Computer Modeling in
Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2022.020331

ARTICLE

B-PesNet: Smoothly Propagating Semantics for Robust and Reliable Multi-Scale Object Detection for Secure Systems

Yunbo Rao[1,2,*], Hongyu Mu[1], Zeyu Yang[1], Weibin Zheng[1], Faxin Wang[1], Jiansu Pu[1] and Shaoning Zeng[2]

[1]School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, 610054, China

[2]Yangtze Delta Region Institute (Huzhou), University of Electronic Science and Technology of China, Huzhou, 313000, China

[*]Corresponding Author: Yunbo Rao. Email: raoyb@uestc.edu.cn

Received: 17 November 2021 Accepted: 26 January 2022

ABSTRACT

Multi-scale object detection is a research hotspot, and it has critical applications in many secure systems. Although the object detection algorithms have constantly been progressing recently, how to perform highly accurate and reliable multi-class object detection is still a challenging task due to the influence of many factors, such as the deformation and occlusion of the object in the actual scene. The more interference factors, the more complicated the semantic information, so we need a deeper network to extract deep information. However, deep neural networks often suffer from network degradation. To prevent the occurrence of degradation on deep neural networks, we put forth a new model using a newly-designed Pre-ReLU, which inserts a ReLU layer before the convolution layer for the sake of preventing network degradation and ensuring the performance of deep networks. This structure can transfer the semantic information more smoothly from the shallow to the deep layer. However, the deep networks will encounter not only degradation, but also a decline in efficiency. Therefore, to speed up the two-stage detector, we divide the feature map into many groups so as to diminish the number of parameters. Correspondingly, calculation speed has been enhanced, achieving a balance between speed and accuracy. Through mathematical demonstration, a Balanced Loss (BL) is proposed by a balance factor to decrease the weight of the negative sample during the training phase to balance the positives and negatives. Finally, our detector demonstrates rosy results in a range of experiments and gains an mAP of 73.38 on PASCAL VOC2007, which approaches the requirement of many security systems.

KEYWORDS

Object detection; Pre-ReLU; CNN; Balanced loss

1 Introduction

In recent years, computer vision has thrived with the support of deep learning [1–7]. Object detection manifests great practical and research value in many subfields of computer vision. Many object detection approaches proposed in earlier years [8,9] were built based on designing features manually. The shortcoming of those detectors is a lack of generalization. Now, more and more state-of-the-art detection models using convolution neural networks (CNNs) [10,11] were presented. By combining with CNN, object detectors display great power by addressing complex tasks. There are two kinds of detectors: (1) two-stage method [12–20] and (2) one-stage method [21–30]. As for the two-stage detector, firstly, the detector tries to find a set of candidate boxes containing objects. These boxes will introduce important semantic information to classification and regression in the second part. Two-stage methods obtained a better accuracy on PASCAL VOC [31] and MSCOCO [32] than one-stage detectors. Object classification and bounding box regression will take place simultaneously in one-stage approaches. These are very important in building a secure system using image detection.

A new framework termed region proposal network (RPN) was proposed by Ren et al. [15] in 2017. They made a detector titled Faster R-CNN, which used RPN to find the bounding box. Object detection was regarded as a regression other than classification task in which you look only once (YOLO) by Redmon et al. [22], which splits the input image into multiple parts for predicting the object. Liu et al. [21] merged features from different layers to make an object prediction in a detector, called single shot detector (SSD). SSD has a better performance in comparison with YOLO. Pang et al. [25] exposed some imbalances during the phase of detection: sample imbalance and feature imbalance. Consequently, the IoU-balanced, feature pyramid and a new loss function termed L1 loss is proposed to moderate those imbalances in Libra R-CNN. Despite its attempts to improve themselves, two-stage detectors were still better than one-stage detectors till the advent of RetinaNet. Lin et al. [24] depicted that the severe imbalance between the foreground and background samples is the critical determinant to influencing the accuracy of one-stage detectors, and a newly-designed loss function termed focal loss was proposed to relieve such imbalance.

However, the one-stage approaches and two-stage approaches mentioned above have some weaknesses. The main limitation of the popular methods is the insensitivity to multi-scale objects. First, in most detection datasets, the scales of objects vary a wide range. What is worse, there are some extremely big or small objects. Such scale variation will make it difficult for detectors to take into account the different scales, hence weakening the accuracy of those detectors. All of the methods summarized above have to reshape the input image to a small size at the beginning of training, which inevitably misses some semantic information. Furthermore, more information will be lost in the following convolution process. Accordingly, traditional detectors have lousy performance for this multi-scale object detection. Second, there is a class imbalance problem in existing datasets. The amount of easily classified samples is much more than those hard to classify. Consequently, the detector is not robust enough.

Based on the above shortcoming of those networks, a new network entitled balanced Pre-ReLU ResNet (B-PesNet) is proposed in our work. As a significant idea, we propose a Pre-ReLU residual block that aims to prevent the degeneration of the deep networks, and then we could build a deeper network without losing semantic information and a new loss function for addressing the extremely class imbalance. The main contributions of our work are from two aspects:

(i) A new residual block within a Pre-ReLU is proposed, which is proven to mitigate the degradation problem of deep neural networks when the network becomes deeper. The reason for the degradation of deep neural networks is that too much semantic information is lost in the repeated convolution process. However, our Pre-ReLU retains shallow semantic information so that the deep neural unit can also use shallow information to calculate, thus solving the degradation problem. This brings a more reliable model for many secure systems.

