electronics

MDPI

Article

Algorithm of Computer Mainboard Quality Detection for Real-Time Based on QD-YOLO

Guangming Tu [1], Jiaohua Qin [1,*] and Neal N. Xiong [2]

[1] College of Computer & Information Engineering, Central South University of Forestry and Technology, Changsha 410004, China; 20192604@csuft.edu.cn
[2] Department of Computer Science and Mathematics, Sul Ross State University, Alpine, TX 79832, USA; neal.xiong@sulross.edu
* Correspondence: qinjiaohua@csuft.edu.cn; Tel.: +86-0731-8562-3008

Abstract: Automated industrial quality detection (QD) boosts quality-detection efficiency and reduces costs. However, current quality-detection algorithms have drawbacks such as low efficiency, easily missed detections, and false detections. We propose QD-YOLO, an attention-based method to enhance quality-detection efficiency on computer mainboards. Firstly, we propose a composite attention module for the network's backbone to highlight appropriate feature channels and improve the feature fusion structure, allowing the network to concentrate on the crucial information in the feature map. Secondly, we employ the Meta-ACON activation function to dynamically learn whether the activation function is linear or non-linear for various input data and adapt it to varied input scenarios with varying linearity. Additionally, we adopt Ghost convolution instead of ordinary convolution, using linear operations as possible to reduce the number of parameters and speed up detection. Experimental results show that our method can achieve improved real-time performance and accuracy on the self-created mainboard quality defect dataset, with a mean average precision (mAP) of 98.85% and a detection speed of 31.25 Frames Per Second (FPS). Compared with the original YOLOv5s model, the improved method improves mAP@0.5 by 2.09% and detection speed by 2.67 FPS.

Keywords: deep learning; YOLO; composite attention; computer mainboard quality detection; real-time detection

Citation: Tu, G.; Qin, J.; Xiong, N.N. Algorithm of Computer Mainboard Quality Detection for Real-Time Based on QD-YOLO. *Electronics* **2022**, *11*, 2424. https://doi.org/10.3390/electronics11152424

Academic Editor: Valentina E. Balas

Received: 8 July 2022
Accepted: 1 August 2022
Published: 3 August 2022

1. Introduction

The electronics industry and its products evolved quickly with the continuous development of industrial technology and industrial level. Computer mainboards have the characteristics of high density, multiple layers, and complicated assembly [1]. For this reason, people are always looking for new mainboard assembly techniques [2], and the assembly of computer mainboards is moving toward smart controllable functionalities [3]. Following the assembly process, quality detection (QD) can aid in the reduction in assembly-related losses. The main challenge in computer mainboard quality-detection tasks is that the defect area is small, and the features are hard to capture. The traditional object-detection algorithm has a high missed-detection rate in these scenarios. Pursuing high precision and employing detection algorithms with complex network structures will reduce real-time performance. Therefore, the computer mainboard assembly industry urgently needs a new solution to improve the detection rate of defects while reducing the time required for quality detection. In this situation, the YOLO algorithm [4] is a fast and accurate one-stage detection method that can effectively be applied to quality-detection scenarios [5].

Currently, the quality-detection methods for computer mainboards can be broadly divided into three categories. The first is the manual detection method: the inspector manually checks the product's quality, and their experience determines the task's success. Therefore, this method is unstable and inefficient.

With developments in industrial technology, quality detection based on electrical characteristics has become more accurate and faster. Yotsuyanagi [6] proposed an electrical test circuit to detect Printed Circuit Board (PCB) defects based on electrical performance. However, because the method requires contact operations, it may result in secondary PCB damage.

Currently, computer vision technology is widely used in various fields, such as image segmentation [7,8], object detection [9], image classification [10], information hiding [11], and industrial quality detection. Zhou et al. designed the AT-YOLO helmet-wearing detection model [12], in which mean average precision (mAP) reaches 96.5%, providing a new solution for helmet-wearing detection. Adibhatla VA et al. used Convolutional Neural Networks (CNNs) for PCB defect detection [13], reducing false detections and increasing productivity. Zhang et al. used CNNs to detect and classify defects on metal surfaces in complex industrial scenarios [14]. Computer-vision-based defect detection has surpassed manual detection in accuracy and speed [15]. However, current computer-vision-based defect detection algorithms can only extract a few features [16]. A two-stage image detector uses candidate frame extract objects. This method first creates image candidate frames and then filters the candidate frames using a classifier and a regressor to find the target. This method is slow and does not satisfy real-time detection requirements.

The above models have improved structures based on their application scenarios, allowing them to detect image targets more accurately. There are few models that have improved mainboard defect detection. Therefore, we propose QD-YOLO for detecting computer mainboards to improve network performance and real-time detection. This paper's main contributions are as follows:

(1) We constructed a dataset for the quality detection of computer mainboards using multiple techniques. We augmented the mainboard quality-detection dataset with selfie data and data augmentation techniques, such as cropping and random rotation, to increase the sample size and diversify the scenes and categories, which avoids overfitting by insufficient data.
(2) We designed a composite attention mechanism to capture both location information and long-range dependencies while focusing on network feature fusion. The composite attention mechanism enables the model to pay attention to both spatial information and channels of feature maps. This attention mechanism enables more accurate detection of small objects in images.
(3) Ghost convolution and Meta-ACON improve model robustness and reduce parameters. The Meta-ACON Activation function adapts network linearity to input conditions, boosting model robustness. Ghost convolution utilizes linear operations to create similar feature maps without fully connected layers, which speeds model detection by reducing model calculation and parameters.

This paper is organized as follows: Section 2 presents research related to object-detection algorithms, Section 3 describes the method proposed in this paper, Section 4 describes the experiments, Section 5 discusses our shortcomings and related research, and Section 6 summarizes the work in this paper.

2. Related Work

Deep-learning-based computer mainboard quality detection aims for automatic optical detection by rapidly and precisely identifying image defects. The object-detection technology involved in defect detection is an important component of computer vision. Most of the existing item detection algorithms are one-stage or two-stage.

YOLO is a classical one-stage object-detection algorithm that reconstructs object-detection as a single regression problem, mapping from the pixel space of the input image to the detection result. The YOLO algorithm performs real-time positioning and classification of the detection target with a delay time of less than 25 milliseconds and a fast detection rate that satisfies the requirements for real-time detection. However, YOLOv1 has significant localization errors and cannot detect nearby objects or small groups. The authors optimized

YOLOv1 and proposed YOLOv2 [17] with the Anchor mechanism to address localization errors and recall. K-means improves the algorithm's recall by clustering the training set. Fine-grained image features help detect small targets. Batch normalization [18] achieves a more efficient and rapid convergence.

