applied sciences

MDPI

Article

A Novel Small Target Detection Strategy: Location Feature Extraction in the Case of Self-Knowledge Distillation

Gaohua Liu [1], Junhuan Li [2], Shuxia Yan [2,*] and Rui Liu [3]

[1] School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, China
[2] School of Electronics and Information Engineering, Tiangong University, Tianjin 300387, China
[3] School of Software, Tiangong University, Tianjin 300387, China
* Correspondence: tjuysx@163.com

Abstract: Small target detection has always been a hot and difficult point in the field of target detection. The existing detection network has a good effect on conventional targets but a poor effect on small target detection. The main challenge is that small targets have few pixels and are widely distributed in the image, so it is difficult to extract effective features, especially in the deeper neural network. A novel plug-in to extract location features of the small target in the deep network was proposed. Because the deep network has a larger receptive field and richer global information, it is easier to establish global spatial context mapping. The plug-in named *location feature extraction* establishes the spatial context mapping in the deep network to obtain the global information of scattered small targets in the deep feature map. Additionally, the attention mechanism can be used to strengthen attention to the spatial information. The comprehensive effect of the above two can be utilized to realize location feature extraction in the deep network. In order to improve the generalization of the network, a new self-distillation algorithm was designed for pre-training that could work under self-supervision. The experiment was conducted on the public datasets (Pascal VOC and Printed Circuit Board Defect dataset) and the self-made dedicated small target detection dataset, respectively. According to the diagnosis of the false-positive error distribution, the location error was significantly reduced, which proved the effectiveness of the plug-in proposed for location feature extraction. The *mAP* results can prove that the detection effect of the network applying the location feature extraction strategy is much better than the original network.

Keywords: location feature extraction; small target detection; self-knowledge distillation; attention mechanism

check for updates

Citation: Liu, G.; Li, J.; Yan, S.; Liu, R. A Novel Small Target Detection Strategy: Location Feature Extraction in the Case of Self-Knowledge Distillation. *Appl. Sci.* **2023**, *13*, 3683. https://doi.org/10.3390/app13063683

Academic Editor: Amerigo Capria

Received: 2 February 2023
Revised: 6 March 2023
Accepted: 9 March 2023
Published: 14 March 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

With the development of deep learning, the ability of neural networks has greatly developed. Because neural networks can automatically extract features and fit them to a large number of data points, many excellent target detection algorithms have emerged, such as mask R-CNN [1], cascade R-CNN [2], hybrid task cascade [3], et al. The technology is becoming more advanced, and the accuracy and speed have significantly improved. Due to the maturity of conventional target detection technology, researchers began to pay more attention to the detection of small targets. The most common definition at present comes from the general data set in the field of target detection. Small targets are defined as having a resolution of less than 32 × 32 pixel target [4]. At the same time, Chen et al. [5] proposed a data set for small targets and made the following definition for small targets: the median of the ratio of the bounding box area to the image area is between 0.08% and 0.58%, which is based on the definition of relative scale. Due to the small size, few pixels, wide distribution, and small number of small targets, the effective features that can be extracted by the network are very few. Many of the networks mentioned above that perform well in conventional target detection do not perform well in small target detection [6–9].

The main method to solve these problems is **multi-scale feature fusion**. The deep network has rich semantic information, but it lacks the fine-grained information of the shallow network. The semantic information and fine-grained information can be used simultaneously by fusing features from multiple scales, providing more comprehensive feature information for small target detections. Feature Pyramid Net (FPN) [10] is applied in many small target detections. It adopts multi-scale feature fusion and uses the fused results to make predictions. In the Single Shot MultiBox Detector (SSD) [11] series, DSSD [12] deconvolutes the deep feature map based on SSD and multiplies it with the shallow feature to obtain a better multi-layer feature map, which is very beneficial to the detection of small objects. In the fast RCNN series, HyperNet [13] fused the feature maps obtained from the first, third, and fifth convolution groups, pooled the shallow features, deconvoluted the deep features, and finally fused them in the way of channel concatenation to complement each other.

In general, multi-scale feature map fusion is helpful to capture detailed information and rich semantic information, which is convenient for object location and classification, respectively. However, the multi-scale representation method does not propose a specific feature. Widely extracting features from multiple layers will inevitably increase the computational burden. In addition, redundant information fusion may lead to background noise and insufficient performance [14].

Because small objects only occupy a small part of the image, the information obtained directly from local areas is very limited. Every object always exists in a specific environment or coexists with other objects. Then, some **context-based detection methods** are proposed to utilize the relationship between small objects and other objects or backgrounds. FASSD [15] uses additional features from different layers as the context and proposes an attention mechanism to focus on the context information from the target layer.

Based on the inspiration of the above two methods, a plug-and-play plug-in called the location feature extraction module (LFE module, for short) is designed that can achieve the extraction of location features in deep layers. The LFE module uses the larger receptive field in the deeper network to establish the global spatial context association and then uses the attention mechanism to strengthen the attention to spatial information so as to implement the location feature extraction process.

In order to enhance the generalization performance of neural networks, regularization methods are proposed to reduce overfitting. Knowledge distillation is one of the most effective regularization strategies, which is used to improve the generalization ability of networks. The existing knowledge distillation methods [16–18] require training a complex teacher network first and then passing it on to the student network, which is both time- and cost-consuming. Therefore, self-distillation learning—learning what you teach yourself—is proposed. Traditional self-distillation is realized under supervision, and supervised learning requires a lot of manually labeled label information [19]. The rise of self-supervised learning has solved this problem. Applying self-supervised contrastive learning to self-distillation may produce desirable results.

Our contributions are:

(1) We found that location feature extraction had a great impact on the improvement of small object detection, and improving location feature extraction can effectively improve detection accuracy.

(2) We proposed a location extraction structure that can extract location features at a deep level, use the larger receptive field to establish a global spatial context mapping, and then use the attention mechanism to strengthen the attention to spatial information to extract location features. Combined with CSPNet [20], it is integrated into the above structure to reduce the use of repeated gradient information.

(3) Based on traditional self-knowledge distillation learning, a self-knowledge distillation model was proposed that can be learned in a self-supervised environment. By using the linear combination of the change information generated after data augmentation and the current prediction results to distill knowledge and soften the target.

This paper was organized as follows: In Section 2, the related work in self-supervised learning, self-knowledge distillation, feature extraction, etc. was generalized. In Section 3, a detailed description of the design and principle of the specific structure of the model was given. In Section 4, the experiment, including the experimental environment, the deployment of the experiment, and the analysis of the corresponding experimental results, were elaborated. In Section 5, conclusions were finally drawn.

2. Related Work

2.1. Small Target Detection

Multi-scale feature fusion. The Single Shot MultiBox Detector (SSD) [11] uses a multi-scale feature map for detection, using a large feature map to detect small targets and a small feature map to detect large targets. SSD detects the feature map obtained from each convolution and completes the target location and classification at one time. However, it detects the target by convolution of the feature map, unlike the full connection layer of YOLOv1 [21], so it will lose a lot of spatial information. In addition, each point on these feature layers constructs six prior frames at different scales. Finally, the output obtained on all feature maps is combined, and the detection results are obtained through NMS (non-maximum suppression). MDSSD [22] adds high-level features with rich semantic information to low-level features through deconvolution Fusion Block. In particular, several high-level features with different proportions were up-sampled at the same time, and then the connection was skipped to form a more descriptive feature map of small objects. Finally, these new fusion features were predicted. Different from the element summation strategy adopted by MDSSD [22], the convolution neural network based on the deconvolution region (DR-CNN [23]) adopted the concatenation strategy to fuse a multi-scale feature map for small traffic symbol detection. The channel-aware deconvolution network (CADNet) [24] was proposed to study the relationship between the deep feature maps in different channels to avoid the simple superposition of feature maps. By using the correlation between different scale features, the recall rate of small objects can be improved at a lower computational cost.