(ii) Balanced Loss (BL) is suggested for solving class imbalance and helping produce a model more reliable than YOLOv2 [27], YOLOv3 [28], and YOLOv4 [26]. Compared with YOLOv5 [33], we found that the accuracy of B-PesNet is comparable, and the speed is slightly inferior. By introducing weight factor α, BL balances the proportions of different samples in the gradient and prevents negative samples from dominating the direction of gradient descent, thereby solving the problem of class imbalance. We give a mathematical deduction, as well as extensive experimental evaluations, to confirm its crucial role for a reliable solution.

The following content in this paper is summarized as follows. We concisely retrospect the research concerned with our work in Section 2. Then, particulars of our approach are given in Section 3, where we introduce our B-PesNet. In Section 4, we will state our datasets and evaluation metrics. Experimental results are listed in Section 5. Finally, Section 6 sums up our outcomes and offers some standpoints for future work.

2 Related Work

With the widespread utilization of deep learning, object detection approaches have evolved from traditional algorithms based on manual design to automatic extraction of features based on deep learning. Modern object detectors can be approximately divided into one-stage [21–24,26–30] and two-stage algorithms [12–20].

2.1 One-Stage Methods

The one-stage method combines extraction and detection into a whole. One-stage approaches will not explicitly extract candidate regions but directly obtain the final detection result. Liu et al. [21] proposed Single Shot Multi-Box Detector (SSD), which utilizes fused features from different layers of the network. Redmon et al. [22] proposed You Only Look Once (YOLO). YOLO integrates the information of the entire image to predict the object at each position. It regards the entire image as input and predicts in the original image: the category and positioning-related information of each part in the area. Lin et al. [23] proposed feature pyramid networks (FPN). The main problem of this method is the insufficiency of object detection in dealing with multi-scale changes. This network architecture can increase the feature map with various semantic information, but rich spatial information is integrated under the premise of less calculation.

2.2 Two-Stage Methods

The working method of the two-stage detection method is to first generate a set of region proposals and then classify and regress these proposals. Compared with the one-stage method, the two-stage method is more accurate but less efficient. Firstly, 2000 region proposals were selected through selective search, and each region proposal will be input into CNN to extract features. And then, the classifier will utilize these features to classify. Subsequently, Girshick et al. [12] proposed an improved version of R-CNN, which is termed Fast R-CNN. For R-CNN, the disadvantage is that the efficiency begins to decline during preprocessing. For instance, use selective search to track potential bounding boxes as input. This search method will make the model spend much time on useless data. Fast R-CNN introduces a new layer of the network, named ROI Pooling, which cleverly puts regression into the network and fuses it with classification into a multi-task model. Ren et al. [15] proposed Faster R-CNN. On the basis of Fast R-CNN, they added a newly-designed structure termed region proposal network (RPN). Therefore, the model will utilize RPN to find candidate boxes, which further improves efficiency.

3 Our Method

In this section, we firstly introduce the B-PesNet. Then BL is proposed to balance the class imbalance during the network training.

3.1 Pre-ReLU

ResNet [34] solved the degradation problem of deep neural networks. However, after deepening the network without limitation, the network degradation problem still exists. It is considering that the residual structure tackles the problem of network degradation by the skip connection, which retains the shallow information. After a large number of experiments and rigorous mathematical verification, it has been found that if we put the ReLU [35] layer in front of the convolution layer, we can get a Pre-ReLU residual block, and improve its performance.

We introduce the Pre-ReLU residual block starting from the standard residual unit:

$$y_l = h(x_l) + F(x_l, W_l). \tag{1}$$

$$x_{l+1} = f(y_l). \tag{2}$$

where x_l is the input of the lth residual unit, and W_l means a set of weight factors associated with the lth residual unit. f denotes the ReLU, h means a skip connection: $h(x_l) = x_l$, and F represents the residual function. As it is known to all, the advantage of ResNet is contributed by the skip connection. This design is able to propagate more lower layers information to the deeper layers. However, in real training, we found that deeper ResNet caused a worse degeneration.

Then, if f is a skip connection: $x_{l+1} = y_l$, we can convert Eq. (2) into Eq. (1):

$$x_{l+1} = x_l + F(x_l, W_l). \tag{3}$$

For the whole network, it becomes:

$$x_L = x_l + \sum_{i=l}^{L-1} F(x_i, W_i), \tag{4}$$

where x_L means the input of any residual unit that is behind the lth unit. Then, we found that, for any unit, the information is able to be propagated from one unit to the others.

In order to build a gate, which is able to propagate information without loss from shallow to deep layer, we put a ReLU layer in front of the convolution layer. As we assumed, the Pre-ReLU residual block builds a clean pathway from shallower to a deeper layer. Then more features can be reserved during convoluting.

3.2 *Feature Map Division*

To decrease the number of parameters and enhance the running speed, we split the feature map into many groups.

First, we divide the feature map into an equal dimension at the dimension level and divide it into multiple groups with the same dimension. The workflow is shown in Fig. 1.

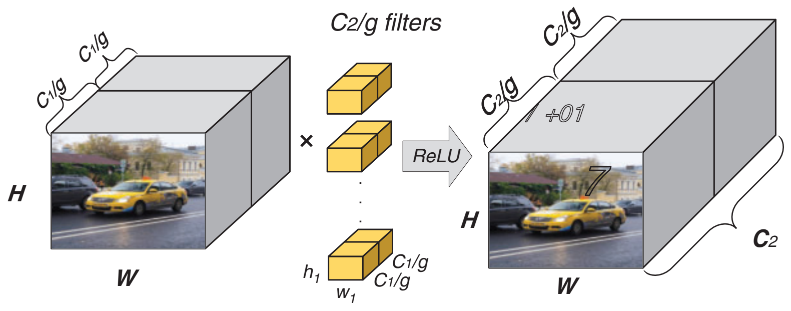

Figure 1: The workflow of dividing. H represents the height of the feature map, and W means the width of the feature map. $\frac{c_1}{g}$ expresses the dimension of each group feature map