YOLOv3 [19] increased network depth while employing darknet53 as the backbone. The inner residual block [20] uses skip connections while mitigating the gradient vanishing issue caused by increasing network depth. All at the same, the multi-scale prediction is realized using the FPN [21] network structure, which improves the detection accuracy and speed while reducing the background error rate.

YOLOv4 [22] modified the backbone to CSPDarknet53 and used PANet [23] as the neck. PANet enhances defect feature extraction while CSPNet [24] improves gradient combination information and reduces computation.

YOLOv5 enhanced input data with Mosaic to improve small-target detection. Using adaptive anchor frames, the best anchor frame was derived for different training sets. FPN and PAN structures aggregate parameters from different backbone layers for different detection layers. In addition to the YOLO family of algorithms, more advanced one-stage detectors currently include YOLOX [25], FCOS [26], FSAF [27], DETR [28], etc.

In addition to one-stage algorithms, two-stage algorithms have continuously improved since their creation. Common two-stage detectors include Faster R-CNN [29], VFNet [30], CenterNet2 [31], Cascade-RCNN [32], etc. The two-stage algorithm remains the predominant algorithm in the detection field, often used to solve complex object-detection problems, such as those that require high precision or multiple scales or do not need to be solved quickly. A summary of the related work is shown in Table 1.

Table 1. Summary of related work on object detection.

Author	Methods	Results
One-stage methods:	-	-
Redmon et al. [4] (2016)	YOLO	66.4% mAP@0.5 on VOC 2007
Redmon et al. [17] (2017)	YOLOv2	76.8% mAP@0.5 on VOC 2007
Redmon et al. [19] (2018)	YOLOv3	57.9% mAP@0.5 on COCO
Tian et al. [26] (2019)	FCOS	65.9% mAP@0.5 on COCO
Zhu et al. [27] (2019)	FSAF	65.2% mAP@0.5 on COCO
Carion et al. [28] (2020)	DERT	62.4% mAP@0.5 on COCO
Bochkovskiy et al. [22] (2020)	YOLOv4	65.7% mAP@0.5 on COCO
Ge et al. [25] (2021)	YOLOX	67.3% mAP@0.5 on COCO
Two-stage methods:	-	-
Ren et al. [29] (2016)	Faster R-CNN	70.0% mAP@0.5 on VOC 2007
Cai et al. [32] (2018)	Cascade-RCNN	67.7% mAP@0.5 on COCO
Zhang et al. [30] (2021)	VFNet	73.0% mAP@0.5 on COCO
Zhou et al. [31] (2021)	CenterNet2	74.0% mAP@0.5 on COCO

3. Materials and Methods

3.1. Abbreviations

The abbreviations used in this article are summarized in Table 2.

Table 2. The abbreviations used in this manuscript.

Abbreviations	Meaning
QD	Quality Detection
mAP	Mean Average Precision
FPS	Frames Per Second
PCB	Printed Circuit Boards
CNNs	Convolutional Neural Network
RCNN	Region-Based Convolutional Neural Network

Table 2. *Cont.*

Abbreviations	Meaning
FPN	Feature Pyramid Network
PAN	Path Aggregation Network
CSP	Cross Stage Partial
FCOS	Fully Convolutional One-Stage
FSAF	Feature Selective Anchor-Free
ACON	Activate Or Not
IoU	Intersection over Union
GPU	Graphics Processing Unit
CPU	Central Processing Unit
SPP	Spatial Pyramid Pooling
ReLU	Rectified Linear Unit
FLOPS	Floating-Point Operations Per Second
GFLOPS	Giga Floating-Point Operations Per Second
CIoU	Complete Intersection over Union
NMS	Non-Maximum Suppression
SE	Squeeze-and-Excitation
CA	Coordinate Attention
P	Precision
R	Recall
TP	True Positive
FP	False Positive
FN	False Negative

3.2. Data Acquisition

This experiment's dataset is partly derived from the computer mainboard quality-detection dataset, which contains 311 images and information on five defect types. This dataset is too small, and the defect types are too few, which can easily lead to network overfitting and poor generalization performance. Therefore, we added four different models of computer mainboards to capture the dataset, resulting in 2639 images with 12 different defect types: loose fan screws, missing fan screws, loose mainboard fixing screws, missing mainboard fixing screws, missed fan wiring, loose fan wiring, wrong screw type, mainboard scratches, fan scratches, chip polishing, interface scratches, and interface oxidation. Various defect types are shown in Figure 1. Moreover, in order to improve the model's generalization performance, this paper applies data enhancement techniques to the original dataset, such as randomly rotating images, modifying contrast and exposure, and removing images that cause augmentation errors. Finally, we obtained 11,300 images, including 8350 augmented images with image data.

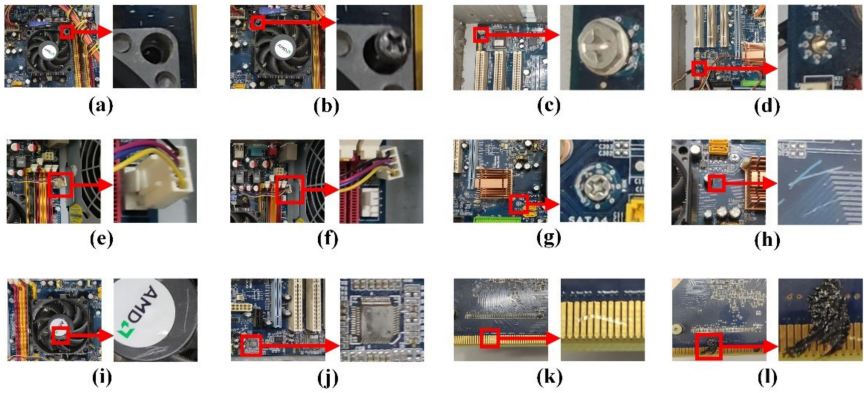

Figure 1. Examples of the 12 types of defects. (**a**) Loose fan screws. (**b**) Missing fan screws. (**c**) Loose mainboard fixing screws. (**d**) Missing mainboard fixing screws. (**e**) Missing fan wiring. (**f**) Loose fan wiring. (**g**) Wrong type of screws. (**h**) mainboard scratches. (**i**) fan scratches. (**j**) chip polishing. (**k**) Interface scratches. (**l**) interface oxidation.

