IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 14, 2021, accepted April 5, 2021, date of publication April 9, 2021, date of current version April 19, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3072211

Small-Object Detection Based on YOLO and Dense Block via Image Super-Resolution

ZHUANG-ZHUANG WANG[1,2,3], KAI XIE[1,2,3], XIN-YU ZHANG[1,2,3], HUA-QUAN CHEN[1,2,3], CHANG WEN[3,4], AND JIAN-BIAO HE[5]

[1]School of Electronic Information, Yangtze University, Jingzhou 434023, China
[2]National Demonstration Center for Experimental Electrical and Electronic Education, Yangtze University, Jingzhou 434023, China
[3]Western Institute of Yangtze University, Karamay 834000, China
[4]School of Computer Science, Yangtze University, Jingzhou 434023, China
[5]School of Computer Science and Engineering, Central South University, Changsha 410083, China

Corresponding author: Kai Xie (pami2009@163.com)

This work was supported in part by the Natural Science Foundation of Xinjiang Uygur Autonomous Region under Grant 2020D01A131, in part by the Fund of Hubei Ministry of Education under Grant B2019039, in part by the Graduate Teaching and Research Fund of Yangtze University under Grant YJY201909, in part by the Teaching and Research Fund of Yangtze University under Grant JY2019011, in part by the Undergraduate Training Programs for Innovation and Entrepreneurship of Yangtze University under Grant 2019099, and in part by the National College Student Innovation and Entrepreneurship Training Program under Grant 202010489007.

ABSTRACT Small-object detection is a basic and challenging problem in computer vision tasks. It is widely used in pedestrian detection, traffic sign detection, and other fields. This paper proposes a deep learning small-object detection method based on image super-resolution to improve the speed and accuracy of small-object detection. First, we add a feature texture transfer (FTT) module at the input end to improve the image resolution at this end as well as to remove the noise in the image. Then, in the backbone network, using the Darknet53 framework, we use dense blocks to replace residual blocks to reduce the number of network structure parameters to avoid unnecessary calculations. Then, to make full use of the features of small targets in the image, the neck uses a combination of SPPnet and PANnet to complete this part of the multi-scale feature fusion work. Finally, the problem of image background and foreground imbalance is solved by adding the foreground and background balance loss function to the YOLOv4 loss function part. The results of the experiment conducted using our self-built dataset show that the proposed method has higher accuracy and speed compared with the currently available small-target detection methods.

INDEX TERMS Small-object detection, image super-resolution, dense block, foreground and background, balance loss function, multi-scale feature fusion.

I. INTRODUCTION

In recent years, although considerable progress has been made in object detection, a significant performance gap remains when detecting small and large targets. Small-object detection plays a key role in many tasks such as identifying traffic signs [1] or pedestrians that are almost invisible in low-resolution images. In medical imaging, early detection of masses and tumors is essential for an accurate early diagnosis. Another application is satellite image analysis [2], in which objects such as cars, ships, and houses must be annotated effectively. In other words, small-target detection requires further attention because increasingly complex systems are being deployed in the real world. To address this problem, in this study, we aim to detect small targets in college classrooms. An increasing number of college students use mobile phones in classroom. Improving the quality of classroom experience and creating a positive learning environment have become a problem that university educators must consider. We propose that schools can estimate learning performance by using cameras to detect the head movements of students in a classroom. They can send the obtained positional information to the head pose estimation [3] model, estimate head postures using deep learning, and determine whether a student's head is down or up to evaluate their listening state. However, before completing the head posture estimation, we need accurate positioning information of the student's head. The minimum resolution of the head size in an image is 15 × 15

The associate editor coordinating the review of this manuscript and approving it for publication was Qingli Li.

pixels, which belongs to the small- target category. Accurate estimation of the head position at low image resolutions has become an urgent problem that must be solved.

In previous research, the method of small-target detection was improved in the following aspects: image or feature scale, anchor, and small-target sample number. Because it is difficult to use the features of the last stage to make predictions, feature pyramid networks (FPN) can be used [4] to 1) predict targets of different scales at multiple scales; 2) enlarge the input image (using methods such as super-resolution);3) adjust the anchor scope as needed from the perspective of scope of anchor range and according to the scope of task detection, or if the target changes too much, use multi-scale detection; and 4) increase the number of small targets in the image. The number of small-target images in the training dataset provides more opportunities to learn the features of small targets.

In the COCO dataset [5], the definition of small targets is given by the size of the target box. Intuitively, when we see a picture, we first pay attention to the more eye-catching areas in the image. Generally, these eye-catching areas often occupy a larger portion of the picture. Small goals are often ignored. This situation also exists in the COCO dataset, and many small objects contained in the images are not marked. In addition, the area where the small target is located is small, and as a result, the feature-extracting process can extract very few features, which is not conducive for small-target detection. In the COCO dataset, many images contain few small objects, and most of the small objects are concentrated in a few areas. As a result, in the training process, half of the time, the model cannot learn the features of small targets. In addition, for small targets, the average number of anchors that can be matched is 1, and the average maximum intersection over union (IOU) [6] is 0.29, which shows that in many cases, some small targets have no or very few corresponding anchors. An analysis of the dataset reveals that there are two major reasons why small targets are not easy to detect: 1) the dataset contains fewer pictures of small targets, which causes the model to be biased toward medium and large targets during training, and 2) the area of the small target is too small, resulting in fewer anchors containing the target, which also means that the probability of detecting a small target becomes smaller. In view of the lack of small targets in the dataset, small sample data can be used for enhancement, and the characteristics of small targets can be fully learned during the training process. In addition to data enhancement, another idea is the feature pyramid network: features at different stages correspond to different receptive fields, and the degree of information abstraction they express is different. The shallow feature map indicated that the field is small and is more suitable for detecting small targets, whereas the deep feature map indicates that the field is large and suitable for detecting large targets. Therefore, some researchers have proposed merging to feature maps of different stages to improve the performance of target detection. Because feature maps of different resolutions can be fused to improve the richness and confidence of features to detect targets of different sizes, at times, only high-resolution feature maps are used to detect small targets and low-resolution rate feature maps are used to detect large targets, such as single-stage headless (SSH) [7] in face detection. The overall concept of fully convolutional networks (FCN) [8] is similar to that of a FPN. The only innovation is abandoning the fully connected layer and replacing the fully connected layer with an equivalent 1×1 convolution kernel so that the image scale of the network input can be inconsistent. Then, we continue up-sampling the stacked feature map to make it the same size as the original image. For the up-sampled stacked feature map, classification prediction is performed on the pixel points mapped to the original image position. In this way, fine image segmentation [9] can be performed based on the original image. For small-target detection, a finer location division can be achieved through pixel classification. In Scale Normalization for Image Pyramids (SNIP) [10] only target samples of appropriate sizes are trained. SNIP is used to train the detector only when the true value scale is close to the anchor scale. If the true value scale is too small or too large, it is discarded. Furthermore, various input images can be used for prediction. There is always an anchor point of suitable size, and the most suitable scale is selected for prediction. Although the SNIP method is simple to implement, it further analyzes the problems of the current detection algorithm in multi-scale detection. During training, only objects within a certain scale are selected for learning. In the COCO dataset, 3% of the detections increased the accuracy. Thereafter, the Scale Normalization for Image Pyramid with Efficient Resampling (SNIPER) network was proposed, the key to which is to reduce the number of SNIP calculations. SNIP draws on the idea of multi-scale training, which uses image pyramids as inputs to the model. Although this approach can improve the performance of the model, the amount of calculation is also very large because the model needs to process each image of each pixel size; moreover, the SNIPER [11] algorithm processes the context area around the ground truth (called chips) at an appropriate scale. The number of chips generated by each image during training adaptively changes according to the complexity of the scene. Because SNIPER runs on low-resolution chips, so it can gain and batch regularization during training, without the need to use synchronous batch normalization between GPUs for statistical information.