Assume the size of the input image is $H \times W \times c_1$, and divide it into g groups, then every group has the same size as $H \times W \times \frac{c_1}{g}$, the corresponding convolution kernel size is $h_1 \times w_1 \times \frac{c_1}{g}$, and there are $\frac{c_2}{g}$ kernels. Then perform the SAME convolution, the size of each output is $H \times W \times \frac{c_2}{g}$. Afterwards, concatenate the results of g groups, we will obtain an output feature map whose size is $H \times W \times c_2$ finally. The amount of parameters of this convolution layer is:

$$h_1 \times w_1 \times \frac{c_1}{g} \times \frac{c_2}{g} \times g = h_1 \times w_1 \times c_1 \times c_2 \times \frac{1}{g}. \tag{5}$$

3.3 *B-PesNet*

Pre-ReLU residual block contains three convolutional layers. The architecture of the block is shown in Fig. 2, and the convolution kernel sizes are 1×1, 3×3, and 1×1, respectively. According to the characteristics of the Pre-ReLU block, we construct a feature extractor named Pre-ReNet, which contains 50 convolutional layers. The net can be divided into five parts based on the size of the output image. The first part is a 7×7 convolution kernel, the second part contains three residual blocks, the third part contains four residual blocks, the fourth part contains six residual blocks, and the fifth part contains three residual blocks. The structure of Pre-ReNet is shown in Table 1.

The number of parameters required for this convolution operation decreases in inverse proportion to the number of groups.

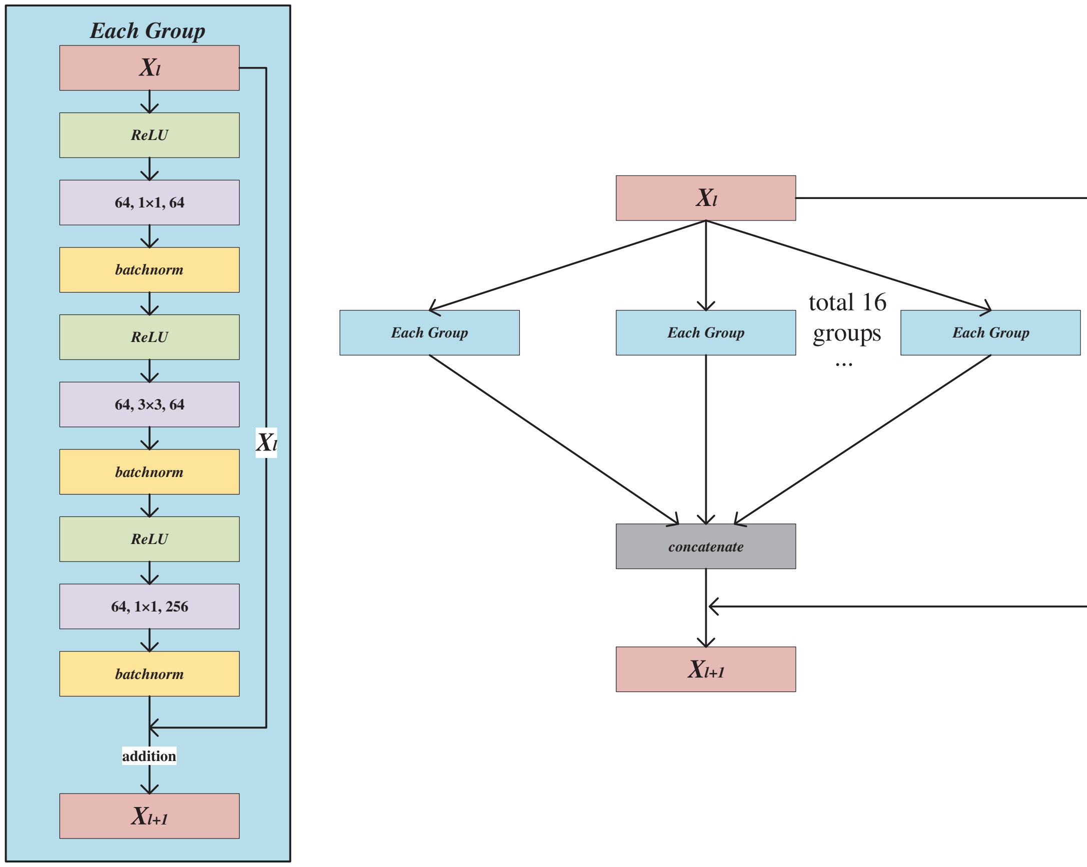

Figure 2: The architecture of the Pre-ReLU residual block

Table 1: The architecture of Pre-ReNet

Layer name	Architecture
Conv1	7×7, stride = 2, padding = 3
Conv2	(Pre-ReLU residual block) $\times$ 3
Conv3	(Pre-ReLU residual block) $\times$ 4
Conv4	(Pre-ReLU residual block) $\times$ 6
Conv5	(Pre-ReLU residual block) $\times$ 3

The output of the lth convolution layer is:

$$x_j^l = f_l\left(\sum_{i \in M_j} x_i^{l-1} \times k_{ij}^l + b_j^l\right). \tag{6}$$

where x_j is the ith output, $f_l\,(g)$ is the ReLU function, k_{ij}^l is the convolution kernel, $\times$ means the convolutional multiplication, and the bias of this layer is b_{ij}.