Context-based detection method. The enhanced R-CNN [5] proposed in the context network can be considered the first detector focusing on small target detection. In this work, a new regional proposal network (RPN) was proposed to encode the context information of small object proposals. Internal and external networks (ION) [25] used spatial recurrent neural networks (RNNs) to search the context information outside the target area; the skip pooling method was then used to obtain internal multi-level feature mapping. Leng et al. [26] integrated the U-V parallax algorithm with the faster R-CNN, which combined internal and contextual information. The new framework of FASSD [15] started from the baseline SSD and then proposed three components to improve the detection accuracy of small targets. First, SSDs and features were fused to obtain context information, named F-SSD. Second, SSDs with reserved modules enabled the network to focus on important components, called A-SSDs. Third, the researcher combined feature fusion and attention modules, called FA-SSD.

Attention Mechanisms. Inspired by the human visual system, the attention mechanism has been introduced into convolutional neural networks in recent years to improve the performance of target detection [27,28]. According to the form of attention acting on the feature map, the attention mechanism was mainly divided into channel attention [29], spatial attention [30], and channel and spatial mixed attention [31]. The SE attention mechanism [29] is a popular attention mechanism at present. It uses two-dimensional global pooling to calculate the channel attention mechanism, but the channel attention mechanism only pays attention to the coding of channel information, ignoring the influence of location information [32], and location information is very critical in visual tasks, so enhancing the extraction of location features is a key technology to improve the detection accuracy.

2.2. Self-Distillation and Self-Supervised Contrastive Learning

Conventional knowledge distillation [33] methods 'distill' the knowledge contained in a teacher model, which has a larger and better performance, into a student model. Then a new knowledge distillation was proposed. The model at the current time was regarded as a student, and the model at the previous time was regarded as a teacher to distill knowledge. Since the model structure of teacher and student is the same, it is called self-knowledge distillation (self KD) [34]. Specifically, a new self-distillation method was proposed [35], which distilled the knowledge from the deeper part of the network to the shallower part of the network. [36] designed a self-distillation that transferred knowledge from the early stage of the network (teachers) to the later stage (students) to support the supervised train in the same network. In order to further reduce the inference time, a distillation-based training scheme [37] was developed, where, the shallow exit layer attempts to simulate the output of the deep layer during the training process. Recently, self-distillation has been theoretically analyzed in [38], and its improved performance has been proven by experiments in the literature [39].

Currently, contrastive learning is widely used in self-supervised learning (SSL) [40–42], where each image is considered a separate class and positive samples are pulled closer while negative samples are pushed away. Its basic principle is to adopt the network structure of Siamese [43] and calculate the output loss of the two branches of the network from the inputting of positive and negative sample pairs of data. So the network can learn the features that can bring similar samples closer and dissimilar samples farther by using the InfoNCE loss [40]. MoCo [41,42] establishes a dynamic dictionary, which stores a large number of negative samples. MoCo emphasized that the size of sample pairs was very important for contrastive learning. However, SimCLR [44] believed that the way to construct negative examples was also very important. SimCLR used more data augmentation and added a projector after the encoder, which can greatly improve the effect. MoCov2 [42] verified the effectiveness of SimCLR by implementing two improvements in the MoCo framework. Contrastive learning requires many negative examples for comparison, which is time-consuming and memory-consuming. Therefore, FAIR and INRIA also launched a new method, SWAV [45], which clusters all kinds of samples and then distinguishes the clusters of each class. MoCo, SimCLR, and other contrastive learning methods rely on negative samples. Without negative samples, BYOL [46] depends on two neural networks, an online network and a target network, which interact and learn from each other. Continuing the ideas of BYOL, Xinlei Chen, and Kaiming He studied the Siamese network [43] and found that stopping the gradient was the key to avoiding collapse, so they proposed SimSiam [47].

2.3. Normalization

In order to make the input data of the neural network independent and identically distributed, normalization was introduced to limit the input to a certain range. BN (Batch Normalization) [48], GN (Group Normalization) [49], LN (Layer Normalization) [50], and IN (Instance Normalization) [51] are several classical normalization algorithms. However, BN was limited by the batch size, and its performance was poor when the batch size was small. Moreover, the above methods are all global normalization, which means that spatial information was not utilized and all features were normalized by the same mean and variance. Anthony Ortiz proposed LCN (local context normalization) [52], which utilized local context information and normalized according to the local neighborhood and corresponding statistical values to improve performance. It is also applicable to various batch sizes and transfer learning. In our experiment, BN and LCN, which were more suitable for this task, were selected as normalization methods, respectively.

2.4. Activation Function

The activation function is an important part of the deep learning network. Nonlinear characteristics were introduced into the network to make the network learn more complex

and deeper nonlinear relationships. Additionally, it was used to enhance the representational ability of the network. At present, the common activation functions are *sigmoid*, *tanh*, and *ReLU* [53], of which *ReLU* is the most widely used. Compared with *sigmoid* and *tanh*, *ReLU* has the characteristic of no saturation, which avoids the risk of gradient disappearance in backpropagation and improves the performance of the network. However, *ReLU* is a zero-negative segmentation function. When the input value is negative, the gradient of neurons will become zero, causing some neurons to "die" and preventing parameters from being updated correctly. The *Swish* [54] activation function is a self-gated activation function. Its simple structure and the similarities with *ReLU* make it often better than *ReLU* in deeper networks. The nonlinear characteristics of the *Swish* activation function can further amplify the nonlinear characteristics of data so that the network can continue to learn. Its unsaturated characteristics can also effectively avoid the boundlessness of gradient disappearance, effectively overcoming the shortcomings of non-negative input in *ReLU*. So the representation ability of the model is further improved, and the learning efficiency of the network is ensured. Based on the high computational cost of the *sigmoid* function, this paper introduces *Swish*'s extended function, *Hard Swish* [54], to activate the function.

3. Materials and Methods

3.1. Self-Supervised Learning for Pre-Training

In order to enhance the generalization of the network, the improved self-distillation named *Simdis2x* in the regularization method was implemented in a self-supervision environment. The self-distillation method in [55] and the method of using only positive samples for self-supervision in SimSiam [47] were borrowed. The biggest difference between the literature [55] and the proposal was mainly the softened hard target, which was used in the supervised learning scenario during self-distillation. The predicted result Pre(x) and target H(x) were linearly combined as a softened hard target to supervise the next epoch. However, the proposed method was a self-supervised method, and only positive samples were used, like SimSiam [47]. In self-distillation, target H(x) was not used; instead, it distilled the next epoch by using the linear combination of the change information brought by the image augmentation [56] and the current prediction result. Like the literature [55], it used the progressive method. At the beginning of the epoch, the weight of self dis loss was small. With the increase of epoch, the proportion of its loss in the total loss increased, and finally it was combined with the cosine loss in SimSiam. The total loss function formula is as follows:

$$L_{(D,S)} = \alpha_t \cdot L_D + L_S, \tag{1}$$

But L_D means the symmetry loss of self-knowledge distillation in each image, as shown in Formula (2) (α_t is a hyperparameter). L_S refers to the loss function of the SimSiam Network.

$$L_D = 0.5\{\cos ine[P_1, f_2(t-1))] + \cos ine[P_2, f_1(t-1)]\}, \tag{2}$$

Here, $t-1$ refers to the last epoch, cosine refers to the minimum of the negative value of the cosine similarity of the two views, P is the current prediction result, and $f(\cdot)$ is the change information generated by the image enhancement of the previous epoch.

