applied sciences

MDPI

Article

A Multi-Scale Rebar Detection Network with an Embedded Attention Mechanism

Yanmei Zheng, Guanghui Zhou * and Bibo Lu *

College of Computer Science and Technology, Henan Polytechnic University, Jiaozuo 454003, China; zhengym02@hpu.edu.cn

* Correspondence: zgh1079937239@outlook.com (G.Z.); lubibo@hpu.edu.cn (B.L.)

Abstract: With the rapid development of computer vision and machine vision, methods based on deep learning have achieved good results in the field of object detection, identification, and tracking. However, for the detection and identification of rebars in smart construction sites, it is very difficult to perform accurate real-time detection of rebars by using object detection technology on the equipment in the field because of the problems of the dense cross-section between bundled bars, the mutual adhesion of cross-section boundaries, and mutual occlusion between cross-sections. To address the above problems, we propose a multi-scale rebar detection network RebarNet with an embedded attention mechanism based on YOLOv5, combining the K-means++ algorithm, attention mechanism, a newly designed SD_IoU Loss, and multi-scale feature fusion, aiming to solve the problems of missed and false detection in dense small object detection. Due to the problems of scarce rebar cross-section datasets, no publicly available large datasets, and weak rebar cross-sectional features, we constructed a new rebar cross-sectional dataset, used a semi-automatic annotation method to annotate part of the dataset, and then used the data enhancement algorithm to expand the rebar dataset. The experimental results show that the average accuracy (mAP) of our proposed RebarNet network is 97.9%, which is comparable to mainstream target detection algorithms such as Faster R-CNN, SSD, RetinaNet, CenterNet, CornerNet, YOLOv3, YOLOv4, and YOLOv5s. mAP0.5 is improved by 8.1%, 13%, 26.4%, 25.8%, 26.2%, 11.7%, 7.6%, and 9%, respectively. In addition, the frames per second (FPS) transmission reaches 89 frames per second, the model weight is only 17.0 MB. In summary, the proposed RebarNet can effectively reduce missed and false detections in the rebar counting detection task based on real-time detection.

Keywords: rebar detection; YOLOv5; K-means++; attention mechanism; SD_IoU loss; multi-scale detection

check for updates

Citation: Zheng, Y.; Zhou, G.; Lu, B. A Multi-Scale Rebar Detection Network with an Embedded Attention Mechanism. *Appl. Sci.* **2023**, *13*, 8233. https://doi.org/10.3390/app13148233

Academic Editor: Laura Cercenelli

Received: 11 May 2023
Revised: 28 June 2023
Accepted: 12 July 2023
Published: 15 July 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The development of new-generation information technology, such as the Internet of Things, cloud computing, big data, and spatial geographic information integration, has been deeply rooted in people's minds, and the concept of a smart city has emerged as a result. One of the core concepts is to embed high technologies, such as robotics, big data, and artificial intelligence, into industrial equipment to improve manufacturing efficiency and bring convenience to people's lives.

Object detection models are often used in industrial scene analysis modules. With the development of convolutional neural networks (CNNs), object detection techniques based on deep learning are widely studied and applied in the industry by a large number of scholars. In recent years, detection algorithms based on anchor boxes have become the mainstream idea in the field of object detection. Some other two-stage object detection algorithms based on a region proposal network (RPN), such as RCNN (Region-CNN) [1], Fast R-CNN (Fast Region-CNN) [2], Faster R-CNN (Faster Region-CNN) [3], and Mask R-CNN (Mask Region-CNN) [4], are constantly updated with higher accuracy for object detection, but these methods need to be supported by a powerful GPU. In this context, for

SSD (Single Shot MultiBox Detector) [5], DSSD (Deconvolutional Single Shot Detector) [6], the YOLO (You Only Look Once) [7–10] series, RetinaNet [11], and other one-stage object detection models, the detection speed of these algorithms can achieve the requirements of both still-picture and real-time video detection speed; therefore, they are widely used in industrial manufacturing.

The end face of bundled rebars is a kind of dense and small target, which has the characteristics of a single target category, concentrated target distribution, and low differentiation between targets in detection:

(1) The size of rebar ends varies. The diameter of the rebar end face varies greatly depending on the specification, and the diameter of the rebar cross-section in this paper is between 12 and 32 mm.
(2) The number of rebars per bundle varies greatly, ranging from 80 to 250 bars in this paper.
(3) The application scenario is complex. The distance, angle, and light of the shot are not fully controlled; the shape of the cut cross-section varies greatly; the color and brightness of the cross-section rust vary greatly and there are boundary adhesions; and some of the rebar cross-sections are scattered or indented, causing obscuration and other problems.
(4) High counting accuracy and precision are required. The production and user base of rebars are very large; if there are misdetections and omissions in the inspection results, they need to be manually identified from a large number of marked points. Only by achieving extremely high counting accuracy and precision can we guarantee the experience of acceptance personnel.

The key to the visual bundle counting technique depends on the ability to detect or segment the rebar cross-sections with the correct method. Xinman Z et al. [12] proposed an online rebar counting and automatic separation system by setting the region of interest, segmenting with template matching, and setting the mutation threshold by preprocessing and through edge detection of rebar images. Wu Y et al. [13] used concave point matching segmentation, K-level fault tolerant counting, and multiple segmentation to accurately count rebars with diameters between 8 mm and 20 mm. Zhao J et al. [14] used an improved Otsu algorithm, Sobel operator, and edge clustering algorithm to detect the number of rebars in a set stable environment.

Wang H et al. [15] combined Canny edge detection, circular Hough transform, and a CNN model to calculate the number of rebars; the highest detection accuracy of this model was 95.99%. Zhu Y et al. [16] proposed a CNN-based fusion model for rebar localization and segmentation. Li Y et al. [17] introduced a feature pyramid and complete cross-joint intersection over union (IoU) in YOLOv3 loss and focal loss to propose a YOLOv3-based rebar counting method in real time.

One essential task in the intelligent application is the detection of the number of rebars, which requires high detection accuracy because a rebar is expensive and there are many of them in actual use, and both misdetection and omission need to be identified manually in a large number of marked points; therefore, current rebars are still manually inventoried. It is proposed in the literature [12,15] that traditional image processing techniques can be used to preprocess the input image and combine the area and morphology for matching counts, but they do not perform well in dense or occluded situations. To further improve the efficiency and accuracy of rebar counting, a study on rebar identification using a neural network was proposed in the literature [18], which significantly improved the detection efficiency but was slightly inferior in accuracy. Zhu et al. [16] proposed a data enhancement method of SWDA (strong–weak distribution alignment), in which each rebar is first positioned for cropping, then a fully convolutional network (FCN) model is input to obtain the semantic mask, and, finally, three FCN models are used to perform high-quality semantic segmentation and combination of the rebar cross-section. Although better accuracy was achieved, this multi-stage network design model is larger and requires higher clarity of the input images.

Therefore, how to improve the detection speed without reducing the detection accuracy becomes an urgent problem.

In the literature [19], it is pointed out that the use of a residual network (ResNet) can effectively solve the problems of gradient dispersion, explosion, and network degradation of neural networks, which enables the input information to propagate further in the network and also improves the performance of the network. The use of pyramid pooling allows the fusion of high- and low-layer features, resulting in a network with better final semantic information features and spatial information features [20]. The literature [21] proposes an automatic detection and localization method using deep learning and migration. The literature [22] proposes a new and accurate counting model based on YOLOv3 to automatically and efficiently count dense steel pipes by using images.

Although the rebar detection counting technique is progressively mature, there is still a possible big boost. In this paper, we propose an improved YOLOv5-based rebar cross-section detection algorithm with the following contributions:

(1) In response to the current problem of scarce rebar datasets and complex practical application scenarios, a new rebar cross-section dataset is created. In order to improve the data labeling efficiency, we propose a new semi-automatic data labeling method.
(2) In order to make the anchor box parameters fit the rebar dataset better, the anchor boxes are re-clustered using the K-means++ algorithm, thus suppressing the redundancy of the anchor box caused by too many small targets and close rebar target sizes.
(3) The channel attention (CA) module was integrated into the backbone network and the attention weight of the feature map was adjusted to make the network more focused on the "interested" targets so as to enhance the feature extraction capability of the network.
(4) The new SD_IoU Loss is designed as the loss function of the bounding box, and the angular deviation and distance between the predicted box and the true box are considered to improve the accuracy of the box detection.
(5) To address the problem that small targets with scales smaller than 8×8 in the network cannot be detected, the detection loss rate of the network for small targets is reduced by fusing high-level semantic information with low-level features and redesigning a new detection layer.

2. Materials

Introduction of YOLO Series Algorithm

The YOLO algorithm is a deep learning-based regression detection method and also the leading algorithm in one-stage object detection. The core idea of YOLOv1 is to use the whole picture as the input of the network to regress the position and category of the box directly in the output layer, and the loss function uses the sum-squared error loss (SEL). The problem with using a single loss is that the IoU error for large objects and the IoU error for small objects contribute close to the loss value in network training. Therefore, for small objects, a very small IoU error can have a significant impact on network optimization, reducing the localization accuracy of object detection.