The output of the Mth pooling layer is:

$$x_j^m = f_s\left(\beta_j^m S\left(x_j^{m-1}\right) + b_j^m\right). \tag{7}$$

where the summation of the entire feature matrix is $S(g)$, $f_s(g)$ means the SoftMax function, the weight coefficient is β_j, the input for this layer is x_j^{m-1}, and the bias is b_j^m. By combining the Pre-ReNet with the loss introduced in the following subsection, we build a detector named B-PesNet. Then the architecture of our B-PesNet is shown in Fig. 3.

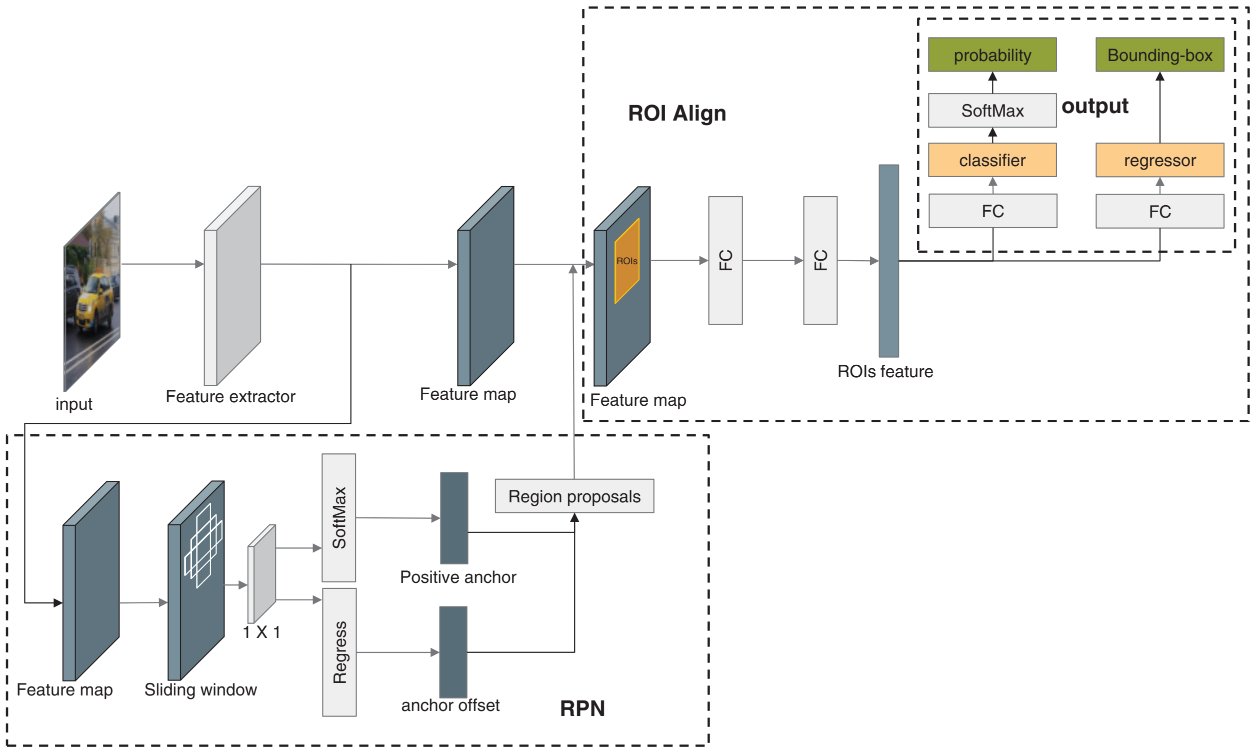

Figure 3: The architecture of B-PesNet

The input of the region proposal network (RPN) is a feature map, which is also used in the region of interest (ROI) Align layer. And the output of RPN is a set of rectangular regions named region proposals. We adopt the sliding window method to select a region titled anchor by sliding a window on the input feature map to obtain the region proposals. In each sliding window, multiple anchors are predicted simultaneously, and the maximum possible anchors of each pixel point are denoted by K. In this paper, K is set to 9 since there are 3 scales anchors, and each scale has 3 anchors with a different shape. Each anchor will be sent to SoftMax and classified it is positive or negative. At the same time, all anchors will perform a regress to get more actual coordinates. Then, the RPN will output the region proposals, which will be sent to the ROI Align layer. ROI Align reduces and normalizes those region proposals, extracts the ROIs feature vectors. And then, sends these vectors into the subsequent layer. The ROIs feature vector obtains the regression results and classification results through the Fully Connected and SoftMax layers.

The classifier and regressor in the output layer work are as follows:

Classifier: This layer will output the probability of objects in region proposals. The classifier will calculate the probability P_i of each element in the feature map i, which contains the object after thoroughly scanning the feature map.

Regressor: In this layer, the intersection of union (IoU) will be utilized to measure the accuracy of the bounding box (b-box). Then, the regressor will output the coordinates (x, y) of the center point of each anchor box, as well as the width w and height h of this box. IoU is calculated by Eq. (8):

$$IoU = \frac{A \cap B}{A \cup B}. \tag{8}$$

where A and B are the area of two region proposals.