The feature information from the last epoch generated through data augmentation and the color changes were processed by a multi-layer perceptron, then linearly combined with the projector in current progress to generate target features.

As shown in Formulas (3) and (4):

$$f_1(t-1) = V_1(t-1) + 0.5 \cdot M[\beta_1(\mathrm{t}-1) + \delta_1(t-1)], \tag{3}$$

$$f_2(t-1) = V_2(\mathrm{t}-1) + 0.5 \cdot M[\beta_2(\mathrm{t}-1) + \delta_2(\mathrm{t}-1)], \tag{4}$$

where, $V(t-1)$ refers projector result of the last epoch, M refers to the MLP process. β is the augmentation difference after cropping, and δ is the augmentation difference caused by color change.

3.2. The Location Feature Extraction Strategy

3.2.1. Location Feature Extraction Structure

The network structure of the location extraction module (referred to as the LFE module) is shown in Figure 1. Based on the idea of CSPNet, this module was divided into two parts. Part I: First, input features go through three times ConvNHS module to downsample. ConvNHS is a convolution module including convolution, normalization [48], and the *Hard Swish* activation function [54]. Normalization accelerates convergence and prevents overfitting. The *Hard Swish* [54] function improved the learning ability of network nonlinear representation without greatly increasing the computational cost. The global spatial context mapping was established through three different dimensional convolution modules. After that, the feature map entered the attention module, including channel attention and spatial attention. Because the location feature extraction of the channel attention mechanism was not strong [32], the spatial attention in the attention module was used to obtain more spatial information. In addition, the cross-channel information interaction of the ConvNHS module can make up for the lack of spatial information in channel attention to a certain extent.

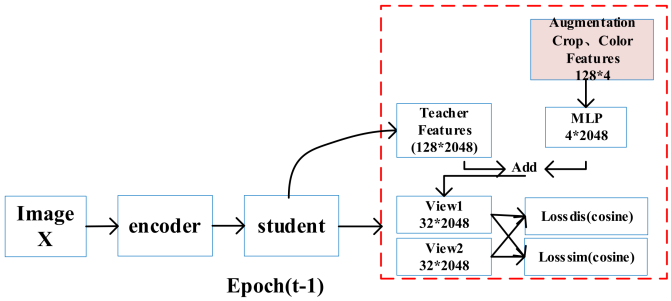


Figure 1. Self−supervised self-distillation module: Simdis2x.

Part II: The second branch retained the initial fine-grained feature information to avoid the loss of global spatial information. The later experiments also proved that this structure does reduce the location errors and achieve the location feature extraction process (Figure 2).

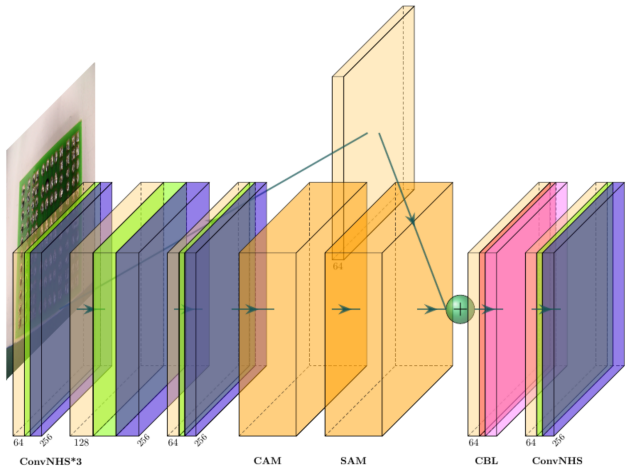


Figure 2. Location feature extraction structure. ConvNHS is a convolution block that contains convolution, normalization, and the *Hard Swish* activation function. CAM: channel attention mechanism. SAM: spatial attention mechanism. CBL, a convolution module, includes convolution, batch normalization, and the *LeakyReLU* function.

The specific model architecture is as follows: The feature map (H × W × C1. H, W: height and width of the feature map. (C1: number of feature map channels) entering the module is processed in two branches. Branch 1, the ConvNHS module, including convolution, normalization, and activation functions, was written in *Hard Swish*. This module used convolution kernels of different sizes to change the number of channels, achieving cross-channel information interaction and the establishment of spatial context mapping. The normalization accelerated convergence, and *Hard Swish* completed the nonlinear modeling of input in a deeper network. Under the condition of keeping the original size of the feature map, the number of channels of the feature map was changed to C2, so the cross-channel information interaction was realized.

Under the condition that the amount of parameters does not increase significantly, the ConvNHS module enhanced the learning ability of nonlinear features and enhanced the learning of the location relationship in the deeper network. The problem that location feature extraction was not strong in the next channel attention mechanism was resolved, and a feature map $F_{context}$ was obtained.

$$F_{context} = \text{ConvNHS}(F), \tag{5}$$

Next, it was the channel attention mechanism: two one-dimensional feature maps were obtained through global max pooling and global average pooling, and a channel attention vector was obtained through a multi-layer perceptron (MLP) and activation function processing. With the previous ConvNHS module processing, the channel attention mechanism could interact with more cross-dimensional information and enhance the attention to spatial information.

$$F_{c,context} = V_c \otimes F_{context}, \tag{6}$$

Here, $F_{c,context}$ is different from the general channel attention feature map, which is the feature map that obtains interactive information and spatial context mapping. $\otimes$ denotes element-wise multiplication.

Deploy spatial attention after channeling attention to enhance attention to spatial information. The feature map $F_{c,context}$ output by the channel attention module was used as its input feature map. First, based on the dimension of the channel, global max pooling and average pooling were carried out to obtain two feature maps (H × W ×1). Then, the two feature maps were concatenated based on the channel dimension, using a 7 × 7 convolution kernel to perform channel dimension reduction. The weights of the spatial dimensions $F_{s,context}$ were generated. At this point, the processing of branch 1 ends.

Use spatial attention for $F_{c,context}$, where V_s is the spatial attention vector obtained by the spatial attention mechanism:

$$F_{s,cross} = V_s \otimes F_{c,context}, \tag{7}$$

Branch 2 performed a simple convolution on the entered feature map, and adjusted the number of channels to the same number as channels C2 in branch 1. The feature maps of branch 1 and branch 2 were concatenated to obtain the feature map with H × W × 2C2. The concatenated feature map was sent into the normalization and activation functions of *LeakyReLU*. Finally, the ConvNHS module integrated the features, and the final feature map F' was the output.

$$F' = \text{ConvNHS}[f_{concate}(F_{s,context}; F_{branch2})], \tag{8}$$

where $f_{concate}$ refers to concatenation, and $F_{branch2}$ refers to the feature map obtained after convolution in branch 2.

Take a practical example to illustrate the dimensional change of this structure. Supposing the input feature map F (8 × 8 × 64. Where 8 × 8 was the height and width of the feature map and 64 is the number of channels) entered the ConvNHS module of branch 1 to interact with cross-dimensional information. The number of channels was changed to

32, the size was left the same, and a new feature map $F_{context}$ ($8 \times 8 \times 32$) was obtained. Then the channel attention module obtained two feature maps ($1 \times 1 \times 32$) through GMP (global max pooling) and GAP (global average pooling). After a series of convolutions, activation functions, additions, and multiplications, the two feature maps were integrated into the channel feature map $F_{c,context}$($8 \times 8 \times 32$). In the spatial attention module, two feature maps were generated after GMP and GAP, and the spatial feature map $F_{s,context}$ ($8 \times 8 \times 32$) was obtained after concatenation. The original feature map entered branch 2 to obtain $F_{branch2}$ after a simple convolution ($8 \times 8 \times 32$). After concatenating the feature maps of branches 1 and 2, going through again ConvNHS processing, a final output feature map F' ($8 \times 8 \times 64$) was generated.

