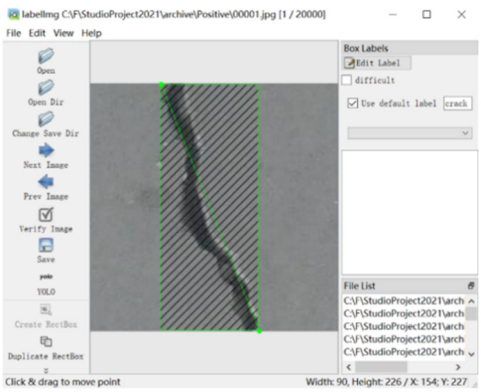

Figure 2 LabelImg

2.3. Mosaic Augmentation

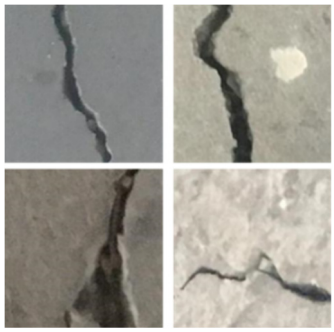

Figure 3 Original images

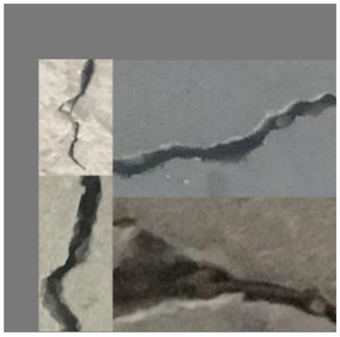

Figure 4 Mosaic Augmentation

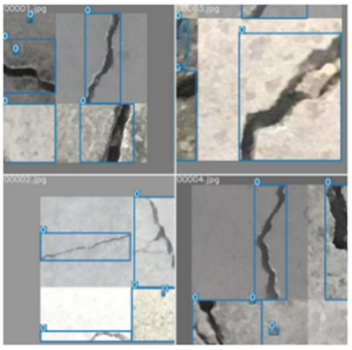

Figure 5 Training process

In order to expand the dataset, this study used Mosaic data augmentation, randomly selecting four images (Figure 3), randomly scaling, and then randomly distributing for stitching, which greatly enriched the detection dataset, especially the random scaling increased many small targets, allowing the network to be more robust, the principle schematic is shown in Figure 5. At the same time, with the Mosaic augmentation, the data from 4 images can be calculated directly during the training process, so that the batch size does not need to be large and a single GPU can achieve better efforts.

2.4. Concrete cracks detection based on YOLO V5s

In order to identify and locate concrete cracks quickly and accurately, the author chose the YOLO V5 model in this study. YOLO V5 is a typical one-stage target detection algorithm that does not need to go through a candidate region stage, but can directly generate the class probability and location coordinates of the target object 16, and can directly obtain the final detection results after a single inspection, therefore, this model has a faster detection speed.

YOLO V5 will pass each batch of training data through the data loader and expand the training data at the same time. Three types of data augmentation are used in the data loader: adaptive image scaling, color space adjustment and Mosaic augmentation. In addition, the YOLO V5 model is analysed by the K-Means algorithm to obtain the suitable anchor box sizes for the custom dataset. These optimization strategies greatly improve the training speed and accuracy of the model. YOLO V5 has a total of four different network structures, and in this study, the author chose the YOLO V5s network structure, which has fewer training parameters, is easy to train, has a fast training speed and is efficient enough for the detection of concrete cracks. The YOLO V5s network structure is shown in Figure 6.