Suppose the coordinates of the center point in the region proposal is (x, y), and the coordinates combined by width and height of this region proposal is (w, h). Then, a region proposal can be denoted by four indexes(x, y, w, h). Moreover, suppose $\left(P_x, P_y, P_w, P_h\right)$, $\left(G_X, G_y, G_w, G_h\right)$, $\left(G'_\chi, G'_y, G'_w, G'_h\right)$, which are depicting the positive anchor, ground-truth box, and predicted box, respectively. IoU can be used to adjust the edges of the predicted box in regression progress. Assume the difference between the region proposal and ground-truth box is $t_d = \left(t_x, t_y, t_w, t_h\right)$, where:

$$t_x = \frac{G_x - P_X}{P_w}, t_y = \frac{G_y - P_y}{P_h}. \tag{9}$$

$$t_w = log\left(\frac{G_w}{P_w}\right), t_h = log\left(\frac{G_h}{P_h}\right). \tag{10}$$

Then the loss function of the regressor can be written as:

$$L_e = \sum_{i=1}^{N} \left(t_d^i - \omega_d^T \phi\left(P_i\right)\right)^2. \tag{11}$$

here the ROI feature vector is $\phi\left(P_i\right)$, the coefficient achieved from the training phase is ω_K^T, the total number of samples is N, i means the ith sample. Thus, the classifier outputs $20k$ scores that denote the probability of 20 objects in the PASCAL VOC2007 dataset for each region proposal, the coordinates of k boxes are output by the regressor, which are $4k$ vectors.

3.4 Balance the Classes with an Improved Cross-Entropy Loss

The balanced loss (BL) is designed to handle the severe imbalance between foreground and background classes during the training phase, and the class imbalance overwhelms the cross-entropy (CE) loss. Quickly classified negatives cover the majority of the loss and dominate the gradient. Whereas, we propose to reshape the loss function to down-weight easy negatives, and then the network focuses on hard negatives during the training stage.

We introduce our BL starting from the standard CE loss:

$$CE\left(p, y\right) = \begin{cases} -\log(p) & \text{if} \quad \text{y=1} \\ -\log(1-p) & \text{otherwise.} \end{cases} \tag{12}$$

here $y \in \pm 1$ indicates the ground-truth class, and $p \in [0, 1]$ means the estimated probability of the model for the class with label $y = 1$. For notational convenience, we define p_t:

$$p_t = \begin{cases} p & \text{if} \quad \text{y=1} \\ 1 - p & \text{otherwise.} \end{cases} \tag{13}$$

then rewrite $CE\,(p, y) = CE\,(p_t) = -log\,(p_t)$.

A common method to address the imbalance of classes is to introduce a hyperparameter. In our method, we introduce a weight coefficient $\alpha \in [0, 1]$. In our experiments, α shows the ability to balance the significance of positive/negative examples. Hence we write the BL as:

$$BL\,(p, y) = \begin{cases} -\alpha \log(p) & \text{if} \quad \text{y=1} \\ -(1-\alpha)\log(1-p) & \text{otherwise.} \end{cases} \tag{14}$$

Then both positive/negative examples will dominate the gradient while $\alpha < 1$, For notational convenience, we define α_t:

$$\alpha_t = \begin{cases} \alpha & \text{if} \quad \text{y=1} \\ -(1-\alpha) & \text{otherwise.} \end{cases} \tag{15}$$

and rewrite BL to $BL\,(p, y) = BL\,(p_t) = -\alpha_t log\,(p_t)$.

The loss is a simple but effective extension to the standard CE loss, as our follow-up experiments show.

4 Data Processing and Evaluation Metrics

4.1 Data Processing

The data set trained, validated, and tested in this paper is the public dataset PASCAL VOC2007. It is composed of 21 classes, 20 foregrounds, and 1 back-ground.

Objects in real life are often under various interferences and are not as simple as the samples in the data set. A model trained with a simple data set will be challenging to apply to complex and changeable real-world scenarios, i.e., generalization is not strong. In order to overcome such a problem, online data enhancement operations will be introduced before training. Whether it is the human eye or the shooting device, it is possible to observe the object from any angle in the real world. The object might present a vastly different appearance due to the perspective. Accordingly, we warp the picture without changing the relevant information of the label. Twist operation can improve the generalization of the model in this respect. Besides this, there may be various interferences in the actual scene, such as intense light, haze, stains, etc. Therefore, we artificially highlight and blur the image to enhance the generalization of the model in such scenarios.

4.2 Evaluation Metrics

We utilize some common evaluation criteria to evaluate our detector in this paper. We utilize IoU, Precision, and Recall to evaluate our detector, and find the average of these experiments through multiple experiments and evaluations, which are

$$Precision = TP/\,(TP + FP)\,, \tag{16}$$

and

$$Recall = TP/\,(TP + FN)\,. \tag{17}$$

In Eqs. (16) and (17), TP indicates the number of samples for which the prediction is a positive example, which is a positive example. FP expresses that the prediction is positive, which is the number of samples for negative cases. FN represents that the prediction is negative, which is the number of samples for positive cases. Accordingly, Precision expresses how many of the predicted positive samples are genuinely correct, and Recall means how many of the predicted samples are genuinely correct.

$$mAP = 1/C \sum_{k=i}^{N} (k = i)^N P(K) \Delta \mathrm{R}(k). \tag{18}$$

In Eq. (18) [7], C represents the number of categories included in the model, N means the number of thresholds used as a reference, and k indicates the currently used threshold. P means the precision rate, R expresses the recall rate, and AP indicates the area under the precision-recall curve. The higher AP represents a better classifier. The average value of APs is mAP, where APs mean the sum of different classes. The value of mAP is within $[0, 1]$. This index is the crucial index of the object detection algorithms.

5 Experimental Results and Analysis

5.1 Experiment Setup

We adopt VOC2007 as the training set and perform data augmentation, which adds artificial noise to each image. Moreover, we scale all inputs into the size of 224×224 to save calculation resources. We utilize stochastic gradient descent (SGD) with momentum as the optimizer with momentum as 0.9 and weight decay as 1e-4. All the experiments are performed on the same computer, and the main hard-ware settings are Intel® Core ™ i7-6700 CPU@3.40 GHz, GeForce GTX 1070 GPU with 8 GB and 16GB RAM. All codes are written in Python, using PyTorch, and running under Ubuntu 16.04.