3.2.2. Instantiation

In the convolution block ConvNHS of the LFE block, we chose the normalization method through careful consideration and experimental verification. In the specific implementation, we chose the most classic BN as the normalization method of the CPU environment to verify the performance of global normalization. At the same time, in the GPU environment, the latest popular LCN was used to verify the performance of local normalization. In addition to the above deployment, the activation function used *Hard Swish*, which can have advantages in deeper networks.

The LFE block can be plug-and-play, which was deployed in the YOLOv4 [57]. Firstly, only one LFE was inserted into the network for training and testing. While inspired by Arunabha M. Roy's team [58], inserting five location feature extraction modules has been tested for training and testing. The insertion positions are shown in Figure 3. After the images enter the backbone network CSPDarknet53 [57] for feature extraction, the SPP module is deployed to increase the receptive field so that any size feature map can be converted into a fixed-size feature vector. Then the PANet [59] is used for feature fusion. Five places were selected after the SPP [60] module and concatenation separately, both of which were rich in information interaction. Finally, the three Yolo heads made predictions, respectively. The later experiments (see Section 4.4 for specific experiment results) also proved that this structure does reduce the location errors and accomplish the location feature extraction process.

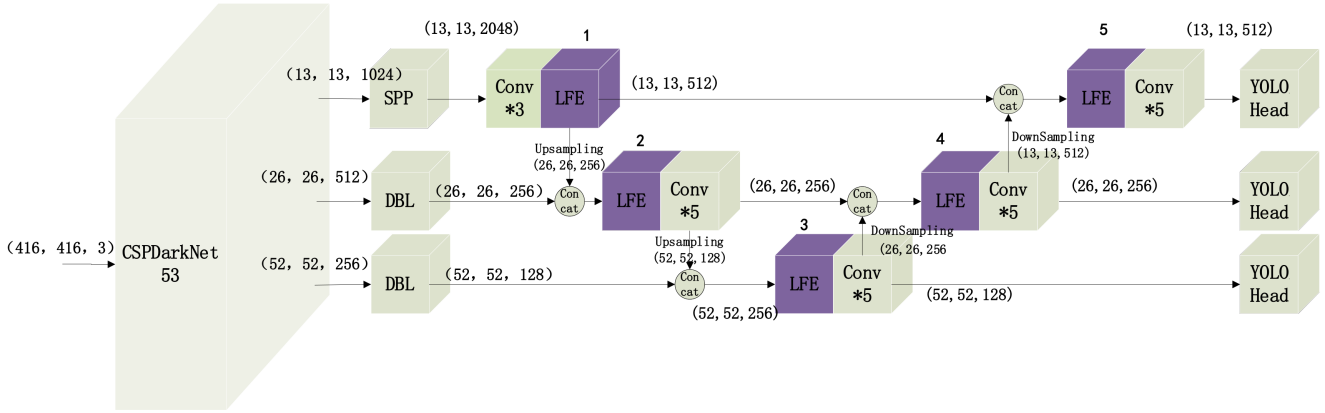


Figure 3. Improved YOLOv4. Scheme I: Insert a position feature extraction module in positions 1–5, respectively. Scheme II: Positions 1–5 are all inserted into the LFE block. DBL: DarkNetConv2D + Batch Normalization + Mish; SPP: Spatial Pyramid Pooling; Conv: Convolutional Layer; Concat: Concatenation.

4. Experiment and Results

4.1. Dataset Pre-Processing

A self-made dedicated dataset for small target detection took images of the solder joints of the circuit board, which included a total of 8600 pictures of solder joint defects. They were divided into training sets and test sets according to the ratio of 9:1 and marked in PASCAL VOC format using the labelImg tool. There were six types of solder joint defects, which were marked as shot out, oxidation, welding missed joint, extensive solver, solder projection, and inveracious solder. We randomly divided 7735 pictures as the training set and 860 pictures as the test set.

The PASCAL VOC dataset [61] is a common public data set used in target detection tasks, including 20 classes of targets such as train, cat, and sofa. The experiment used VOC2012 for training, including 23,080 pictures and 54,900 targets, and used VOC2007 for testing, including 9963 pictures and 24,640 targets.

Public PCB dataset: The PCB defect dataset has collected a total of 693 pictures. They were divided into training sets and test sets according to the ratio of 9:1.

In order to expand the training samples, various forms of data augmentation are used for the training set. Firstly, the traditional methods of data augmentation are adopted, for instance, rotation, noise, brightness transformation, random clipping, and amplification, to simulate the differences in shooting time, angle, and definition. So the robustness and generalization of the model improve, and the network can fully learn the detailed characteristics of solder joint defects. In addition, mosaic data enhancement was also introduced.

4.2. Experimental Environment Setting

To prove the effectiveness of our strategy in different environments, different experimental verifications were carried out on the GPU and CPU. Under the CPU, the TensorFlow framework was deployed to build a model of YOLOv4 with the backbone network of CSPDarkerNet53 [57]. At the same time, based on the GPU, the PyTorch framework was used to build a model for the same training and testing. Specific parameter settings may vary depending on the environment. For example, the normalization method was selected according to the specific task and environment configuration. Here we chose two methods for experiments: batch normalization under the TensorFlow framework and local context normalization under the PyTorch framework. See Table 1 for the specific training parameters of the network.

Table 1. Training parameters and environment.

Training Parameters	Parameter Value
Input_shape	[416, 416]
Initial learning rate	0.01
Min learning rate	Initial learning rate * 0.01
Optimizer_type	sgd
Learning rate_decay_type	cos
CPU	Intel(R) Xeon(R) CPU E5-2640 v3 @ 2.60GHZ (4 processors)
GPU	GeForce GTX 1080Ti/PCIe/SSE2

4.3. Evaluation Metrics

The most important detection metric for object detection is mean average precision (*mAP*), which is the mean value of *AP* for each class. The results of object detection were presented in binary classification, and there were four categories: true positive (*TP*), false positive (*FP*), true negative (*TN*), and false negative (*FN*). Based on the number of the four categories, precision and recall were defined as

$$P = \frac{TP}{TP + FP} \tag{9}$$

$$R = \frac{TP}{TP + FN} \tag{10}$$

AP is the area under the precision-recall(PR) curve:

$$AP = \int_0^1 P(R)dR \tag{11}$$

mAP is defined as

$$mAP = \frac{1}{N}\sum_N AP \tag{12}$$

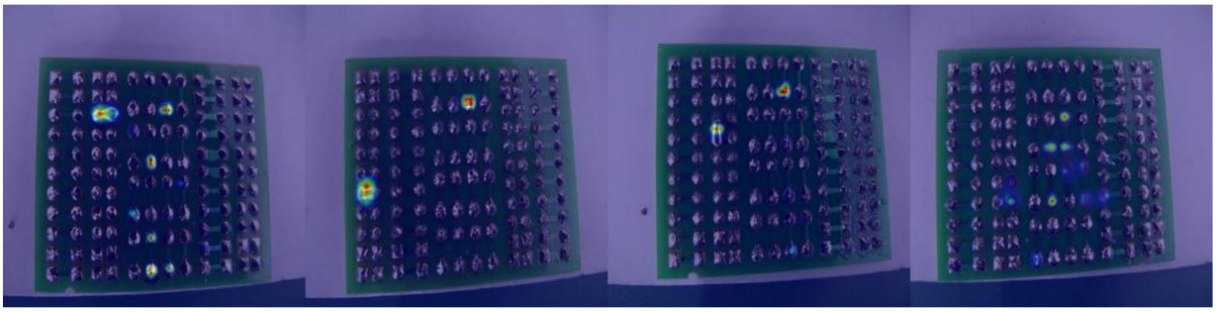
where N represents the number of classes.

