Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 9937038, 11 pages
https://doi.org/10.1155/2021/9937038

Research Article

Lightweight Automatic Identification and Location Detection Model of Farmland Pests

Kunpeng Li,[1] Junsheng Zhu,[2] and Nianqiang Li[1]

[1]*University of Jinan, Jinan 250000, China*
[2]*Plant Protection Station, Jinan 250000, China*

Correspondence should be addressed to Nianqiang Li; ise_linq@ujn.edu.cn

Received 23 August 2021; Revised 15 September 2021; Accepted 20 September 2021; Published 15 October 2021

Academic Editor: Deepak Gupta

Automatic identification and location of farmland pests are an important direction of target detection research. The wide variety of pests and the similarity between pest categories make the automatic identification of farmland pests have some problems, such as high error rate and difficult identification. In order to achieve a better target for automatic identification and location of farmland pests, this paper proposes a lightweight pest detection model, and the network is the EfficientNet proposed by Google, which achieves the detection of 26 pests, the idea based on the classical Yolo target detection algorithm. First of all, features were extracted through the lightweight backbone, and then multiscale feature fusion is performed by PANet; finally, three feature matrices with different sizes were output to predict pests of different sizes. Using CIOU as the loss function of regression prediction better reflects the relative position of the prior box and the real box. The experimental results are compared with other lightweight algorithms, and the results show that the accuracy rate of the algorithm for identification and localization of agricultural pest in this paper is the highest and could reach 93.73%. Moreover, the model is lightweight and can be deployed on low-cost equipment, which reduces the cost of equipment and accurately predicts the status of pests and diseases in farmland. In practice, it is shown that the algorithm can effectively solve the problems of large number of pests, pest accumulation, background interference, and has strong robustness.

1. Introduction

Wheat and corn are the main food crops in North China. The growth of crops often suffers from pests, which cause enormous economic losses to wheat and corn yields every year. There are many kinds of pests on agricultural crops, which attack the growth of crops and often erupt into disasters; so, there is a need for real-time and accurate monitoring of wheat and corn pests, develop reasonable prevention, and control measures to reduce economic losses. Traditional wheat versus corn pest detection primary methods still require staff from the base layer to enter the field to observe pest type characteristics, visually observe, and diagnose pest status in the area. This method has the characteristics of heavy workload and low efficiency. It cannot predict the occurrence of diseases and insect pests in real time, meet the needs of current pest monitoring, reduce the accuracy of agricultural pest monitoring, and is not conducive to the scale and automation of pest detection [1–3].

With the growth of computing resources, deep learning has developed rapidly, especially in the field of image, which provides a technical basis for lightweight farmland pest detection [4]. In the early stage of pest identification, artificial neural network, support vector machine, and other methods were used to realize pest identification, mainly based on the color, texture, morphology, and other characteristics of pests. It has high requirements for the body shape characteristics of pests in the data set and can only complete several categories. The identification results are very unstable. This method is essentially a classification problem. Only one pest can be solved in one picture, which is not in line with the actual environment [5]. With the growing maturity of deep learning technology [6, 7], a large number of excellent detection models have emerged in the field of target detection, such as SSD, Fast-RCNN, and Yolo [8–10]. These excellent target detection models extract features through convolutional neural networks. These algorithms are based on anchors to achieve target positioning. Recently, some target detection algorithms without anchors have emerged, such as Centernet [11]. However, for small target objects such as pests, the performance of no a priori frame algorithm is not very ideal. The target detection model has been widely used in pedestrian detection, vehicle detection, face detection, driverless, and other fields. It is also applicable to the target detection of pests. For example, Wei Yang and others have realized the automatic identification of pests by using the two-stage faster-RCNN detection model [12]. Yuan and others have realized the automatic recognition and counting of 8 types of insects by using the yolov3 model, and the recognition rate can reach 70.98% [13], while the accuracy of pest recognition still needs to be improved.

Most of the existing recognition methods use network pictures as datasets for training. Although they have good recognition rate, the pictures collected on the network can only identify one side of the pests [14]. There is a big gap in practical applications. The robustness of the model is not high, and the deployment needs a high computing resources device, which cannot meet the current actual needs [15]. In this paper, a lightweight detection model is proposed to solve the problem that the existing target detection model requires a large amount of computing resources. The model is deployed on low-cost devices, mainly to monitor pests in farmland, so as to achieve the scale and automation of pest monitoring. The detection model in this paper mainly refers to the idea of the Yolo algorithm. Because each pest has a different size, the model outputs three different size feature matrices, and sets three anchors with different size for each feature matrix, and the regression predicts pests with different size. The deployment of the detection model on the local device is implemented, which reduces the waste of computing resources and greatly reduces the cost.

2. Related Work

2.1. Image Processing. The nature of depth learning is end-to-end; so, the construction of datasets is the basis of indepth learning. Because of the scarcity of public pest datasets, there is a big difference from the actual situation. The pest dataset used in this paper was obtained by using a telemetry lamp device in Shandong Province. The device mainly uses light to attract pests, kills pests through heating chamber, falls on the insect board, and takes pictures of pests through high-definition camera. A total of 10,000 pictures of pests were collected. 6,144 useful pictures of pests were manually screened out. A total of 26 pests were identified. Some of the samples were shown in Figure 1. The category and number distribution of pests were shown in Table 1 below. The labelImg tool was used for labeling manually to generate VOC2007 format [16].

The input of the pest detection model in this paper is a 416×416 size picture, while the size of the data set is not the same, the data need to be adjusted to a uniform size to be the input of the network, if the direct resize the picture is distorted, and may lose the original characteristics of the picture; so, the method of adding padding to the picture is adopted to prevent the distortion of the data set. For a good target detection model, it requires massive data sets for training to avoid overfitting of the network and enhance the robustness of the model, while the number of this data set is obviously insufficient, a total of seven methods have been used to further augment the data set, namely, rotation, horizontal translation, vertical translation, perspective transformation, and scaling horizontal inversion as well as brightness enhancement, thereby enhancing the generalization ability of the model, and the dataset was expanded to more than 20000 sheets [17].

