Hindawi
Journal of Electrical and Computer Engineering
Volume 2022, Article ID 5774306, 9 pages
https://doi.org/10.1155/2022/5774306

Research Article

Real-Time Forestry Pest Detection Method Based on Enhanced Feature Fusion with Deep Learning

Rui Li, Tong Liu, and Yalu Ren

School of Computer and Communication, Lanzhou University of Technology, Lanzhou, Gansu, China

Correspondence should be addressed to Tong Liu; wwlt1031@126.com

Received 8 June 2022; Revised 4 July 2022; Accepted 28 July 2022; Published 28 August 2022

Academic Editor: Yang Li

The aim of this study is to address the real-time requirements of forestry pest detection and the problem of a low detection rate caused by anchor box redundancy of existing detection methods. This paper proposes a real-time forestry pest detection method based on theanchor-free method that can balance the detection rate and detection accuracy. Based on the TTFNet method, a mobile feature extraction network is introduced, and the effective feature weights are increased by one-dimensional convolution before feature output to suppress invalid features. For pest detection, data are mostly small-scale targets. An enhanced feature fusion method is proposed to introduce an asymmetric convolution module in multi-scale feature fusion to feature-enhance the feature maps extracted by the backbone network and connect across layers to improve the detection accuracy. To address the degradation of the anchor box position regression loss in the original method, DIOULoss is introduced to optimize the position regression loss function of the anchor box. Finally, data augmentation is performed on a relatively small number of samples in the dataset, the accuracy of the model is improved by 1.94%, the FPS is improved to 1.6 times of the original one, and the training time is slightly increased compared with the preinnovation model. Ablation experiments are designed to demonstrate the effectiveness of the proposed algorithm while being more conducive to deployment on edge devices.

1. Introduction

Forestry pest detection can play an early preventive role in forestry control, where pests can be extremely destructive to trees. Conventional identification relies on manual work, which is often time-consuming and difficult to achieve accurate control in real time due to the knowledge base and identification time. Traditional target detection algorithms are based on the manual annotation of features and consist of six stages: preprocessing, suggested regions, feature extraction, feature selection, feature classification, and postprocessing. Detection models generally focus on the extraction of target features and the selection of region classification algorithms. The main drawbacks of traditional detection algorithms are as follows: (1) a large number of redundant candidate boxes are generated in the candidate region generation stage; (2) the traditional feature extractor is unable to learn the high-level semantic information of the input image; (3) global tuning is not possible because the algorithm is divided into multiple stages [1].

Anchor-base target detection algorithms: since 2012, deep learning-based target detection algorithms have dramatically improved detection accuracy as hardware computing power has increased. Compared to traditional algorithms, deep learning automatically learns features in the data to obtain high-level semantic and contextual content of the image. Models can be classified into one-stage methods and two-stage algorithms based on the method of model training. The former has a faster detection rate than the latter, and the latter has higher detection accuracy than the former. The two-stage algorithm first generates region proposals and then performs feature extraction. The feature map generated from all the candidate regions is fed into the classifier to determine the category to which the object belongs, and the location of the object is determined by the regression loss function based on bounding boxes. Typical ones are the RCNN(Region-CNN, RCNN) method, which proposes a new approach to candidate region target detection with significant accuracy improvement but causes

positive and negative sample imbalance as well as long training time due to phasing [2]; SPPNet(Spatial Pyramid Pooling Networks, SPPNet) proposes a method to fix the input image size while improving the accuracy of model detection, but its training cannot backpropagate to update the convolutional layer parameters [3]. Fast RCNN proposes ROI pooling to optimize the selection of region features, but its positive and negative samples are unbalanced and cannot meet the demand for real-time detection [4]. Mask RCNN fills noninteger location pixels by bilinear interpolation and uses additional branches to output mask predictions of candidate regions to achieve higher accuracy detection. However, it can lead to positive and negative sample imbalance as well as failing to meet the real-time requirements [5]. The one-stage algorithm uses the original image directly to extract features and predicts the target class and position regression, so the detection rate is fast. Due to the simple structure of the algorithm, there are cases of missed detection for large-scale targets and densely distributed targets. Typical ones are the YOLO (You Only Look Once, YOLO) series[6], SSD (Single Shot Detector, SSD) [7], and so on. Since feature extraction is relatively inadequate in two stages, multiscale fusion is available to improve the accuracy of detection.