4.4. Experimental Result

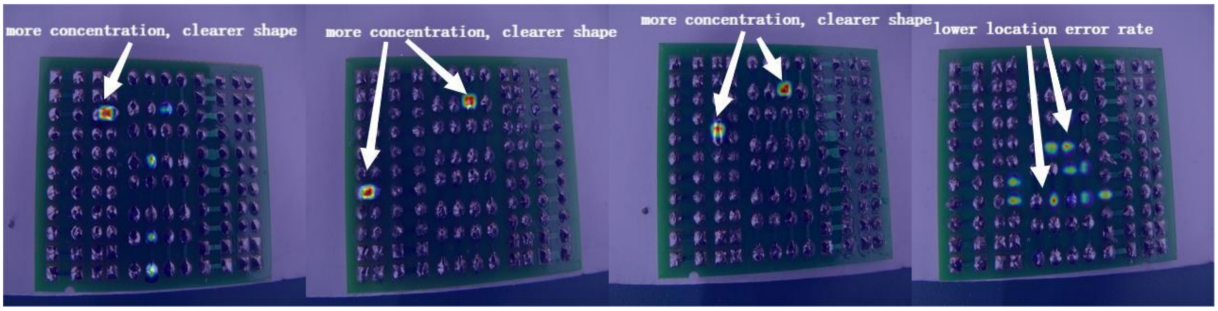
In order to verify the performance of the LFE module, a visual thermodynamic diagram, a distribution diagram of false positive errors, and a series of comparisons between the proposed network and the original network on mAP were conducted.

4.4.1. Visual Experiment on the Effect of Location Feature Extraction

Heat map visualization is a tool commonly used in image analysis. It aggregates a large amount of data and uses a progressive color band to show the saliency effect. The more the color tends toward red, the better the recognition effect. Under CPU conditions, a heat map was used to visualize the recognition effect of the improved network with five LFE blocks inserted. In addition, the improved network was compared with the original YOLOv4. The results shown in Figure 4a show the results of the original YOLOv4, and Figure 4b shows the results of the improved YOLOv4. Compared with Figure 4a, the red area on the defect in Figure 4b has a darker color, a larger area, more concentration, a clearer shape, and a lower location error rate, indicating the improved network can more accurately locate the defect target, and the contour of the identified defect is more detailed. On the whole, the error rate of the improved network was lower, which shows that the LFE structure was effective for the extraction of location features.



(**a**) Thermal visualization of the original YOLOv4

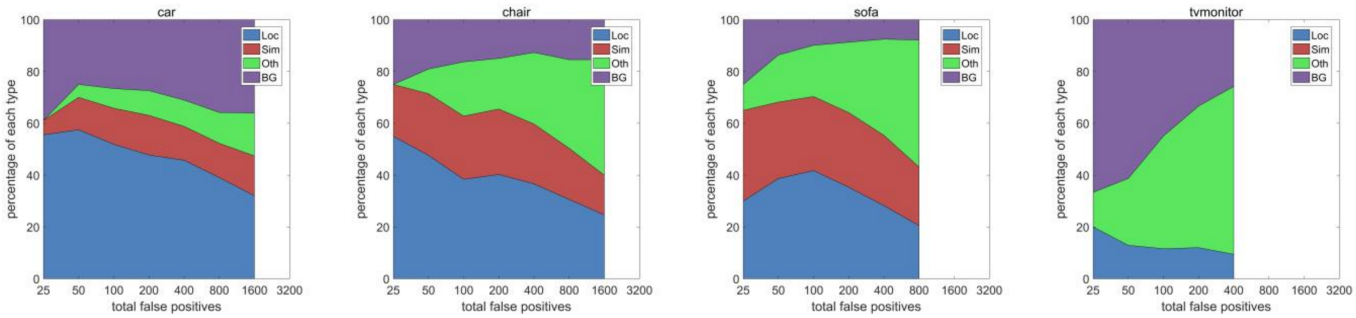


(**b**) Thermal visualization of the improved YOLOv4

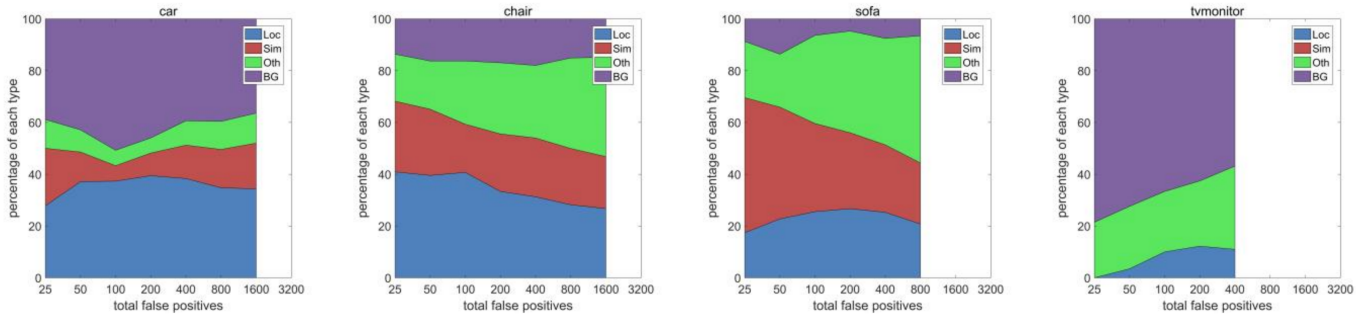
Figure 4. Visualization results of the heat map.

4.4.2. False Positive Error Analysis Experiment of Characteristics

In order to further verify the LFE module effect, the diagnosing methods for false positive (*FP*) errors in object detection proposed by Hoiem et al. [62] were used. Figure 5 was one of the methods that showed the evolution of the proportion of four false positive errors with the increase in the total number of false positives. In Figure 5, the network with five LFE blocks inserted (Figure 5b compared with the original YOLOv4 Figure 5a) in terms of the error distribution on the PASCAL VOC 07+12 datasets. The *x*-axis represents the increasing number of false positive labels, and the *y*-axis represents the proportion of each error under a certain number of false positive labels, expressed in area. Through a change in the area occupied by the four errors, it can be seen that the diminution of the area occupied by the location errors was the most obvious, thus proving the effectiveness of our method for location feature extraction.



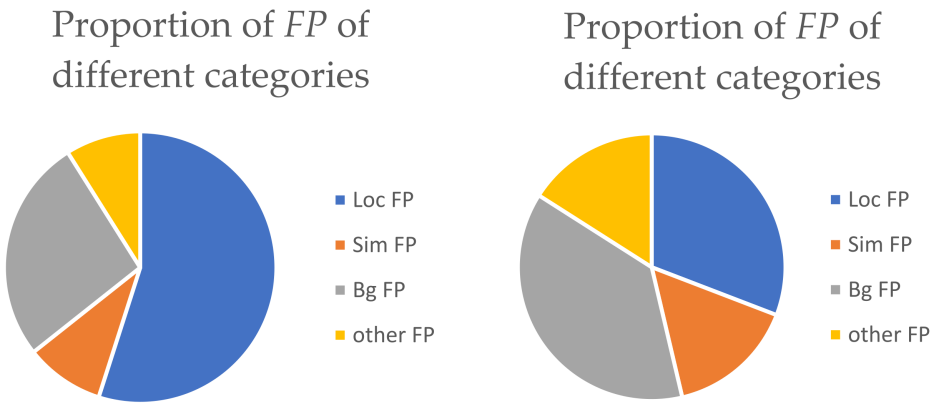
(**a**) Diagnosis result of the original YOLOv4.