In order to further improve the robustness of the model and enhance its generalization ability, the mosaic data enhancement method is used when loading the data set. The mosaic data enhancement refers to the data enhancement method of Cutmix [18]. The Cutmix data enhancement method is to splice two images, but mosaic uses four images to enrich the background of the object and increase the diversity of the data. When calculating in BN (batch normalization) layer, the larger the setting of batch size is, the closer the mean value and variance of the whole data set will be, and the better the effect will be. Due to the limitation of GPU memory, it is impossible to train multiple pictures at one time. When we put four pictures together and input them into the network, the batch size of the input network will be increased in disguise. As shown in Figure 2, the image is enhanced by mosaic data.

2.2. Target Detection Algorithm. The object detection algorithm in deep learning consists of three parts: backbone, neck, and head. Backbone is mainly used for feature extraction to generate feature map, such as VggNet, ResNet, and Densenet [19–21]. The function of neck is to fuse feature maps of different scales for further feature extraction, such as FPN, PAN, and BiFPN [22–24]. Finally, the head is used for classification and regression prediction to complete the target recognition and positioning. The head is mainly divided into two parts. One is based on anchor, such as SSD, Yolo, and Retinanet. It sets anchor box in feature points in feature map in advance and locates the target by adjusting the size and position of anchor box. There are two main problems: the preset anchor box size is fixed, and the other is based on anchor free, such as Cornernet [25] and Centernet. When building the model, it takes the target as a point; that is, the center point of the target BBox uses key points to find the center point and returns to other target attributes; in the experimental study, the accuracy of anchor is higher than that of anchor free; so, this paper proposes a lightweight target detection algorithm based on Yolo's idea and improves the two shortcomings of two anchor bases.

2.3. Model Structure. The network architecture draws lessons from the Yolo model structure and uses the one stage method to build the model. The Yolo series detection model has been very perfect after three generations of iteration. It has the advantages of high calculation speed and high accuracy and is widely used. In this paper, the network

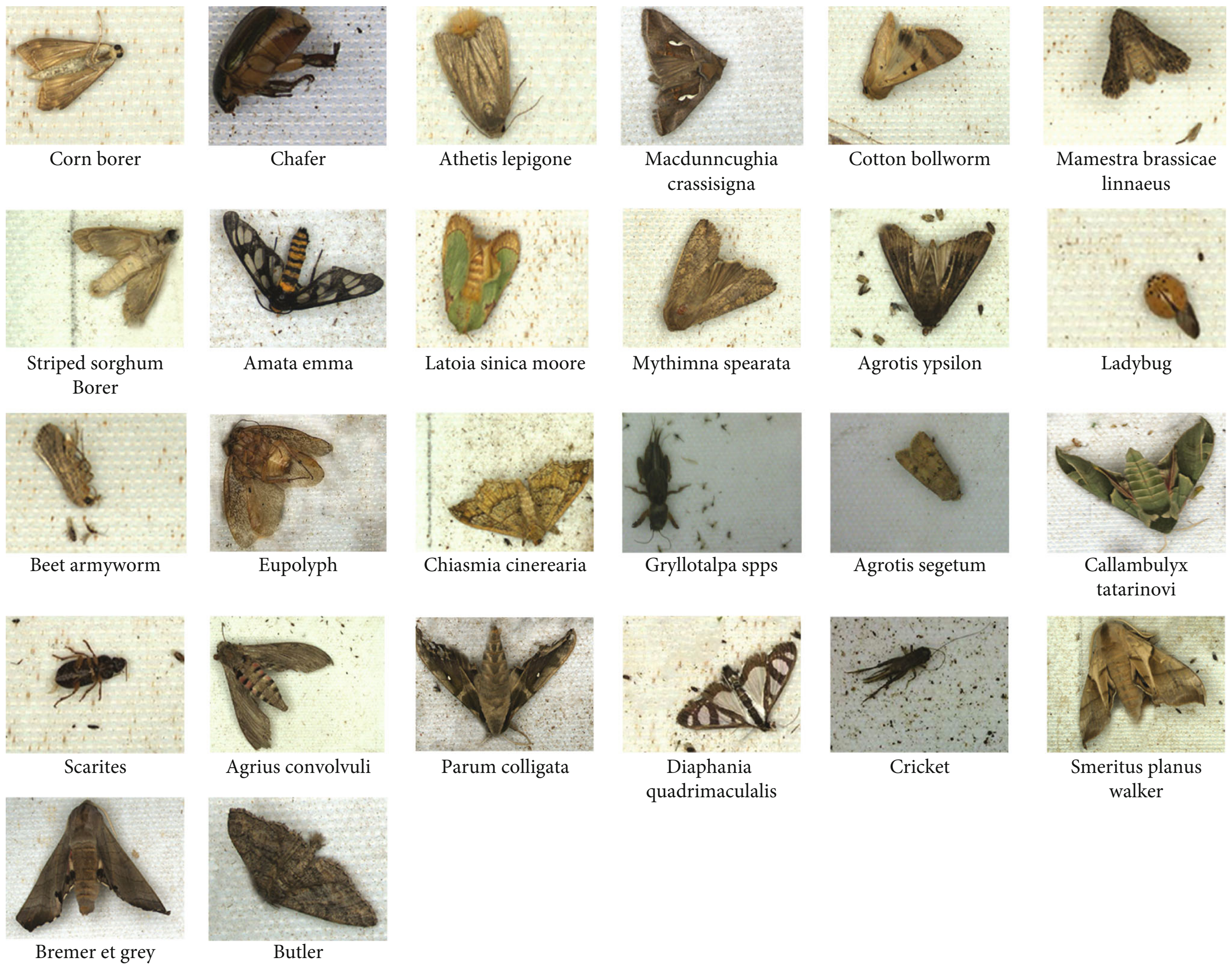


Figure 1: Insect samples of each classification target.

Table 1: Sample and target quantity of each classification.