Anchor-base target detection algorithms have a large number of redundant boxes in forwarding inference, which causes an additional overhead and thus affects the rate of object detection. The CornerNet method uses the upper left and the lower right corner points as a set of corner points to represent a target, using the idea of corner pooling and embedding vectors to improve accuracy. However, it has a high false detection rate due to the increased computational difficulty [8]. The CenterNet method adds center points to the CornerNet method, the upper left corner point, the lower right corner point, and the middle point as a set of corner points to represent an object, which takes into account the internal information and reduces the false detection rate compared to the former [9]. The FCOS (fully convolutional cne-stage, FCOS) method represents a target in terms of points and the distance from the point to the anchor, thus improving detection performance under NMS (nonmaximal suppression, NMS) [10].

The nature of the difference between anchor-base design and anchor-free design is as follows: frist, the two methods differ in the definition of positive and negative samples, the former usually sets sampling anchor boxes on the feature map based on a priori knowledge, but most of the anchor boxes have no objects or background areas, so there are a large number of negative samples and no useful response for detector learning; second, for the size of the anchor box, the number of hyperparameters affects the recall and detection rate of detection and needs to be set artificially; third, because of the presence of a large number of negative samples of anchor box, in the IOU (intersection of union, IOU) calculation will occupy a lot of memory and computing time.

The detection accuracy of forestry pests is affected by the small size of the box and the resolution of the camera; in addition, due to the high complexity of the above-mentioned model, the detection accuracy is high, but the detection rate is low, so it is difficult to meet the demand for real-time forestry detection. Therefore, it is difficult to deploy the above algorithms in a low computing power environment, so this paper proposes a detection method with balanced detection accuracy and detection rate, which will be beneficial to the deployment of deep learning algorithms in agricultural engineering.

2. TTFNet Target Detection Algorithm

TTFNet uses Gaussian kernels to encode training samples, designs active sample weights to make better use of information, balances the training time, and provides significant improvements in the inference rate and accuracy. TTFNet uses DarkNet53 [11] as the network feature extractor, which uses a stack of convolutional residual blocks to reduce the phenomenon of gradient disappearance due to the deep number of layers in the network. TTFNet borrows ideas from CornerNet [12], where center localization and size regression for object detection are achieved through a two-dimensional Gaussian kernel $K_m(x, y) = \exp(-(x - x_0)^2/2\sigma_x^2 - (y - y_0)^2/2\sigma_y^2)$, $\sigma_x = \alpha w/6$, and $\sigma_y = \alpha h/6$, (x_0, y_0) is the central position. The center is positioned to generate a heat map from the Gaussian kernel $\widehat{H} \in R^{N\times C\times H/R\times W/R}$, where N is the number of single training samples, C is the number of categories, H and W are the height and width of the input image, and r is the downsampling rate, which allows the model to focus more on the center of the object, and also the aspect ratio of the anchor box is further considered in the Gaussian kernel. $L_{loc} = 1/M\sum_{xyc}\begin{cases}(1-\widehat{H}_{ijc})^{\alpha_f\log(\widehat{H}_{ijc})} & \text{if } H_{ijc=1}\\ (1-\widehat{H}_{ijc})^{\beta_f \hat{H}_{ijc}^{\alpha_f}\log(1-\widehat{H}_{ijc})} & \text{elsewise}\end{cases}$ is the heatmap loss, where α_f and β_f are the cross-entropy hyperparameters. Size regression defines all pixels in the Gaussian region $\hat{S} \in R^{N\times 4\times H/r\times W/r}$ as the training sample and predicts the height and width of the object. Loss of width and height is determined by the equation, $L_{reg} = 1/N_{reg}\sum_{(i,j)\in A_m} GIoU(\hat{B}_{ij}, B_m)\times W_{ij}$, where $\hat{B}_{ij}, B_m$ are the prediction boxes and the actual label boxes, respectively. W_{ij} indicates sampling weights. After normalizing the loss of all samples, the loss of small targets is almost negligible, so for the overall detection performance, W_{ij} is used to alleviate the problem of sample imbalance between small and large targets by allowing the information of small samples to be retained. Assuming that (i, j) is in the subregion A_m of the m-th labeled box. Then, $W_{ij} = \begin{cases}\log(a_m)\times G_m(i,j)/\sum_{(x,y)\in A_m} G_m(x,y) & (i,j)\in A_m\\ 0 & (i,j)\notin A_m\end{cases}$, where $G_m(i, j)$ is the Gaussian probability at the position (i, j), and a_m is the area of m th boxes. In summary, the total loss is $L_{\text{total}} = w_{\text{loc}}L_{\text{loc}} + w_{\text{reg}}L_{\text{reg}}$. In addition, weights calculated in terms of object size and Gaussian probability

are applied to the samples so that the model makes full use of the information. For the forestry pest prediction covered in this paper, the TTFNet method is improved upon to meet the needs of forestry pest detection, taking into account the real-time requirements of the detection model and the sensitivity of the model to small samples.