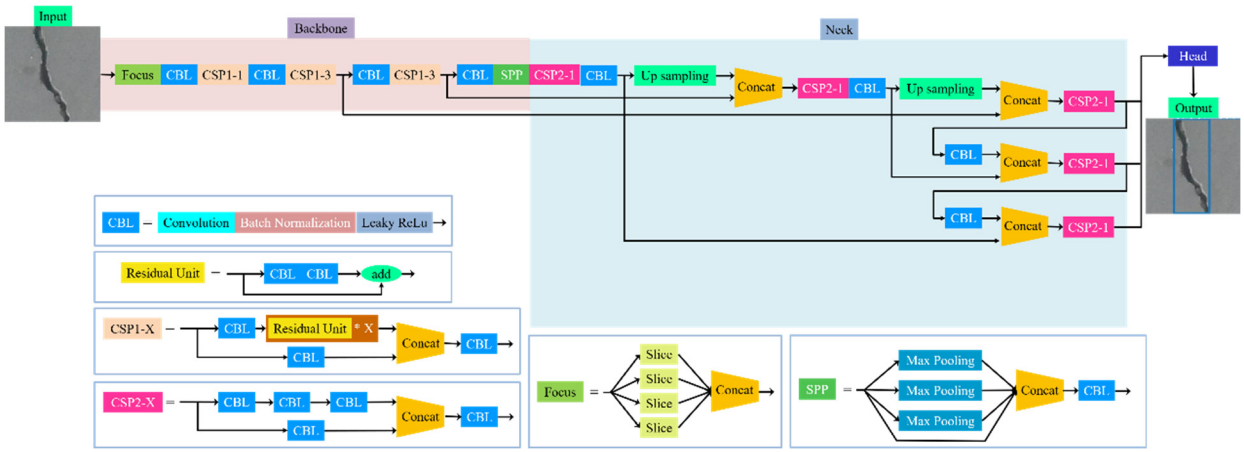

Figure 6 The structure of YOLO V5s

In the Backbone structure of YOLO V5s network, Focus, CBL, SPP and CSPDarkNet modules are used. In the Neck structure, YOLOv5 uses the FPN combined with PAN modules as YOLOv4 to enhance the multi-scale feature fusion capability of the network features. In the output layer, GIoU_Loss17 is used as the loss function for the prediction as well as non-maximum suppression (NMS) is used to filter the multi-target boxes.

The basic operations and the functions of the modules of the YOLO V5s network are as follows.

(1) Concat operation: splicing tensors to obtain a new tensor whose dimension will be expanded from the original.

(2) Adding operation: tensors are added to obtain a new tensor, and the new tensor's dimension will not be expanded.

(3) CBL module: this module is composed of a Convolution layer, a Batch Normalization layer and a Leaky ReLu activation function, whose role is to extract the features of the image.

(4) Focusing module: the function of this module is to slice the image four times when it enters the trunk section, and then use concat operation to expand the input channel. Compared with the original RGB three channel mode, the new input becomes 12 channels, and then convolution operation is carried out. Finally, a down sampling feature map without information loss is obtained.

(5) CSP (Cross Stage Partial) module: Two CSP network structures are used in the YOLO V5s network. The CSP1_X structure is applied in the Backbone section, and X refers to using X residual units in this structure, which serves to strengthen the learning ability of the CNN, reduce the computational bottleneck and lower the memory cost18. The CSP2_X structure is applied to the Neck of the network, borrowing from the structure of CSPNet. X refers to the fact that there are 2*X CBL modules in the module, whose role is to strengthen the network feature fusion.

(6) SPP (Spatial Pyramid Pooling) module: The SPP module consists of four parts, which are three Max Pooling operations with different convolutional kernel sizes and a Concat operation. The main function of this part is to fuse local and global features.

(7) FPN (Feature Pyramid Network) + PAN (Path Aggregation Network) module: The FPN19 module is top-down form, which expands the dimensionality of the feature maps by up-sampling, and then these feature maps are fused with the same dimension feature maps in the backbone part of the network, so that the stronger feature information in the deep layer of the network is transferred to the shallow layer. Then the bottom-up PAN20 module is used to reduce the dimensionality of the deeper feature maps by down-sampling, and then these feature maps are fused with the same dimensional feature maps in the FPN, so that the location information in the shallow layer of the network can be transferred to the deeper layer of the network. Finally the feature maps of differe dimensions contain both stronger feature information and location information, making the network capable of detecting target objects of different dimensions.