No.	Name	Sample quantity	No.	Name	Sample quantity
0	Corn borer	545	13	Amata emma	492
1	Cotton bollworm	1565	14	Gryllotalpa spps	167
2	Chafer	1799	15	Chiasmia cinerearia	221
3	Macdunncughia crassisigna	159	16	Eupolyph	256
4	Athetis lepigone	928	17	Callambulyx tatarinovi	340
5	Mamestra brassicae linnaeus	727	18	Scarites	229
6		265	19	Cricket	179
7	Striped sorghum borer	73	20	Diaphania quadrimaculalis	238
8	Agrotis ypsilon	624	21		200
9	Mythimna Separata	189	22	Agrius convolvuli	200
10	Latoia sinica Moore	366	23	Smeritus planus walker	203
11	Beet armyworm	183	24	Parum colligata	190
12	Agrotis segetum Ladybug	374	25	Bremer et Grey Butler	147

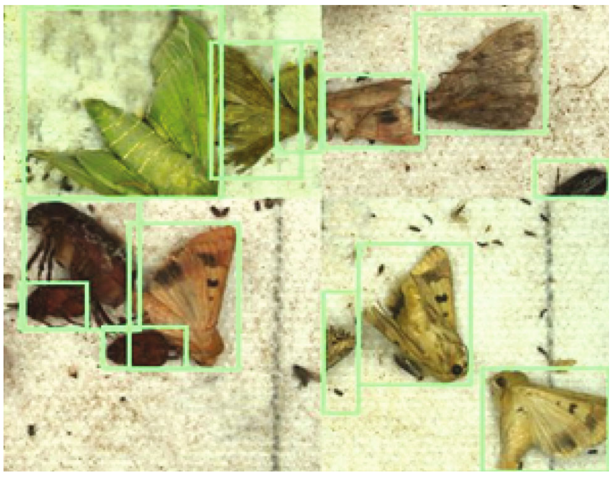


Figure 2: Mosaic enhanced image.

architecture constructed by the algorithm according to its idea is shown in Figure3. The darknet-53 of the backbone in yolov3 is replaced by the improved Efficientnet [26], and the PAN is replaced by the FPN in neck to improve the feature fusion.

The backbone in this algorithm is based on Efficientnet-B2 [26] network, which was proposed by Google in 2019 for image classification. The input of the Efficientnet-B2 network is 260×260. In order to better extract the characteristics of the image, the input size is changed to 416×416, and the SPP network structure is added in the last block, so as to further sample the feature map. The Efficientnet-B2 network is mainly composed of seven MBConv blocks. The structure is shown in Figure 4. It uses 1×1 ordinary convolution for dimension raising, then BN and swish activation functions, and then uses deep separable volume for down sampling. After an SE module, it uses a 1×1 convolution for dimension reduction, and normalizes through a BN layer. Finally, the input characteristic matrix is added with the main channel characteristic matrix through the shortcut branch to complete the output of the characteristic matrix. Only when the dimension of the input MBConv structure characteristic matrix is the same as that of the output characteristic matrix, the splicing operation is carried out. In the first ascending dimension of 1×1 convolution layer, the input MBConv structure characteristic matrix is connected with the output characteristic matrix, and the number of convolution kernels is N times of the input characteristic matrix channel.

The changed network parameters are shown in Table 2, rechanging the input size and adding the SPP structure, enhancing the generalization ability of the algorithm and having a broader vision of features.

Deep separable convolution [27] is a deformation of traditional convolution. It is different from traditional convolution in that the number of channels of its convolution core is equal to the number of channels of the input characteristic matrix, the number of channels of the output characteristic matrix, the number of convolution cores, and the size of the convolution core is a matrix of 1×1. The structural schematic of the ordinary convolution is shown in Figure 5, the deep separable convolution is shown in Figure 6, assuming that D_F represents the size of the input feature matrix, M is the number of feature matrices, D_K is the size of the convolution kernel, N is the number of output feature matrices, and the computational comparison of the deep separable convolution with ordinary convolution is shown by equation (1).

$$\frac{D_k \cdot D_k \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F}{D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F} = \frac{1}{N} + \frac{1}{D_K^2}. \quad (1)$$

Assuming that the size of the convolutional kernel is 3×3, the formula is equal to $1/N + 1/9$; so, the computation of the ordinary convolution is theoretically 8 to 9 times the depth, which is visible to be much less than the ordinary convolution.

The structure of SE module is shown in Figure 7. The feature map performs a global average pooling and transforms the size of the feature matrix to 1×1, which performs two full connection layers. The first full connection uses the swish activation function, and the number of channels becomes 1/4 of the original number. The second full connection uses the sigmoid activation function, the number of nodes is equal to the number of channels of the output characteristic matrix of the depth separable convolution layer, and the final output is obtained by multiplying with the input feature map. The SE module is similar to the self-attention [28] mechanism and increases the interesting features through the output of the sigmoid function.

In the main feature network, a SPP (spatial pyramid pooling) structure is added; that is, three maximum pooling samplings are conducted after the last MBConv block, because the stride is all one, the size of the padding, and feature matrix in the feature matrix is added not to change. The three obtained feature matrices and the input feature matrix are splicing to obtain the feature matrix of 4 times the depth. The structure is shown in Figure 8.

In the stage of feature fusion, the idea of the PANet (path aggregation network) structure is used for multiscale feature fusion. In the backbone feature network, three feature matrices with different sizes are collected, as shown in Figure 3. The three feature matrices are MBConv5, MBConv7, and MBConv8. First, convolution and SPP operations with convolution kernel size of 3×3 are performed on MBConv8, and then up sampling and stacking with MBConv7 are performed. The feature layer continues to perform up sampling and fusion stacking with MBConv5. Through two times of length and width expansion, the up sampling operation is completed, and the feature layer with high semantics is obtained. The convolution operation with convolution kernel size of 5×5, and down sampling are carried out, respectively, to ensure the feature information of the target, which is more conducive to detecting objects of different sizes.