(**b**) Diagnosis result of the YOLOv4 added with LFE module.

Figure 5. Error distribution and diagnosis diagram. The first row is the diagnosis result of the original YOLOv4, and the second row is the diagnosis result of the YOLOv4 added with the location feature extraction structure. Loc: position error; Sim: similarity error; BG: background error; Oth: other error.

According to the change in the number of various errors, the proportions of location errors, similarity errors, background errors, and other errors (see Figure 6 for the results) were analyzed. We found that location errors accounted for the largest proportion in the original YOLOv4, they decreased most after the LFE modules insertion, and their proportion of the total errors also decreased most. It proved the effectiveness of our LFE module in reducing location errors and extracting location features.



(**a**) YOLOv4, *mAP* = 84.24% (**b**) YOLOv4_LFE, *mAP* = 84.47%

Figure 6. Proportion of *FP* in different categories.

4.4.3. Experiment on Exploring the Effect of LFE Blocks Inserted into the Network

Experimental results carried out on the original YOLOv4 and the improved YOLOv4 with 5 LFE blocks inserted under the CPU are shown on the left side of Table 2. Compared with the original YOLOv4, the *mAP* of the improved YOLOv4 on all three datasets was improved. Due to the small number of samples in the PCB dataset itself, *mAP* was relatively small. However, results were still improved, indicating that the LFE module was also effective for other datasets and had good generalization ability. Meanwhile, in the GPU environment, one LFE block was integrated into YOLOv4 at the third location in Figure 3, and experiments were conducted on three different datasets for comparison. The experimental results are shown on the right side of Table 2. Through the verification of two different experimental environments, it can be proven that the location feature extraction structure can improve network accuracy no matter the GPU or CPU conditions. In addition, the LFE module is universal across different datasets, which can prove the existence of generality.

Table 2. *mAP* of original YOLOv4 and improved YOLOv4 in different environments.

CPU			GPU				-
Network	**Training Parameters**	***mAP*@0.5**	**Network**	**Training Parameters**	***mAP*@0.5**	***mAP*@0.5:0:95**	**Dataset**
YOLOv4	Epoch = 50 Batch size = 32	43.65%	YOLOv4	Epoch = 50 Batch size = 32	80.2%	58.1%	Small target
YOLOv4 + Five LFE blocks	Epoch = 50 Batch size = 32	**45.45%**	YOLOv4 + one LFE block	Epoch = 50 Batch size = 32	**81.6%**	**60%**	
YOLOv4	Epoch = 50 Batch size = 32	36.29%	YOLOv4	Epoch = 150 Batch size = 64	62%	24.7%	PCB
YOLOv4 + Five LFE blocks	Epoch = 50 Batch size = 32	**38.89%**	YOLOv4 + one LFE block	Epoch = 150 Batch size = 64	**67.3%**	**27.7%**	
YOLOv4	Epoch = 50 Batch size = 32	84.24%	YOLOv4	Epoch = 100 Batch size = 64	82.1%	57.1%	Pascal VOC 07 + 12
YOLOv4 + Five LFE blocks	Epoch = 50 Batch size = 32	**84.47%**	YOLOv4 + one LFE block	Epoch = 100 Batch size = 64	**82.6%**	**58.7%**	

The bold font in the table is the better result in each group of comparison. Same as Tables 3 and 4.

4.4.4. The Joint Experiment of Self-Supervised Self-Knowledge Distillation and LFE Block under GPU Condition

In addition to the above independent network exploration, a joint experiment of self-knowledge distillation, pre-training, and downstream tasks was conducted. Improved self-distillation was used to do pre-training on the Simdis2x network and was then trained and tested on the Pascal VOC, PCB, and our small target dataset. At the same time,

during the downstream task training, the backbone of the training network was changed to ResNet18. Table 3 shows our experimental results.

Table 3. Joint experimental results of improved self-distillation pre-training and location feature extraction structure.

Pretext Dataset	Pretext Task	Downstream Dataset	Baseline	Training Parameters	*mAP*@0.5	*mAP*@0.5:0:95
Random initialization	-	PCB	ResNet18	Epoch = 50 Batch size = 32	48.2%	18.3%
ImageNet100	**Simdis2x**		ResNet18	Epoch = 50 Batch size = 32	59.7%	23.6%
Random initialization	-		**ResNet18 + LFE block**	Epoch = 150 Batch size = 64	48.7%	19.1%
ImageNet100	**Simdis2x**		**ResNet18 + LFE block**	Epoch = 150 Batch size = 64	**63.1%**	**23.9%**
Random initialization	-	Small target	ResNet18	Epoch = 100 Batch size = 64	80.5%	59.2%
ImageNet100	**Simdis2x**		ResNet18	Epoch = 100 Batch size = 64	80.6%	59%
Random initialization	-		**ResNet18 + LFE block**	Epoch = 50 Batch size = 32	80.5%	59.7%
ImageNet100	**Simdis2x**		**ResNet18 + LFE block**	Epoch = 50 Batch size = 32	**81.5%**	**61.5%**
Random initialization	-	Pascal VOC	ResNet18	Epoch = 150 Batch size = 64	83.6%	60.3%
ImageNet100	**Simdis2x**		ResNet18	Epoch = 150 Batch size = 64	83.6%	60.5%
Random initialization	-		**ResNet18 + LFE block**	Epoch = 100 Batch size = 64	83.6%	60.8%
ImageNet100	**Simdis2x**		**ResNet18 + LFE block**	Epoch = 100 Batch size = 64	**83.6%**	**61.2%**

The training conditions are in bold to highlight the changes based on the initial conditions, and the mAP results are in bold to highlight better training effects. Same as Table 4.

Comparing the experimental results of different groups, it can be concluded that the proposed self-distillation and LFE modules have improved the network's *mAP*, respectively, and that the combined use of the two modules is the best. The location feature extraction strategy is also suitable for self-supervised scenarios and has advantages for the improvement of downstream tasks.

In addition, after inserting the module into the two latest baseline networks [15,63], the detection accuracy was improved compared with the original networks in Table 4, which proves the effectiveness of the two LFE blocks.

Table 4. Effect of the LFE module in other advanced algorithms.

Baseline	*mAP*@0.5	*mAP*@0.5:0:95
YOLOv7	90.3%	71.2%
YOLOv7 + two LFE blocks	**90.5%**	**71.2%**
FA_SSD	70.3%	67.5%
FA_SSD + two LFE blocks	**71.1%**	**68.4%**

4.5. Ablation Study

A series of ablation experiments were conducted to explore how to build LFE blocks and how to insert them into a network to achieve the best detection accuracy.

4.5.1. Design of the ConvNHS Module

In the first branch of the LFE block, before the feature map enters the attention mechanism, it will be preprocessed by the convolution block ConvNHS. Whether effective location information can be extracted from the later attention mechanism depends largely on this module. Therefore, the ConvNHS module played a key role. The ConvNHS module is mainly composed of convolution, normalization, and activation functions. Different normalization methods were tested in the ConvNHS module under the GPU environment, and LCN achieved the best effect, followed by BN in Figure 7a. Therefore, LCN with the best effect was used in the GPU, and BN with the best effect in global normalization was used in the CPU.