3.3. The QD-YOLO Network Model

The QD-YOLO model presented in this paper is intended to optimize network detection speed and accuracy. To improve upon YOLOv5s, we replace the Swish activation function [33] with the Meta-ACON activation function [34] and add a composite attention mechanism to the backbone. Meanwhile, Ghost convolution blocks replace convolution blocks, and Ghost Bottleneck is applied to the backbone and neck. Figure 2 depicts the network structure of the QD-YOLO system.

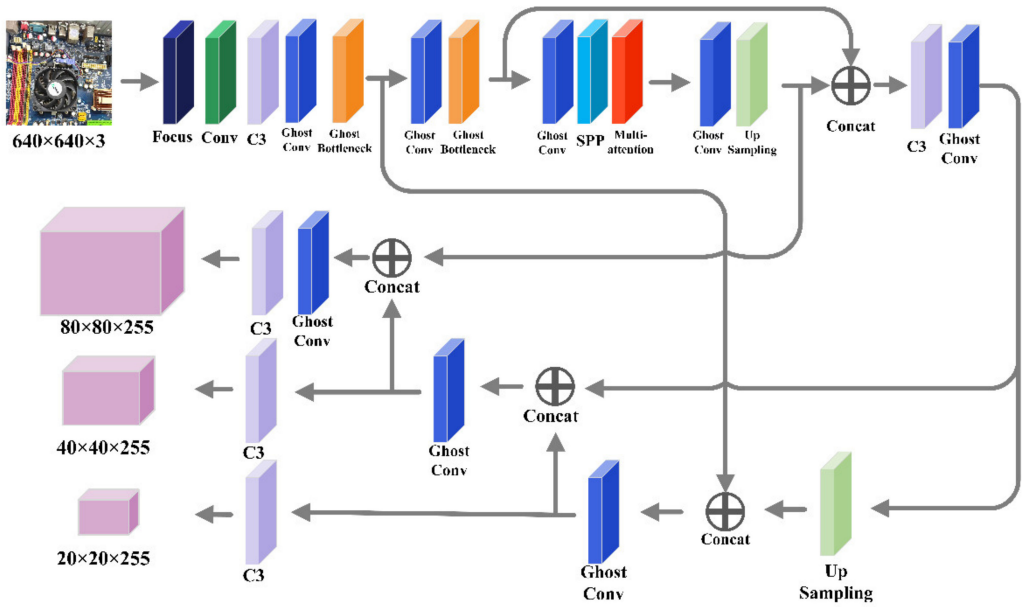

Figure 2. QD-YOLO network structure.

3.4. Meta-ACON Activation Function

We hope that the QD-YOLO model will be able to be generalized to a wider variety of datasets while achieving greater accuracy without the addition of excessive parameters and computation. ACON (Activate Or Not), proposed by Ma et al., allows each neuron to activate or deactivate adaptively, which helps improve generalization and transfer performance. Since the Meta-ACON activation function can determine the activation degree explicitly, we attempt to use it in the QD-YOLO network model, which can be formulated as:

$$Meta - ACON(x) = (p_1 - p_2)x \cdot \sigma[\beta(p_1 - p_2)x] + p_2 x \tag{1}$$

The learning switching factor β_c depends on the input sample $x, x \in R^{C \times H \times W}$, and the switching factor β_c is determined as:

$$\beta_c = \sigma W_1 W_2 \sum_{h=1}^{H} \sum_{w=1}^{W} x_{c,h,w} \tag{2}$$

where $W_1 \in R^{C \times C/r}$, $W_2 \in R^{C/r \times C}$.

3.5. Composite Attention Mechanism

The attention mechanism integrates correlations, allowing the model to dynamically focus on important input parts to more effectively complete the task. We introduce an attention mechanism module into QD-YOLO to improve network performance by improving the network's recognition of features. A single attention mechanism will cause the model to pay more attention to a subset of the information, and it cannot capture all useful content. For example, channel attention typically ignores some position information, limiting the scope for accuracy improvement, whereas Coordinate Attention [35] can address the issue

of ignoring position information. As a result, we combine the Squeeze-and-Excitation (SE) block [36] and the Coordinate Attention (CA) module to optimize the network.

3.5.1. Coordinate Attention Layer

Coordinate Attention, a new network attention mechanism proposed by HOU et al. [35], encodes channel relationships and long-term dependencies via precise location data. As shown in Figure 3, the overall module can be divided into two steps: coordinate information embedding and Coordinate Attention generation.

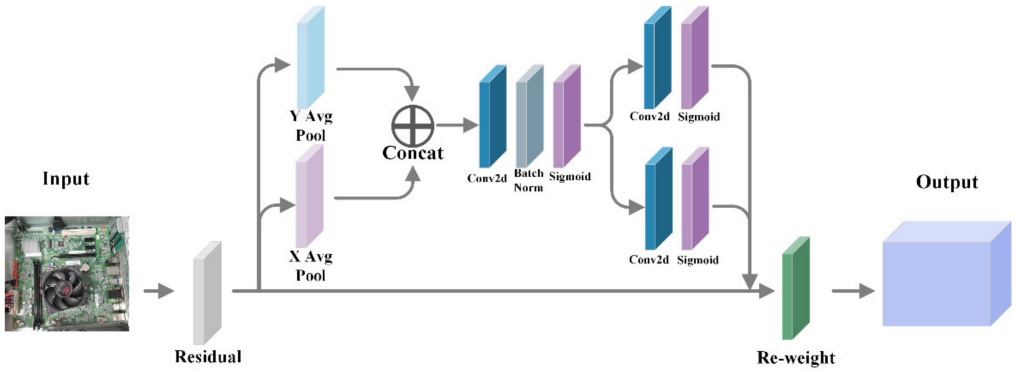

Figure 3. Coordinate Attention layer.

(1) Coordinate information embedding

To motivate the attention module to capture remote spatial interactions with precise location information, Coordinate Attention decomposes global pooling into a one-to-one feature encoding operation.

$$Z_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} X_c(h, j) \tag{3}$$

Each channel is encoded along with the horizontal and vertical coordinates for a given input X, using a pooling kernel of size $(H,1)$ or $(1,W)$, respectively. Thus, the output of the cth channel with height h can be expressed as:

$$Z_c^h(h) = \frac{1}{W} \sum_{0 \le i \le W} X_C(h, j) \tag{4}$$

Similarly, the output of channel C with width W can be written as:

$$Z_c^w(h) = \frac{1}{H} \sum_{0 \le i \le H} X_C(j, w) \tag{5}$$

These two transformations aggregate features along two spatial directions to obtain a pair of direction-aware feature maps, enabling the attention module to capture long-term dependencies and precise location information along different spatial directions to locate objects of interest accurately.