In neck, after completing the fusion stack, the three characteristic matrices perform the convolution operation with the convolution kernel size of 5×5, respectively, and output three characteristic matrices with different sizes, namely, (13,13,93), (26,26,93), and (52,52,93). The first two

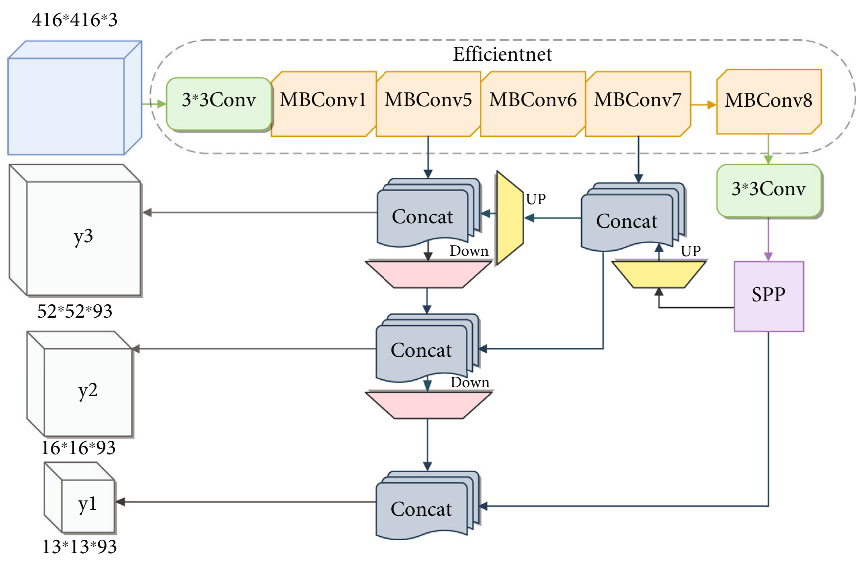


Figure 3: Model structure.

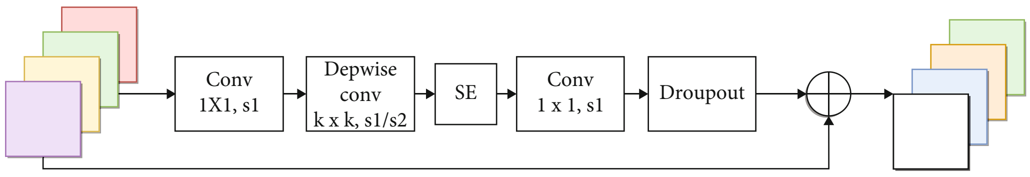


Figure 4: MBConv structure.

Table 2: Network parameters of features extraction.

Stage	Operation type/ convolutional core size	Matrix size	Number of channels	Number of repetitions
1	Conv, $k3 \times 3$	416×416	32	1
2	MBConv, $k3 \times 3$	208×208	16	1
3	MBConv, $k3 \times 3$	208×208	24	2
4	MBConv, $k5 \times 5$	104×104	48	2
5	MBConv, $k3 \times 3$	52×52	80	3
6	MBConv, $k5 \times 5$	26×26	120	3
7	MBConv, $k5 \times 5$	26×26	192	4
8	MBConv, $k3 \times 3$	13×13	350	1
9	SPP, $k5$, $k9$, $k13$	13×13	1408	1

dimensions represent the size of the feature layer and are used to detect objects of different sizes, while 93 represents that each feature point has three a priori boxes. Each a priori box contains five parameters, namely, length, width, center point, and classification probability. A total of 26 pests are detected this time; so, this dimension is 3×31.

Swish activation function is used in MBconv block. Swish is an improved version of sigmoid and Relu, similar to the combination of Relu and Sigmoid, which contains a parameter β, and β can be set as a constant or a trainable parameter. It has the characteristics of no upper bound but lower bound, smoothness, and nonmonotonicity, as shown in formulae (2) and (3).

$$\mathrm{Sigmoid}(x) = \frac{1}{1 + e^{-x}}, \tag{2}$$

$$f(x) = x \cdot \mathrm{sigmoid}(\beta x). \tag{3}$$

The Swish activation function not only has the advantages of the Relu and Sigmoid functions but also is superior to Relu in the deep model. It can be seen as a smoothing function between the linear function and the Relu function.

2.4. Loss Function. The loss function in target detection can be roughly divided into three parts: confidence loss, classification loss, and location loss. In the confidence loss function, IOU (intersection over union) is used to judge the relative position relationship between the prediction frame and the real frame, but IOU cannot judge the overlapping area, center distance, and aspect ratio between the prediction frame and the real frame. Therefore, this paper uses CIOU (complete, IOU) [29] to replace IOU and adds a penalty term, so that the regression loss function tends to converge and reduce the divergence of loss function in the training process. The loss function of the model is shown in formula (4).

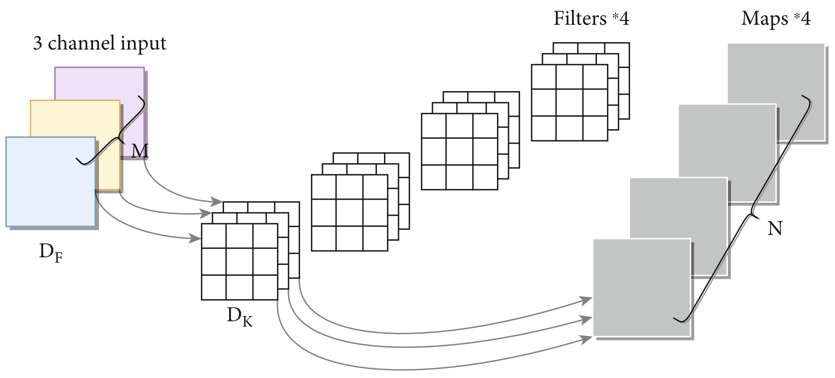


Figure 5: Ordinary convolution.

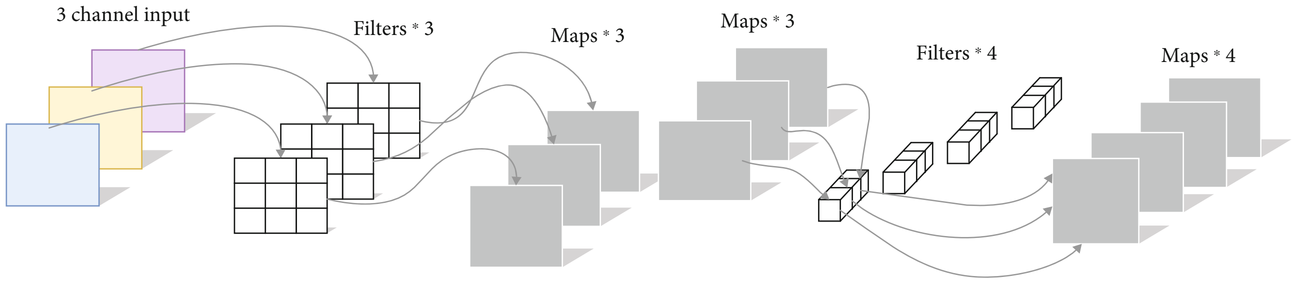


Figure 6: Depthwise separable convolution.