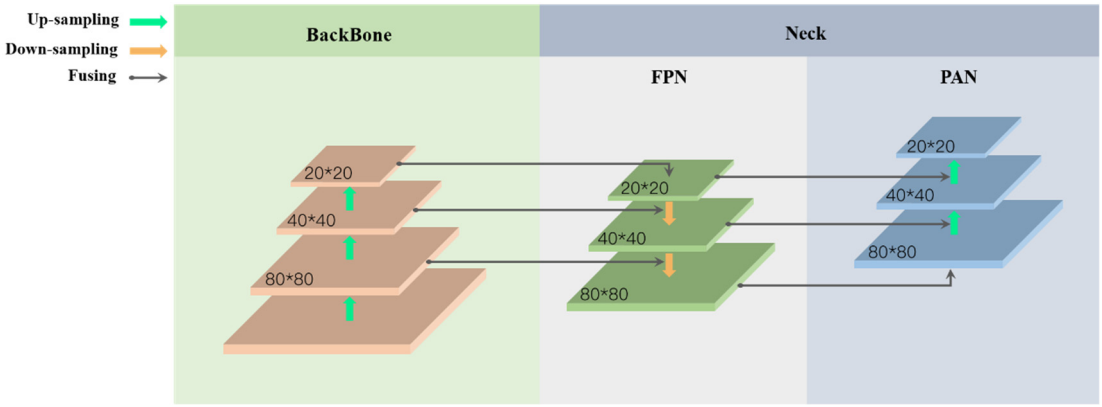

Figure 7 PAN + FPN module

3. METHODS

3.1. Denoising concrete crack images

In the process of engineering structural crack detection, the actual acquired crack images have complex characteristics such as much noise, weak crack information and large differences in crack morphology due to surface morphology and environmental interference, making it difficult to extract the crack targets from the background.

The crack image dataset the author used contains several types of concrete, where the noise points in the concrete images can cause serious interference to the crack detection, leading to a decrease of the accuracy of the model. To address the impact of noisy points on the model, the author used a threshold segmentation method based on Otsu's maximum inter-class variance 21, which achieves denoising by constructing a connected domain for grey-scale change points, thereby it can fuse the noisy points with the background. The method can better preserve the edge information of cracks and is insensitive to image brightness and contrast, with fast computation and high error tolerance.

3.2. Adapting the optimal anchor box size

Anchor box is the bridge between the bounding box and the groundtruth box. The bounding box is predicted on the association between anchor box and groundtruth box through GIoU and on the size and position of the anchor box. The large size anchor box is suitable for detecting large target tasks, the medium size anchor box is suitable for detecting medium size target tasks, and the small size anchor box is suitable for detecting small target tasks. Depending on the actual size of the target in the training

dataset, a set of suitable initial anchor box sizes can be chosen to speed up the training of the model and improve the accuracy. The model weight file used in this study is yolov5s.pt, and its corresponding conFigureuration file yolov5.yaml gives the initial parameters of the model, as well as a set of initial anchor box sizes, which were obtained by training on the COCO128 image dataset and are suitable for the detection of 80 categories of targets, but there is only one crack class in this study, so it is clear that the initial anchor boxes are not suitable for our training data.

In order to obtain an initial anchor box that is suitable for our concrete crack dataset, this study chose to use the K-Means clustering method22. Firstly, K cluster boxes were randomly given as input, and the crack information matrix were also used as input, then the correlation between each cluster box and each groundtruth box was calculated by using the GIoU method. Then the author divided all the groundtruth boxes into K clusters, calculated the average size of groundtruth boxes in each K clusters, assigned each average value to a closest cluster box, and repeated the above operation with these new clusters until these average size value stabilized. Finally the author got K clusters boxes, which were set to the initial anchor box sizes. In this study, the value of K to 9 was set, meaning that there will be 9 initial anchor boxes.

3.3. Evaluation criteria

In this study, the author used Precision and Recall as the criteria for evaluating model performance. However, as Precision and Recall are contradictory criteria, in general, when Precision is high, Recall is low, and when Recall is high, Precision is low, the author has also used F1-Score to combine the performance of Precision and Recall, and therefore used the evaluation parameters Precision(P), Recall(R), and F1-Score. The formulae for these parameters are given in Eqs. 1, 2 and 3

$$P = \frac{TP}{TP+FP} \tag{1}$$

$$R = \frac{TP}{TP+FN} \tag{2}$$

$$F1 - Score = \frac{2PR}{P+R} \tag{3}$$

where TP, FP, and FN denote True Positive, False Positive, and False Negative, respectively.

In addition, mAP (mean Average Precision) is used as an additional metric for the evaluation of the crack detection model. More specifically, the author adopted mAP@0.5, it represents the mAP value when the threshold is set to 0.5 for determining whether an IoU is a positive or negative.