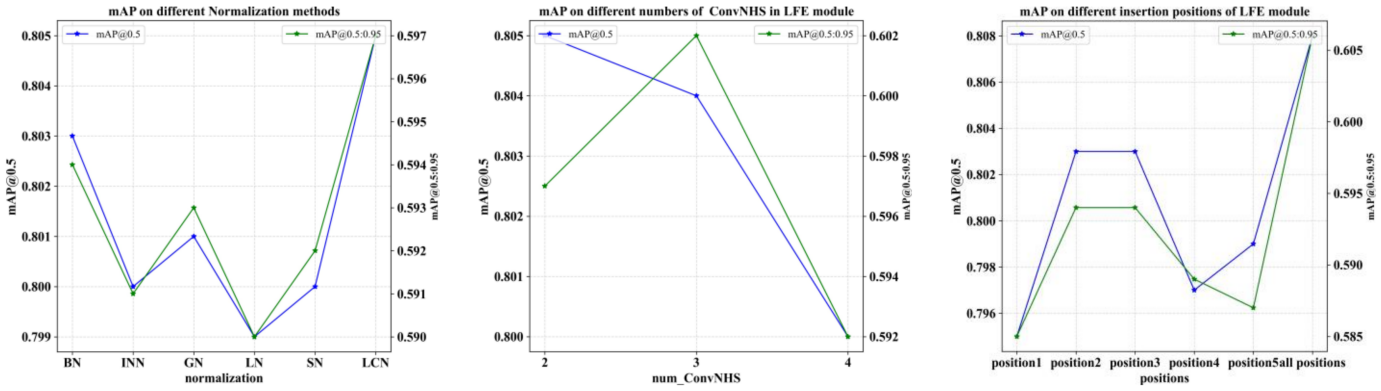


Figure 7. *mAP* line charts of the different integration schemes of LFE. (**a**) shows the experimental results of different normalization methods in the LFE module. (**b**) shows the experimental results of different numbers of ConvNHS in the LFE module. (**c**) shows the experimental results of LFE modules at different insertion positions of YOLOv4.

Based on the above experiments, LCN was selected for the normalization method. Next, the number of ConvNHS modules was further explored. As shown in Figure 7b, when three ConvNHS modules were inserted into the network, the detection effect was the best. Therefore, three ConvNHS modules were designed for subsequent experiments.

4.5.2. Position and Quantity of LFE Block Inserted into YOLOv4

One LFE block was respectively inserted at the five locations in Figure 3 for the experiment under GPU conditions, and then all five positions were inserted for the experiment to explore the impact of insertion position and number on the recognition effect. As shown in Figure 7c, for single insertion, it was best to insert at the second or third position. We determined that the third position was the connection point of the whole PANet, from bottom to top or from top to bottom, which had rich interactive information. The location feature extraction at this position was the most effective and abundant. From a global perspective, the five full insertion results were the best, indicating that the effect of LFE structures inserted into the network can be accumulated.

Based on the above experiments, the best LFE deployment strategy was determined; that is, each LFE block contains three ConvNHS structures, using LCN or BN for normalization. The insertion location is the location with the most abundant interactive information.

5. Conclusions

A novel positioning strategy was proposed: location feature extraction, which established spatial context mapping and improved attention mechanisms with the CSPNet idea to strengthen the network's attention to location features, obtain more space information, and increase the accuracy of network detection. In addition, the proposed self-knowledge distillation was used for pre-training, which strengthened the generalization ability of the

network. Self-made solder joint defects as the object of small target detection and public datasets (Pascal VOC, PCB) used for general verification. The LFE module was integrated into the YOLOv4 network, and through the visualization of a heat map, diagnosis of the false positive error of location, and comparison of *mAP*, the superiority of the improved network was verified. In the future, the extraction structure of location features can be more in-depth and perfected, and it can be extended from location features to other features so as to design more targeted strategies.

Author Contributions: Conceptualization, G.L., J.L., S.Y. and R.L.; methodology, G.L., J.L., R.L. and S.Y.; writing—original draft preparation, G.L. and J.L.; writing—review and editing, G.L., J.L. and S.Y. All authors have read and agreed to the published version of the manuscript.

Funding: We acknowledge the support from the China Scholarship Council, which enabled the author Gaohua Liu to study at the Singapore University of Technology and Design. This research was funded by the [China Postdoctoral Science Foundation] grant number [2020M680883].

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Public Pascal VOC dataset official website link: http://host.robots.ox.ac.uk/pascal/VOC/ and public PCB dataset link: https://robotics.pkusz.edu.cn/resources/dataset/.

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

References

1. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
2. Cai, Z.; Vasconcelos, N. Cascade R-CNN: Delving into high quality object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 6154–6162.
3. Chen, K.; Pang, J.; Wang, J.; Xiong, Y.; Li, X.; Sun, S.; Feng, W.; Liu, Z.; Shi, J.; Ouyang, W.; et al. Hybrid task cascade for instance segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4974–4983.
4. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common objects in contex. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Springer: Cham, Switzerland, 2014; pp. 740–755.
5. Chen, C.; Liu, M.Y.; Tuzel, O.; Xiao, J. R-CNN for small object detection. In Proceedings of the Computer Vision–ACCV 2016: 13th Asian Conference on Computer Vision, Taipei, Taiwan, 20–24 November 2016; Springer: Cham, Switzerland, 2016; pp. 214–230.
6. Liu, Y.; Sun, P.; Wergeles, N.; Shang, Y. A Survey and Performance Evaluation of Deep Learning Methods for Small Object Detection. *Expert Syst. Appl.* **2021**, *172*, 114602. [CrossRef]
7. Han, W.; Chen, J.; Wang, L.; Feng, R.; Li, F.; Wu, L.; Tian, T.; Yan, J. Methods for Small, Weak Object Detection in Optical High-Resolution Remote Sensing Images: A survey of advances and challenges. *IEEE Geosci. Remote Sens. Mag.* **2021**, *9*, 8–34. [CrossRef]
8. Gao, X.; Mo, M.; Wang, H.; Leng, J. Recent Advances in Small Object Detection. *J. Data Acquis. Process.* **2021**, *36*, 391–417.
9. Chen, G.; Wang, H.; Chen, K.; Li, Z.; Song, Z.; Liu, Y.; Chen, W.; Knoll, A. A Survey of the Four Pillars for Small Object Detection: Multiscale Representation, Contextual Information, Super-Resolution, and Region Proposal. *IEEE Trans. Syst. Man Cybern. Syst.* **2020**, *52*, 936–953. [CrossRef]
10. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017. [CrossRef]
11. 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 Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
12. Fu, C.Y.; Liu, W.; Ranga, A.; Tyagi, A.; Berg, A.C. DSSD: Deconvolutional Single Shot Detector. *arXiv* **2017**, arXiv:1701.06659.
13. Kong, T.; Yao, A.; Chen, Y.; Sun, F. HyperNet: Towards Accurate Region Proposal Generation and Joint Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 845–853.
14. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, Columbus, OH, USA, 23–28 June 2014.
15. Lim, J.S.; Astrid, M.; Yoon, H.J.; Lee, S.I. Small Object Detection using Context and Attention. In Proceedings of the 2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), Jeju Island, Republic of Korea, 13–16 April 2021; pp. 181–186.