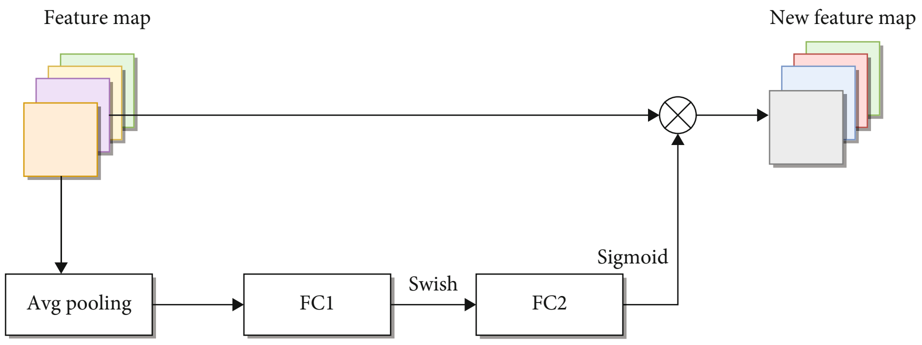


Figure 7: SE block.

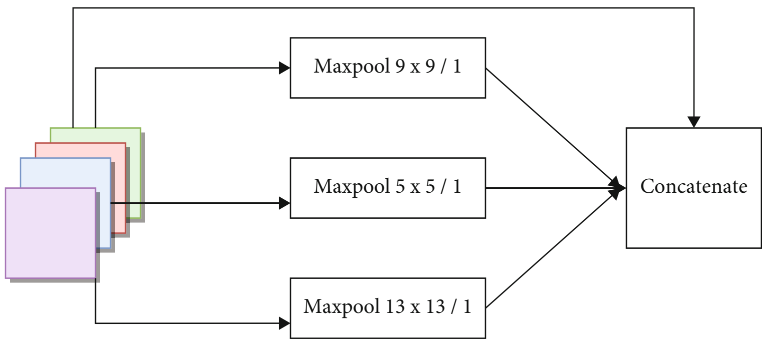


Figure 8: Spatial pyramid pooling.

$$L(o, c, O, C, l, g) = \lambda_1 L_{\mathrm{conf}}(o, c) + \lambda_2 L_{\mathrm{cla}}(O, C) + \lambda_3 L_{\mathrm{loc}}(l, g). \tag{4}$$

Among them, λ_1, $\lambda_{2,}$ and λ_3 are equilibrium coefficients. The weight of each loss function is adjusted by setting the size of each y. The specific formulas of confidence loss, classification loss, and positioning loss are shown in formula (5), (6), and (7).

$$L_{\mathrm{conf}}(o, c) = -\frac{\sum_i (o_i \ln(\hat{c}) + (1 - o_i) \ln(1 - \hat{c}_i))}{N}, \tag{5}$$

$$L_{\mathrm{cla}}(O, C) = -\frac{\sum_{i \in \mathrm{pos}} \sum_{j \in \mathrm{cla}} \left(O_{ij} \ln\left(\hat{C}_{ij}\right) + (1 - O_{ij}) \ln\left(1 - \hat{C}_{ij}\right)\right)}{N_{\mathrm{pos}}}, \tag{6}$$

$$L_{\mathrm{loc}}(l, g) = \frac{\sum_{i \in \mathrm{pos}} \sum_{m \in \{x,y,w,h\}} \left(l\wedge_i^m - g\wedge_i^m\right)^2}{N_{\mathrm{pos}}}. \tag{7}$$

In formula (5), $o_i \in [0, 1]$, which represents the CIOU between the prediction frame and the real frame, c is the prediction value, $\hat{c}$ is the prediction confidence obtained by c through the Sigmoid function, and N is the number of positive and negative samples. In formula (6), $O_{ij} \in \{0, 1\}$ indicates whether there is a class j target in the prediction frame i, C_{ij} is the prediction value, $\hat{C}_{ij}$ is the target probability obtained by C_{ij} through the Sigmoid function, and N_{pos} is the number of positive samples. In formula (7), $\hat{l}_i^m$ is the center point coordinate and length and width in the prediction frame, and $\hat{g}_i^m$ is the information of the real frame. The formula of CIOU is shown in (8).

$$\mathrm{CIOU} = \mathrm{IOU} \text{-} \frac{\rho^2\left(b, b^{gt}\right)}{c^2} - \alpha v, \tag{8}$$

where b represents the coordinates of the distance between the center point of the prediction box, b^{gt} represents the coordinates of the distance between the center point of the real box, $\rho(.)$ represents the calculation of Euclidean distance, c represents the distance between the prediction box and the diagonal line of the minimum bounding box of the real box, and penalty factors α and v are added to it, as shown in (9) and (10).

$$\alpha = \frac{v}{1 - IOU + v}, \tag{9}$$

$$v = \frac{4}{\pi^2} \left(\arctan \frac{w^{gt}}{h^{gt}} - \arctan \frac{w}{h}\right)^2. \tag{10}$$

$w^{gt}$and h^{gt} represent the width and height of the real box. w and h are the width and height of the prediction box. This penalty is mainly to make the width of the prediction box as fast as possible to be close to the width and height of the real box. Finally, we get the regression loss function as shown in formula (11).