YOLOv2 has been improved in three aspects: more accurate prediction, faster speed, and more recognized objects, while maintaining the processing speed. A joint training algorithm is proposed, which mixes two datasets, uses the idea of a hierarchical classification of objects, and expands the detection dataset with the huge data from the classification dataset to train a better detection model.

YOLOv3 uses multi-scale prediction and is designed in such a way that it still uses the clustering algorithm to obtain 9 clustering centers, which are divided into 3 detection layers according to their sizes. Also, the backbone of the network uses Darknet-53, and YOLOv3 has a significant improvement in detection accuracy compared to YOLOv2.

YOLOv4 mainly replaces the Darknet-53 backbone with CSPDarknet-53 and adds the spatial pyramid module (SPP) and path aggregation network (PANet). Adding a dropblock module prevents overfitting while using the Mish and Leaky Relu activation functions.

In 2020, Ultralytics released the YOLOv5 algorithm, which improves the backbone of C3Darknet, making the algorithm more sensitive to feature information and thus better able to extract feature information. YOLOv5 performs one-stage object detection with high detection accuracy and fast inference, and the fastest detection rate is 140 frames/sec. On the other hand, the model weight of the YOLOv5 network is smaller, reduced by nearly 90% compared to YOLOv4; therefore, the YOLOv5 model is suitable for deployment to embedded devices for real-time detection. Meanwhile, YOLOv5 contains four size architectures: s, m, l, and x. The main difference between them is the different number of feature extraction modules and convolutional kernels of the network. The size of the model and model parameters increase sequentially in the four architectures. In summary, the advantages of YOLOv5 are high detection accuracy and a fast detection rate. Meanwhile, a large number of experiments show that YOLOv5 works well on many open-source datasets, but the performance in the smart rebar detection task still needs to be improved. Since the accuracy and real-time performance of rebar detection are crucial to the accuracy and efficiency of rebar recognition, especially for densely arranged targets in the dataset whose target boundary features are not obvious, small targets, and those with a complex overall image background, this study intends to improve the rebar target recognition network based on YOLOv5 and verify the effectiveness of the improved algorithm on the test set. Figure 1 shows the structure of the YOLOv5 network.

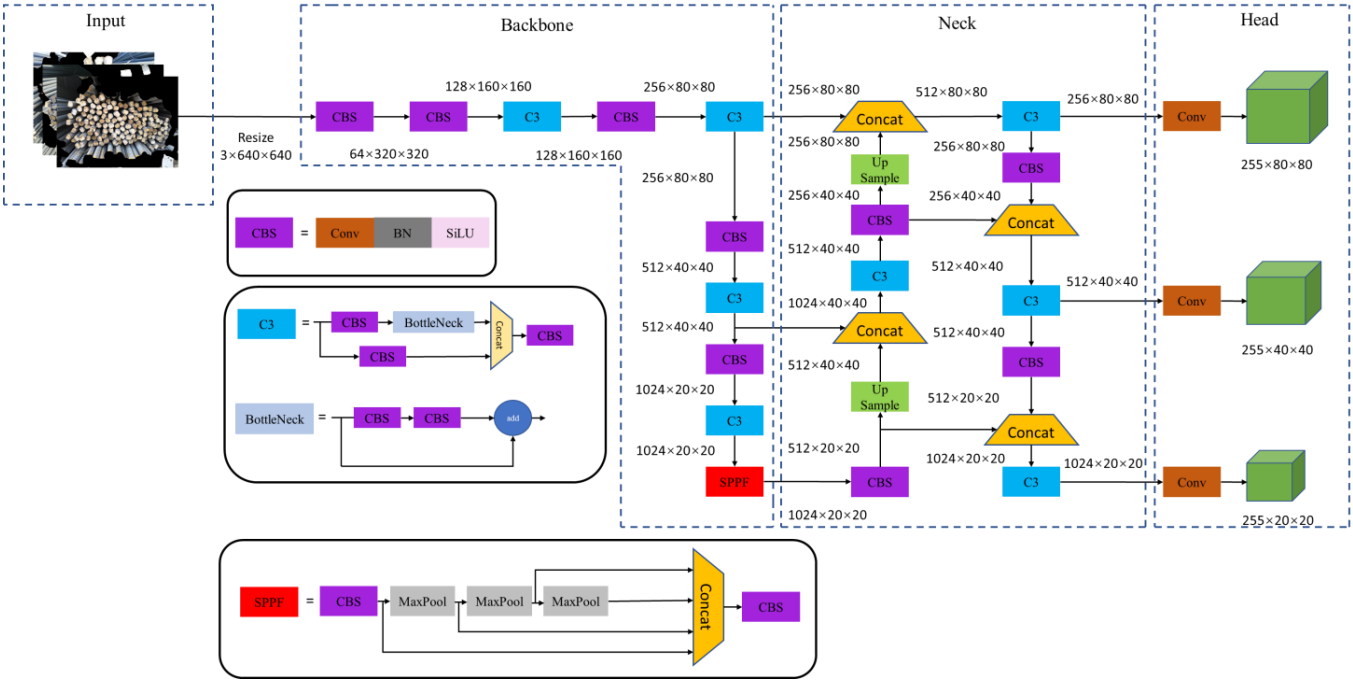


Figure 1. YOLOv5 network structure diagram.

The Conv module includes convolution, batch norm, and activation function operations, where the swish activation function is used in the new version of YOLOv5. The Bottleneck module first uses the convolution operation with a convolution kernel size of 1 × 1 to reduce the number of channels of the input feature map, then doubles the number of channels through 3 × 3 convolution, and finally performs feature fusion through a shortcut connection, with the number of channels of the fused feature map remaining unchanged; Figure 2 shows the Bottleneck module. The C3 module contains multiple Bottleneck modules that can adjust the depth of the network by setting the size of n. Figure 3 shows the C3 module. The SPPF module is improved based on SPP by first obtaining the number of input channels through a convolution module and then maximum pooling down-sampling with convolution kernel sizes of 5, 9, and 13, coating the down-sampled feature map with the feature map without down-sampling. The number of channels after

splicing is twice the original, and then the output is obtained through a convolutional module; Figure 4 shows the SPPF module.

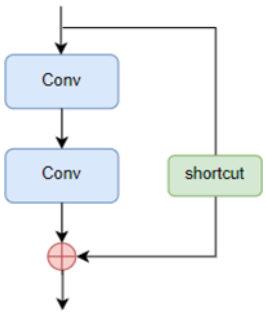


Figure 2. Bottleneck module.

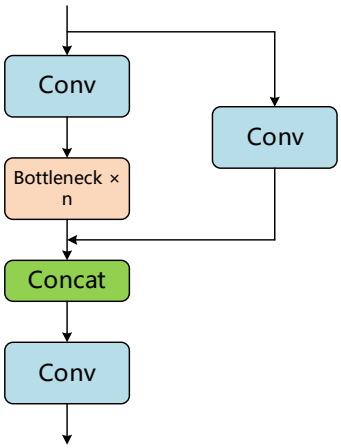


Figure 3. C3 module.

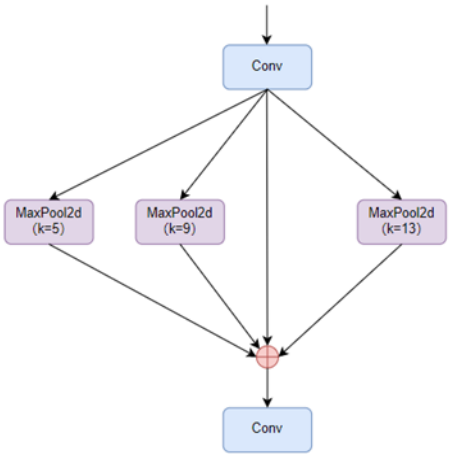


Figure 4. SPPF module.

3. Methods

3.1. Semi-Automated Labeling

Since the original data with multiple targets are high-resolution (2666 × 2000 and 2464 × 2056) images, the labeling operation is so time consuming for traditional supervised learning methods. Currently, image segmentation labeling tools, such as LabelImg, are very laborious and inefficient to use for labeling. Therefore, Castrejon [23] proposed a labeling tool, PolyRNN, and then Acuna [24] improved PolyRNN to PolyRNN++ as a vehicle semantic segmentation labeling tool for autonomous driving. PolyRNN and PolyRNN++ are mostly applicable to images with a small number of targets and data samples with

obvious features. However, usually, the rebar cross-sections are irregular, concave, and convex, which makes it difficult to find the convex points. Therefore, a new semi-automatic labeling method is proposed in this paper.

LabelImg is an open-source image annotation tool that can be used for classification and target detection. It is written in Python, uses Qt as its graphical interface, and is simple and easy to use. Its annotations are saved in PASCAL VOC format as XML files, which is the format used by ImageNet. In addition, it supports the COCO dataset format.

In order to improve the efficiency, we use the already labeled intelligent inventory rebar dataset, use YOLOv5 to train the preliminary model, use the preliminary model to detect the unlabeled self-built dataset, generate the XML labeling file, and then use labelImg software (version 1.8.6) to manually correct the labeling box. Figure 5 shows the semi-automatic data-labeling process flow chart.