3. Improved Lightweight TTFNet Target Detection Algorithm

3.1. Lightweight Attention Module. MobileNet inevitably loses some feature information during the feature map extraction process, especially for small-scale targets. The extracted feature maps not only contain the response detection results but also contain information that interferes with the detection effect [13]. Using ResNet [14] as a benchmark, the validity of the attention module is verified separately by placing it at a position before the output feature map of ResNet. The comparative performance is shown in Table 1.

As can be seen from Table 1, the SE (squeeze and excitation, SE) module performs global average pooling in both width and height dimensions of the feature map, increases the global receptive field, compresses the feature map dimension through the first fully connected layer, and then expands to the original dimension with a subsequent fully connected layer. The feature expression power of the channel dimension is enhanced, which is beneficial to the accuracy of detection [15]. Unlike the SE module, the ECA (efficient channel attention, ECA) module uses a one-dimensional convolution with the kernel size k (in this paper, $k = 5$) after global average pooling to avoid losing the dependencies between the former channels, and the resulting weights are then multiplied by the corresponding positions of the input feature maps. It can improve the detection accuracy of the baseline model with a small loss in the detection rate [16].

3.2. Backbone Network Enhancements. MobileNet is used as a feature extraction network and as a detection model for mobile devices. It uses deep separable convolution to first convolve the channels of the image separately; however, channel-by-channel convolution will lose the information between channels, so each channel is stitched together by 1×1 convolution to reduce the computational effort of convolution, as shown in Figure 1. This facilitates the deployment of mobile devices. For small-scale target detection and enhanced feature extraction, four scales of feature maps (56×56, 28×28, 14×14, 7×7) are output after adding the ECA module to MoblieNet before outputting the feature maps, which will be used for subsequent feature map fusion. The improvement of the feature extraction network by a more efficient attention module enables the network as a whole to focus on important information with high weights and ignore irrelevant information and filter noise with low weights. This improves the ability of the feature extraction network to generate high-quality feature maps.

Table 1: Performance metrics of different types of attention on the output of feature extraction networks.

Modules	Accuracy (%)	FPS
ResNet	77.86	22
+SE module	78.43	17
+ECA module	79.46	20

3.3. Asymmetric Convolution Modules. The module, shown in Figure 2, is introduced to improve the robustness of the detection network to different angles, relying on the superimposability of the convolution operation for the detection of pests, whose data are collected from different angles. In addition, to address the indistinguishability of pest samples, the feature extraction of the neural network has to be enhanced. Borrowing ideas from MobileNet, ACNet (asymmetric convolution)[17] replaces each 3×3 convolution with 3×3, 3×1, and 1×3 convolutions by fusing the computational results of the three convolutions as the output of each 3×3 convolution. Since several convolution kernels of the same size are computed in the same step on the same input feature map, the resulting feature maps are of the same size, and they are summed in the corresponding positions, as shown in the following equation:

$$I * K^{(1)} + I * K^{(2)} = I * \left(K^{(1)} \oplus K^{(2)}\right), \tag{1}$$

where I is the input feature map, $K^{(1)}$ and $K^{(2)}$ are convolution kernels with the same scale, and $\oplus$ is a summation operation at the corresponding position. This improves the robustness of the network model to rotations and flips with a small increase in training cost, while enhancing feature extraction and optimizing feature reuse, thus improving the accuracy of the detection network.