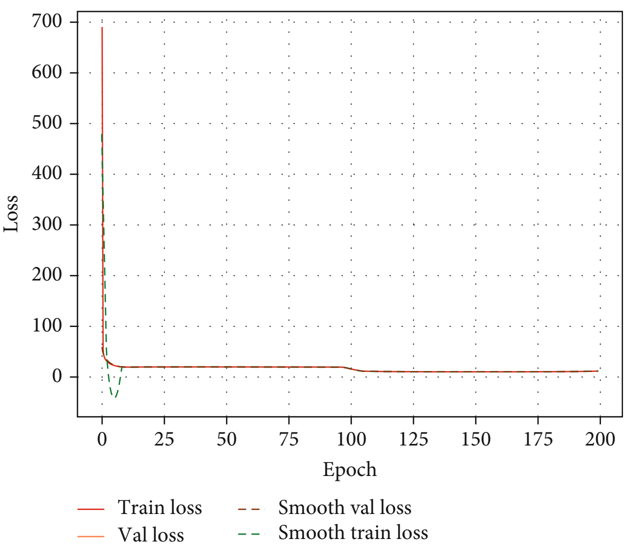


FIGURE 9: Training loss function curve.

$$L_{\mathrm{ciou}} = 1 - IOU + \frac{\rho^2\left(b, b^{gt}\right)}{c^2} + \alpha v. \tag{11}$$

The present algorithm uses anchor base to predict the target position of the object. The main idea is to set anchor, of different sizes in each feature point in the feature matrix, although the problem of positive and negative sample imbalance during training, leading to reduced model accuracy. For example, an image may produce tens of thousands of candidate boxes, but only few parts contain the target; the target box is positive sample, and the negative sample with no candidate box. The focal loss [30] function solves the problem of positive and negative sample imbalance and also controls the weights of easily classified and difficult classified samples. The formula of the focal loss function is as follows: (12), (13), and (14).

$$p_t \begin{cases} p \text{ if } \mathrm{y} = 1 \\ 1 - p \text{ otherwise} \end{cases}, \tag{12}$$

$$\alpha_t = \begin{cases} \alpha & \text{if } \mathrm{y} = 1 \\ 1 - \alpha & \text{otherwise} \end{cases}, \tag{13}$$

$$\mathrm{FL}(p_t) = -\alpha_t (1 - p_t)^{\Upsilon} \log(p_t). \tag{14}$$

Among them, $(1 - p_t)^{\gamma}$ is called the adjustment coefficient. When p_t tends to 0, the adjustment coefficient tends 1, the contribution to loss increases, and the adjustment coefficient tends 0, equivalent to a small contribution to the total loss. When the coefficients $\Upsilon = 0$, the traditional crossentropy loss function realizes the adjustment coefficient by adjusting the Υ.

3. Results and Discussion

3.1. Experimental Environment. This experiment uses Pytorch as a deep learning framework to accelerate the

Table 3: Prediction effect of target detection model in this paper on insects in each test set.

Name	AP (%)	Name	AP (%)
Corn borer	99.36	Amata emma	99.13
Cotton bollworm	99.27	Gryllotalpa spps	95.50
Chafer	99.59	Chiasmia cinerearia	96.56
Macdunncughia crassisigna	71.43	Eupolyph	96.67
Athetis lepigone	99.01	Callambulyx tatarinovi	99.58
Mamestra brassicae Linnaeus	96.15	Scarites	94.85
Striped sorghum borer	98.20	Cricket	100.00
Agrotis ypsilon	62.86	Diaphania quadrimaculalis	97.50
Mythimna Separata	98.32	Agrius convolvuli	99.58
Latoia sinica Moore	99.94	Smeritus planus walker	86.96
Beet armyworm	92.75	Parum colligata	88.60
Agrotis segetum	81.90	Bremer et Grey	89.81
Ladybug	95.82	Butler	97.66
mAP			93.73

training model. The hardware configuration is: R5-3600 processor, 16GB memory, and Nvidia RTX3070 graphics card; the software environment is Windows10 system, Python3.7, Pytorch1.9, CUDA version 11.0, and cuDNN version 8.0.1.

3.2. Evaluation Criteria. Average precision (AP) is a mainstream measure of the target detection model. AP is calculated by calculating the AP of each target category. AP is calculated by using the area under precision-recall (P-R) curve as the AP value, where precision and recall formulas (15) and (16) show.

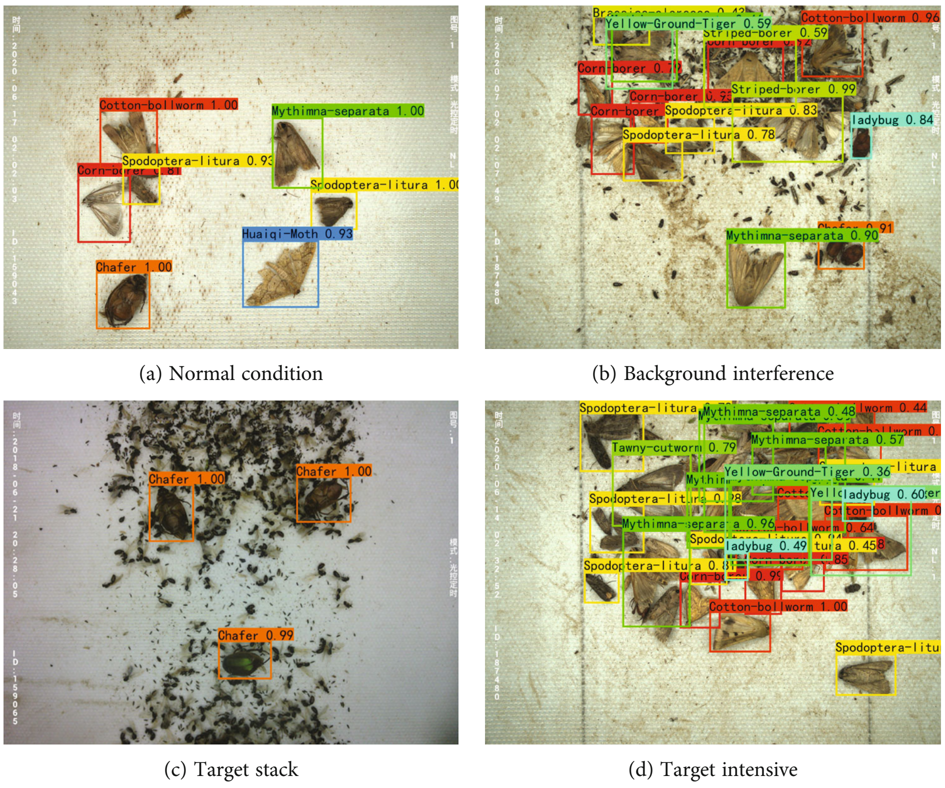
$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{15}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{16}$$

TP (true positives) represents the correct target class classification and is a positive sample, FN (false negatives) represents the wrong result of model classification, and the sample is negative, FP (false positives) represents the wrong model classification, the sample is negative, and map has become a recognized method of target detection and is widely used.