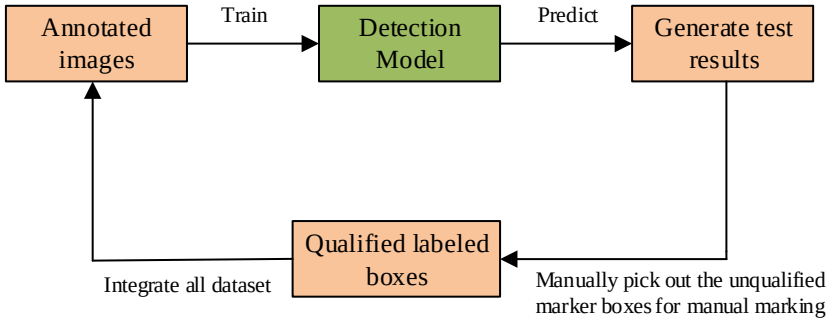


Figure 5. Semi-automatic labeling data processing flow.

3.2. Anchor Box Parameter Optimization

YOLOv5 adds the a priori information of the target box to the network prediction information. Before training the network, the information range of the target box, such as the range of box center coordinates, is obtained first, then the network is trained to learn this range of values.

The idea of adaptive box anchors (auto-learning bounding box anchors) is used in YOLOv5, and since the target size of the rebar dataset differs significantly from the COCO dataset [25], we need to recalculate the anchor box size. The specific function T for the calculation of K-means [26] clustering is shown in Equation (1).

$$T = \min \sum_{i=1}^{n} \sum_{j=1}^{k} [1 - \frac{box_i \cap cen_j}{box_i \cup cen_j}] \tag{1}$$

The evaluation clustering is expressed by the best possible recall (BPR), which is calculated as shown in Equation (2).

$$BPR = \frac{1}{n} \sum_{i=1}^{n} \max(\frac{box_i \cap cen_1}{box_i \cup cen_1}, \frac{box_i \cap cen_2}{box_i \cup cen_2}, \ldots, \frac{box_i \cap cen_k}{box_i \cup cen_k}) \tag{2}$$

where box_i is the area of the ith labeled box in the target, cen_j is the area of the jth clustering center, n is the number of detected targets, and k is the number of clustering centers.

The adaptive setting of K-means should conform to the anchor size of the dataset, which makes the training model easier to converge. But the clustering results are influenced by the selection of initial points, and the data need to be clustered several times until the clustering centers do not change; finally, the width and height of the anchor are derived. Therefore, we use the K-means++ [27] algorithm to perform a re-clustering analysis on the rebar cross-section dataset, which optimizes the problem that the algorithm is sensitive to the initial points, and find that the clustering results are better when the initial points are selected with a greater distance.

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

(1) Randomly select samples from the dataset as cluster centers.
(2) Calculate the distance $D(x)$ between the remaining samples in the dataset and the nearest clustering center.
(3) Calculate the probability P of whether the sample is the next clustering center, as shown in Equation (3). Select the clustering center based on the probability.

$$P = \frac{D(x)^2}{\sum_{x \in X} D(x)^2} \tag{3}$$

(4) Repeat steps (2) and (3) to select all clustering centers.
(5) Iterate using the original K-means method to obtain the final results based on the obtained clustering centers.

To match the re-clustered anchor with the multi-scale detection head of YOLOv5, the value of K is fixed to 16. We finally determined the size of the anchors after several iterations and optimizations. The final anchor boxes are assigned according to the different scales of detection layers, as shown in Table 1:

Table 1. Anchor box assignment table.

Feature Map	20 × 20	40 × 40	80 × 80	160 × 160
Receptive Field	**Larger**	**Large**	**Middle**	**Small**
Anchor boxes	(36, 32) (39, 39) (45, 45) (56, 54)	(23, 21) (26, 27) (31, 28) (31, 33)	(15, 14) (15, 16) (18, 17) (19, 20)	(7, 7) (9, 8) (8, 10) (11, 11)

3.3. Add CA Attention Module

The small and dense rebar targets make it difficult for the network to extract the feature information of the rebar cross-section. Meanwhile, the background in the image produces extensive redundancy when the network performs convolution, and the iterative accumulation swamps part of the target information, which leads to the low accuracy of model detection. To address the above problems, we add a coordinate attention (CA) mechanism that embeds location information into the backbone's channel [28]. CA can capture not only cross-channel information but also direction-aware and location-aware information, combining channel attention and spatial attention to make the model locate and identify targets more accurately. In addition, adding CA has almost no computational overhead to the network, and the model weights hardly increase. CA consists of two steps: the coordination of information embedding and attention generation. Figure 6 shows the structure of CA.

Coordinate information embedding: The channel relations and long-term dependencies are encoded by precise location information, and the channel attention is decomposed into two one-dimensional feature-encoding processes that aggregate features along two spatial directions. After decomposition, the channel can capture long-range dependencies along one spatial direction, while location information can be retained along the other spatial direction. The specific steps are to input a feature map X of dimension $C \times H \times W$ and encode the channel along the horizontal and vertical directions with two pooling kernels of dimensions $(H, 1)$ and $(1, W)$, respectively. The cth channel outputs feature maps of $z_c^h(h)$ and $z_c^w(w)$, respectively, as shown in Equations (4) and (5).

$$z_c^h(h) = \frac{1}{W} \sum_{0 \le i < W} x_c(h, i) \tag{4}$$

$$z_c^w(w) = \frac{1}{H} \sum_{0 \le j < H} x_c(j, w) \tag{5}$$

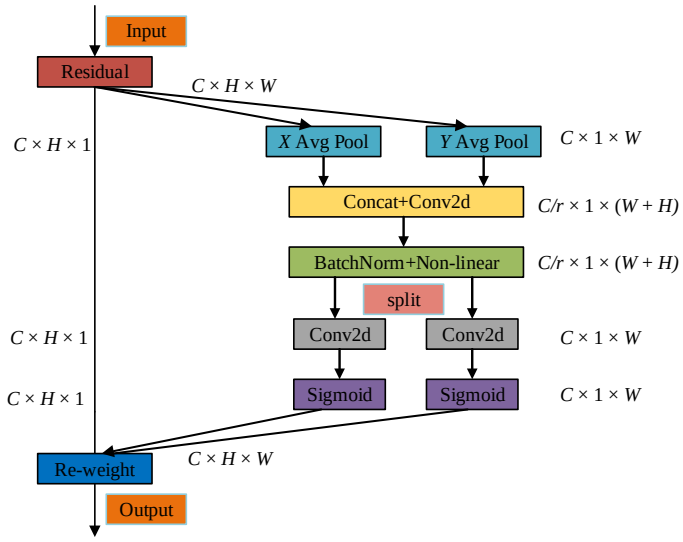


Figure 6. CA structure diagram.

Attention generation: The spatial information in the vertical and horizontal directions is encoded using batch normalization and nonlinear operations, and then the encoded information is segmented using convolution, while the channels of the attention map are adjusted to the number of channels of the input feature map. This is performed by cascading the feature maps z^h and z^w generated by the coordinate-information-embedding process and performing an F_1-transform using the shared 1×1 convolution operation, as shown in Equation (6).

$$f = \delta(F_1([z^h, z^w])) \tag{6}$$

where $f \in R^{C/r \times (H+W)}$ is an intermediate feature map encoding spatial information in the horizontal and vertical directions, r denotes the compression ratio, f is divided into separate tensors $f^h \in R^{C/r \times H}$ and $f^w \in R^{C/r \times W}$, and then two 1×1 convolutions F_h and F_w convert f^h and f^w to the same number of channels as the feature map X. The weights g^h (Equation (7)) and g^w (Equation (8)) are obtained by normalization and weighted fusion using the Sigmoid function.

$$g^h = \sigma(F_h(f^h)) \tag{7}$$

$$g^w = \sigma(F_w(f^w)) \tag{8}$$

Finally, the attention weights g^h and g^w are multiplied with the feature map X to obtain the weighted feature map Y output by the attention module (Equation (9)).

$$y_c(i,j) = x_c(i,j) \times g_c^h(i) \times g_c^w(j) \tag{9}$$

The resulting feature maps are encoded as direction-aware and position-sensitive attention maps, which can be applied complementarily to the input feature maps to enhance the representation of objects of "interest".

CA is added to the convolutional layer module of the network before the end of the feature extraction process, i.e., before the SPPF module. Therefore, the original feature extraction network is changed from 10 layers to 11 layers, which can effectively extract the information of the target features and reduce missed detection. Table 2 shows the structure of the backbone network after adding CA.

Table 2. Backbone network structure after joining CA.