4. RESULT

4.1. Experimental results and analysis

To test the crack detection ability of the model, the author used 3500 crack images with their corresponding labels as the training data, and the remaining 500 crack images with their corresponding labels as the test data, the size of the input images is 640 pixels × 640 pixels. The author set the time, batch size and learning rate to 100, 8 and 0.01 respectively, and then compared the prediction results with no optimization process, using the optimal initial anchor box process and denoising process. The experimental results in real are shown in Figure 8-11, and the experimental data is presented in Table 1. Figure 12, 13, and 14 show the Precision curves, Recall curves and mAP@0.5 curves, respectively.

Figure 8 Original cracks image

Figure 9 Results without optimization

Figure 10 Results with adopting the optimal initial anchor box size

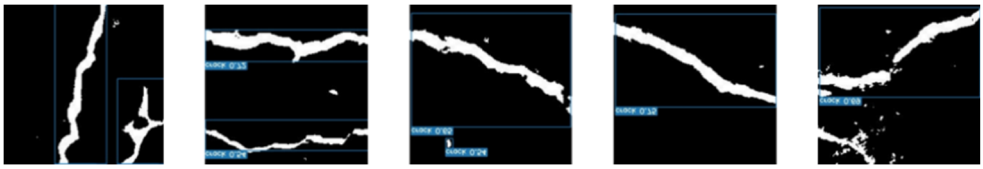

Figure 11 Results with denoising optimization

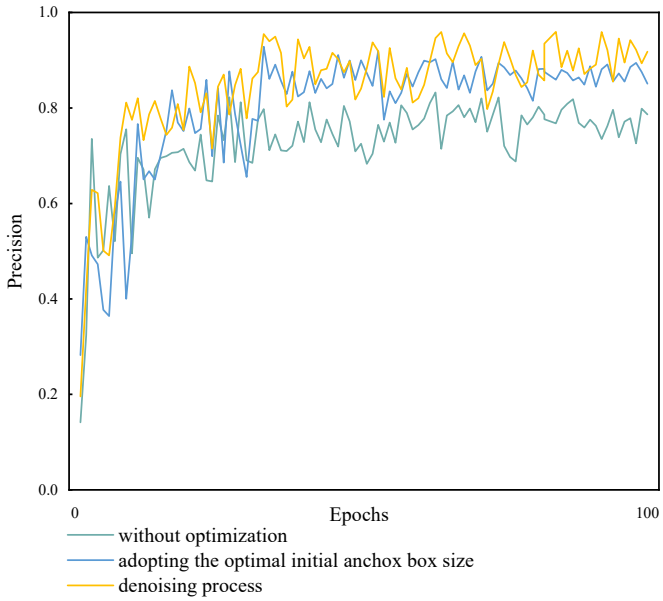

Figure 12 Precision curves

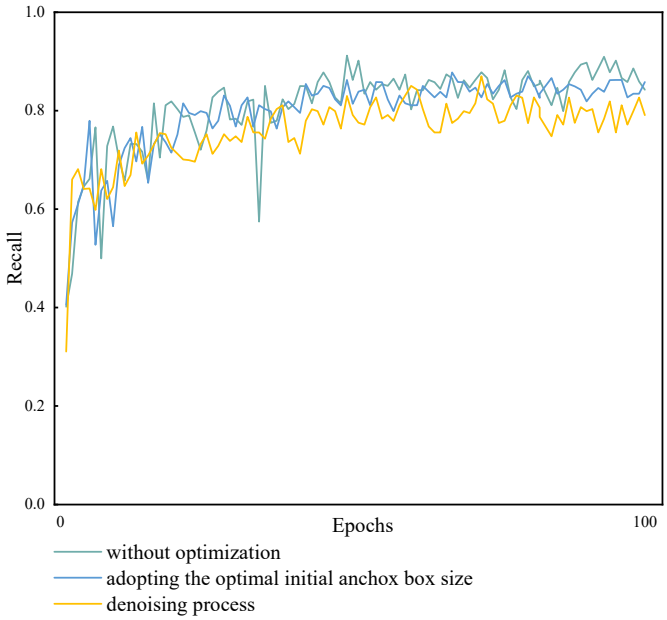

Figure 13 Recall curves

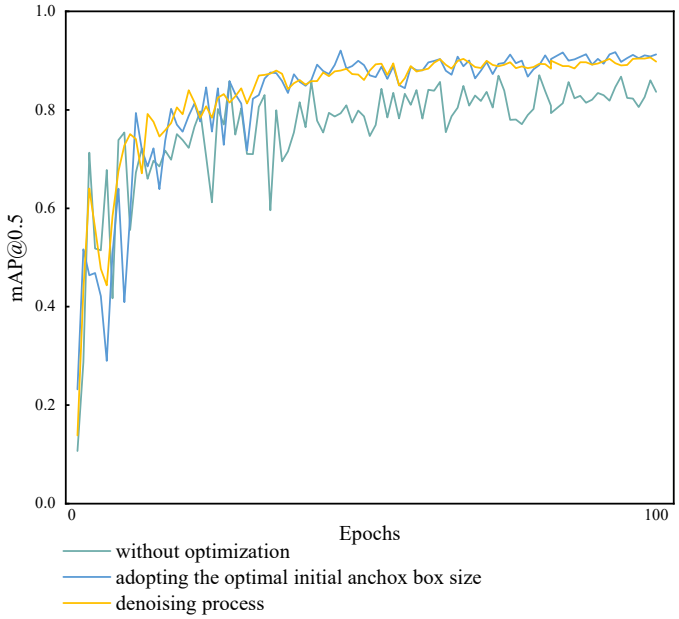

Figure 14 mAP@0.5 curves

4.2. Results without optimization

Figure 8 shows the manually marked cracks in one of the batches during our testing, and Figure 9 shows the cracks predicted by YOLO V5s. In comparison, the author finds that our model is better at detecting some crack details than the manual detection. For example, for the identification and detection of small cracks, the manual markers may be overlooked, but our model can detect them well, indicating that the model is not only good at detecting large target objects, but also has excellent detection capability for small targets.

After 100 epochs of processing, the model achieved an Average Precision (AP) of 72.75% and an Average Recall (AR) of 80.82% on the test dataset, with an average F1-Score of 76.57%.

4.3. Results with adapting the optimal initial anchor box size