5.2 Calculation Results Using Different Backbones

To test the impact of different backbones used in the detector, we have carried out experiments on different feature extractors to compare the difference of mAP. The mAP and calculating speed in the different condition is shown in Table 2. It is evident that the accuracy of our method is higher than the others except for YOLOv5. The accuracy of our proposed B-PesNet is inferior to YOLOv5 cause the network structure of YOLOv5 is more complex than B-PesNet. From this point of view, our B-PesNet is still valuable and meaningful. The B-PesNet obtain a balance between accuracy and speed.

Table 2: Results with different methods

Detector	Network	Metrics	
		mAP	Speed(ms/image)
Faster R-CNN [15]	ResNet18	0.6100	80
	ResNet34	0.6305	92
	ResNet101	0.6582	145
	ResNext101	0.6754	120
Fast R-CNN [12]	VGG16	0.6229	280

(Continued)

Table 2 (continued)

Detector	Network	Metrics	
		mAP	Speed(ms/image)
YOLO [22]	GoogLeNet	0.6228	60
YOLOv4 [26]	CSPDarknet53	0.6927	**31**
YOLOv5 [33]	CSPDarknet53	**0.7134**	51
B-PesNet	Pre-ReNet	0.7086	132

5.3 Results by Different Detection and Recognition Approaches

To demonstrate the effect of the BL, we test some networks using BL or not. Table 3 shows the mAP for each condition. In this table, the value of α in every BL is 0.75. The reason for choosing 0.75 as the value of α will be discussed in subsequent sections. It is easily found that our BL is able to be inserted into any network as a plug-in. What is more, BL improves the performance of these methods. Furthermore, our B-PesNet is better than the other methods when using ordinary CE. The mAP of B-PesNet will increase to 73.34% while using BL as the loss function. The efficiency of the BL is shown in Fig. 4.

Table 3: Comparison with the different loss

Detector	Network	Loss function	Results in mAP
Faster R-CNN [15]	ResNet18	CE	0.6100
		BL	0.6326
	ResNet34	CE	0.6305
		BL	0.6506
	ResNet50	CE	0.6460
		BL	0.6686
	ResNet101	CE	0.6582
		BL	0.6803
	ResNext50	CE	0.6678
		BL	0.6876
	ResNext101	CE	0.6754
		BL	0.6942
Fast R-CNN [12]	VGG16	CE	0.6229
		BL	0.6431
YOLO [22]	GoogLeNet	CE	0.6228
		BL	0.6482
YOLOv2 [27]	Darknet19	CE	0.6464
		BL	0.6716
YOLOv3 [28]	Darknet53	CE	0.6566
		BL	0.6745
YOLOv4 [26]	CSPDarknet53	CE	0.7027
		BL	0.7256

(Continued)

Table 3 (continued)

Detector	Network	Loss function	Results in mAP
YOLOv5 [33]	CSPDarknet53	CE	0.7121
		BL	0.7356
B-PesNet	**Pre-ReNet**	CE	0.7086
		BL	0.**7334**

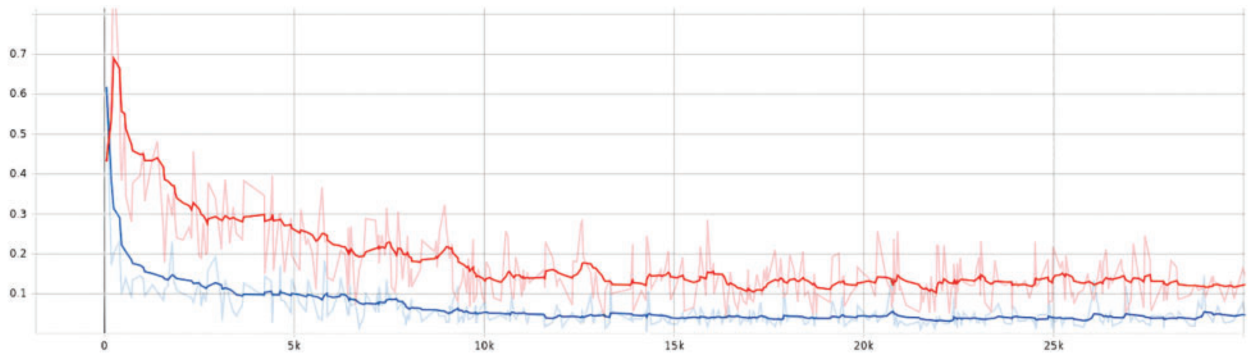

Figure 4: The loss curve of object classification. The blue one represents our B-PesNet with BL, and the orange one means our B-PesNet with CE. Obviously, the loss of B-PesNet with BL is lower and converges more quickly

5.4 Results by Using Different Hyper-Parameters

We chose 20 different values of α to verify the effect of the various hyper-parameter settings on BL performance. The result shows that when α is too small, the capability of BL will be inferior, even exacerbating the problem of class imbalance. When α is close to 1, BL will lose its effect and degenerate into ordinary CE. In this paper, we finally found that when $\alpha = 0.75$, the effect is the best. The experimental results are shown in Table 4.