Module Name	Module Number	Parameters	Output
Conv	1	3520	320 × 320
Conv	1	18,560	160 × 160
C3	3	18,816	160 × 160
Conv	1	73,984	80 × 80
C3	6	115,712	80 × 80
Conv	1	295,424	40 × 40
C3	9	625,152	40 × 40
Conv	1	1,180,672	20 × 20
C3	3	1,182,720	20 × 20
CA	**1**	**25,648**	**20 × 20**
SPPF	1	656,896	20 × 20

3.4. SD_IoU Bounding Box Loss Function

The loss function of YOLOv5 consists of classification loss, localization loss, and confidence loss, where the CIoU loss function is used for the bounding box loss of localization loss. The principle of CIoU loss [29] is shown in Figure 7 and Equations (10)–(13), where A is the prediction box, B is the true box, and C is the smallest outer rectangle of the two boxes A and B.

$$L_{CIoU} = 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v \tag{10}$$

$$\alpha = \frac{v}{(1 - IoU) + v} \tag{11}$$

$$IoU = \frac{A \cap B}{A \cup B} \tag{12}$$

$$v = \frac{4}{\pi^2}(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h})^2 \tag{13}$$

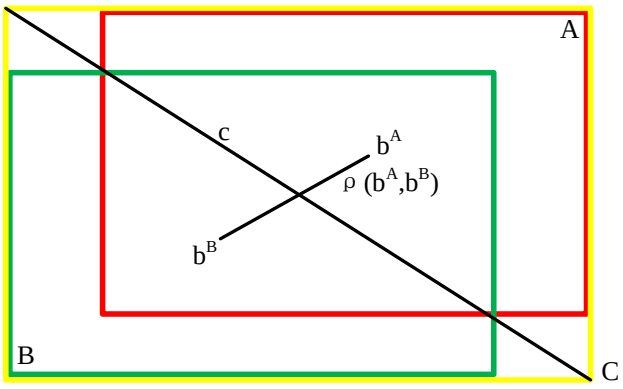


Figure 7. CIoU diagram.

In the above equation, IoU is the intersection ratio of the prediction box and the true box. ρ is the Euclidean distance between the center points of the prediction box and the true box; b and b^{gt} represent the center points of the prediction box and the true box, respectively; c represents the diagonal length of the smallest external rectangle of the prediction box and the true box; α is the weighting factor; w and w^{gt} represent the width of the prediction box and the true box, respectively; and h and h^{gt} represent the width of the prediction box and the true box width, respectively.

The boundary box loss of YOLOv5 uses CIoU loss because it considers the overlapping area, centroid distance, and aspect ratio of the boundary box regression, but the rebar cross-section targets are dense and it is difficult to optimize the error between the prediction

box and the true box during the network training. Inspired by SIoU Loss [30], this paper considers the angle deviation and distance relationship between the prediction box and the true box, and we design the SD_IoU Loss bounding box loss function, which reduces the degrees of freedom of the prediction box by adding angle and distance penalty costs to prevent it from "wandering around". In addition, the shapes of the rebar cross-sections studied in this paper are all circular and of the same shape, and the sizes of the anchor boxes are fixed using the K-means++ clustering algorithm before the network training. Therefore, the shape difference between the prediction box and the true box is not considered a factor affecting the regression of the bounding box, which can increase the training speed of the network.

The SD_IoU Loss is calculated as shown in Equation (14).

$$L_{SD_IoU} = 1 - IoU + \frac{\Gamma}{2} + \frac{\Delta}{2} \tag{14}$$

In the above equation, Γ is the angle loss and Δ is the distance loss.

The angular loss is calculated as shown in Equations (15)–(18).

$$\Gamma = \cos(2 \times (\arcsin(\frac{c_h}{\sigma}) - \frac{\pi}{4})) \tag{15}$$

$$x = \frac{c_h}{\sigma} = \sin(\alpha) \tag{16}$$

$$\sigma = \sqrt{(b_{c_x}^{gt} - b_{c_x})^2 + (b_{c_y}^{gt} - b_{c_y})^2} \tag{17}$$

$$c_h = \max(b_{c_y}^{gt}, b_{c_y}) - \min(b_{c_y}^{gt}, b_{c_y}) \tag{18}$$

In the above equation, σ is the distance between the center point of the prediction box and the true box; c_h is the height difference between the center point of the prediction box and the true box; $b_{c_x}^{gt}$ and $b_{c_y}^{gt}$ are the center coordinates of the true box; and b_{c_x} and b_{c_y} are the center coordinates of the prediction box. Figure 8 shows the schematic diagram of SIoU.

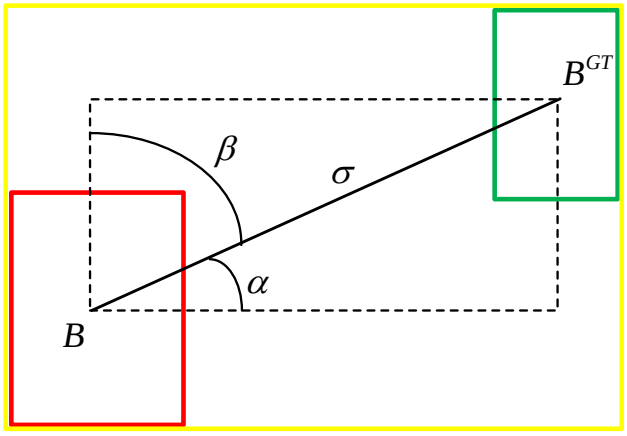


Figure 8. SIoU diagram.

In Figure 8, B and B^{GT} are the prediction box and the true box, respectively, and α is the angle between the line connecting the center point of the prediction box and the true box and the rectangle formed by the center point of the two boxes as the vertex on one side. When α is less than 45 degrees, α is minimized; otherwise, β is minimized. When α is minimized to 0, Γ is equal to 0, and at this time, the contribution of the angle loss Γ is zero. In order to avoid the angle loss disappearing to zero, we also consider the distance

factor between the prediction box and the true box. Therefore, we design a new distance loss function, calculated as shown in Equations (19)–(22).

$$\Delta = 2 - e^{-\gamma\rho_x} - e^{-\gamma\rho_y} \tag{19}$$

$$\rho_x = \left(\frac{b_{c_x}^{gt} - b_{c_x}}{d_w}\right)^2 \tag{20}$$

$$\rho_y = \left(\frac{b_{c_y}^{gt} - b_{c_y}}{d_h}\right)^2 \tag{21}$$

$$\gamma = 2 - \Gamma \tag{22}$$

In the above equations, d_w and d_h are the width and height of the smallest outer rectangle of the predicted and true boxes, respectively; ρ_x and ρ_y in Equations (20) and (21) are assigned to the distance loss Δ using powers of two, thus indicating that the distance loss is larger than the angle loss for the overall SD_IoU Loss contribution.

In this paper, by designing SD_IoU Loss, the angular deviation and distance relationship between the predicted box and the true box are taken into consideration, which greatly helps the convergence process and effect of training by improving the accuracy of the detected box. To achieve the purpose of optimizing and improving the accuracy of the model, we use the SD_IoU Loss bounding box function with better performance. Figure 9 shows the comparison of the bounding box loss curves before and after using SD_IoU Loss, where the horizontal coordinates are the number of iterations and the vertical coordinates are the loss values. It can be seen that the algorithm converges faster after using SD_IoU Loss, and the loss value is smaller than the original YOLOv5s; therefore, it proves that the SD_IoU Loss designed in this paper has better performance.

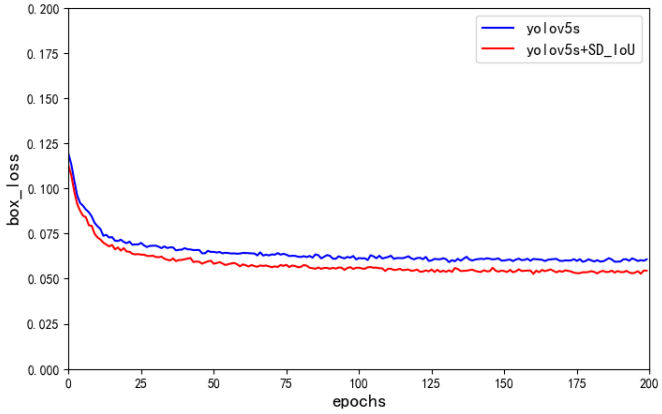


Figure 9. Training comparison graph before and after using SD_IoU.

3.5. Multi-Scale Detection

Because the low-level network has a smaller receptive field, although the resolution is high, the semantic information characterization ability is weak. The receptive field of the high-layer network is larger, and the network can perform feature extraction of the image from a more global perspective; thus, it can generate higher-level semantic information. How to fuse the geometric information of the low-level network with the semantic information of the high-level network is an important measure to improve detection performance.

YOLOv5 uses different levels of pyramidal layer feature maps for prediction and uses a multi-scale feature fusion method to extract features of different sizes from different network layers for prediction; this fusion method does not add additional computational effort [31]. The input image's size is 640 × 640, and the feature maps are processed using 32, 16, and 8 down-sampling to detect different target sizes from different network feature

layers. We counted a total of 1068 images from the training and validation sets, and a total of 155,946 rebars true (Ground True, GT) boxes were distributed during YOLOv5 detection, which contains 91.53% (142,738) of the rebar targets that can be classified as small targets. Using the K-means algorithm, we calculated that there are 1800 targets with sizes smaller than 3 pixels, which is not acceptable for a smart rebar detection task. Using the original YOLOv5 for detecting such targets is highly likely to cause missed detections; therefore, we added a new detection layer to cope with the targets with pixel values less than 3, thus reducing the model's missed detection for such small targets [32].