(2) Coordinate Attention Generation

In order to utilize the representation produced by information embedding, we adopt Coordinate Attention to generate and concatenate the information embedding result and then use the 1×1 convolution transformation function F_1 to transform it:

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

δ is the nonlinear activation function, and f is the intermediate feature mapping that encodes spatial information in horizontal and vertical directions. The transform

decomposes f into two separate tensors: f^h and f^w along the spatial dimension, where $f^h \in R^{C/r \times H}$, $f^w \in R^{C/r \times w}$.

We transform f_h and f_w into tensors with the same number of channels to the input X using the other two 1×1 convolutional transform F_h and F_w, respectively:

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

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

Finally, the output y of the Coordinate Attention block can be written as:

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

3.5.2. SE Attention Layer

SE-Net, a network structure proposed by Jie Hu et al., focuses on feature fusion between channels in convolutional operations in the backbone network. Channel calibration of the original feature maps is done through squeeze, excitation, and reweight, adaptively learning the importance of each feature channel, and assigning different weights. The structure of the SE attention mechanism can be seen in Figure 4.

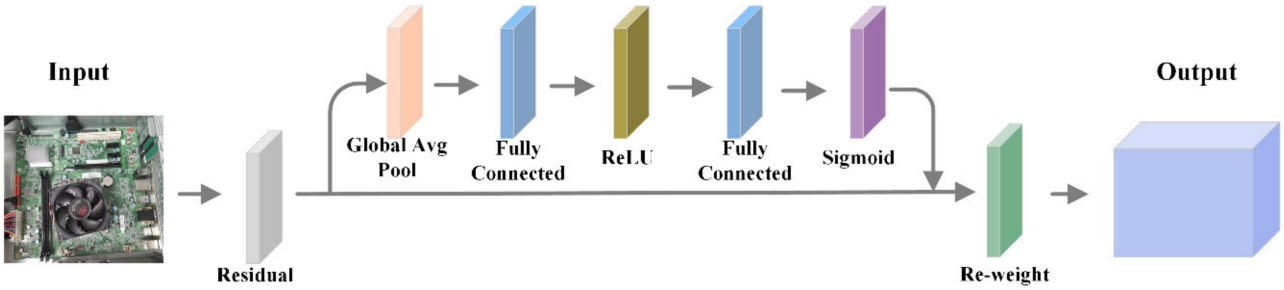

Figure 4. SE layer.

(1) Squeeze operation

First, we encode the entire spatial feature of the channel as a global feature using global average pooling.

$$Z^C = F_{sq}(u_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i,j), z \in R^C \tag{10}$$

(2) Excitation operations

Then, a global description of the features is obtained by the Squeeze operation, using a gating mechanism in the form of sigmoid to correlate the information between channels.

$$S = F_{ex}(Z, W) = \sigma(g(Z, W)) = \sigma(W_2 ReLU(W_1 Z)) \tag{11}$$

where $W_1 \in R^{\frac{C}{r} \times C}, W_2 \in R^{C \times \frac{C}{r}}$.

A bottleneck structure with two fully connected layers is used. The first FC layer acts as a dimensionality reduction, with the dimensionality reduction factor r being a hyperparameter, and then ReLU activation is used. The final FC layer restores the original dimensionality by multiplying the obtained activation values of each channel by the original features on U to reduce the complexity of the model and improve the generalization capability.

$$xc = Fscale(u_c, s_c) = u_c \times s_c \tag{12}$$

We connect the SE attention mechanism and CA mechanism to form a composite attention mechanism added to the backbone of the YOLOv5 network, as shown in Figure 5.

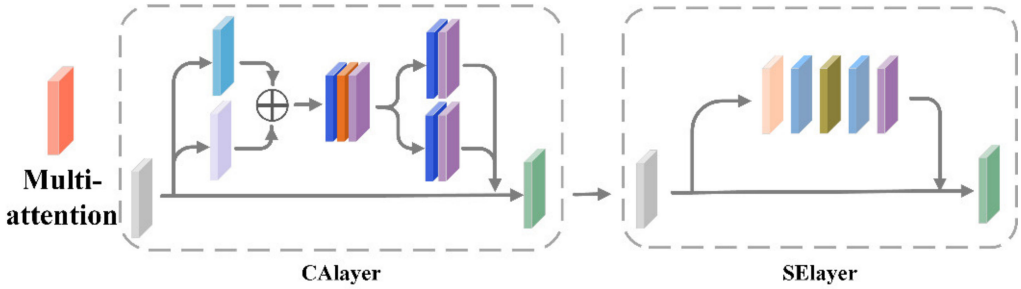

Figure 5. Composite attention mechanism.

3.6. Ghost Convolution Blocks and Ghost Bottleneck

Conventional feature extraction methods can capture a wealth of feature information and generate redundant data. Many redundant feature maps will be generated by stacking multiple convolutional layers, requiring many parameters and computations to process. Therefore, some researchers proposed methods to compress the model, such as pruning, quantization, and knowledge distillation, that effectively reduce the number of parameters. Unfortunately, they suffer from the problem of complex model design and training difficulties. Other methods concentrate on network structure optimization, such as MobileNet [37] and ShuffleNet [38], which are simple, effective, and straightforward to implement, although 1×1 convolutional layers continue to consume a significant amount of memory and FLOPs.

In order to better achieve real-time detection, we adopt Ghost convolution blocks and Ghost Bottleneck [39] in this paper to reduce the number of parameters and operations in the network.

Ghost convolution allows extracting feature maps with as few operations as possible, as shown in Figure 6. We use different convolution kernels to extract features from the input feature maps and perform simple linear transformation operations on these feature maps to generate more small feature maps, which can be formulated as:

$$Y' = X \times f' \tag{13}$$

where $Y' = R^{h' \times w' \times m}$, $f' \in R^{c \times k \times k \times m}$.