To compensate for the loss of small object information, it is important to increase the feature resolution. In this paper, a small-target detection method based on super-resolution (SR) reconstruction technology is proposed. Among the previous deep learning models, the SR convolutional neural network (SRCNN) [12], which is the first proposed model of SR technology, is mainly based on a single-image low-resolution reconstruction method. It uses only a three-layer network structure to achieve SR. The first layer uses the properties of the convolutional network to extract the characteristics of the image block, the second layer is used for nonlinear mapping, and the last layer uses the convolution operation

for SR reconstruction. However, for single-image SR, the reconstruction performance of the neural network model is very sensitive to small changes in the architecture, and the performance of the same model under different initialization and training techniques is limited. In response to this problem, Enhanced Deep Residual Networks (EDSR) [13] have been proposed. The author deletes unnecessary modules in the SR residual network (SRResNet) [14] architecture through analysis to ensure that the training model is more stable and the computational efficiency is better than that of the original network.

Considering that ordinary SR model training only uses the mean square error as the loss function, although a high peak signal-to-noise ratio (PSNR) can be obtained, the recovered image usually loses high-frequency details. The SR generative adversarial network (SRGAN) [15] uses perceptual loss and adversarial loss to enhance the realism of the restored image. Perceptual loss is the feature extracted by the convolutional neural network. By comparing the features of the generated image and the target image using the convolutional neural network, the generated image and target image are semantically and styled. The above method is more similar, the adversarial loss is provided by a generative adversarial network (GAN) [16], and the discriminant network is trained according to whether the image can be fooled.

The purpose of the method proposed in this paper is to better detect and locate the students' head for assessing the students' concentration level in the classroom. As most datasets have low-resolution images and many small targets, a small-target detection method based on image SR is proposed, which uses the improved small target features to complete the small-target detection task. On the basis of this, the present study introduces the FFT module [17] to complete images SR and uses Darknet53 [18] combined with Dense block to extract small target features. Using the neck of YOLOv4 [19] for reference, spatial pyramid pooling in deep convolutional networks (SPPnet) [20] and path aggregation network (PANet) [21] are used to complete multi-scale feature fusion. Furthermore, we add a foreground-background balance loss function to the YOLOv3 head to solve the problem of unbalanced image in the foreground and background of the detector and increase the weight of the image in the foreground to improve the effectiveness of the detector.

We trained and tested the model using our self-built dataset. The results show that the detector performs better than the previous one- and two-stage detectors in detecting small targets, and the detection speed is also close to that of YOLOv4. The contributions of the present study are as follows.

1. We provide a more small-target feature information that, is then used in the feature texture transfer (FTT) module to improve the resolution of small target features and remove noise in the image.
2. We design an efficient backbone network to extract small target features. This structure improves the feature extraction capability while reducing the number of parameters of the network structure and avoiding unnecessary calculations.
3. Considering a series of imbalance problems of the detector from the foreground and background of the picture, the prediction result is obtained in the final part of the head, and the foreground-background balance loss function is added to solve the foreground-background imbalance problem.
4. Compared with the previous deep learning target detection models, the proposed method shows better accuracy and speed of detecting small targets.

The remainder of this paper is organized as follows. Section 2 introduces the algorithm of the proposed model. Section 3 presents the experimental setup, training details, and analysis of the results. In Section 4, we provide our conclusions.

II. METHOD

The process of the proposed small-object detection algorithm is divided into four parts: input, backbone network, neck network, and head. The input part performs SR processing on the image, the backbone network is used to extract the features of small target objects in the image, the neck is used to fuse multi-scale features, and the head uses multi-scale feature maps to detect small targets and determine their location. The structure of the algorithm is illustrated in Fig. 1.

We added the FTT module to the input to capture the regional details of the small targets. The main structure contains two extractors: a content extractor and a texture extractor. The content extractor was used for image enhancement, and the texture extractor was used for image detection. In the backbone network, we use a connection method similar to each layer of DenseNet [22] to connect the blocks in Darknet53. This dense connection mode facilitates the training of deeper network structures and the concatenation of feature maps learned at different levels. It requires fewer parameters than other networks and can prevent overfitting. In the neck, the original spatial pyramid pooling and PANet structure were maintained. As the feature fusion module of this part, PANet combines the features of different scales. The spatial pyramid module is a structure attached to the neck to increase the receptive field of the network. In the head, the YOLOv3 [18] head was selected, and the loss function was added to the foreground-background balance loss from bounding box regression, confidence loss, and classification loss, thereby increasing the accuracy of small-object detection.

A. USEING FTT MODULE FOR IMAGE SR AT THE INPUT

At the input, the image is usually transformed to a given size. In addition to such processing, we propose adding an FTT module, to achieve the SR of features and to extract regional textures from reference features. The FTT combines strong semantic features with upper low-resolution reference features and important local details in lower high-resolution reference features at the output.

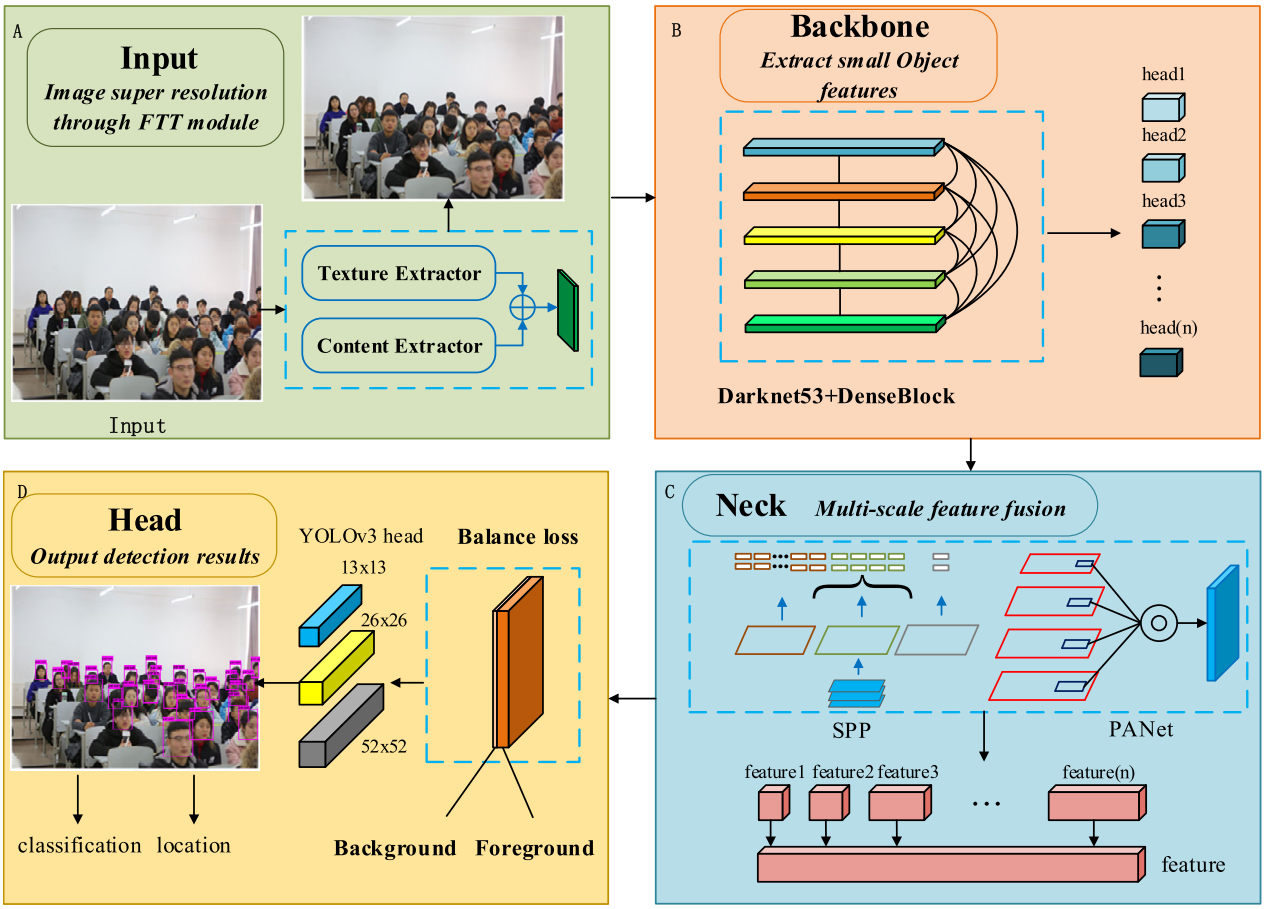


FIGURE 1. Flow of proposed algorithm.