YOLOv5 has three detection layers, corresponding to three sets of a priori boxes at the same time. The detection layer corresponding to the backbone's eighth layer has a size of 20 × 20, and it can be used to detect targets above the 32 × 32 scale; the detection layer corresponding to the backbone's sixth layer has a size of 40 × 40 and can detect targets above the 16 × 16 scale; and the detection layer corresponding to the backbone's fourth layer has a size of 80 × 80, which corresponds to the smallest receptive field at this point and is used to detect targets above the 8 × 8 scale. However, small targets cannot be detected by the network when the target scale is smaller than 8 × 8. To address this problem, after the 18th layer of the network's Neck, the feature map continues to be up-sampled, and the up-sampling scales the smaller feature map to the same size as the larger feature map to satisfy the condition of feature fusion. Meanwhile, in layer 21, the acquired feature map of size 160 × 160 is fused with the layer 2 feature map of the backbone, and in layer 22 of the network, a new detection layer is added. The improved network has 4 detection layers, and the feature information of the target can be better extracted. This method, by transmitting the higher-level semantic features down to supplement the lower-level semantics, can obtain high-resolution and strong semantic features, which can facilitate the detection of small targets. The overall network structure is shown in Figure 10; the red dashed box part in the figure shows the structure of the newly added detection layer.

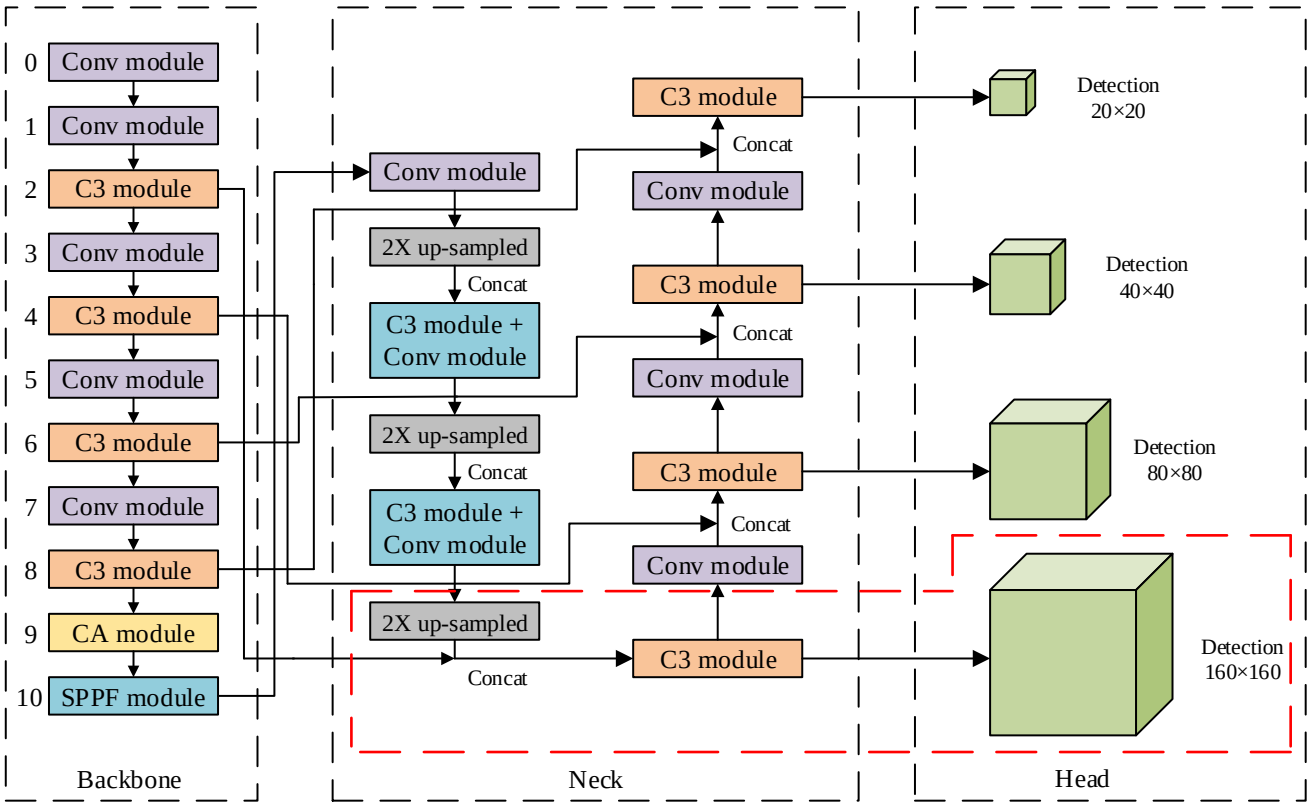


Figure 10. Overall Network Structure.

4. Experimental Results and Analysis

4.1. Training Environment and Parameters

The experimental environment is Ubuntu 20.04, IntelXeon(R) Silver 4214R CPU @ 2.40 GHz x48 processor, 128 GB RAM, NVIDIA Corporation TU102GL (Quadro RTX 8000) graphics card, and CUDA11.4.2 accelerated GPU. The Python version is 3.8.12, and the deep learning framework is Pytorch 1.9.1.

In this paper, we use YOLOv5s as the benchmark network, trained with the same parameters: an initial learning rate of lr0 = 0.01, lrf cyclic learning rate of 0.1, momentum parameter of 0.937, weight_decay decay rate of 0.0005, warmup_epochs of 3.0, warmup_momentum of 0.8, and warmup_decay of 0.8. Momentum is 0.8, warmup_bias_lr is 0.1, the batch size is 8, i.e., 8 images per input network, the number of iterations (epochs) is set to 200, and the network is trained using stochastic gradient descent SGD [33].

4.2. Dataset Acquisition and Analysis

The dataset used in this paper consists of two parts—one is the dataset labeled by the rebar AI recognition contest, with 250 labeled images and an image size of 2666 × 2000, and another part of the data is provided by Angang Iron and Steel Group. According to the needs of the production process, the acquisition equipment is a depth camera, and the image to be processed is the image obtained after mapping the depth point cloud data to speed up the processing speed; this dataset contains a total of 229 images of rebar cross-sections of 28 mm, 22 mm, and 14 mm, with an image size of 2464 × 2056.

To address the insufficient number of datasets, the dataset was expanded by using horizontal flip and translation. The expanded sample number was 1187, with 80–250 rebar targets in each image, and the total data had 173,274 rebar cross-sections. The images of the dataset are shown in Figure 11. Labeling was performed using LabelImg software. The 1187 images were divided in a ratio of 8:1:1 to obtain 949 training set images, 119 validation set images, and 119 test set images.



Figure 11. Example datasets.

Figure 12 counts the number of labeled boxes on each image. The horizontal coordinate is the number of labeled boxes on one image, and the vertical coordinate is the number of images. It can be seen that the number of rebars in each image varies widely, which poses a challenge to the rebar detection task. Second, we use Python to count the width and height of the rebar in the training images, and the corresponding scatter diagram is shown in Figure 13, which shows that both the width and height of the rebar are mainly distributed in the range of 50–250, with a relatively compact distribution range and few abnormal width and height values. It can also be seen that the targets of individual rebars can be regarded as small targets, which also fits the research theme of small targets in this paper.

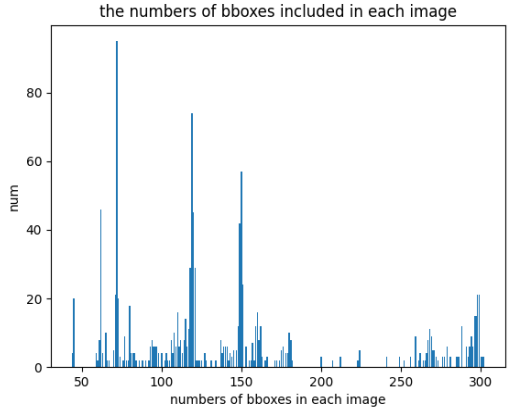


Figure 12. Number of labeled boxes per image.

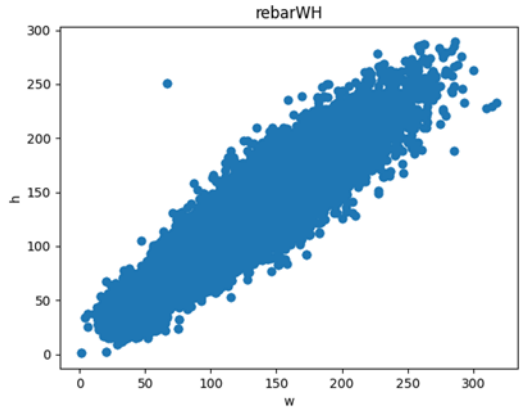


Figure 13. Scatterplot of marker box width and height.

4.3. Evaluation Indicators

In this paper, we use evaluation metrics such as precision (P), recall (R), mean average precision (mAP), mAP@0.5, mAP@0.5:0.95, and frames per second (FPS) parameters and weight. The above metrics are calculated as shown in Equations (23)–(26).

$$Pression = \frac{TP}{TP + FP} \tag{23}$$

$$Recall = \frac{TP}{TP + FN} \tag{24}$$

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AceP(i) \tag{25}$$

$$FPS = \frac{1}{time} \tag{26}$$

Precision represents the degree of prediction accuracy in the positive sample results. The denominator is the predicted positive class, and precision is proposed to make the existing prediction results of the model as error-free as possible.