16. Aguinaldo, A.; Chiang, P.Y.; Gain, A.; Patil, A.; Pearson, K.; Feizi, S. Pearson and S. Feizi. Compressing GANs using Knowledge Distillation. *arXiv* **2019**, arXiv:1902.00159.
17. Ahn, S.; Hu, S.X.; Damianou, A.; Lawrence, N.D.; Dai, Z. Variational Information Distillation for Knowledge Transfer. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019.
18. Aditya, S.; Saha, R.; Yang, Y.; Baral, C. Spatial Knowledge Distillation to Aid Visual Reasoning. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa Village, HI, USA, 7–11 January 2019.
19. Jing, L.; Tian, Y. Self-Supervised Visual Feature Learning With Deep Neural Networks: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *43*, 4037–4058. [CrossRef]
20. 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 (CVPRW), Seattle, WA, USA, 14–19 June 2020; pp. 1571–1580.
21. 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 (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788. [CrossRef]
22. Cui, L.; Ma, R.; Lv, P.; Jiang, X.; Gao, Z.; Zhou, B.; Xu, M. MDSSD: Multi-scale deconvolutional single shot detector for small objects. *arXiv* **2018**, arXiv:1805.07009. [CrossRef]
23. Liu, Z.; Li, D.; Ge, S.S.; Tian, F. Small traffific sign detection from large image. *Appl. Intell.* **2020**, *50*, 1–13. [CrossRef]
24. Duan, K.; Du, D.; Qi, H.; Huang, Q. Detecting small objects using a channel-aware deconvolutional network. *IEEE Trans. Circuits Syst. Video Technol.* **2020**, *30*, 1639–1652. [CrossRef]
25. Bell, S.; Zitnick, C.L.; Bala, K.; Girshick, R. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2874–2883.
26. Leng, J.; Liu, Y.; Du, D.; Zhang, T.; Quan, P. Robust obstacle detection and recognition for driver assistance systems. *IEEE Trans. Intell. Transp. Syst.* **2020**, *21*, 1560–1571. [CrossRef]
27. Chen, X.L.; Gupta, A. Spatial Memory for Context Reasoning in Object Detection. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017. [CrossRef]
28. Hu, H.; Gu, J.; Zhang, Z.; Dai, J.; Wei, Y. Relation Networks for Object Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 3588–3597.
29. Jie, H.; Li, S.; Gang, S. Squeeze-and-Excitation Networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 2011–2023.
30. Mo, K. Spatial Transformer Network. *Neural Inf. Process. Syst.* **2017**, 2017–2025.
31. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. *ECCV* **2018**, 3–19.
32. 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 (CVPR), Virtual, 19–25 June 2021. [CrossRef]
33. Hinton, G.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. *arXiv* **2015**, arXiv:1503.02531.
34. Zhang, L.; Bao, C.; Ma, K. Self-Distillation: Towards Efficient and Compact Neural Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 4388–4403. [CrossRef] [PubMed]
35. Zhang, L.; Song, J.; Gao, A.; Chen, J.; Bao, C.; Ma, K. Be Your Own Teacher: Improve the Performance of Convolutional Neural Networks via Self Distillation. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019. [CrossRef]
36. Yang, C.; Xie, L.; Su, C.; Yuille, A.L. Snapshot Distillation: Teacher-Student Optimization in One Generation. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 2854–2863.
37. Phuong, M.; Lampert, C. Distillation-Based Training for Multi-Exit Architectures. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019. [CrossRef]
38. Mobahi, H.; Farajtabar, M.; Bartlett, P. Self-Distillation Amplifies Regularization in Hilbert Space. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 3351–3361.
39. Zhang, Z.; Sabuncu, M.R. Self-Distillation as Instance-Specific Label Smoothing. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 2184–2195.
40. Oord, A.V.D.; Li, Y.; Vinyals, O. Representation Learning with Contrastive Predictive Coding. *arXiv* **2018**, arXiv:1807.03748.
41. He, K.; Fan, H.; Wu, Y.; Xie, S.; Girshick, R. Momentum Contrast for Unsupervised Visual Representation Learning. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020. [CrossRef]
42. Chen, X.; Fan, H.; Girshick, R.; He, K. Improved Baselines with Momentum Contrastive Learning. *arXiv* **2020**, arXiv:2003.04297.
43. Chopra, S.; Hadsell, R.; LeCun, Y. Learning a similarity metric discriminatively, with application to face verification. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–26 June 2005; Volume 1, pp. 539–546.
44. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A Simple Framework for Contrastive Learning of Visual Representations. In Proceedings of the International Conference on Machine Learning, Virtual, 13–18 July 2020; p. 119.

45. Caron, M.; Misra, I.; Mairal, J.; Goyal, P.; Bojanowski, P.; Joulin, A. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. In Proceedings of the 34th Conference on Neural Information Processing Systems, Online, 6–12 December 2020; Volume 33, pp. 9912–9924.
46. Grill, J.B.; Strub, F.; Altché, F.; Tallec, C.; Richemond, P.; Buchatskaya, E.; Doersch, C.; Avila Pires, B.; Guo, Z.; Gheshlaghi Azar, M.; et al. Bootstrap Your Own Latent: A New Approach to Self-Supervised Learning. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 21271–21284.
47. Chen, X.; He, K. Exploring Simple Siamese Representation Learning. *Comput. Vis. Pattern Recognit.* **2021**, 15745–15753.
48. 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; Volume 37, pp. 448–456.
49. Wu, Y.; He, K. Group Normalization. *Int. J. Comput. Vis.* **2019**, *128*, 742–755. [CrossRef]
50. Ba, J.L.; Kiros, J.R.; Hinton, G.E. Layer Normalization. *arXiv* **2016**, arXiv:1607.06450.
51. Ulyanov, D.; Vedaldi, A.; Lempitsky, V. Instance Normalization: The Missing Ingredient for Fast Stylization. *arXiv* **2016**, arXiv:1607.08022.
52. Ortiz, A.; Robinson, C.; Morris, D.; Fuentes, O.; Kiekintveld, C.; Hassan, M.M.; Jojic, N. Local Context Normalization: Revisiting Local Normalization. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020. [CrossRef]
53. Glorot, X.; Bordes, A.; Bengio, Y. Deep Sparse Rectifier Neural Networks. In Proceedings of the fourteenth international conference on artificial intelligence and statistics, Ft. Lauderdale, FL, USA, 11–13 April 2011; Volume 15, pp. 315–323.
54. Howard, A.; Sandler, M.; Chu, G.; Chen, L.C.; Chen, B.; Tan, M.; Wang, W.; Zhu, Y.; Pang, R.; Vasudevan, V.; et al. Searching for MobileNetV3. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324.
55. Kim, K.; Ji, B.; Yoon, D.; Hwang, S. Self-Knowledge Distillation with Progressive Refinement of Targets. *Int. Conf. Comput. Vis.* **2021**, *6*, 6567–6576.
56. Lee, H.; Lee, K.; Lee, K.; Lee, H.; Shin, J. Improving Transferability of Representations via Augmentation-Aware Self-Supervision. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 17710–17722.
57. Bochkovskiy, A.; Wang, C.Y.; Liao, H. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
58. Roy, A.M.; Bose, R.; Bhaduri, J. A fast accurate fine-grain object detection model based on YOLOv4 deep neural network. *Neuural Comput. Appl.* **2021**, *34*, 3895–3921. [CrossRef]
59. 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]
60. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*; IEEE: Piscataway, NJ, USA, 2015; Volume 37, pp. 1904–1916. [CrossRef]
61. Everingham, M.; Eslami SM, A.; Van Gool, L.; Williams CK, I.; Winn, J.; Zisserman, A. The Pascal Visual Object Classes Challenge: A Retrospective. *Int. J. Comput. Vis.* **2015**, *111*, 98–136. [CrossRef]
62. Hoiem, D.; Chodpathumwan, Y.; Dai, Q. Diagnosing Error in Object Detectors. In Proceedings of the 12th European Conference on Computer Vision, Florence, Italy, 7–13 October 2012. Part III.
63. Wang, C.Y.; Bochkovskiy, A.; Liao, H. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv* **2022**, arXiv:2207.02696.

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.