3.3. Results and Analysis. In the process of regression prediction, nine candidate frames are set according to three characteristic matrices with different sizes. Because of different training data sets, the sizes of candidate frames are also different. In this paper, K-means clustering algorithm is used to find the appropriate size of prior frames in the training set. The K-means clustering algorithm is different from the standard one. It calculates the distance between candidate frames through CIOU, and the final nine candidate frames are (29, 35), (55, 70), (72, 117), (87, 83), (94, 149), (115, 110), (124, 186), (145, 146), and (182, 220), which fit 81% of the frames of the dataset. During training, the transfer learning method is adopted. The preloaded model is the model of VOC data set, and the partially frozen backbone method is used to iterate 50 times, then thaw the training, and then iterate 50 times. As shown in Figure 9, the decline curve of the loss function after 200 times of model training is displayed. The cosine annealing method is used to reduce the learning rate. Among them, the learning rate adopts the cosine annealing method to reduce the learning rate through the cosine function. In the cosine function, with the increase of x, the cosine value first decreases slowly, then accelerates the decline and decreases slowly again. It is easier for the model to find the best advantage, and the label smoothing method is added, which is mainly to punish the classification so that the model cannot be classified too accurately and prevent overfitting

We divide the dataset into training set, validation set, and test set according to $7:1:2$ ratio and then calculate the AP value of the pest species through the test set. Table 3 shows that the mAP value of the model can reach 93.73%, which proves that the detection model has good performance. We can see that the accuracy of Scarite pests can reach 100%, while the AP of Agrotis ipsilon pests is only 62.86%. From the table, we can know that this may be caused by the uneven distribution of pest species, or it may make the characteristics of pests have little difference, the number of samples is small, and the model is not enough to extract the features. To further demonstrate the robustness of the model, we tested the detection of the algorithm in real-world situations. As shown in Figure 10, the algorithm detection results in different scenarios. In Figure 10(a), there is little background interference, the number of pests is always constant, and there is no stacking. The algorithm accurately predicts the type and location of pests. In Figure 10(b), although the number of pests has decreased, the background has been seriously disturbed, and the pests have been flipped. The model still finds the location and classification of pests accurately. In Figure 10(c), stacks of pests appear, and the number and species increase. Although pests are identified as two species, this may be due to the low IOU threshold, but most pests are accurately identified. In Figure 10(d), there are a large number of pests, and some


(a) Normal condition

(b) Background interference

(c) Target stack

(d) Target intensive

Figure 10: Model detection results in different scenarios.

Table 4: Average recognition accuracy, detection speed, and model size of different models for insects.

Models	mAP(%)	Speed (s)	Parameters (million)	Size (MB)
Yolov3	89.71	1.90	61.66	235.21
ShuffleNet-yolo	75.71	0.42	9.60	36.60
MobileNetv1-yolo	84.66	0.60	12.40	47.31
MobileNetv2-yolo	81.65	0.63	10.51	40.09
MobileNetv3-yolo	81.93	0.54	11.13	43.63
GhostNet-yolo	82.49	0.53	11.14	42.49
EfficientNet-yolo	88.16	0.69	15.12	59.23
Our model	93.73	0.72	15.66	59.78

of them are stacked, which is compared with the actual number. Despite the missed pest detection, the model still detects the location of pests and their corresponding species. Through these complex cases, it can be shown that the target detection algorithm has excellent robustness, can cope with a variety of complex environments, and has wide application.

3.4. Comparison of Several Models. To evaluate the performance of the algorithm, this paper is compared with yolov3, yolov3 algorithm adopts Darknet-53 as the backbone feature network, it uses the original EfficientNet-B2, for comparison, and finally, on the basis of this algorithm replaced different lightweight backbone, such as Google MobileNet [31, 32] series, and more lightweight ShuffleNet [33, 34] and Huawei GhostNet [35], these are lightweight classification networks. The same training set is used in the experiments, finally, test evaluation using the same test set. Finally, the model is deployed on industrial tablets, its CPU is adopted as a J1900 processor, 4GB of running memory, and test the inference speed of the model, and results are shown in Table 4. We can see that the yolov3 model has high accuracy, but the number of participants can reach 60 million, the inference of the model is the longest time-consuming, high performance requirements for the equipment, not convenient for practical deployment, and using a lightweight backbone, we can see that the ShuffleNet parameter is minimal, the model is only 36 MB, and the detection speed is also the fastest, but the model really has the lowest accuracy. Compared with MobileNet series and GhostNet, the comprehensive performance of explicit GhostNet is the best, with the volume size of only 42 MB, the accuracy can be up to 82%, and the detection speed can basically reach two pictures per second, with good timeliness. Although these models are small, the accuracy does not meet the commercial standard. For this algorithm compared with EfficientNet-B2, the accuracy rate can reach 93% and improve by almost 5 percentage points, and the volume of the algorithm is only a quarter more than GhostNet; so, this algorithm has lightweight and high sex characteristics.

4. Conclusion

In order to achieve automatic recognition and classification of farmland pests, a lightweight target detection model is proposed. Based on the idea of yolov3, the EfficientNet-B2 classification network is used as the main feature extraction network and improved. PANet is added to THE neck, and CIOU is used as the loss function of target detection to highlight the relative position between the prediction box and the real box. The problem that the low confidence prediction box is filtered due to the overlap of the target box, and the prediction box is avoided. Focal loss function is used to solve the imbalance between positive and negative samples during training. In order to increase the diversity of datasets, the Mosaic data enhancement method is used to increase the diversity of data and improve the robustness of the model. Experiments show that the mAP value of this algorithm can reach 93% accuracy, and it has good recognition ability. The algorithm also has strong recognition ability in complex environment. Compared with other algorithms, this algorithm not only recognizes many kinds of classes but also has high accuracy and wide application.

Data Availability

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

Conflicts of Interest

The authors declared that they have no conflicts of interest regarding this work.

Acknowledgments

This work is supported by the Major Agricultural Application Technology Innovation Project of Shandong Province. The project number is SD2019ZZ007.