Recall is for the original sample, and its meaning is the probability of being predicted as a positive sample out of an actual positive sample.

mAP@0.5 represents the average detection accuracy of all target classes at an IoU threshold of 0.5, where mAP@0.5 reflects the combined classification ability of the algorithm for different classes of targets. *mAP@0.5:0.95* represents the average of the detection accuracy under all 10 IoU thresholds calculated, with a step size of 0.05 and an IoU threshold from 0.5 to 0.95. In general, the higher the IoU threshold, the higher the requirement for

the regression capability of the model. The higher the detection index at high thresholds, the more closely the detection results of the model fit the actual target.

FPS is the number of images that can be detected per second, which is used to measure the real-time performance of the model.

mAP is the average detection accuracy of all categories.

4.4. Ablation Experiments

In order to verify the effectiveness of the improved method, five sets of experiments are designed to analyze the results of different improvement methods; the experiments use the same training parameters, and the comparative results of each method are shown in Table 3. "√" in the table indicates that the corresponding policy in the model is used.

Table 3. Comparison of ablation experimental results.

Model	K-Maens++	CA	SD_IoU	Multi-Scale	Precision	Recall	*mAP*@0.5	mAP@0.5:0.95	Parameters	Weight
YOLOv5s					95.2%	79.5%	88.9%	58.9%	7,022,326	14.4 MB
Improvement1	√				94.9%	79.6%	89.4%	59.6%	7,022,326	14.4 MB
Improvement2	√	√			94.7%	79.8%	89.8%	59.1%	7,211,814	14.8 MB
Improvement3	√	√	√		90.4%	91.4%	94.2%	62.3%	7,211,814	14.8 MB
Improvement4	√	√	√	√	96.3%	96.6%	97.9%	65.2%	7,862,352	17.0 MB

From Table 3, it can be seen from the experimental results of improvement 1 that after re-clustering using the K-means++ algorithm, all indicators improved slightly; from the experimental results of improvement 2, it can be seen that the attention module brought a slight performance improvement compared with the original YOLOv5s on various indicators, among which mAP@0.5 increased by 0.9%. Compared with the experimental results of improvement 2 and improvement 3, it can be seen that by using the SD_IoU loss function, angle and distance loss are introduced. Compared with improvement 2, all indicators except accuracy are improved; mAP@0.5 increased by 4.4% and the recall rate increased by 11.6%. Compared with the experimental results of improvement 3 and improvement 4, the new detection layer is added to the network, and the indicators of the algorithm are also improved, among which mAP@0.5 reaches 97.9%, which is 3.7% higher than that of improvement 2, which greatly reduces the missed detection rate of small targets. From the five sets of ablation experiments, it can be seen that the proposed RebarNet can effectively improve the effect of rebar detection by using K-means++ clustering, adding the CA, using a new SD_IoU loss, and adding a new detection layer.

4.5. Training Curve Comparison Graph

According to the set parameters for training the rebar dataset, Figure 14 shows the comparison curves of accuracy, recall, average precision (mAP_0.5 and mAP_0.5:0.95), bounding box loss, confidence loss of YOLOv5s, and the improved algorithm during the training process. In the figure, "yolov5s" is the training curve of YOLOv5s, "yolov5s+c" indicates the addition of CA attention, "yolov5s+c+s" denotes adding CA attention and using SD_IoU loss, and "yolov5s+c+s+m" denotes adding CA attention, using SD_IoU and loss, using the multi-scale method on the network.

From Figure 14a–d, the values of P, R, mAP0.5, and mAP0.5:0.95 of the improved algorithm are higher than those of the original YOLOv5s. From Figure 14e,f, the confidence loss values of the improved algorithm on the training and validation sets are always smaller than those of the YOLOv5s, which indicates that the prediction box of the improved algorithm RebarNet is closer to the true box.

The F_1 score considers both the precision and the recall, allowing both to be maximized and a balance to be achieved. Figure 15 shows the graph of F_1. The horizontal coordinate is the value of the confidence, and the vertical coordinate is the value of F_1.

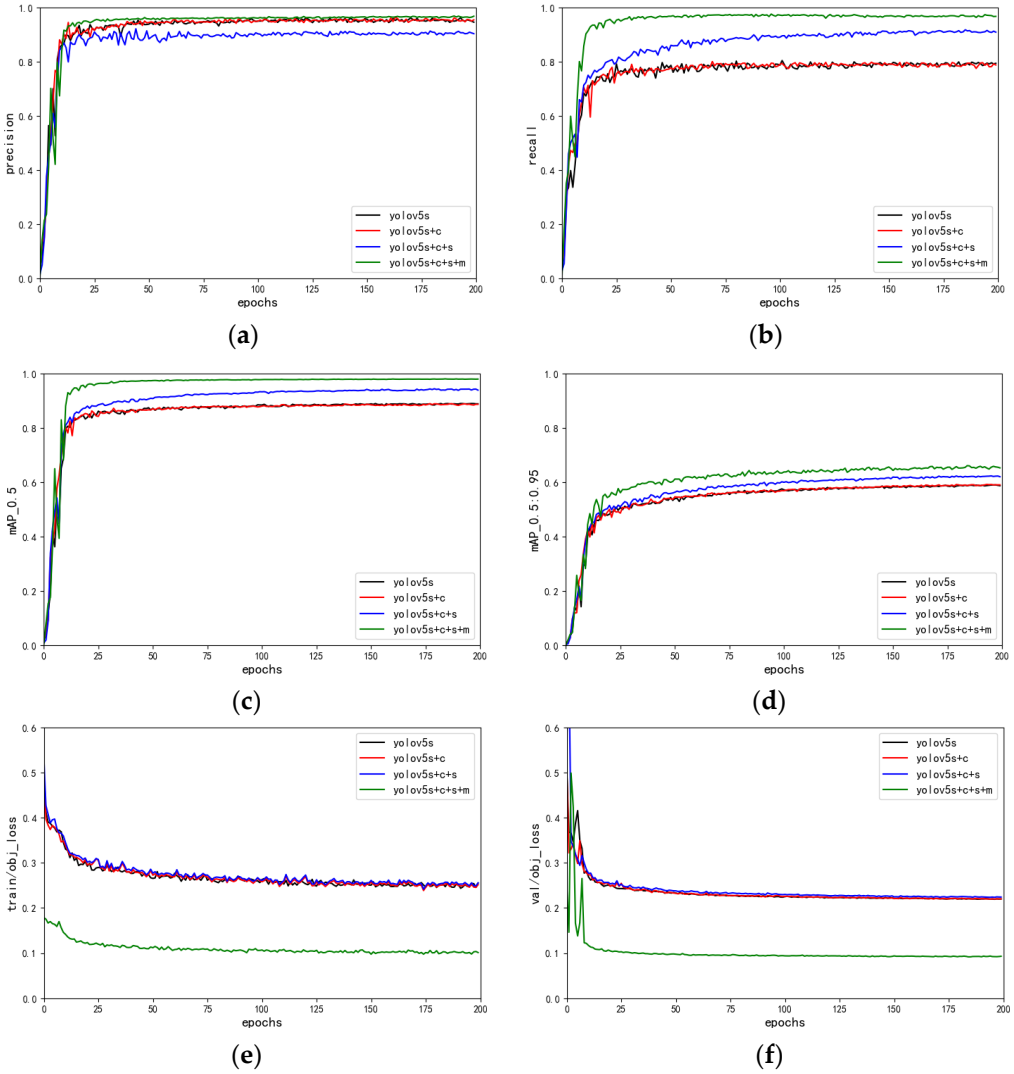


Figure 14. Training curve comparison graph. (**a**) Precision, (**b**) Recall, (**c**) mAP_0.5, (**d**) mAP_0.5:0.95, (**e**) train/obj_loss, and (**f**) val/obj_loss.

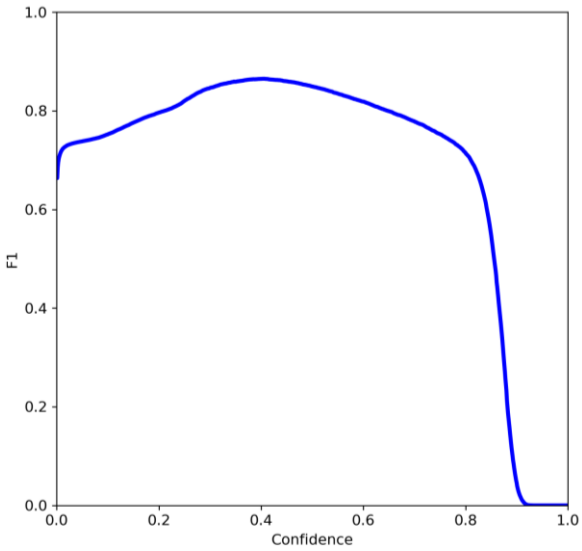


Figure 15. F_1 graph.

4.6. Comparison Experiments

In order to further verify the overall performance of the improved algorithm, we conducted a total of nine sets of comparison experiments, using precision (Precision), average precision (mAP), frames per second (FPS), and model weights as evaluation metrics. To ensure the fairness of the experimental process, all models were set with a batch size of 8 and epoch of 200, and other detailed parameters are the same. The comparison results of different indexes are shown in Table 4 below.

Table 4. Comparison of experimental results of different detection algorithms.