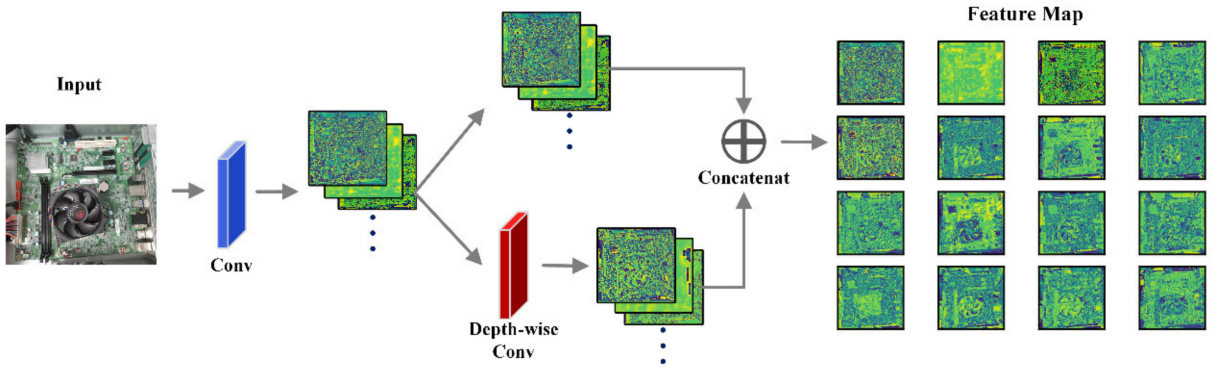

Figure 6. Ghost convolution structure.

Ignoring m, n, and bias terms, other hyperparameters remain the same as ordinary convolutions to keep feature maps consistent. To further obtain the desired n mappings, simple linear operations are applied to each feature in Y' to extract features and increase the number of channels. In Ghost convolution, the identity map is parallelized with a linear transformation to preserve the inherent feature map.

The Ghost Bottleneck comprises two stacked Ghost convolution blocks. The first Ghost convolution block is used to increase the number of channels of the input feature map and expand it for subsequent operations. Following this, the second Ghost convolution block

reduces the number of channels of the output feature map to match the network's structure. Then, the shortcut is used to connect the input and output of these two Ghost modules.

We use the Ghost convolution block to replace the ordinary convolution block in the backbone and neck of the YOLOv5s network and utilize the Ghost Bottleneck to replace the C3 block in the backbone and neck of the YOLOv5s network. In this way, the scale of the network's parameters and the number of computing resources it uses can be optimized without changing how well it detects things.

4. Experiments and Results

4.1. Experimental Evaluation Criteria

In this paper, we mainly use mAP@0.5 to reflect and evaluate detection performance, which is the average AP@0.5 value of all classification detection results that overcomes the limitation of single-point Precision (P) and Recall (R) values and can effectively reflect global performance. The AP@0.5 value denotes the closed area of the precision and recall curves when the Intersection over Union (IoU) threshold is 0.5. P, R, $F1$, and mAP@0.5 are the performance metrics used in this paper.

$$P(precision) = \frac{TP}{TP + FP} \tag{14}$$

$$R(recall) = \frac{TP}{TP + FN} \tag{15}$$

$$F1 = \frac{2 \times P \times R}{P + R} \tag{16}$$

True Positive (TP) represents accurately detected positive samples, False Positive (FP) represents negative samples that were misclassified as positives, and False Negative (FN) represents positives that were misclassified as negatives.

4.2. Experimental Environment

This experiment conducts all training and testing on the same hardware and software platform. The experiment is configured on Windows 10, using Pytorch as the deep-learning framework and Pycharm as the programming environment for the proposed approach. The experiment's hardware configuration is as follows: Central Processing Unit (CPU): Intel Core I5-9300H @2.4GHz. Graphics Processing Unit (GPU): NVIDIA GeForce GTX1650. The environment software consists of CUDA 10.2, CUDNN 7.6, and Python 3.9.

The following are the training's specific configurations: Before entering the network, the input image size is scaled to 640 $\times$ 640 pixels, and the batch size is 8. The network's initial learning rate is 0.01, its ultimate learning rate is 0.002, and the optimizer uses adam with a weight decay value of 5×10^{-4}. Three hundred iterations are performed on each model. The starting momentum value of the warmup is set to 0.8, and the first 3 epochs are trained as warmups with a slow learning rate. In subsequent epochs, the learning rate is varied using the cosine annealing training method [40]. The classification loss and localization loss are computed with BCEWithLogitsLoss, whereas the confidence loss is computed with CIoU. The experiment's parameter settings had a significant impact on the results. Both the YOLOv5s and the QD-YOLO networks make use of identical values for their training parameters. This is done to ensure that the experiment results are correct and accurate.

4.3. Experimental Results

In order to accurately reflect the detection performance and convergence performance of the network before and after improvement, the evaluation results in this section are mainly based on the loss function curve and mAP. In the network training process, the loss change curve [41] intuitively reflects whether the network model can converge stably as the number of iterations increases.

As seen in Figure 7, the curves of both the QD-YOLO algorithm and the original YOLOv5s show a decreasing training trend, with the loss function's value gradually decreasing with the number of iterations. When the model is iterated 100 times, the QD-YOLO algorithm has a loss function value of 0.018, while YOLOv5s still have a loss function value of 0.023. When the model is iterated 300 times, the loss values of both algorithms drop to close to 0, and the network converges. At this time, the loss function value of the QD-YOLO algorithm is 0.011, while the loss value of YOLOv5s is still 0.014. Compared with YOLOv5s, the QD-YOLO model has a faster loss decay rate and smaller decay function value, which indicates that our model has a faster convergence performance.

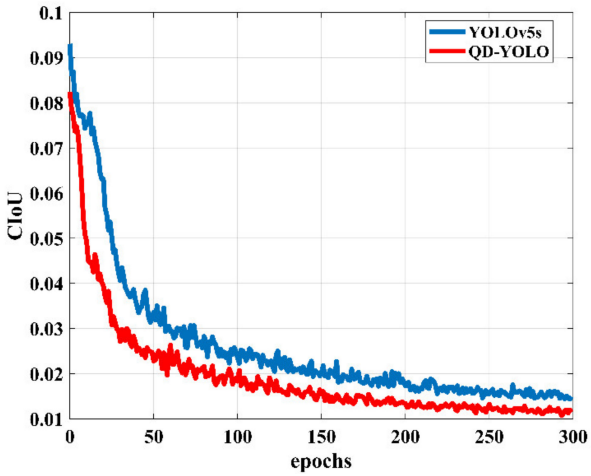

Figure 7. Comparison of CIoU values in network training before and after improvement.

As shown in Figure 8, The mAP@0.5 of the original YOLOv5s requires about 100 iterations of training to reach 90%, and the mAP@0.5 in the iterative training is only 96.76% at the highest, while QD-YOLO only needs 30 iterations of training to reach about 90%. At the same time, the highest accuracy can reach 98.85% of QD-YOLO. These figures show that QD-YOLO has a higher accuracy rate than YOLOv5s and greatly improves convergence speed. Since we adopted the Meta-ACON activation function, the network has different characteristics and properties for different samples, making the network better applicable in different scenarios. Meanwhile, introducing the composite attention mechanism can make the network pay more attention to the critical samples in the sample. The improved model is able to capture the correct information in a short training period to a certain extent, showing the accuracy and efficiency of the model.