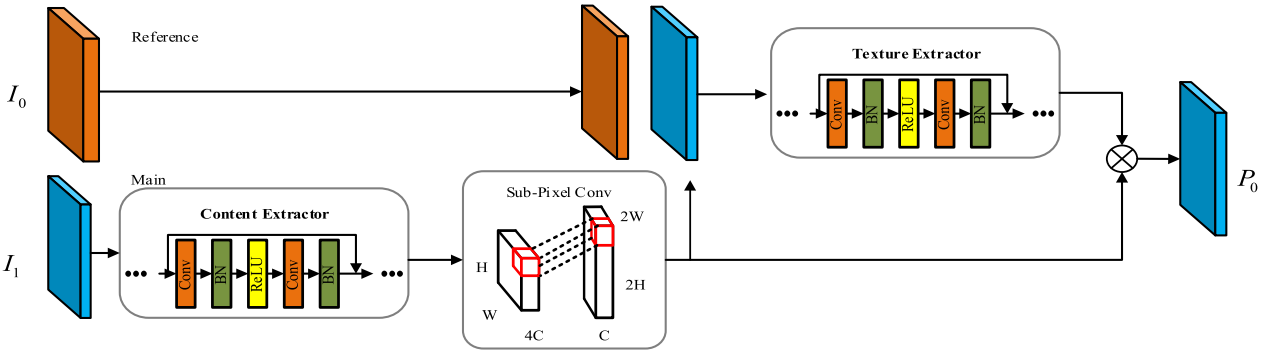


FIGURE 2. The structure of FTT module. mainly composed of two extr actors: content extractor and texture extractor.

The FTT module input is divided into two parts, as shown in Fig. 2: content and regional texture. First, it is extracted by the content extractor, and then the resolution of the content feature is doubled using sub-pixel convolution. The texture extractor selects credible regional textures from the main and reference and features and splices the two parts to the output terminal while removing the noise in the reference feature.

P_0 represents the output of the FTT module and is defined as

$$P_0 = R_t(I_0 \otimes R_c(I_1) \uparrow_{2\times}) + R_c(I_1) \uparrow_{2\times} \tag{1}$$

I_0 is the regional texture input, I_1 is the content input, $R_t(\cdot)$ is the texture extraction component, $R_c(\cdot)$ is the content extraction component, $\uparrow_{2\times}$ represents secondary upscaling through sub-pixel convolution, and $\otimes$ represents feature stitching. Both the content extractor and texture extractor are composed of residual blocks.

In the main method, we use sub-pixel convolution to perform advanced spatial resolution processing on the content features of the main input I_0. Sub-pixel convolution is used to enhance the pixels in the width and height dimensions by transferring the pixels in the channel dimension. The fea-

ture generated by the convolutional layer is expressed as: $F \in \mathbb{R}^{H\times W\times C\cdot r^2}$. The pixel-shuffling operation in the sub-pixel convolution rearranges the features into $rH \times rW \times C$. This operation is mathematically defined as follows.

$$\mathrm{PS}(F)_{x,y,z} = F_{\lfloor x/\lambda\rfloor,\lfloor y/\lambda\rfloor,C\cdot\lambda\cdot \bmod (y,\lambda)+C\cdot \bmod (x,\lambda)+z} \quad (2)$$

Here, $\mathrm{PS}(F)_{x,y,z}$ represents the output feature pixel on coordinate (x, y, z) after $\mathrm{PS}(\cdot)$, which is the pixel-shuffling operation, and r is the up-scaling factor. In the FTT module, we use $\lambda = 2$ to double the spatial scale.

In the texture area, the area texture input I_0 and content input I_1 are sent to the texture extractor. The purpose of the texture extractor is to obtain credible textures for small-target detection. Adding to the texture and content, element by element, ensures that the output integrates semantic and regional information from the input and references. Therefore, P_0 has a reliable texture selected from shallow features I_0 and similar semantics from a deeper level I_1.

B. BACKBONE: COMBINATION OF DARKNET53 AND DENSE BLOCK

The backbone network is mainly used to extract the features of small targets in a picture. Based on the YOLOv4 network, we discarded the cross-stage partial (CSP) [23] part of the CSPDarknet53. In the original Darknet53 structure, residual blocks were used to connect the convolutional layer. After we opted to use dense blocks to connect, the network was narrower, its parameters were fewer, and the overfitting phenomenon was also reduced. It improves the speed of feature extraction and the ability of the network to extract deep features.

1) DARKNET53

Darknet53 contains 53 convolutional layers. Drawing on the idea of residual connections in the residual network, some layers are connected by shortcut links. It abandons the traditional pooling and fully connected layers, uses the increased step size of the convolution kernel to reduce the feature map, and uses full convolution to achieve the up- sampling of the feature map. The structure is mainly composed of a series of 1×1 and 3×3 convolutional layers. Each convolutional layer is followed by a batch normalization and a LEAKYReLU layer. The LEAKYReLU activation function is as follows:

$$y_i = \begin{cases} x_i & \text{if } x_i \geq 0 \\ \dfrac{x_i}{a_i} & \text{if } x_i < 0 \end{cases} \qquad a_i \in (1, +\infty) \quad (3)$$

The Darknet53 structure is illustrated in Fig. 3. The middle res module follows the order: convolutional layer, batch normalization layer, LEAKYReLU layer, convolutional layer, batch normalization [24] layer, LEAKYReLU layer, and the final module output layer. In this section, we use the Mish activation function to replace the LEAKYReLU activation function. To avoid the problem of gradient saturation, the effect of gradient descent improves. The Mish function

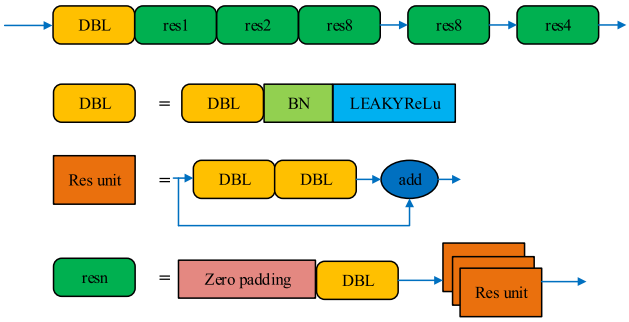


FIGURE 3. The simple structure of Darknet53.

formula is expressed as (4) and (5).

$$\tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (4)$$

$$Mish = x^* \tanh(\ln(1 + e^x)) \quad (5)$$

The middle layer uses the shortcut connection method in ResNet [22], and the res8, res8, and res4 layers output $52\times52\times256$, $26\times26\times512$, and $13\times13\times1024$ feature maps, respectively.

In Fig.3, the DBL layer includes a convolutional layer, batch normalization layer, and LEAKYReLu layer; the Res unit represents that each block is connected by the residual block; the res includes a convolution layer, batch normalization layer, LEAKYReLu layer, convolution layer, batch normalization layer, LEAKYReLu layer, and block output layer.