Algorithm	Precision	Recall	mAP0.5	Parameters	Weight	FPS
Faster R-CNN	87.8%	70.3%	89.8%	91,050,000	182.1 MB	45
SSD	66.7%	50.9%	84.9%	45,300,000	90.6 MB	37
RetinaNet	65.8%	52.4%	71.5%	59,000,000	118 MB	49
CenterNet	56.6%	48.7%	72.1%	77,000,000	154 MB	56
CornerNet	62.5%	50.8%	71.7%	81,500,000	163 MB	63
YOLOv3	95.7%	84.5%	86.2%	9,500,000	19.0 MB	81
YOLOv4	76.6%	68.4%	90.3%	244,200,000	488.4 MB	85
YOLOv5s	95.2%	79.5%	88.9%	7,022,326	14.4 MB	92
RebarNet	96.3%	96.6%	97.9%	7,862,352	17.0 MB	89

As can be seen from Table 4, RebarNet outperforms other algorithms in terms of accuracy and recall in terms of the mAP0.5 of detection results. mAP0.5 of RebarNet is 97.9%; compared to other mainstream target detection algorithms including Faster R-CNN, SSD, RetinaNet, CenterNet, CornerNet YOLOv3, YOLOv4, and YOLOv5s, mAP0.5 improved by 8.1%, 13%, 26.4%, 25.8%, 26.2%, 11.7%, 7.6%, and 9%, respectively. In terms of model weight size, the model weight obtained by training with RebarNet is 17.0 MB, which is 1/11, 1/5, 1/7, 1/9, 1/10, 7/8, and 1/29 of the model weights of Faster R-CNN, SSD, RetinaNet, CenterNet, CornerNet, YOLOv3, and YOLOv4, respectively. Although the model weights are slightly increased compared with the previous YOLOv5s, mAP0.5 is significantly improved by 9%; in terms of frames per second transmission, the FPS of RebarNet is 89 frames, which is slightly inferior to YOLOv5s in terms of detection rate but still satisfies the real-time performance of the rebar detection task. In summary, RebarNet achieves a balance of detection accuracy and detection speed, and its overall performance is better than that of other object detection algorithms, which can meet the basic requirements of rebar cross-section detection.

4.7. Comparison of Test Results

In order to demonstrate more intuitively the performance of the RebarNet, the trained model is tested on the test set (confidence parameter conf = 0.45, non-maximum suppression threshold IOU = 0.6), the detection results of both are compared, and results are shown in Figure 16. In Figure 16, the green box is the effect graph of the original YOLOv5s detection, the purple box is the detection effect graph of RebarNet, and the blue circles marked are missed by the original YOLOv5s and accurately detected by the corresponding RebarNet. In Figure 16a,b, the rebar arrangement is denser, and the original YOLOv5s missed rebar detection, while RebarNet detected all of the rebars. Figure 16c,d show the comparison of YOLOv5s and RebarNet detection in the case of uneven, adhering, small rebar cross-sections in the bundled rebar. RebarNet still performs well under non-uniform illumination conditions as shown in Figure 16e,f. When the shooting angle of the image deviates, the detection performance of the original YOLOv5s also has occurrences of missed detection, while RebarNet performs excellently, as shown in Figure 16g,h. On the images with smaller rebar diameters, the original YOLOv5s also showed missed detection, while RebarNet was able to correctly detect the location of the rebars, as shown in Figure 16i. In summary,

RebarNet can accurately detect all rebars in different detection environments, which can basically achieve the requirement of rebar detection in a smart detection environment.

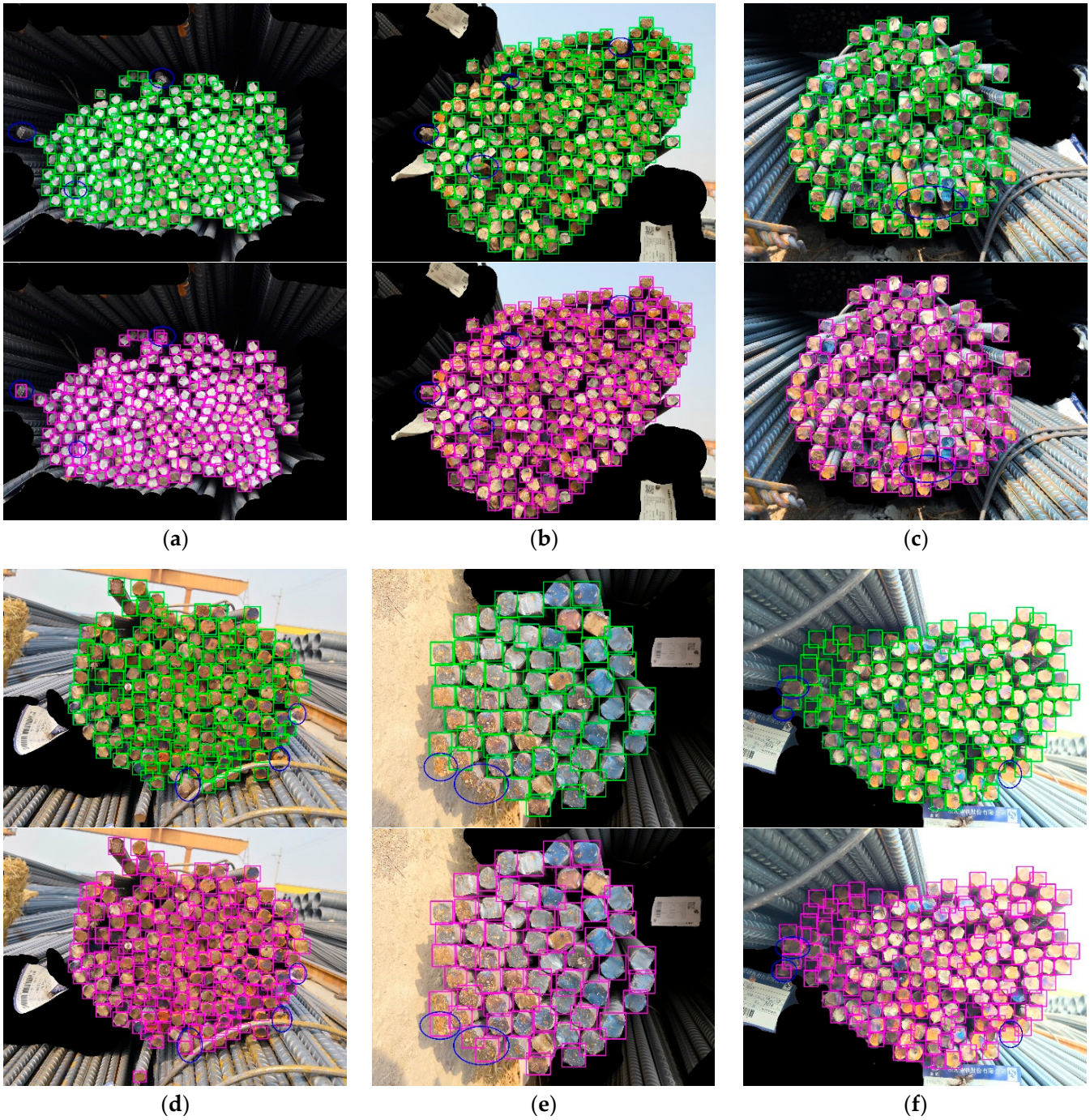
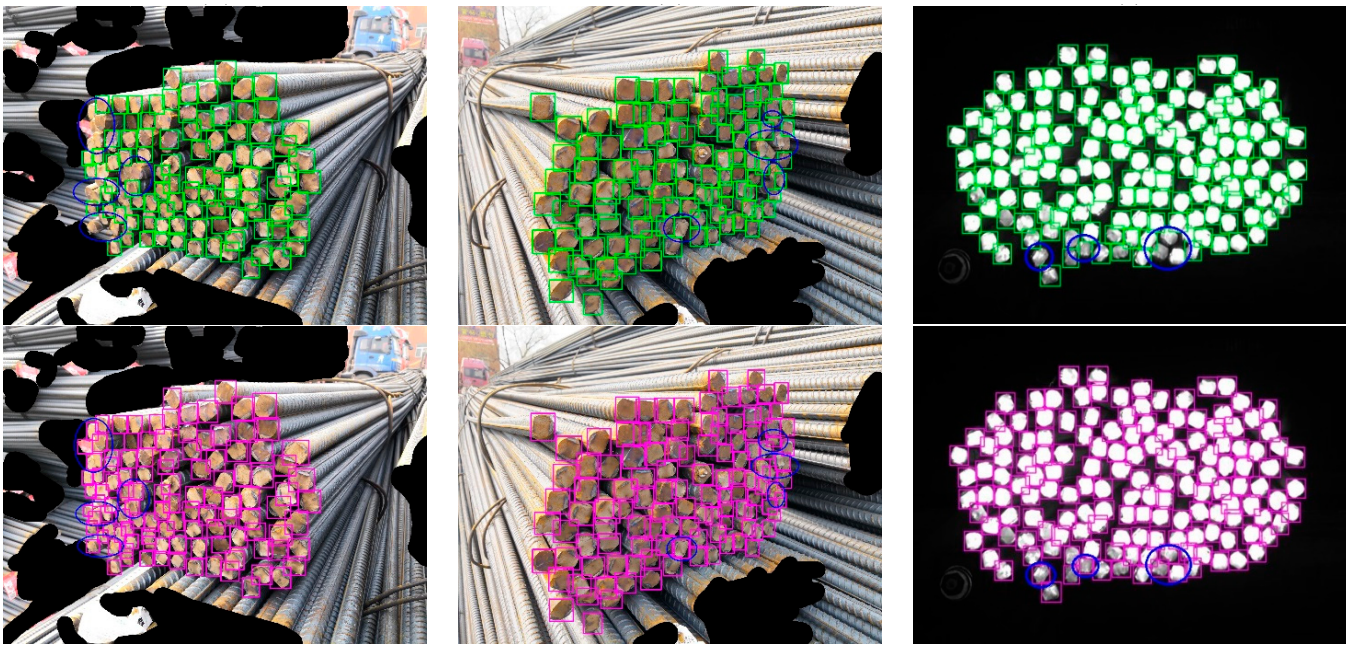