Table 4: Results by using different hyper-parameters

Loss	α	mAP
BL	0.25	0.2753
	0.30	0.3248
	0.35	0.3746
	0.40	0.4354
	0.45	0.4987
	0.50	0.5559
	0.55	0.5884
	0.60	0.6100

(Continued)

Table 4 (continued)

Loss	α	mAP
	0.65	0.6736
	0.70	0.7145
	0.75	**0.7268**
	0.80	0.7017
	0.85	0.6874
	0.90	0.6752
	0.95	0.6621
	0.99	0.6432
	0.999	0.6289

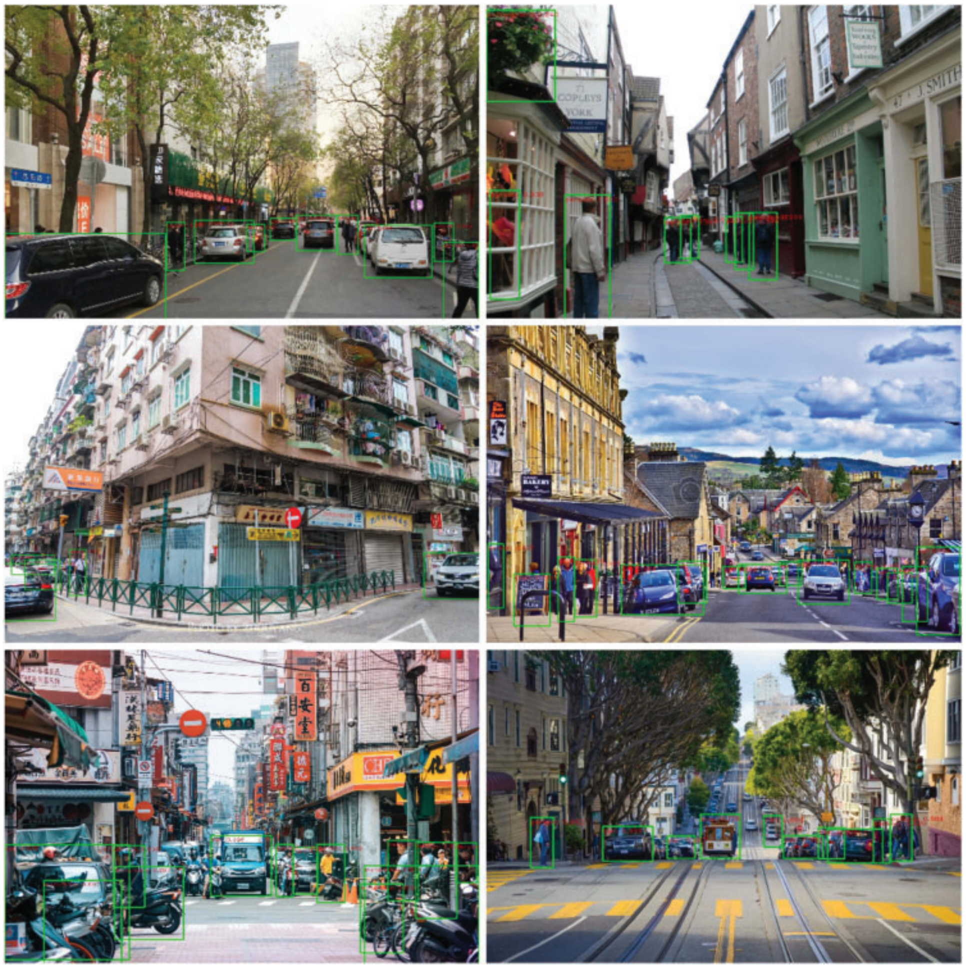

Figure 5: The detection results of B-PesNet. The detected target in the figure is marked by a green bounding box, and the mAP of the target is displayed on the upper part of the bounding box. The detection results of these actual scenes show that our B-PesNet has a good performance in a complex environment with interference factors such as occlusion and deformation

5.5 *Results and Discussions*

According to the results, we can infer that the semantic information is completely extracted, and the objects are accurately detected. From Table 3, it can be seen that our B-PesNet gets a balance between speed and accuracy. Table 3 shows that BL can tackle the class imbalance problem and improve the accuracy of the detector from the perspective of data. This shows that the research of deep learning can still start from data. As shown in Table 4, the parameters have a significant influence on BL performance. It is considering the definition of the formula, α should be set at a little larger value to have a better balance effect. Fig. 5 shows the detection results. In this way, we demonstrate that the new proposed loss plays an essential role in building a reliable detection for the security system.

6 Conclusions

We propose a method that improves the performance of the existing state-of-the-art two-stage object detection framework by enriching semantic information in shallow layers and introducing a new loss function, then demonstrate its effectiveness on the benchmark datasets. Our newly-designed network outperforms the previous neural network frameworks. The proposed B-PesNet shows above 1% mAP better than those typical object detectors. Compared with the existing approaches, the image processing speed of our method is also boosted. Greater efficiency imparts a higher practical value to the proposed method. This leads to a reliable solution for the applications of detection in security systems. In the future, we are interested in applying this detector to more specific scenarios, e.g., Internet of Things (IoT) applications in healthcare and health [36] and Blockchain-enabled IoMT [37].