3.4. Multiscale Feature Fusion for Cross-Layer Connectivity. Since the introduction of the lightweight feature extraction network, it accelerated the inference rate of the network, the features were replicated to compensate for the loss of accuracy as its network extracted limited features, and the deep feature maps and shallow feature maps had different perceptual fields and semantic information. As shown in Figure 3, after obtaining the four sets of features extracted by the feature extraction network, the high-level feature map extracted by the feature extraction network is first upsampled by a factor of two, and then the cross-layer connection of the AC module performs feature enhancement on the previous four scales of the feature map, which forms an hourglass-like feature fusion network through the superposition of the convolution operation and adds up with the upsampled feature map. The final feature map is obtained by adding the upsampled and reinforced cross-layer connections point by point, thus enriching the semantics of the feature map. The final feature map is then used for target identification and location regression.

After the input image has been extracted by MobileNet, a four-scale feature map is generated, which is summed point by point with the results generated by the AC module during

Figure 1: Depthwise separable convolution.

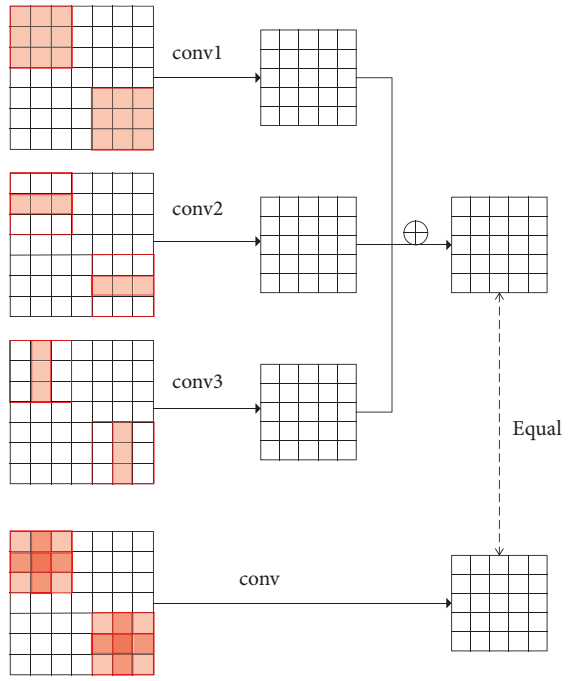

Figure 2: Asymmetric convolution module.

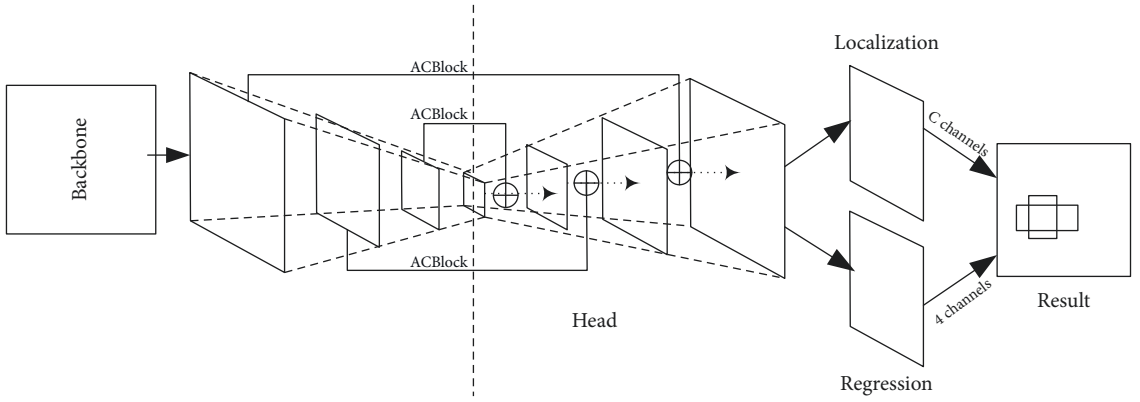

Figure 3: Improved network structure diagram.

upsampling to produce the final result, resulting in C-class object recognition and position regression of the centroid and the aspect.