2) DENSE BLOCK

If some layers that can learn the identity mapping are added to a certain network to form a new network, then the worst result is that these layers in the new network become identity-mapping layers after training without affecting the performance of the original network. A similar assumption was made when DenseNet was proposed: Instead of learning redundant features multiple times, feature reuse is a better extraction method. In the CNN [24], as the depth increases, the problem of gradient disappearance becomes more obvious. DenseNet connects all layers directly on the premise of ensuring the maximum information transmission between the network layers. In previous research, the shortcut connection method proposed by ResNet [25] played a very positive role in solving the problem of gradient dispersion, and also reduced the calculation and parameter burden of the deep network. As expressed in (6), the number of connections between layers in ResNet's connection mode is much less than that of DenseNet, where l represents the layer, X_l represents the output of the layer, and H_l represents a nonlinear transformation. The output of layer l is the output of layer $l-1$ and the nonlinear transformation of layer $l-1$.

$$X_l = H_l(X_{l-1}) + X_{l-1} \quad (6)$$

In a traditional convolutional neural network, if there are L layers, then there will be L connections, and in DenseNet,

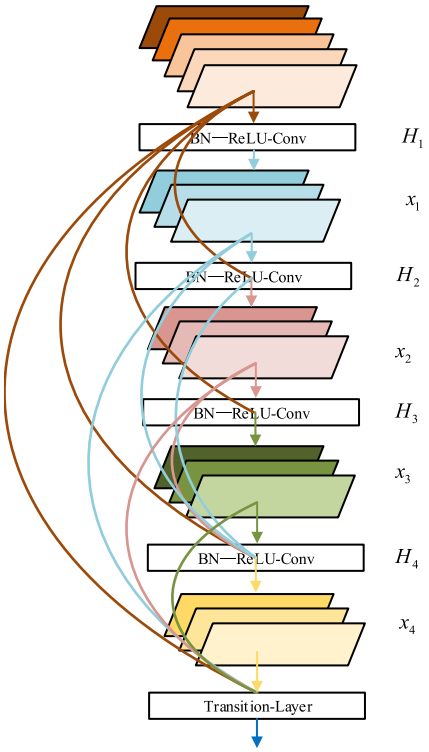


FIGURE 4. The connection method of dense block.

there will be L(L + 1)/2 connections. In general, the input of each layer is derived from the outputs of all previous layers.

$$X_l = H_l([X_0, X_1, \ldots, X_{l-1}]) \tag{7}$$

$[X_0, X_1, \ldots, X_{l-1}]$ means cascading the output feature maps of layers 0 to $l - 1$, similar to Inception [26].H_l includes batch normalization, ReLU activation function, and 3×3 convolution. The dense block structure is shown in Fig. 4.

The design of the dense block reduces the number of output feature maps of each convolutional layer (to less than 100), instead of hundreds or thousands of widths, as in other networks. This connection method also enhances the transfer of features and gradients, and the network is easier to train. Because the problem of gradient disappearance usually occurs when the input information and gradient information are transferred between many layers, this method of dense connection is equivalent to direct input and loss for each layer, which can reduce the phenomenon of gradient disappearance. This connection method can also produce a regularization effect and suppresses over-fitting.

C. NECK: SPATIAL PYRAMID POOLING AND PANET FOR MULTI-SCALE FEATURE FUSION

In the neck, we continue to use PANet and the spatial pyramid pooling layer structure to fuse the feature information of feature maps of different sizes for the fused small target features to be detected more easily. The purpose of the SPP network in the proposed network is to increase the receptive field of the network, whereas PANet uses the precise positioning signal at the bottom layer to shorten the information path, enhance the feature pyramid, and create a bottom-up path enhancement algorithm that spreads through the bottom layer to enhance the entire feature hierarchy.

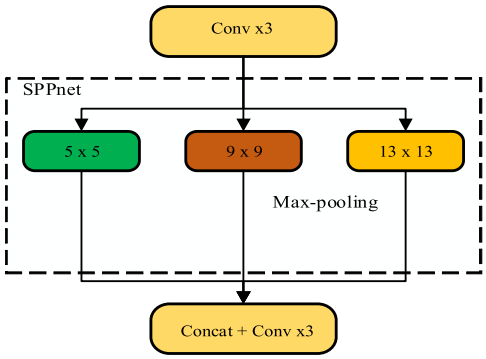


FIGURE 5. The structure of Spatial pyramid pooling layer.

1) SPATIAL PYRAMID POOLING

There is usually a problem when training a CNN: the general CNN has a fixed size requirement for the input image, which places certain restrictions on the aspect ratio and the ratio of the input image. When inputting an image of any size, the current method is mainly used to fit the input image to a fixed size by cutting or warping. However, the cropped area may not contain the entire object, and warping may cause unwanted geometric distortions. The final detection accuracy may be affected due to content loss or distortion. Using the spatial pyramid, the input image can be of any size. This allows arbitrary aspect ratios and arbitrary scaling. When the input image is of different scales, the network (the same filter size) extracts feature of different scales. We use a spatial pyramid layer to eliminate the fixed-size constraint of the network. Specifically, we added an SPP layer to the final convolutional layer. The SPP layer pools features and generates a fixed-length output, which is then input to the fully connected layer. Otherwise, we perform some information "aggregation" in the deeper stages of the network hierarchy (between the convolutional layer and the fully connected layer) to avoid the need for cropping or distortion in the beginning.

We perform a maximum pooling of 5×5, 9×9, and 13×13 on the 107th layer of the network, and obtain the 108th, 110th, and 112th layers, respectively. After pooling is completed, the 107th, 110th, and 112th layers are cascaded, which is connected to a 114th layer feature map and is reduced to 512 channels through 1×1 convolution. The structure of the SPP module is shown in Fig. 5.

We then feed the features extracted using the backbone network into the SPP layer through 3×3 convolution, which is then used to obtain the output and complete a multi-scale feature concatenation.

2) PATH AGGREGATION NETWORK

Because the path from the bottom structure to the top feature is very long, which increases the difficulty of obtaining accurate positional information, PANet uses a bottom-up path

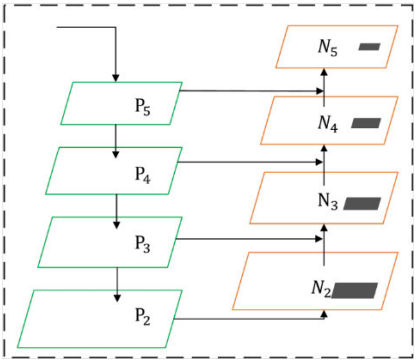


FIGURE 6. Bottom-up path augmentation.

enhancement method to enhance the entire feature level with a lower-level accurate positional signal, shortening the lower level and the information path of the top-level feature. In addition, the use of adaptive feature pooling allows each proposal to access information from various levels for prediction. This new structure produces satisfactory performance.

The framework completes the bottom-up path expansion using the FPN to generate the same spatial feature map layer at the same network stage. Each feature level corresponds to a stage. With ResNet as the basic structure, P_2, P_3, P_4, P_5 is used to represent the feature level generated by the FPN. The expanded path gradually approaches P_5 from the lowest P_2. From P_2 to P_5, the space size is gradually down-sampled by a factor of 2. We use N_2, N_3, N_4, and N_5 to represent the newly generated feature map, which corresponds to P_2, P_3, P_4, and P_5.The path is shown in Fig. 6.

In the FPN, each proposal is assigned to different levels of feature maps according to the size of the proposal. For example, a large-sized proposal is allocated to a high-level map, and a small one is allocated to a low-level map, but this cannot maximize the use of high-level semantic information and low-level location information. This above problem can be solved using adaptive feature pooling. Feature fusion is performed after ROIAlign [4] pooling in multiple layers, and the fused features are input to the detection task.

The specific adaptive feature pooling (AFP) calculation process of the bounding box branch is as follows: ROIAlign pooling first obtains four feature maps of equal size, and then uses the same fully connected layer (fc1) to calculate the four feature maps separately. The four groups of features are fused, and then a fully connected layer (fc2) is used to calculate the classification and bounding box regression results.