After obtaining the optimal initial anchor box size by using K-Means algorithm, Figure 10 shows the crack information predicted by the model. Figure 12-14 contain the Precision curve, recall curve and mAP@0.5 curve for 100 training epochs with using the optimal initial anchor box size. After 100 epochs, the AP and AR of our method were 80.05% and 79.69% respectively, which was 7.3% higher and 1.11% lower than that without any optimization process. The average F1-Score (79.87%) and mAP@0.5 (81.98%) was 3.3% and 5.72% higher than the correspond results without any optimization process.

4.4. Results with denoising optimization

After the denoising process, most of the interference information among the crack images was filtered out, retaining clearer crack information, which helped the model for crack detection, and the experimental detection results are shown in Figure 11.

Figure 12-14 show the Precision curve, Recall curve and mAP@0.5 for in 100 tests after the denoising process. After 100 epochs, the AP and AR of our method are 84.37% and 76.01% respectively, which are 11.62% higher and 4.81% lower than the method without any optimization process. The average F1-Score (79.97%) and mAP@0.5 (83.13%) was 3.4% and 6.87% higher than that without any optimization process.

Table 1 Concrete cracks detection results of model trained without using optimization and with using optimization

YOLO V5s	AP (%)	AR (%)	F1-Score (%)	mAP@0.5 (%)
Model trained without any optimization	72.75%	80.82%	76.57%	76.26%
Model trained with adapting the optimal initial anchor box size	80.05%	79.69%	79.87%	81.98%
Model trained with denoising optimization	84.37%	76.01%	79.97%	83.13%

5. CONCLUSION

Achieving automatic detection of cracks in engineering structures plays a very important role in ensuring the safety of people's lives and property. Considering the complexity of the crack image background which affects the detection of cracks by the model, and the importance of selecting the appropriate anchor box in the YOLO algorithm, this paper adopted a threshold segmentation method based on Ostu's maximum inter-class variance to denoise the crack images, and used K-Means method to obtain the initial anchor box sizes that are suitable for the crack image dataset of this research. The experimental results showed that both approaches were able to improve the detection capability of YOLO V5s for cracks.