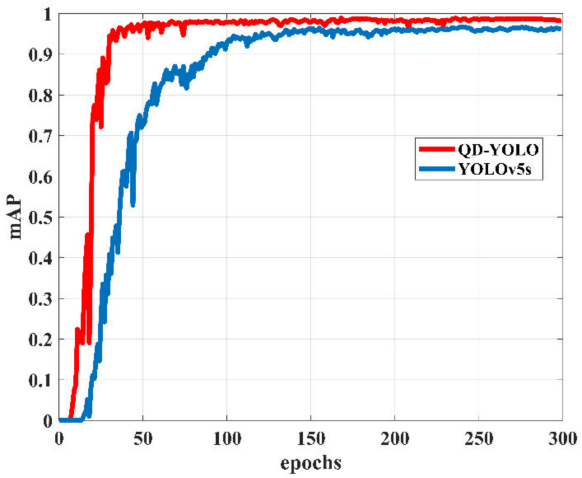

Figure 8. Comparison of mAP values in network training before and after improvement.

In addition, the actual detection performance of QD-YOLO is better than that of YOLOv5s in terms of small-target and background recognition. Figure 9 shows the detection results of QD-YOLO and YOLOv5s for the same image.

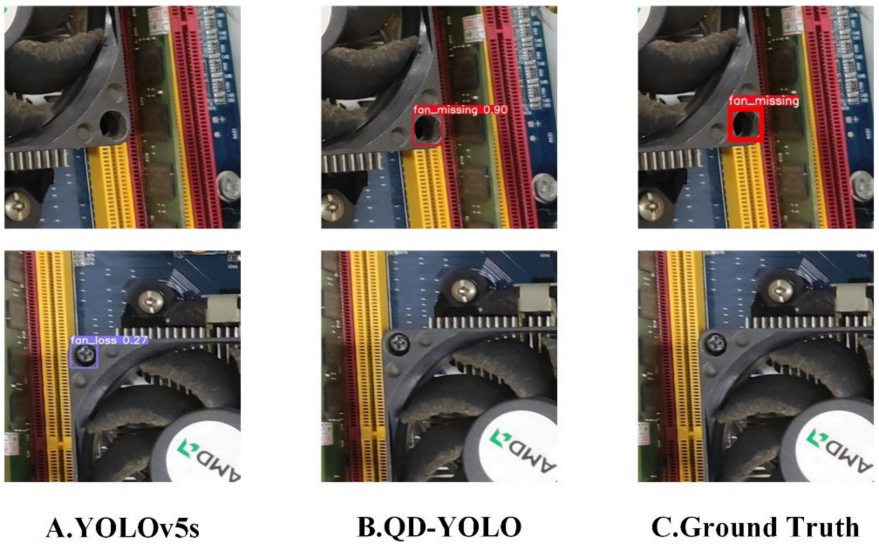

Figure 9. Comparison of actual detection results.

By observing the detection results of the two networks, we can find that QD-YOLO can detect minor defects in high-resolution images, return the positioning frame more accurately, and avoid the background being detected as a wrong sample effectively, which is difficult for YOLOv5s to do. It also shows the effectiveness of the improved scheme in this paper.

4.4. Ablation Experiment of Attentional Mechanisms

In order to explore the impact of different attention mechanisms on the network model, we set up ablation experiments, as shown in Table 3, to demonstrate their impact on the network's overall performance.

Table 3. Comparison of performance-optimization effects of different attention modules.

Model	Precision (%)	Recall (%)	*F*1 (%)	mAP@0.5 (%)
YOLOv5s	97.19	93.94	95.53	96.76
YOLOv5s + Ghost	95.26	94.26	94.75	96.13
YOLOv5s + Meta-ACON	94.13	96.70	95.39	97.02
YOLOv5s + CA	97.67	96.10	96.88	97.47
YOLOv5s + SE	96.55	95.22	95.88	97.51
YOLOv5s + CA + SE	97.26	97.67	97.46	98.09
QD-YOLO	95.93	99.08	97.48	98.85

Compared with the original YOLOv5s, the mAP increased by 0.71% after adding the Coordinate Attention (CA) mechanism, the mAP increased by 0.75% after adding the Squeeze-and-Excitation (SE) attention mechanism, and the mAP increased by 1.33% after adding SE and CA at the same time. We found that the precision of the network with SE and CA is lower than that of the network with only CA. However, the recall rate is greatly improved. On the other hand, the *F*1 of the network with SE and CA is higher than that with only CA, indicating that the network has better overall detection performance.

The SE attention mechanism pays attention to the relationship between channels and automatically learns the importance of different channel features. The CA attention mechanism captures cross-channel, orientation-aware, and position-sensitive information, locating target regions more accurately. Therefore, combining the CA and SE can effectively

improve the network's recognition of features and improve the network's performance in detection.

4.5. Ablation Experiment of Ghost Convolution

Aiming to explore the impact of using Ghost convolution on the network, we set up experiments on model size and detection speed, as shown in Table 4.

Table 4. Comparison of Ghost convolution optimization performance.

Model	Precision (%)	Recall (%)	*F*1 (%)	mAP@0.5 (%)
YOLOv5s	97.19	93.94	95.53	96.76
YOLOv5s + Ghost	95.26	94.26	94.75	96.13
YOLOv5s + Meta-ACON	98.10	96.47	97.27	97.02
YOLOv5s + CA	97.67	96.10	96.88	97.47
YOLOv5s + SE	96.55	95.22	95.88	97.51
YOLOv5s + CA + SE	97.26	97.67	97.46	98.09
QD-YOLO	95.93	99.08	97.48	98.85

According to Table 4, when the composite attention mechanism and the Meta-ACON activation function are added to YOLOv5s, the number of network layers is increased by 157, the number of parameters is increased by 0.9 M, the amount of computation is increased by 1 GFLOPS, and the FPS is decreased by 5.3. Compared with the original YOLOv5s, QD-YOLO has more network layers and smaller parameters. Specifically, the amount of computation is reduced by 6.3 GFLOPS, and the FPS is increased by 2.7, which shows better real-time performance.

4.6. Comparison with Other Networks

We compare the performance of the QD-YOLO model and the current mainstream detection algorithms applied to the quality detection of computer mainboards, including Faster R-CNN, Cascade R-CNN, YOLOv3, etc., as shown in Table 5.