Figure 16. *Cont.*



(**g**) (**h**) (**i**)

Figure 16. Comparison of results before and after algorithm improvement. (**a**,**b**) Detection comparison of results before and after algorithm improvement in the rebar dense situations. (**c**,**d**) Detection comparison of results before and after algorithm improvement in the case of rebar section adhesion and uneven levels. (**e**,**f**) Detection comparison of results before and after algorithm improvement in the non-uniform lighting situations. (**g**,**h**) Detection comparison of results before and after algorithm improvement in the camera angle deviation case. (**i**) Detection comparison of results before and after algorithm improvement in the smaller rebar diameters.

5. Conclusions

Rebars have an irreplaceable role in construction, manufacturing, and road building. The final process in the rebar production process is the separation of rebars and bundling them into individual units. For commercial management reasons, the quantity of rebars must be counted and accounted for before they leave the factory, before they arrive in the market, before they leave the market, and when they arrive at the building construction site. Rebar counting is mostly performed manually, but this method is highly error-prone. Thus, it is urgent to research an object detection method for rebar counting, which also plays an important role in reducing workers' work intensity, improving the efficiency and accuracy of rebar counting.

In the paper, the current development status of mainstream object detection algorithms based on deep learning is sorted out, and then the idea of using computer vision technology to solve the problem of rebar counting is proposed. According to the demand for static counting of rebar targets in pictures, we propose a rebar detection network RebarNet with an embedded attention mechanism, which includes the K-means++ algorithm, CA module, SD_IoU Loss, and multi-scale detection. Through various experimental analyses, the proposed RebarNet can better cope with the task of rebar counting and detection at smart construction sites. Compared with the mainstream object detection models, the accuracy and real-time performance of RebarNet detection are better, and the balance of detection accuracy and detection rate is basically achieved.

Although deep-learning-based rebar detection has achieved certain results, there are still some problems to be solved, such as the type of data being not enough, and the network back-end computation being large. In addition, efficient and low-consumption detection algorithms can be better applied to smart devices, and further improvements to the network structure will be considered in the future to lighten the deep network and

improve the detection speed on the basis of ensuring accuracy, making it more suitable for the requirements of the lightweight deployment model of mobile devices.

Author Contributions: Conceptualization, G.Z.; methodology, G.Z.; software, Y.Z.; validation, G.Z., B.L. and Y.Z.; formal analysis, G.Z.; investigation, Y.Z.; resources, B.L.; data curation, B.L.; writing—original draft preparation, G.Z.; writing—review and editing, G.Z.; visualization, Y.Z.; supervision, Y.Z.; project administration, Y.Z.; funding acquisition, B.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (41773024); the 2022 Henan Provincial Key R&D and Promotion Special Project (Science and Technology Research) (222102210131); the Henan University National Student Innovation and Entrepreneurship Training Program (202110460078); and the Henan Provincial Higher Education Key Research Project Funding Program (21A520016); and the Doctoral Foundation of Henan Polytechnic University under Grant B2016-40.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available from the corresponding author. Send a request to the corresponding author's email, and then you will receive the data.

Acknowledgments: We thank the Artificial Intelligence Laboratory, Henan Polytechnic University, for supporting our work and providing funding to us. We also thank all of the authors of the primary studies includes in this article. We also wish to thank anonymous reviewers for their kind advice.

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

References

1. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
2. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
3. Ren, S.Q.; He, K.M.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
4. He, K.M.; Gkioxari, G.; DOLLÁR, P.; Girshick, R. Mask R-CNN. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
5. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. SSD: Single shot multibox detector. In Proceedings of the 2016 European Conference on Computer Vision, LNCS 9905, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Cham, Switzerland, 2016; pp. 21–37.
6. Fu, C.Y.; Liu, W.; Ranga, A.; Tyagi, A.; Berg, A.C. DSSD: Deconvolutional single shot detector. *arXiv* **2017**, arXiv:1701.06659.
7. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
8. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525.
9. Redmon, J.; Farhadi, A. YOLOv3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
10. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. YOLOv4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
11. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollar, P. Focal loss for dense object detection. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2999–3007.
12. Zhang, X.M.; Ma, M.; He, T.T.; Xu, X. Steel bars counting method based on image and video processing. In Proceedings of the 2017 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), Xiamen, China, 6–9 November 2017; pp. 304–309.
13. Wu, Y.; Zhou, X.; Zhang, Y. Steel bars counting and splitting method based on machine vision. In Proceedings of the 2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), Shenyang, China, 8–12 June 2015; pp. 420–425.
14. Zhao, J.; Xia, X.; Wang, H.; Kong, S. Design of real-time steel bars recognition system based on machine vision. In Proceedings of the 2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, China, 27–28 August 2016; Volume 1, pp. 505–509.
15. Wang, H.; Polden, J.; Jirgens, J.; Yu, Z.; Pan, Z. Automatic rebar counting using image processing and machine learning. In Proceedings of the 2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER), Suzhou, China, 29 July–2 August 2019; pp. 900–904.

16. Zhu, Y.; Tang, C.; Liu, H.; Huang, P. End-face localization and segmentation of steel bar based on convolution neural network. *IEEE Access* **2020**, *8*, 74679–74690. [CrossRef]
17. Li, Y.; Lu, Y.; Chen, J. A deep learning approach for real-time rebar counting on the construction site based on YOLOv3 detector. *Autom. Constr.* **2021**, *124*, 103602. [CrossRef]
18. Qu, F.; Li, C.M.; Peng, K.; Qu, C.; Lin, C. Research on detection andidentification of dense rebar based on lightweight network. In Proceedings of the 2020 International Conference of Pioneering Computer Scientists, Engineers and Educators, CCIS 1257, Taiyuan, China, 18–21 September 2020; Springer: Singapore, 2020; pp. 440–446.
19. He, K.M.; Zhang, X.Y.; Ren, S.Q.; Sun, J. Deep residual learning for image recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
20. Neubeck, A.; Van Gool, L. Efficient non-maximum suppression. In Proceedings of the 18th International Conference on Pattern Recognition, Hong Kong, China, 20–24 August 2006; pp. 850–855.
21. Liu, H.; Lin, C.; Cui, J.; Fan, L.; Xie, X.; Spencer, B.F. Detection and localization of rebar in concrete by deep learning using ground penetrating radar. *Autom. Constr.* **2020**, *118*, 103279. [CrossRef]
22. Li, Y.; Chen, J. Computer Vision–Based Counting Model for Dense Steel Pipe on Construction Sites. *J. Constr. Eng. Manag.* **2022**, *148*, 04021178. [CrossRef]
23. Castrejon, L.; Kundu, K.; Urtasun, R.; Fidler, S. Annotating object instances with a polygon-rnn. In Proceedings of the IEEE conference on computer vision and pattern recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5230–5238.
24. Acuna, D.; Ling, H.; Kar, A.; Fidler, S. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 859–868.
25. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Proceedings, Part V 13. Springer International Publishing: Cham, Switzerland, 2014; pp. 740–755.
26. Likas, A.; Vlassis, N.; Jverbeek, J. The global k-means clustering algorithm. *Pattern Recognit.* **2003**, *36*, 451–461. [CrossRef]
27. Arthur, D.; Vassilvitskii, S. K-Means++: The Advantages of Careful Seeding. In Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, Philadelphia, PA, USA, 7–9 January 2007.
28. Hou, Q.; Zhou, D.; Feng, J. Coordinate attention for efficient mobile network design. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 13713–13722.
29. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. Distance-IoU loss: Faster and better learning for bounding box regression. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 12993–13000.
30. Gevorgyan, Z. SIoU Loss: More Powerful Learning for Bounding Box Regression. *arXiv* **2022**, arXiv:2205.12740.
31. Liang, Z.; Shao, J.; Zhang, D.; Gao, L. Small object detection using deep feature pyramid networks. In Proceedings of the Pacific Rim Conference on Multimedia, Hefei, China, 21–22 September 2018; Springer: Cham, Switzerland, 2018; pp. 554–564.
32. Deng, Z.; Sun, H.; Zhou, S.; Zhao, J.; Lei, L.; Zou, H. Multi-scale object detection in remote sensing imagery with convolutional neural networks. *ISPRS J. Photogramm. Remote Sens.* **2018**, *145*, 3–22. [CrossRef]
33. Ketkar, N. Stochastic gradient descent. In *Deep Learning with Python*; Apress: Berkeley, CA, USA, 2017; pp. 113–132.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.