However, the analysis of the experimental results revealed that the actual performance of the network was not excellent, and the detection accuracy for cracks in some cases performed poorly, so the following improvements are needed for this project in the future.

(1) The detection effectiveness of the model for small cracks should be improved by enhancing the multi-scale feature fusion capability of the model.

(2) The detection ability of the model for overlapping and discontinuous cracks should be improved by selecting suitable Non-Maximum Suppression (NMS), such as soft NMS or DIoU_NMS, to filter the overlapping target frames, while the threshold value of NMS should be appropriately increased to retain more prediction results and improve the detection rate.

REFERENCES

1. Nama, Pooja, et al.. Study on causes of cracks & its preventive measures in concrete structures [J]. International Journal of Engineering Research and Applications, 2015, 5(5): 119-123.
2. Yang X, Li H, Yu Y, et al.. Automatic pixel-level crack detection and measurement using fully convolutional network[J]. Computer-Aided Civil and Infrastructure Engineering, 2018, 33(12): 1090-1109.
3. Gavilán M, Balcones D, Marcos O, et al.. Adaptive road crack detection system by pavement classification[J]. Sensors, 2011, 11(10): 9628-9657.
4. Fujita Y, Hamamoto Y. A robust automatic crack detection method from noisy concrete surfaces[J]. Machine Vision and Applications, 2011, 22(2): 245-254.
5. Zou Q, Cao Y, Li Q, et al.. CrackTree: Automatic crack detection from pavement images[J]. Pattern Recognition Letters, 2012, 33(3): 227-238.
6. Prasanna P, Dana K J, Gucunski N, et al.. Automated crack detection on concrete bridges[J]. IEEE Transactions on automation science and engineering, 2014, 13(2): 591-599.
7. Guo Y, Liu Y, Oerlemans A, et al.. Deep learning for visual understanding: A review[J]. Neurocomputing, 2016, 187: 27-48.
8. Zhang L, Yang F, Zhang Y D, et al.. Road crack detection using deep convolutional neural network[C]//2016 IEEE international conference on image processing (ICIP). IEEE, 2016: 3708-3712.
9. Zhang A, Wang K C P, Li B, et al.. Automated pixel-level pavement crack detection on 3D asphalt surfaces using a deep-learning network[J]. Computer-Aided Civil and Infrastructure Engineering, 2017, 32(10): 805-819.
10. Sun M, Song Z, Jiang X, et al.. Learning pooling for convolutional neural network[J]. Neurocomputing, 2017, 224: 96-104.
11. Redmon J, Divvala S, Girshick R, et al.. You only look once: Unified, real-time object detection [C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 779-788.
12. Redmon J, Farhadi A. YOLO9000: better, faster, stronger[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 7263-7271.
13. Redmon J, Farhadi A. Yolov3: An incremental improvement [J]. arXiv preprint arXiv:1804.02767, 2018.
14. Bochkovskiy A, Wang C Y, Liao H Y M. Yolov4: Optimal speed and accuracy of object detection[J]. arXiv preprint arXiv:2004.10934, 2020.
15. 2018 – Özgenel, Ç.F., Gönenç Sorguç, A. “Performance Comparison of Pretrained Convolutional Neural Networks on Crack Detection in Buildings”, ISARC 2018, Berlin.
16. Tian Z, Shen C, Chen H, et al.. Fcos: Fully convolutional one-stage object detection [C]// Proceedings of the IEEE/CVF international conference on computer vision. 2019: 9627-9636.
17. Rezatofighi H, Tsoi N, Gwak J Y, et al.. Generalized intersection over union: A metric and a loss for bounding box regression[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 658-666.
18. Wang C Y, Liao H Y M, Wu Y H, et al.. CSPNet: A new backbone that can enhance learning capability of CNN[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. 2020: 390-391.
19. Lin T Y, Dollár P, Girshick R, et al.. Feature pyramid networks for object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 2117-2125.
20. Liu S, Qi L, Qin H, et al.. Path aggregation network for instance segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 8759-8768.
21. Jiao S, Li X, Lu X. An improved Ostu method for image segmentation[C]//2006 8th international Conference on Signal Processing. IEEE, 2006, 2.
22. Likas A, Vlassis N, Verbeek J J. The global k-means clustering algorithm[J]. Pattern recognition, 2003,

36(2): 451-461.