Table 5. Comparison of results of different target detection algorithms.

Model	mAP@0.5 (%)	FPS
Faster-RCNN	94.15	1.322
Cascade R-CNN	99.20	7.69
FPN	92.08	6.46
YOLO v3	89.16	21.47
YOLO v4	95.84	25.46
YOLO v5	96.76	28.58
QD-YOLO	98.85	31.25

Faster-RCNN networks have many parameters that are much more computationally intensive than one-stage algorithms, which results in detection algorithms that are less than real-time. Meanwhile, Faster-RCNN cannot be combined with the whole picture for comprehensive analysis in the computer mainboard quality-detection scene, which causes serious network misjudgment of the background. This leads to lower accuracy than QD-YOLO for Faster-RCNN deployed under the same conditions. The Cascade R-CNN has the highest accuracy in this scenario. It aims to optimize the prediction results by cascading several detection networks continuously. However, the cascade of multiple sub-detection networks increases the structural complexity of the network, which has a particular impact on the detection speed. It takes 0.13 s to detect a picture, which is insufficient in real-time. FPN fuses the shallow layers with high resolution and the deep layers with rich semantic information. In actual use, the FPN network has not achieved outstanding results. The network has a poor effect on small-target detection and low accuracy. At the same time,

in terms of real-time performance, the speed is only 6.46 FPS, which is far lower than the requirements of real-time detection.

The network structure of YOLOv3 is relatively simple, which brings good speed performance and poor performance in detection accuracy. YOLOv3 makes it difficult to detect small objects in the computer mainboard defect dataset, so it is not suitable for quality detection of computer mainboards. YOLOv4 introduces some tricks, which means it has higher detection accuracy. Some speed-optimization techniques introduced also mean that YOLOv4 has a good detection speed during testing. However, compared to QD-YOLO, YOLOv4 is lower in both accuracy and speed. Compared with YOLOv5s, QD-YOLO improves mAP by 2.09%. This paper adds some structure to the network, which makes the QD-YOLO network develop in a complex direction. However, due to the use of Ghost convolution instead of ordinary convolution, the number of parameters is significantly reduced, and the detection speed is accelerated. The speed of 31.25 FPS satisfies the requirement of real-time detection.

5. Discussion

Table 3 shows the results of ablation experiments on the detection performance of the network with different structural improvements. We have discovered that the network's performance diminishes after using Ghost convolution. The mAP is reduced by 0.63% when compared to the original YOLOv5s. We believe this is because Ghost convolution uses a linear operation to generate a part of the feature map, which leads to the loss of some information and the degradation of network detection performance. When combined with Table 4, Ghost convolution is very effective at improving real-time performance, and the detection performance loss in this area is low-cost. Ghost convolution can effectively reduce the network's computational load, can significantly accelerate detection speed, and has convenient use characteristics.

In Section 4, we discuss the performance of QD-YOLO, which achieved satisfactory detection results in the application scenario of computer mainboard quality detection. The QD-YOLO can reach 31.25 FPS, close to real-time detection. In terms of detection effect, QD-YOLO achieves a mAP of 98.85% and has good robustness with variable inputs, making it suitable for real-time quality detection. However, the detection algorithm could not produce accurate results for partially occluded objects. This is also the issue that the object-detection model should address [42]. Identifying occluded objects can be achieved using a variety of methods, such as optimizing the loss function for application scenarios [43], optimizing the Non-Maximum Suppression (NMS) method [44], optimizing the network structure [45], and so on. The structure of the QD-YOLO model will be further optimized in future work.

6. Conclusions and Future Work

This paper proposes a QD-YOLO mainboard quality detection model. Meta-ACON replaces swish activation function in the YOLOv5 network to improve generalization. Then, a composite attention mechanism is embedded in the network's backbone to model channel and location information. In the meantime, Ghost convolution has replaced conventional convolution to ensure efficient real-time detection. In addition, a computer mainboard quality detection dataset with 11,300 images and 12 different defect types was established. Experiments show that the method in this paper achieves mAP of 98.85% and FPS of 31.25 in the self-made dataset test. The QD-YOLO network outperforms other methods in both accuracy and speed.

Currently, our QD-YOLO network focuses on enhancing detection precision and small-target detection effect. The improvement in the speed of detection is relatively modest. In the future, we will make every effort to reduce the number of parameters and the amount of computation required for the network to run smoothly on mobile devices. Alternatively, we will combine the correlation of time series in order to optimize the defect detection problem. In the meantime, future research will introduce the model to larger datasets to

improve its generalizability. Furthermore, the way small targets are found and modeled will be tweaked to improve the accuracy of the detection model.

Author Contributions: Project administration, G.T.; data curation, G.T.; writing—original draft, G.T.; writing—review and editing, J.Q. and N.N.X.; funding acquisition, J.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Natural Science Foundation of Hunan Province under Grant (No.2022JJ31019, No.2021JJ31164) and the Soft Science Research Project of Guangdong Digital Government Reform and Construction Expert Committee (No.ZJWKT202204).

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Acknowledgments: The authors would like to thank all the anonymous reviewers for their insightful comments and constructive suggestions.