3) COMBINATION OF SPP AND PANET

In the combination of SPP and PANet, processes one to three are all up-sampled to obtain feature maps, which are stacked on the output layer of the backbone network, and the output of the neck is obtained through a series of Darknet-Con v2D_BN_LEAKYReLU modules. This module includes a two-dimensional convolution layer, batch normalization layer, and LEAKYReLU activation function layer. Processes four and five are based on the bottom-down feature fusion of the FPN, and one more bottom-up feature fusion is added. In this step, variable y76 is first down-sampled to 38 × 38 in size and then stacked with variable y38.

D. YOLO HEAD WITH FOREGROUND-BACKGROUND BALANCE LOSS

In this section, we use YOLOv3's head as the output of the detection end and use a multi-scale fusion method (similar to the FPN) for prediction. To enhance the accuracy of small-target detection, predictions were made on the feature maps at three scales. We also modified the loss function by adding a foreground-background balance function to the original YOLOv4 loss function. The purpose is to increase the weight of the foreground of the image and eliminate the interference of the image's background in the detection result.

1) FOREGROUND-BACKGROUND BALANCE LOSS

The problem of foreground-background imbalance [27] is widespread in target detectors, and data have shown that imbalance problems hinder the detection accuracy of the detector. We seek a solution from the one-stage target detector because the target only occupies a small part of the entire picture, and the loss function of the original network will cause the network to learn the characteristics of the small target image insufficiently. In actual operations, both the key points and the central area of the object only occupy a small part of the image, and most of the image forms the background.

We use the foreground-background balance function to improve the quality of the foreground and background features. The loss function is divided into two parts: global SR loss and foreground enhancement loss. Because background pixels constitute the major part of the image, the global loss is to enhance mainly enhances the similarity with the real background features. Here, we use the common loss in image SR as the global SR loss L_{gsr}:

$$L_{gsr}(G, G^f) = \left\| G^f - G \right\|_1 \tag{8}$$

where G is the generated feature map and G^f represents the object feature map.

The foreground enhancement loss emphasizes the positive pixels because a severe imbalance of positive and negative pixels affects the performance of the detector. We use the loss of the foreground area as the foreground enhancement loss L_{pse} :

$$L_{pse}(G, G^f) = \frac{1}{M} \sum_{(a,b)\in P_{gt}} \left\| G^f_{a,b} - G_{a,b} \right\|_1 \tag{9}$$

P_{gt} is a patch of ground truth, M is the total number of positive pixels, and (a, b) are the coordinates of the pixels on the feature map. The foreground enhancement loss imposes stronger constraints on the area where the object is located and forces the true expression of these areas to be learned. The foreground-background balance loss function is defined

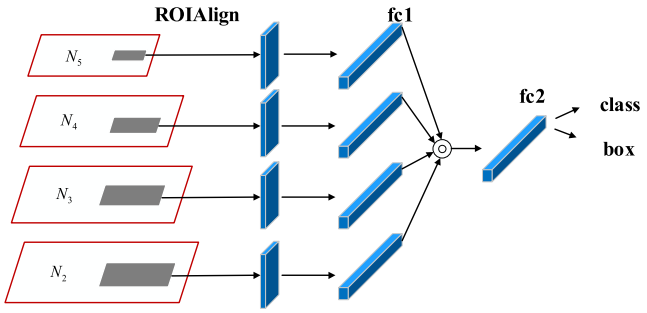


FIGURE 7. The process of Adaptive feature pooling on box branch.

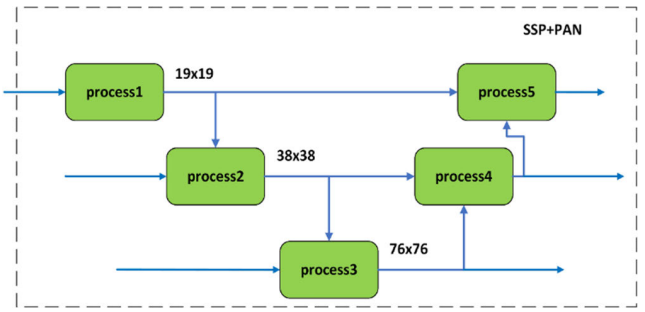


FIGURE 8. The combination of Spatial pyramid pooling layer and Path aggregation network. Process 4 and 5 mainly completes multi-scale feature fusion.

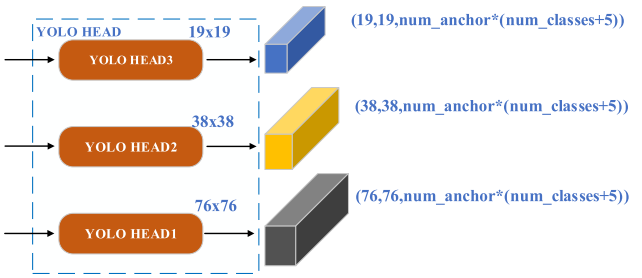


FIGURE 9. The output of YOLO Head.

as follows:

$$L_{fbb}(G, G^f) = L_{gsr}(G, G^f) + \mu L_{pse}(G, G^f) \tag{10}$$

where μ is the weighting factor. The balanced loss function mines the "true value" by improving the feature quality of the foreground area and eliminating false feedback by improving the feature quality of the background area.

2) YOLO HEAD

To improve the accuracy of small-target detection, a multi-scale fusion method is used to make predictions. The size of the feature map of the same layer is 13×13, and the function of this layer is to combine the 26×26 feature map of the previous layer when this layer is connected. Finally, three scales of feature maps were generated with sizes of 13×13, 26×26, and 52×52, the smallest scale being used to detect large targets, and the largest scale to detect small targets.

As shown in Fig. 9, three convolutional layers were used: YOLO HEAD1, YOLO HEAD2, and YOLO HEAD3. YOLO HEAD1 finally uses 1×1 convolution to output the largest feature map with dimensions of $76 \times 76 \times 18$; YOLO HEAD2 and YOLO HEAD3 also perform a series of convolution operations, the dimensions of which are $38 \times 38 \times 18$ and $19 \times 19 \times 18$, respectively. The value of the anchor point was set to 3 by default, and the number of categories was set to 1.

III. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, we describe the details of the experiments. The entire experimental process was divided into three parts. First, we provide the experimental settings, evaluation criteria, and working platform of the experiment, which includes the dataset we collected. Then, we introduce the details of the experiment: the image SR evaluation result, the parameter setting of the backbone network, and the design of the loss function. Finally, we compare the performance of the detector with other methods, display the small target results, and make a final evaluation of the model.

A. EXPERIMENTAL CRITERION

The model detection performance was evaluated mainly using the mean average precision (mAP). Other indicators, such as accuracy, f1 score, and frames per second (FPS), will also help us to further evaluate the model performance. The accuracy, f1 score, sensitivity, and mAP were as follows:

$$Accuracy(\%) = \frac{TP + TN}{TP + TN + FP + FN} \tag{11}$$

$$Precision(\%) = \frac{TP}{TP + FP} \tag{12}$$

$$Senitivity(\%) = \frac{TP}{TP + FN} \tag{13}$$

$$F1score(\%) = \frac{2}{\frac{1}{Sensitivity} + \frac{1}{Precision}} \tag{14}$$

$$AP(\%) = \frac{1}{|classes|} \sum_{c} \left(\frac{1}{|threholds|} \sum_{t} \frac{TP}{TP + FP} \right) \tag{15}$$

$$mAP = \frac{AP}{M} \tag{16}$$

Here, TP (true positive) is the prediction error (the algorithm predicts a non-existent object), FN (false negative) means no prediction (the algorithm does not predict the object within the specified range), TP (true positive) means that the prediction is correct (the algorithm predicted within the specified range of the object), and TN (true negative) means that no object is predicted. M represents the number of object categories. The F1 score was derived using (12) and (13).

We used the peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) to evaluate the similarity between the two pictures. These two indices are also used for

TABLE 1. Comparison of the similarity between the original image and the processed image.