References

[1] H. Liu, "Driving agricultural modernization with photography agriculture to accelerate the digital transformation of modern agriculture," *China Agricultural Resources and Regional Planning*, vol. 40, 2019.

[2] Y. Peng, Y. Yang, W. Yang, and S. Chen, "Design of intelligent agricultural management system based on Internet of Things technology," *Modern Agricultural Science and Technology*, vol. 4, no. 19, 2020.

[3] J. Nie, R. Sun, and X. Deng, "Processing of uncertain complex events in precision agriculture based on data lineage management," *Transactions of the Chinese Society of Agricultural Machinery*, vol. 47, no. 5, pp. 245–253, 2016.

[4] H. Feng and Q. Yao, "Automatic identification and monitoring technology of agricultural pests," *Plant Protection*, vol. 44, no. 5, 2018.

[5] X. Shi, M. Zhao, H. Li, and Z. Wen, "Research on corn pest species recognition system based on image processing," *Agriculture and Technology*, vol. 41, no. 12, pp. 28–31, 2021.

[6] Y. Shen, *Research on stored-grain pest detection algorithm based on deep learning, Master thesis*, Beijing University of Posts and Telecommunications, 2018.

[7] Y. Zhang, M. Jiang, and P. Yu, "Image recognition method of agricultural pests based on multi-feature fusion and sparse representation," *China Agricultural Sciences*, vol. 51, no. 11, pp. 67–76, 2018.

[8] W. Liu, D. Anguelov, and D. Erhan, "SSD: Single shot multibox detector," in *in Proc. European conference on computer vision*, pp. 21–37, Springer, Cham, 2016.

[9] S. Ren, K. He, and R. Girshick, "Faster R-CNN: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[10] J. Redmon, S. Divvala, and R. Girshick, "You only look once: unified, real-time object detection," in *in Proc. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, Las Vegas, USA, 2016.

[11] X. Zhou, D. Wang, and P. Krähenbühl, *Objects as points*, 2019, https://arxiv.org/abs/1904.07850.

[12] Y. Wei, B. I. Xiu-Li, and B. Xiao, "Agricultural insect pest detection method based on regional convolutional neural network," *Computer Science*, vol. 45, no. 11A, pp. 226–229, 2018.

[13] Z. Yuan, H. Yuan, Y. Yan, S. Liu, and S. Tian, "Automatic recognition and classification algorithm of field Insects Based on lightweight deep learning model," *Journal of Jilin University (Engineering and Technology Edition)*, vol. 51, no. 3, pp. 1131–1139, 2021.

[14] W. An, L. Zhao, X. Zhang, and Y. Han, "Phenological prediction and prediction of major pests in orchards in Baoding area," *Modern rural Science and Technology*, vol. 12, no. 3, pp. 36-37, 2020.

[15] X. Liu, *Research on lightweight insect detection model based on deep learning, Mater thesis*, Beijing Forestry University, 2019.

[16] M. Everingham, L. V. Gool, and C. K. I. Williams, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

[17] Y. Gao, B. Zhou, and X. Hu, "Research on convolutional neural network image recognition based on data enhancement," *Computer Technology and Development*, vol. 28, no. 8, pp. 62–65, 2021.

[18] J. Xing, M. Jia, F. Xu, and J. Hu, "Recognition method of small defects on workpiece surface based on Cut Mix and YOLOv3 (English)," *Journal of Southeast University (English Edition)*, vol. 37, no. 2, pp. 128–136, 2021.

[19] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *Computer Science*, 2014, https://arxiv.org/abs/1409.1556v6.

[20] K. He, X. Zhang, and S. Ren, "Deep residual learning for image recognition," in *in Proc Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, USA, 2016.

[21] G. Huang, Z. Liu, and L. Van Der Maaten, "Densely connected convolutional networks,," in *in Proc. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4700–4708, Hawaii, USA, 2017.

[22] T. Y. Lin, P. Dollar, and R. Girshick, "Feature pyramid networks for object detection," in *in Proc. 2017 IEEE conference on computer vision and pattern recognition (CVPR), IEEE Computer Society*, Hawaii, SA, 2017.

[23] S. Liu, L. Qi, and H. Qin, "Path aggregation network for instance segmentation," in *in Proc. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8759–8768, Salt Lake City, USA, 2018.

[24] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: scalable and efficient object detection," in *in Proc. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10781–10790, Seattle, USA, 2020.

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

[26] M. Tan and Q. Le, "Efficientnet: rethinking model scaling for convolutional neural networks," in *in Proc. International Conference on Machine Learning*, pp. 6105–6114, PMLR, 2019.

[27] A. G. Howard, M. Zhu, and B. Chen, "Mobilenets: efficient convolutional neural networks for mobile vision applications," 2017, ar Xiv preprint ar Xiv: 1704.04861.

[28] S. Khan, M. Naseer, and M. Hayat, "Transformers in vision: a survey," 2021, ar Xiv preprint ar Xiv: 2101.01169.

[29] Z. Zheng, P. Wang, and D. Ren, "Enhancing geometric factors in model learning and inference for object detection and instance segmentation," 2020, ar Xiv preprint ar Xiv: 2005.03572.

[30] T. Y. Lin, P. Goyal, and R. Girshick, "Focal loss for dense object detection," in *in Proc. Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, Hawaii, USA, 2017.

[31] A. Howard, A. Zhmoginov, and L. Chen, "Inverted residuals and linear bottlenecks: Mobile networks for classification," *detection and segmentation*, 2018, https://arxiv.org/abs/1801.04381v2.

[32] A. Howard, M. Sandler, and G. Chu, "Searching for mobilenetv 3," in *in Proc. Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1314–1324, Long Beach, USA, 2019.

[33] X. Zhang, X. Zhou, and M. Lin, "Shufflenet: an extremely efficient convolutional neural network for mobile devices," in *in Proc. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6848–6856, Salt Lake City, USA, 2018.

[34] N. Ma, X. Zhang, and H. Zheng, *Shuffle Net V2: Practical Guidelines for Efficient CNN Architecture Design*, in Proc. Springer, Cham. Springer, Cham, 2018.

[35] K. Han, Y. Wang, and Q. Tian, "Ghostnet: more features from cheap operations," in *in Proc. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1580–1589, Seattle, USA, 2020.