3.5. Improvement of the Detection Head. After feature fusion, the algorithm performs class prediction and anchor box position regression on the object. Anchor-based algorithms usually generate a large number of predicted boxes during forward inference, find the intersection ratio of the predicted boxes to the actual labelled boxes as part of the loss, and then perform anchor-frame correction by back-propagation, so anchor frames that do not contain targets may cause some unnecessary computational effort. Unlike anchor-base algorithms, the TTFNet method generates a heat map at the center of the target to predict the object, thus eliminating the computational effort associated with nonmaximum suppression and requiring only maximum pooling to be able to locate the target. Also, the distance from the center of the heat map to the four edges of the border can be predicted directly. In the TTFNet detection head, part object detection loss uses CTFocalLoss, and border loss uses GIOULoss, as shown in equation (3). GIOU Loss is to mitigate the IOU Loss and does not take into account the value of 0 for IOU

when the real box and the predicted box do not overlap affecting the model convergence. GIOU loss based on IOU Loss adds a penalty item, as shown in Figure 4 to solve the problem of IOU Loss being 0 when the prediction box and detection box do not overlap, which accelerates the convergence of the model compared to IOU Loss.

$$IoU = \frac{|A \cap B|}{|A \cup B|}, \tag{2}$$

$$GIoU \quad Loss = 1 - IoU + \frac{C - (A \cup B)}{C}. \tag{3}$$

Here, A is the prediction box, B is the true box, and C is the smallest enclosing box that minimally contains the two overlapping boxes. The penalty term is$(C - A \cup B)/C$, but if the real box and the predicted box contain each other, the GIoU Loss will be equivalent to the IoU Loss. For small-scale forestry pest detection, the DIOU Loss is introduced, as shown in equation (4), and the Euclidean distance between the centroids of the two boxes is calculated in the loss function, which avoids the problem of GIoU Loss degradation and facilitates the convergence of the model at the same time.

$$DIoU \quad Loss = 1 - IoU + \frac{\rho^2\left(b, b^{gt}\right)}{c^2}, \tag{4}$$

where $\rho_2(b, b_{gt})$ is the Euclidean distance between the center point of the predicted box and the real box, and c is the diagonal distance of the smallest external box.

4. Experimental Results

4.1. Datasets. The dataset uses the publicly available forestry bollworm dataset, which has approximately 2,200 samples, with a training set of approximately 1,700 and both a validation and test set of 245 in Petri dishes so that the pests are randomly combined, with six classes of Boerner, Leconte, Acuminatus, Armandi, Coleoptera, and Linnaeus [18].

The target data in the sample were counted, and due to the low volume of the Acuminatus sample, it was subjected to the copy-paste method of data amplification [19], where the existing missing sample was copied and rotated and mirror-flipped. The sample size after amplification was 1429. The results of the data amplification are shown in Table 2 and Figure 5.

4.2. The Experimental Environment. The training environment is Ubuntu 16.04, the processor is Intel(R) Xeon(R) Gold 6148 CPU @ 2.40 GHz x 2, the RAM is 16G, the GPU is Tesla V100 16G, and the deep learning framework is Paddle. To test the inference effect of the proposed method on edge devices, the testing environment uses mobile devices, Ubuntu 16.04, where the processor is Intel(R) Core i7-6700HQ, the RAM is 8G, the GPU is Nvidia GTX 960M (4G), and the deep learning framework is Paddle. The training environment was Ubuntu 16.04, CPU: Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHzx2, RAM: 16G, GPU: Tesla V100 16G. To test the inference effect of the proposed method on edge devices, the testing environment used a mobile device, Ubuntu 16.04, CPU Intel(R) Core i7-6700HQ, RAM: 8G, GPU: Nvidia GTX 960M (4G). The deep learning framework used for the experiments was PadlePaddle, CUDA Version: 11.0.

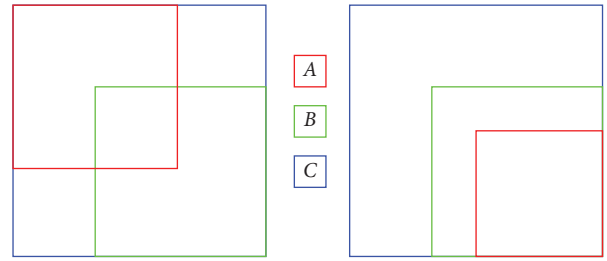

Figure 4: GIOU loss diagram.

Table 2: Pest sample count.

Category	Count
Boerner	1595
Leconte	2216
Acuminatus	953 (1429)
Armandi	1764
Coleoptera	2091
Linnaeus	1728

The initial learning rate is 0.001, and the batch size is 24. The model initializes the weights at the beginning of training, and to prevent the gradient of the weights from oscillating back and forth, the learning rate preheating method is used. After the 80th epoch and 110th epoch, the learning rate is decayed, the decay coefficient is 0.5, and the model is optimized to achieve the globally optimal weights by training with a smaller learning rate. The optimizer is trained iteratively by using Adam. This is shown in Figure 6 and Figure 7. The training loss is made up of two components, and they are heatmap loss and anchor box position loss. After training to 9000iters, there is no more oscillation in the loss values and the model converges. However, the improved model has slightly higher loss values than those before the improvement.