Funding Statement: This research was supported by the Science and Technology Project of Sichuan (Nos. 2019YFG0504, 2021YFG0314, 2020YFG0459), and the National Natural Science Foundation of China (Grant Nos. 61872066 and U19A2078).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Zeng, N., Wang, Z., Zhang, H., Liu, W., Alsaadi, F. E. (2016). Deep belief networks for quantitative analysis of a gold immunochromatographic strip. *Cognitive Computation, 8(4),* 684–692. DOI 10.1007/s12559-016-9404-x.
2. Zeng, N., Zhang, H., Song, B., Liu, W., Li, Y. et al. (2018). Facial expression recognition via learning deep sparse autoencoders. *Neurocomputing, 273,* 643–649. DOI 10.1016/j.neucom.2017.08.043.
3. Jiao, L., Zhang, F., Liu, F., Yang, S., Li, L. et al. (2019). A survey of deep learning-based object detection. *IEEE Access, 7,* 128837–128868. DOI 10.1109/Access.6287639.
4. Wang, H., Gong, D., Li, Z., Liu, W. (2019). Decorrelated adversarial learning for age-invariant face recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3527–3536. Long Beach.
5. Huang, H., Feng, Y., Zhou, M., Qiang, B., Yan, J. et al. (2021). Receptive field fusion retinanet for object detection. *Journal of Circuits, Systems and Computers, 30(10),* 2150184. DOI 10.1142/S021812662150184X.
6. Zheng, S., Wu, Z., Xu, Y., Wei, Z., Xu, W. et al. (2021). Pillar number plate detection and recognition in unconstrained scenarios. *Journal of Circuits, Systems and Computers, 30(11),* 2150201. DOI 10.1142/S0218126621502017.
7. Wan, S., Goudos, S. (2020). Faster R-CNN for multi-class fruit detection using a robotic vision system. *Computer Networks, 168,* 107036. DOI 10.1016/j.comnet.2019.107036.

8. Dalal, N., Triggs, B. (2005). Histograms of oriented gradients for human detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 886–893. San Diego.
9. Felzenszwalb, P., McAllester, D., Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8. Anchorage.
10. Wu, X., Sahoo, D., Hoi, S. C. (2020). Recent advances in deep learning for object detection. *Neurocomputing, 396,* 39–64. DOI 10.1016/j.neucom.2020.01.085.
11. Oksuz, K., Cam, B. C., Kalkan, S., Akbas, E. (2020). Imbalance problems in object detection: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 43(10),* 3388–3415. DOI 10.1109/TPAMI.2020.2981890.
12. Girshick, R. (2015). Fast R-CNN. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1440–1448. Santiago.
13. He, K., Zhang, X., Ren, S., Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 37(9),* 1904–1916. DOI 10.1109/TPAMI.2015.2389824.
14. Cai, Z., Fan, Q., Feris, R. S., Vasconcelos, N. (2016). A unified multi-scale deep convolutional neural network for fast object detection. *European Conference on Computer Vision*, pp. 354–370. Amsterdam.
15. Ren, S., He, K., Girshick, R., Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6),* 1137–1149. DOI 10.1109/TPAMI.2016.2577031.
16. Cheng, B., Wei, Y., Shi, H., Feris, R., Xiong, J. et al. (2018). Revisiting RCNN: On awakening the classification power of faster RCNN. *European Conference on Computer Vision*, pp. 473–490. Munich.
17. Girshick, R., Donahue, J., Darrell, T., Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587. Columbus.
18. Cao, J., Cholakkal, H., Anwer, R. M., Khan, F. S., Pang, Y. et al. (2020). D2det: Towards high quality object detection and instance segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11485–11494. Seattle.
19. Lu, X., Li, B., Yue, Y., Li, Q., Yan, J. (2019). Grid R-CNN. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7363–7372. Long Beach.
20. He, Y., Zhu, C., Wang, J., Savvides, M., Zhang, X. (2019). Bounding box regression with uncertainty for accurate object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2888–2897. Long Beach.
21. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S. et al. (2016). Ssd: Single shot multibox detector. *European Conference on Computer Vision*, 21–37. Amsterdam.
22. Redmon, J., Divvala, S., Girshick, R., Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788. Las Vegas.
23. Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B. et al. (2017). Feature pyramid networks for object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2117–2125. Honolulu.
24. Lin, T. Y., Goyal, P., Girshick, R., He, K., Dollár, P. (2017). Focal loss for dense object detection. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2980–2988. Venice.
25. Pang, J., Chen, K., Shi, J., Feng, H., Ouyang, W. et al. (2019). Libra R-CNN: Towards balanced learning for object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 821–830. Long Beach.
26. Bochkovskiy, A., Wang, C. Y., Liao, H. Y. M. (2020). YOLOV4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*.
27. Redmon, J., Farhadi, A. (2017). YOLO9000: Better, faster, stronger. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7263–7271. Honolulu.

28. Redmon, J., Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.
29. Zhang, S., Wen, L., Bian, X., Lei, Z., Li, S. Z. (2018). Single-shot refinement neural network for object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4203–4212. Salt Lake City.
30. Pang, Y., Wang, T., Anwer, R. M., Khan, F. S., Shao, L. (2019). Efficient featurized image pyramid network for single shot detector. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7336–7344. Long Beach.
31. Everingham, M., van Gool, L., Williams, C. K., Winn, J., Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision, 88(2),* 303–338. DOI 10.1007/s11263-009-0275-4.
32. Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P. et al. (2014). Microsoft coco: Common objects in context. *European Conference on Computer Vision*, pp. 740–755. Zurich.
33. Jocher, G., N. K. M. T. V. R. (2020). YOLOV5. https://github.com/ultralytics/yolov5.
34. He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778. Las Vegas.
35. Nair, V., Hinton, E. G. (2010). Rectified linear units improve restricted boltzmann machines. *International Conference on Machine Learning*, pp. 807–814. Haifa.
36. Xiong, H., Hou, Y., Huang, X., Zhao, Y., Chen, C. M. (2021). Heterogeneous signcryption scheme from IBC to PKI with equality test for WBANS. *IEEE Systems Journal, Early Access*, *(1),* 1–10. DOI 10.1109/JSYST.4267003.
37. Xiong, H., Jin, C., Alazab, M., Yeh, K. H., Wang, H. et al. (2021). On the design of blockchain-based ecdsa with fault-tolerant batch verication protocol for blockchain-enabled iomt. *IEEE Journal of Biomedical and Health Informatics, Early Access(1),* 1–9. DOI 10.1109/JBHI.2021.3112693.