Number of image frames	The original image		Image processed by FTT module	
	PSNR	SSIM	PSNR	SSIM
50th	29.54	0.841	27.32	0.823
100th	28.41	0.851	26.21	0.815
200th	29.12	0.824	25.32	0.791
400th	30.12	0.812	27.14	0.784
500th	26.54	0.832	24.52	0.802

image SR, and are expressed as follows:

$$PSNR(P,\hat{P}) = 10\log_{10}\frac{\max_{\hat{P}}^2}{\frac{1}{N}\sum_{i=0}^{N}(P_i-\hat{P}_i)^2} \quad (17)$$

$$SSIM(P,\hat{P}) = \frac{(2\mu_P\mu_{\hat{P}}+c_1)(2\sigma_{P\hat{P}}+c_2)}{(\mu_P^2+\mu_{\hat{P}}^2+c_1)(\sigma_P^2+\sigma_{\hat{P}}^2+c_2)} \quad (18)$$

Here, P represents the original image; $\hat{P}$ represents the image processed by the image SR, μ_P and $\mu_{\hat{P}}$ represent the mean value of P and $\hat{P}$, respectively; σ_P^2, and $\sigma_{\hat{P}}^2$ represent the variances of P and $\hat{P}$, respectively; $\sigma_{P\hat{P}}$ represents the covariance of $P\hat{P}$; and $c_1 = (k_1L)^2$, $c_2 = (k_2L)^2$ are two constants, where k_1 is usually set to 0.01, k_2 is 0.03, and L is the range of image pixels. The larger the value of the SSIM, the higher is the similarity of the image.

In addition to detection accuracy, speed is an important evaluation index for object detection algorithms. FPS is used to evaluate object detection, that is, the number of pictures that can be processed per second.

B. EXPERIMENTAL SETTING

1) SMALL OBJECT DATASETS

The images of the small targets for our training and evaluation were taken from a university classroom video record. Five videos approximately 3-4 min long were collected. They included scenes from different classrooms. A total of 2,200 images were acquired, and we performed the labeling manually, selecting to use the students' heads as the small targets for detection. The experiment applied a cross-validation method, using 550 images for training, 1100 images for testing, and 550 images for verification.

2) EXPERIMENTAL PLATFORM

In this study, all experiments were conducted on a platform with the Ubuntu18.04 operating system, an NVIDIA GeForce GTX 1660Ti with 8 GB graphics memory, and Intel Core i7-9750H with 8 GB memory. The software platform was Python 3.7.0, based on the TensorFlow 1.15.0.

C. EXPERIMENTAL DETAILS

We used a self-built dataset for training and fixed the image size at the input to 416×416 pixels. The Mish activation function is used in the backbone network, and the LAEKYReLU activation function is used in other convolutional layers, the DropBlock regularization [31] method is randomly used in the convolutional layer to optimize the generalization ability of the model, and the DIOU-NMS [32] method is used to improve the boundary when processing the bounding box frame suppression accuracy; this experiment sets the maximum number of training batches to 60,000, the initial learning rate is set to 1e-4, the optimal momentum coefficient is set to 0.9, the weight decay regular term is 0.0005, and the beta nms is 0.4. Considering the capacity of the GPU, the batch size was set to 64. After 20,000 iterations, the single-scale training method was transformed into multi-scale until the end of training. In the iterative training process of 0 to 20,000 times, the best model is saved every 1000 times. After 20,000 iterations, the model is saved every 5000 times. After the training was completed, the last saved model was selected for testing. The detection performance of the model was estimated based on two factors: accuracy and speed. The mAP was used to evaluate accuracy and the FPS was used to estimate speed.

1) IMAGE SR

We chose to use an image SR module that is more suitable for small-object detection to obtain deeper feature information for detection. By comparing the original input image and the processed image (the results are shown in Table 1), it can be seen that the module minimizes the difference from the original image and significantly improves the image quality. While the signal-to-noise ratio (PSNR) continues to decrease, the SSIM of the image is significantly improved. Compared with the direct input of the original image, after the SR processing, the image has less target feature information to be extracted, which provides a good foundation for feature extraction by the backbone network. Concurrently, richer features help the final detector distinguish between positive

TABLE 2. Comparison of model calculation and parameters.

Model	Backbone	BFLOPS	Parameters
YOLOv3[18]	Darknet53	141.5	40.9M
YOLOv4[19]	CSPDarknet53	128.4	34.2M
SSD[28]	ResNet-101[25]	152.6	43.1M
Fatser R-CNN[29]	VGGNet[30]	332.6	134.7M
Our Method(k=16)	**Darknet53+Denseblock**	**110.7**	**28.7M**
Our Method(k=32)	Darknet53+Denseblock	118.6	30.6M

TABLE 3. Weight factor setting for small object detection.

Weighting factor(μ)	Acc(%)	Rec(%)	F1 score
Original loss	80.4	86.9	0.834
0.5	81.9	87.0	0.839
1.0	**82.7**	**87.4**	**0.844**
1.5	83.8	86.2	0.856

TABLE 4. The influence of different components of the detector on the detection of different targets.

Backbone	Balanced Loss	FTT	Small ACC.(%)	Medium ACC.(%)	Large ACC.(%)
			83.2	89.5	90.7
✓			83.5	89.8	90.9
	✓		83.4	90.1	91.4
		✓	83.8	91.2	92.0
✓	✓	✓	**85.4**	**91.8**	**92.3**

TABLE 5. The performance of different detectors on different targets.

Model	$F1_S$	$F1_M$	$F1_L$
FPN[4]	0.845	0.871	0.892
YOLOv4[19]	0.850	0.864	0.883
SNIP[10]	0.837	0.858	0.881
Faster R-CNN[29]	0.839	0.852	0.895
SSD[28]	0.806	0.841	0.887
Our method	**0.862**	**0.869**	**0.893**

and negative examples, thereby providing better positioning and classification.

2) DARKNET53 WITH DENSE BLOCK

We chose two different growth rates ($k = 16$ and 32) for application in the experiment. The growth rate is defined as if each function F produces k feature maps, which obtains $k_0 + k \times (l-1)$ input feature maps,k_0 is the number of channels in the input layer. The growth rate regulates the amount of new information that contributes to the global state by each layer of the network. Once the global state is determined, it can be accessed anywhere in the network, unlike in the traditional network architecture, which copies layer by layer. As shown in Table 2, among the current mainstream methods, YOLOv4 has the fewest network structure parameters and requires fewer calculations. Compared with the method we currently propose, the network structure parameters are reduced by 3.6 (million) M and 5.5 M compared with YOLOv4. When $k = 16$, the total number of network parameters and calculations were optimized, which increased the accuracy and speed of the backbone network in the feature extraction stage.

3) LOSS FUNCTION

We add the foreground and background balance loss to the loss function part of the network and set the foreground and background weights of the balance training loss to 0.5, 1, and 1.5. In the experiment, because the number of small targets in each picture is uncertain, it is impossible to estimate the picture background's degree of influence on the detector. Here, we adjust the background of the picture to different color depths to improve the robustness of the detector. The balance loss between the foreground and background affects the performance of the final detector. Table 3 shows that the balance loss increases the accuracy of small-target detection by 3.4%, thus increasing the F1 score by 0.5%. This demonstrates that the loss of foreground and background balance promotes meaningful changes in the positive region of the extended feature map. We further studied the different configurations of the balanced hyperparameter μ. When μ was set to 0.5, 1.0, and 1.5, the small target F1 score was 0.839, 0.844, and 0.856, respectively. Therefore, we used $\mu = 1.0$ to achieve a better balance between accuracy and recall.

D. EXPERIMENTAL RESULT AND ANALYSIS

1) DETECTOR PERFORMANCE ANALYSIS

To evaluate the performance of the detector, the accuracy and speed tests we conducted are shown in Fig. 11. The detection speed of our proposed algorithm is equal to that of YOLOv4. Moreover, and the accuracy test surpasses some of the previous mainstream one- and two-stage object detection algorithms, and the mAP is close to 90%. For small-target detection tasks that most of the current detectors cannot complete, our proposed algorithm guarantees the speed advantage of the one-stage method while continuously improving the accuracy of the detector.