The trained models were tested, and the anchor-free models based on models such as CornerNet and CenterNet as well as common anchor-base such as YOLO series and SSD were selected as comparison models. The evaluation metrics are based on average accuracy, as shown in (5), with FPS and training cost as their test metrics. The inference rate is measured in terms of frames per second (FPS) greater than or equal to 30 as a metric for real-time detection.

$$\begin{aligned} \text{Precision}\,(P) &= \frac{TP}{TP + FP} \\ \text{Recall}\,(R) &= \frac{TP}{TP + FN} \\ AP &= \int_0^1 PdR \\ mAP &= \frac{\sum_{i=0}^{n} AP\,(i)}{n}, \end{aligned} \tag{5}$$

Figure 5: Data before augmentation (top) and data after augmentation (bottom).

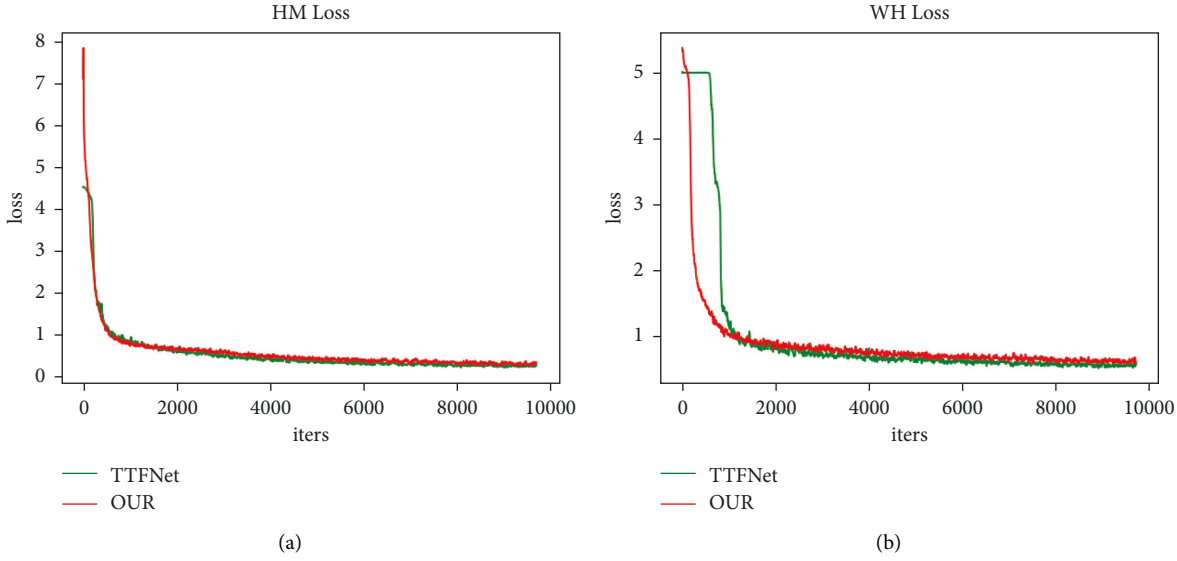

Figure 6: (a) Heatmap loss and (b) anchor WH regression loss.

where TP is the correct prediction box, FP is the incorrect prediction box, FN is the missed detection box, TN is the correct background, AP is the detection accuracy for a particular category, and mAP is the average detection accuracy across multiple categories.

Analysis of the data in Table 3 shows that this paper compares whether the models are based on anchor boxes or not. The anchor-base detection method YOLOv5 has the highest detection accuracy, but its detection rate does not meet the demand for real-time detection, and the training cost of all anchor-base detection methods is higher than that of anchor-free detection methods. The improved method improves the accuracy by 1.94% compared with the previous method with a small increase in training cost, and the detection accuracy is higher than that of the other methods, while the detection rate is 1.6 times higher than the original method, 4.9 times higher than the CornerNet method, and 4 times higher than the CenterNet method. The overall number of computational parameters increased by 6.84% compared to the previous method, and the training time increased by 12%, but the model size was reduced to 46.5% of the previous method. Both were much smaller than the comparable method. It can be seen that although the improved method in this paper is slightly less accurate than the individual anchor-base detection algorithms, the detection rate is higher than the compared