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

References

1. Arabian, J. *Computer Integrated Electronics Manufacturing and Testing*; CRC Press: Boca Raton, FL, USA, 2020. [CrossRef]
2. Reyes, A.C.C.; Del Gallego, N.P.A.; Deja, J.A.P. Mixed reality guidance system for motherboard assembly using tangible augmented reality. In Proceedings of the 2020 4th International Conference on Virtual and Augmented Reality Simulations, Sydney, Australia, 14–16 February 2020; ACM: New York, NY, USA, 2020; pp. 1–6. [CrossRef]
3. Grieco, L.A.; Boggia, G.; Piro, G.; Jararweh, Y.; Campolo, C. *Ad-Hoc, Mobile, and Wireless Networks: Proceedings of the 19th International Conference on Ad-Hoc Networks and Wireless, ADHOC-NOW 2020, Bari, Italy, 19–21 October 2020*; Springer Nature: Berlin/Heidelberg, Germany, 2020; Volume 12338.
4. 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. [CrossRef]
5. Li, Y.; Huang, H.; Chen, Q.; Fan, Q.; Quan, H. Research on a product quality monitoring method based on multi scale PP-YOLO. *IEEE Access* **2021**, *9*, 80373–80387. [CrossRef]
6. Yotsuyanagi, H.; Ono, A.; Takagi, M.; Roth, Z.; Hashizume, M. A built-in electrical test circuit for interconnect tests in assembled PCBs. In Proceedings of the 2012 2nd IEEE CPMT Symposium Japan, Kyoto, Japan, 10–12 December 2012; pp. 1–4. [CrossRef]
7. Zhou, Q.; Qin, J.; Xiang, X.; Tan, Y.; Ren, Y. MOLS-Net: Multi-organ and lesion segmentation network based on sequence feature pyramid and attention mechanism for aortic dissection diagnosis. *Knowl.-Based Syst.* **2022**, *239*, 107853. [CrossRef]
8. Hou, G.; Qin, J.; Xiang, X.; Tan, Y.; Xiong, N.N. Af-net: A medical image segmentation network based on attention mechanism and feature fusion. *CMC-Comput. Mater. Contin.* **2021**, *69*, 1877–1891. [CrossRef]
9. Ma, W.; Zhou, T.; Qin, J.; Zhou, Q.; Cai, Z. Joint-attention feature fusion network and dual-adaptive NMS for object detection. *Knowl.-Based Syst.* **2022**, *241*, 108213. [CrossRef]
10. Esteva, A.; Kuprel, B.; Novoa, R.A.; Ko, J.; Swetter, S.M.; Blau, H.M.; Thrun, S. Dermatologist-level classification of skin cancer with deep neural networks. *Nature* **2017**, *542*, 115–118. [CrossRef]
11. Luo, Y.; Qin, J.; Xiang, X.; Tan, Y. Coverless image steganography based on multi-object recognition. *IEEE Trans. Circuits Syst. Video Technol.* **2020**, *31*, 2779–2791. [CrossRef]
12. Zhou, Q.; Qin, J.; Xiang, X.; Tan, Y.; Xiong, N.N. Algorithm of helmet wearing detection based on AT-YOLO deep mode. *CMC Comput. Mater. Contin.* **2021**, *69*, 159–174. [CrossRef]
13. Adibhatla, V.A.; Chih, H.-C.; Hsu, C.-C.; Cheng, J.; Abbod, M.F.; Shieh, J.-S. Defect detection in printed circuit boards using you-only-look-once convolutional neural networks. *Electronics* **2020**, *9*, 1547. [CrossRef]
14. Tao, X.; Zhang, D.; Ma, W.; Liu, X.; Xu, D. Automatic metallic surface defect detection and recognition with convolutional neural networks. *Appl. Sci.* **2018**, *8*, 1575. [CrossRef]
15. Jiao, L.; Zhang, F.; Liu, F.; Yang, S.; Li, L.; Feng, Z.; Qu, R. A survey of deep learning-based object detection. *IEEE Access* **2019**, *7*, 128837–128868. [CrossRef]
16. Jian, C.; Gao, J.; Ao, Y. Automatic surface defect detection for mobile phone screen glass based on machine vision. *Appl. Soft Comput.* **2017**, *52*, 348–358. [CrossRef]
17. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271. [CrossRef]
18. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 448–456. [CrossRef]
19. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767. [CrossRef]
20. He, K.; Zhang, X.; Ren, S.; 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. [CrossRef]

21. Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125. [CrossRef]
22. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934. [CrossRef]
23. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768. [CrossRef]
24. Wang, C.-Y.; Liao, H.-Y.M.; Wu, Y.-H.; Chen, P.-Y.; Hsieh, J.-W.; Yeh, I.-H. CSPNet: A new backbone that can enhance learning capability of CNN. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 390–391. [CrossRef]
25. Ge, Z.; Liu, S.; Wang, F.; Li, Z.; Sun, J. Yolox: Exceeding yolo series in 2021. *arXiv* **2021**, arXiv:2107.08430. [CrossRef]
26. Tian, Z.; Shen, C.; Chen, H.; He, T. Fcos: Fully convolutional one-stage object detection. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019; pp. 9627–9636. [CrossRef]
27. Zhu, C.; He, Y.; Savvides, M. Feature selective anchor-free module for single-shot object detection. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 840–849. [CrossRef]
28. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-end object detection with transformers. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; pp. 213–229. [CrossRef]
29. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1137–1149. [CrossRef]
30. Zhang, H.; Wang, Y.; Dayoub, F.; Sunderhauf, N. Varifocalnet: An iou-aware dense object detector. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 8514–8523. [CrossRef]
31. Zhou, X.; Koltun, V.; Krähenbühl, P. Probabilistic two-stage detection. *arXiv* **2021**, arXiv:2103.07461. [CrossRef]
32. Cai, Z.; Vasconcelos, N. Cascade r-cnn: Delving into high quality object detection. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6154–6162. [CrossRef]
33. Ramachandran, P.; Zoph, B.; Le, Q.V. Searching for activation functions. *arXiv* **2017**, arXiv:1710.05941. [CrossRef]
34. Ma, N.; Zhang, X.; Liu, M.; Sun, J. Activate or not: Learning customized activation. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 8032–8042. [CrossRef]
35. Hou, Q.; Zhou, D.; Feng, J. Coordinate attention for efficient mobile network design. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 13713–13722. [CrossRef]
36. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141. [CrossRef]
37. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861. [CrossRef]
38. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6848–6856. [CrossRef]
39. Han, K.; Wang, Y.; Tian, Q.; Guo, J.; Xu, C.; Xu, C. Ghostnet: More features from cheap operations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 1580–1589. [CrossRef]
40. Loshchilov, I.; Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv* **2016**, arXiv:1608.03983. [CrossRef]
41. Zheng, Z.; Wang, P.; Ren, D.; Liu, W.; Ye, R.; Hu, Q.; Zuo, W. Enhancing geometric factors in model learning and inference for object detection and instance segmentation. *IEEE Trans. Cybern.* **2021**, *52*, 8574–8586. [CrossRef]
42. Zhang, K.; Xiong, F.; Sun, P.; Hu, L.; Li, B.; Yu, G. Double anchor R-CNN for human detection in a crowd. *arXiv* **2019**, arXiv:1909.09998. [CrossRef]
43. Wang, X.; Xiao, T.; Jiang, Y.; Shao, S.; Sun, J.; Shen, C. Repulsion loss: Detecting pedestrians in a crowd. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7774–7783. [CrossRef]
44. Huang, X.; Ge, Z.; Jie, Z.; Yoshie, O. Nms by representative region: Towards crowded pedestrian detection by proposal pairing. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 10750–10759. [CrossRef]
45. Xie, J.; Pang, Y.; Cholakkal, H.; Anwer, R.; Khan, F.; Shao, L. PSC-Net: Learning part spatial co-occurrence for occluded pedestrian detection. *Sci. China Inf. Sci.* **2021**, *64*, 120103. [CrossRef]