2) SMALL-OBJECT DETECTION RESULT DISPLAY

To test the model's ability to generalize, we used three images acquired from five videos as a dataset and selected images from the remaining two videos to test the trained model. The results in Fig. 12 show that our model produces good results

TABLE 6. Evaluation of the results of detecting small objects with different models.

Model	Accuracy(%)	F1 score	mAP(%)	FPS(s)
YOLOv3[18]	82.1	0.867	79.26	5.4
YOLOv4[19]	84.5	0.843	87.54	7.9
SSD[28]	80.1	0.821	78.19	5.6
Faster R-CNN[29]	81.2	0.835	82.51	4.9
SPP-net[20]	83.1	0.863	84.15	6.1
Our method	**83.4**	**0.884**	**89.91**	**8.0**

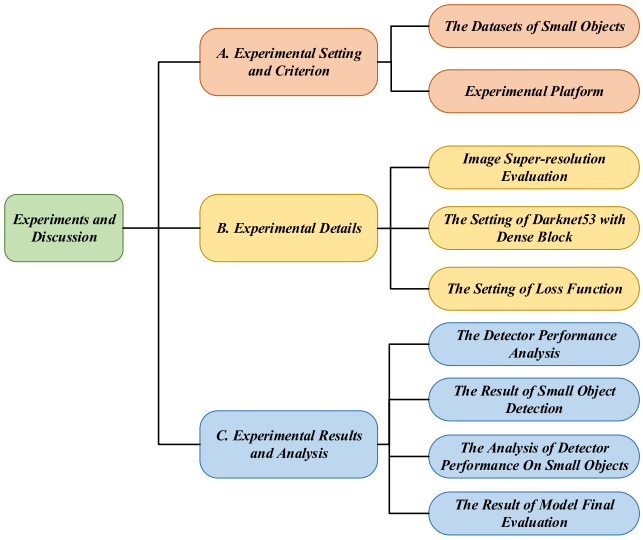


FIGURE 10. Overview of experimental results and discussion. (A) Experimental setting and criterion(B)Experimental details (C) Experimental result and analysis.

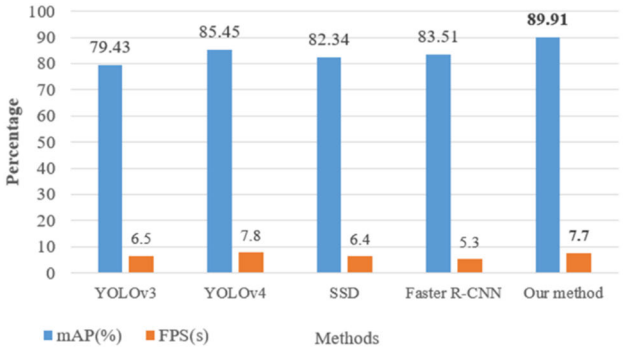


FIGURE 11. Comparison of models' accuracy and speed.

in detecting small targets in images. It can be said that the problem of small targets being easily occluded and difficult to detect is solved, which confirms that even if the feature information of small targets in the image is minimal and the resolution is low, the classification and positioning task can be completed.

FIGURE 12. Small object detection results from different videos.

3) THE PERFORMANCE OF THE DETECTOR IN A SMALL TARGET

Considering that the detector detects targets of different sizes, we designed a detection accuracy experiment for different types of targets. According to the COCO dataset, pixels smaller than 32×32 are defined as small targets, between 32 and 96 pixels as medium targets, and those larger than 96×96 pixels are defined as large targets. This part of the experiment is shown in Table 4. After using our proposed backbone network structure, FTT module, and balance loss,

the performance of the detector in small, medium, and large objects has been improved to a certain extent. In contrast, if one of the components is used alone, the effect will be improved when detecting a certain type of object, but the overall effect is still not as good as the effect obtained using all the components.

We made similar comparisons on other detectors and included the best-performing model in our method, where $F1_s$ represents the f1 score of the small target, $F1_M$ represents the f1 score of the medium target, and $F1_L$ represents the f1 score of the large target. As shown in Table 5, our model has the highest f1 score among the three objects of different scales. The accuracy and recall of the model were in good agreement. The results show that our model can not only improve the classification accuracy of small targets, but also maintain other classification accuracy of the target type.

4) FINAL RESULT EVALUATION

In Table 6, we compare and evaluate the current mainstream target detection algorithms. Compared with the latest YOLOv4, our proposed algorithm shows a 2.37% increase in mAP and a 0.1 s faster FPS in the small-target detection task. The accuracy rate was 1.1% lower. Prejudice against the previous one-stage algorithm, the notion that it can only improve the detection speed but not the accuracy of the detector, has changed since the emergence of YOLOv4.

Our method, based on YOLOv4, is a breakthrough in the field of small-target detection as it significantly improves the accuracy of our algorithm in small-target detection tasks. Cases where the previous small-target detection approached failed to produce ideal or favorable results, our algorithm identified the highest number of small objects in the image.

IV. CONCLUSION AND FUTURE WORK

In this study, we designed an algorithm specifically to detect small targets for use in university classrooms. The pictures captured from the video were introduced at the input end of the network, and image SR processing was completed using the FTT module. During this process, the noise in the input image was also eliminated. For the feature extraction portion of the backbone network, we discarded the CSP portion in CSPDarknet53 and changed the connection mode between each block from the residual block to the dense block, reducing the network parameters and calculations, and improving the accuracy of feature extraction. The neck still uses the structure of the SPP block plus PANet to complete the multi-scale feature fusion task. Finally, in the prediction part of the head, we add the foreground and background balance functions based on the three-part loss functions of YOLOv4 to enhance the weight of the image foreground and weaken the image background's influence on the detector.

Before our proposed method, some researchers proposed the FPN method, which uses multi-scale feature fusion to make predictions on different feature maps by fusing high-level semantic information and low-level location information. Some scholars choose to cascade R-CNN [33] and train high-quality detectors while ensuring the quality and quantity of samples by continuously increasing the threshold of IOU. It is also believed that improving the accuracy of small-target detection by enhancing the resolution of the image will increase the number of calculations of the network and that the use of multi-scale feature representation will produce unknowable results. Others proposed PGAN [34] to improve the detection rate by increasing the feature representation of small objects and designed a perceptual loss function.

Finally, the results of our experiments show that the proposed algorithm is effective for when detecting targets down to 32×32 pixels in size. However, this method requires improvement for small targets with a very low resolution (such as 10×10 pixels), that is, when the resolution is too low and the target features are blurred. In the future, we will continue to explore small-target detection methods, and we intend to explore head pose estimation in our follow-up work.

AUTHOR CONTRIBUTIONS

(Zhuang-Zhuang Wang and Kai Xie contributed equally to this work.) Zhuang-Zhuang Wang conceived the algorithms, and designed the experiments; Kai Xie reviewed the paper; Xin-Yu Zhang conducted the comparative experiment; Hua-Quan Chen is responsible for software design; Chang Wen is responsible for data collection; Jian-Biao He checked the spelling and made suggestions.

REFERENCES

[1] Á. Arcos-García, J. A. Álvarez-García, and L. M. Soria-Morillo, "Evaluation of deep neural networks for traffic sign detection systems," *Neurocomputing*, vol. 316, pp. 332–344, Nov. 2018.

[2] K.-Y. Lee and J.-Y. Sim, "Cloud removal of satellite images using convolutional neural network with reliable cloudy image synthesis model," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep.2019, pp. 3581–3585.

[3] N. Ruiz, E. Chong, and J. M. Rehg, "Fine-grained head pose estimation without keypoints," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 2074–2083.

[4] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2117–2125.

[5] M. Kisantal, Z. Wojna, J. Murawski, J. Naruniec, and K. Cho, "Augmentation for small object detection," 2019, *arXiv:1902.07296*. [Online]. Available: http://arxiv.org/abs/1902.07296

[6] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang, "Acquisition of localization confidence for accurate object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 784–799.