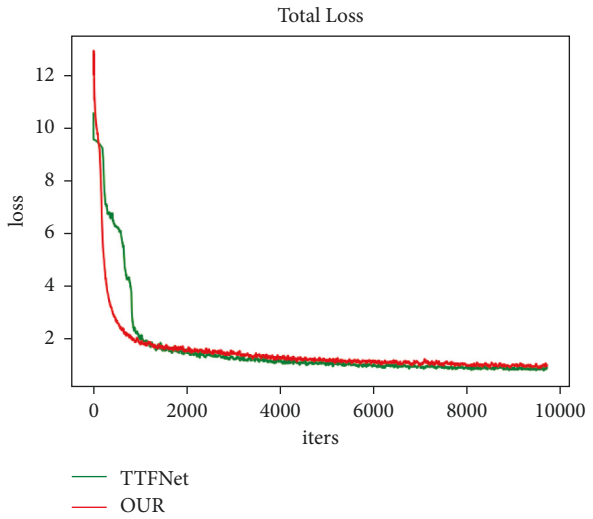

Figure 7: Total loss.

Table 3: Comparison of several models.

	Methods	mAP (%)	FPS	Paras/M	FLOPs/B	Training time (h)
Anchor-base	SSD	75.19	31	238.51	42.36	13.6
	YOLOv4	84.16	24	235.14	54.36	18.7
	YOLOv5	86.73	16	251.76	58.63	19.5
Anchor-free	CornerNet	76.93	9	161.22	44.98	14.5
	CenterNet	77.45	11	150.56	43.17	13.6
	TTFNet	81.91	27	77.36	22.64	7.5
	Our	83.85	44	35.97	24.19	8.4

Table 4: Ablation experiment table.

Backbone	Attention	AcMoudle	mAP (%)	FPS
MobileNet			72.61	57
	√		76.23	51
		√	81.47	48
	√	√	83.85	44

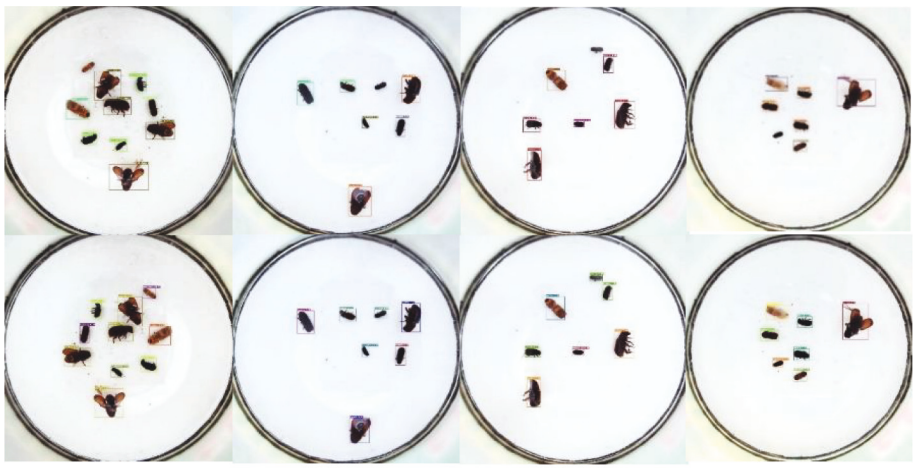

Figure 8: Diagram of the test results (top) before and (bottom) after improvement.

detection algorithms, and it can meet the demand for real-time detection of edge devices. Thus, this experiment improves the detection of very small targets by enhancing the feature extraction of cross-layer connections and achieves a balance between the detection accuracy and the detection rate.

4.3. The Ablation Experiment. To assess the impact of different modules on the experiment, ablation experiments were designed.

As can be seen from Table 4, the average detection accuracy improves by 3.62%, and the detection rate decreases by 10.5% with the placement of an efficient ECA module in the baseline backbone network under the same experimental environment. Enhancing the feature extraction of cross-layer connections with the AC module in the baseline backbone network improves the average detection accuracy by 8.86% and also reduces the detection rate by 15.8%. When both modules are present, the detection accuracy is 83.85%, and the detection rate can be detected in real time.