[7] M. Najibi, P. Samangouei, R. Chellappa, and L. S. Davis, "SSH: Single stage headless face detector," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4875–4884.

[8] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

[9] P. O. Pinheiro, R. Collobert, and P. Dollár, "Learning to segment object candidates," 2015, *arXiv:1506.06204*. [Online]. Available: http://arxiv.org/abs/1506.06204

[10] B. Singh and L. S. Davis, "An analysis of scale invariance in object detection SNIP," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3578–3587.

[11] B. Singh, M. Najibi, and L. S. Davis, "SNIPER: Efficient multi-scale training," 2018, *arXiv:1805.09300*, [Online]. Available: http://arxiv.org/abs/1805.09300

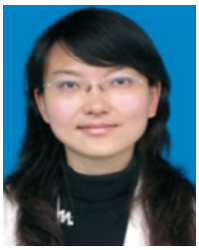
[12] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 184–199.
[13] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 136–144.
[14] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 295–307, Jun. 2015.
[15] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 4681–4690.
[16] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014, *arXiv:1406.2661*. [Online]. Available: http://arxiv.org/abs/1406.2661
[17] C. Deng, M. Wang, L. Liu, and Y. Liu, "Extended feature pyramid network for small object detection," 2020, *arXiv:2003.07021*. [Online]. Available: http://arxiv.org/abs/2003.07021
[18] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: http://arxiv.org/abs/1804.02767
[19] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*. [Online]. Available: http://arxiv.org/abs/2004.10934
[20] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.
[21] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8759–8768.
[22] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.
[23] C.-Y. Wang, H.-Y. Mark Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "CSPNet: A new backbone that can enhance learning capability of CNN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 390–391.
[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1097–1105.
[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[26] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-ResNet and the impact of residual connections on learning," in *Proc. AAAI Conf. Artif. Intell.*, 2017, vol. 31, no. 1.
[27] J. Chen, Q. Wu, D. Liu, and T. Xu, "Foreground-background imbalance problem in deep object detectors: A review," in *Proc. IEEE Conf. Multi-Media Inf. Process. Retr. (MIPR)*, Aug. 2020, pp. 285–290.
[28] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 21–37.
[29] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," 2015, *arXiv:1506.01497*. [Online]. Available: http://arxiv.org/abs/1506.01497
[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556
[31] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "DropBlock: A regularization method for convolutional networks," 2018, *arXiv:1810.12890*. [Online]. Available: http://arxiv.org/abs/1810.12890
[32] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU loss: Faster and better learning for bounding box regression," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 7, pp. 12993–13000.
[33] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6154–6162.
[34] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan, "Perceptual generative adversarial networks for small object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1222–1230.
[35] B. Xue and N. Tong, "DIOD: Fast and efficient weakly semi-supervised deep complex ISAR object detection," *IEEE Trans. Cybern.*, vol. 49, no. 11, pp. 3991–4003, Nov. 2019.
[36] B. Xue and N. Tong, "Real-world ISAR object recognition using deep multimodal relation learning," *IEEE Trans. Cybern.*, vol. 50, no. 10, pp. 4256–4267, Oct. 2020.
[37] Z. Khan, T. Hussain, A. Ullah, S. Rho, M. Lee, and S. Baik, "Towards efficient electricity forecasting in residential and commercial buildings: A novel hybrid CNN with a LSTM-AE based framework," *Sensors*, vol. 20, no. 5, p. 1399, Mar. 2020.
[38] F. U. M. Ullah, A. Ullah, I. U. Haq, S. Rho, and S. W. Baik, "Short-term prediction of residential power energy consumption via CNN and multi-layer bi-directional LSTM networks," *IEEE Access*, vol. 8, pp. 123369–123380, 2020.
[39] Y. Zhao, K. Xie, Z. Zou, and J.-B. He, "Intelligent recognition of fatigue and sleepiness based on InceptionV3-LSTM via multi-feature fusion," *IEEE Access*, vol. 8, pp. 144205–144217, 2020.
[40] Z.-Z. Zou, K. Xie, Y.-F. Zhao, J. Wan, L. Lan, and C. Wen, "Intelligent assessment of percutaneous coronary intervention based on GAN and LSTM models," *IEEE Access*, vol. 8, pp. 90640–90651, 2020.
[41] J. Li, T. Qiu, C. Wen, K. Xie, and F.-Q. Wen, "Robust face recognition using the deep C2D-CNN model based on decision-level fusion," *Sensors*, vol. 18, no. 7, p. 2080, Jun. 2018.
[42] C. Wen, K. Xie, Y. Hu, and J. He, "Fast recovery of weak signal based on three-dimensional curvelet transform and generalised cross validation," *IET Signal Process.*, vol. 12, no. 2, pp. 149–154, Apr. 2018.
[43] Y.-X. Yang, C. Wen, K. Xie, F.-Q. Wen, G.-Q. Sheng, and X.-G. Tang, "Face recognition using the SR-CNN model," *Sensors*, vol. 18, no. 12, p. 4237, Dec. 2018.
[44] T. Qiu, C. Wen, K. Xie, F. Wen, G. Sheng, and X. Tang, "Efficient medical image enhancement based on CNN-FBB model," *IET Image Process.*, vol. 13, no. 10, pp. 1736–1744, Aug. 2019.
[45] F. Chen, C. Wen, K. Xie, F. Wen, G. Sheng, and X. Tang, "Face liveness detection: Fusing colour texture feature and deep feature," *IET Biometrics*, vol. 8, no. 6, pp. 369–377, Nov. 2019.
[46] F. Yi, Y.-F. Zhao, G.-Q. Sheng, K. Xie, C. Wen, X.-G. Tang, and X. Qi, "Dual model medical invoices recognition," *Sensors*, vol. 19, no. 20, p. 4370, Oct. 2019.
[47] K. Xie, Z. Bai, and W. Yu, "Fast seismic data compression based on high-efficiency SPIHT," *Electron. Lett.*, vol. 50, no. 5, pp. 365–367, Feb. 2014.

ZHUANG-ZHUANG WANG was born in Hubei, China, in 1995. He is currently pursuing the master's degree with Yangtze University, Jingzhou, China. In 2019, he joined the National Demonstration Center for Experimental Electrical and Electronic Education with the intent to research deep learning and image processing. He has been conducting research projects on object detection. His research interests include image processing, artificial intelligence, and machine learning.

KAI XIE received the M.S. degree in electronic engineering from the National University of Defense Technology, Changsha, China, in 2003, and the Ph.D. degree in pattern recognition and intelligent system from Shanghai Jiao Tong University, Shanghai, China, in 2006. He is currently a Professor with the School of Electronic Information, Yangtze University, Jingzhou, China. He also works in the field of image processing and signal processing.

XIN-YU ZHANG was born in Sichuan, China, in 2001. She joined the laboratory with the intent to research deep learning and image processing. She is currently an Assistant Researcher with Yangtze University, Jingzhou, China. She has been conducting research projects on image recognition and video prediction. Her primary interests include image processing and artificial intelligence.

HUA-QUAN CHEN was born in Guangxi, in 2000. In 2020, he joined the National Demonstration Center for Experimental Electrical and Electronic Education to study image processing and deep learning. He is committed to research in the laboratory image recognition and image classification and other scientific research projects. His main interest includes artificial intelligence.



CHANG WEN received the B.S. degree in computer science from the Naval University of Engineering, Wuhan, China, in 2002, and the M.S. degree in computer science from Yangtze University, Jingzhou, China, in 2008. She is currently an Assistant Professor with the School of Computer Science, Yangtze University, Jingzhou, China. She also works in the field of image processing and signal processing.

JIAN-BIAO HE received the B.S. and M.S. degrees from the Huazhong University of Science and Technology, Wuhan, China, in 1986 and 1989, respectively. He is currently an Associate Professor with the School of Computer Science and Engineering, Central South University. His research interests include artificial intelligence, the Internet of things, pattern recognition, mobile robots, and cloud computing.