4.4. Test Results. The detection results on the test set are shown in Figure 8. The original method is very prone to miss detection of very small targets, the top four graphs are the detection results of the original method, and the bottom one is the detection result of the improved method. From the comparison results, it can be seen visually that the improved method has better detection results for very small targets than the original method.

5. Conclusion

This paper proposes a real-time forestry pest detection method based on anchor-free enhanced feature fusion to address the real-time requirements of forestry pest detection and the problem of missing some samples by using the TTFNet method directly. It is demonstrated that the proposed method improves the detection accuracy and detection rate compared with the original method, especially the detection rate meets the demand of real-time detection and solves the problem of target underdetection in the original method. The proposed method is important for preventing and reducing the incidence of forestry pests and improving pest management. In addition, given that the dataset used in this paper is a small target sample dataset, good detection accuracy can be achieved. Therefore, the module proposed in this paper can be used in other detection tasks to improve the accuracy of the detection of small targets. In future, it will be the next step of research to improve the detection accuracy even more closely.

Data Availability

The data used to support the findings of this paper are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was only supported by the National Natural Science Foundation of China [grant no., 62161020]

References

[1] K. Li, C. Yan, L. J. Chen, and X. Mou, "A review of deep learning-based target detection algorithms [J/OL]," *Computer Engineering*, 2022.

[2] H. Liang, Q. Wang, Q. Zhang, and C. Li, "A review of research on small target detection techniques[J]," *Computer Engineering and Applications*, vol. 57, no. 01, pp. 17–28, 2021.

[3] P. Purkait, C. Zhao, and C. Zach, "SPP-net: deep absolute pose regression with synthetic views[J]," *Computer Vision and Pattern Recognition*, 2017.

[4] R. Girshick, "Fast R-CNN[J]," 2015, http://arXiv.org/abs/1504.08083.

[5] K. He, G. Gkioxari, and P. Dollár, "Mask R-CNN[J]," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 2017, pp. 2961–2969, 2017.

[6] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger[J]," in *IEEE Conference on Computer Vision & Pattern Recognition*, pp. 6517–6525, 2017.

[7] W. Liu, D. Anguelov, and D. Erhan, *SSD: Single Shot MultiBox Detector[J]*, Springer, Cham, 2016.

[8] H. Law and J. Deng, "CornerNet: detecting objects as paired keypoints," *International Journal of Computer Vision*, vol. 128, no. 3, pp. 642–656, 2020.

[9] K. Duan, S. Bai, L. Xie, H. Qi, and T. Qi, "Centernet: keypoint triplets for object detection[C]," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6569–6578, Seoul, Korea (South), 27 October-02 November 2019.

[10] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: fully convolutional one-stage object detection[C]," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9627–9636, Seoul, Korea (South), 27 October-02 November 2019.

[11] J. Redmon and A. Farhadi, "Yolov3: an incremental improvement[J]," 2018, http://arXiv.org/abs/1804.02767.

[12] Z. Liu, T. Zheng, G. Xu, Z. Yang, H. Liu, and D. Cai, "Training-time-friendly network for real-time object detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, pp. 11685–11692, 2020.

[13] A. G. Howard, M. Zhu, and B. Chen, "MobileNets: efficient convolutional neural networks for mobile vision applications [J]," *Computer Vision and Pattern Recognition*, 2017.

[14] K. He, X. Zhang, and S. Ren, "Deep residual learning for image recognition[C]," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[15] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks [C]," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

[16] Q. Wang, B. Wu, P. Zhu, P. Li, and W. Zuo, "ECA-net: efficient Channel attention for deep convolutional neural networks[C]," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Seattle, WA, USA, 13-19 June 2020.

[17] X. Ding, Y. Guo, G. Ding, and J. Han, "Acnet: strengthening the kernel skeletons for powerful CNN via asymmetric convolution blocks[C]," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1911–1920, IEEE, Seoul, Korea (South), 27 October-02 November 2019.

[18] Y. Hua, J. Wang, J. Rong, H. Guodong, and L. Li, "Research on the identification of small stupid insects based on ResNet network[J]," *Forestry and Environmental Science*, vol. 37, no. 06, pp. 124–129, 2021.

[19] G. Ghiasi, Y. Cui, A. Srinivas, R. Qian, and T. Lin, "Simple copy-paste is a strong data augmentation method for instance segmentation[C]," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2918–2928, IEEE, Nashville, TN, USA, 20-25 June 2021.