A Rich Feature Fusion Single-Stage Object Detector

KAII ZHANG1, YASENJIANG MUSHA1, AND BINGLONG SI1

School of Mechanical Engineering, Xinjiang University, Urumqi 830046, China

Corresponding author: Yasenjiang Musha (8416293@qq.com)

This work was supported in part by the Natural Science Foundation of Education Department of Xinjiang Uygur Autonomous Region under Grant XJEDU2017M009, and in part by the Natural Science Foundation of Xinjiang University under Grant BS180264.

ABSTRACT

Single-stage object detectors are quick and highly accurate. Based on the way training model is developed, single-stage object detectors either adopt a training model based on a pre-trained backbone network model, or a model trained from the scratch. The pre-trained backbone network model is associated with the propagation sensitivity both in classification and detection. This leads to deviations in learning goals, and results in an architecture which is limited by the classification network, hence not easy to modify. Training from the scratch is not as efficient as using a pre-trained network, mainly due to the limitations of the predefined network system. In this paper, we combine these two approaches to overcome the above-mentioned shortcomings. In our proposed method a top-down concatenated feature pyramid is built upon a basic FSSD network. The experiments in this paper are conducted on MS COCO and PASCAL VOC data sets. Moreover, we apply VGG16 as the backbone network to further indicate the effectiveness of our proposed method which reaches 33.1 AP on MS COCO benchmark.

INDEX TERMS

Single-stage object detectors, training from the scratch, the pre-trained network.

I. INTRODUCTION

Object detection is a rapidly developing research area as it is used in a wide range of applications. The object detection techniques can be divided into single-stage and two-stage object detection categories. In the single-stage object detectors, see, e.g., [1]–[6], extracting the candidate regions are carried out without explicitly giving the final detection result. Combining extraction and detection is then used to detect the object, therefore, single-stage object detectors are faster than that of the two-stage object detectors. In two-stage object detector, see, e.g., [7]–[12], a two-stage cascaded network is used, where in the first stage the candidate boxes are proposed and then the object detection is decided on the candidate boxes.

There are two types of single-stage object detectors: (i) an object detector based on a pre-trained convolutional neural network, see, e.g., [13]–[15], and; (ii) an object detector trained from the scratch. Yolo algorithm in YOLOv3 [16] utilizes the CNN model to achieve the end-to-end object detection. In Yolo algorithm, first, the input image is resized to a fixed size, then it is feed as an input to the CNN network. Then the network predictions are processed to obtain the detected object. In SSD [1] inherits the idea of converting detection into a regression problem as in YOLO, and directly completes target positioning and classification; it is inspired by the anchor in Faster R-CNN [8], and proposes a similar Prior box; by adding FPN, predict the target on the feature map of different receptive fields. The FSSD [17] is for the improvement of the SSD model, adopting shallow feature fusion to improve the recognition of SSD in small object samples. In the object detection, positive and negative sample areas are extremely unbalanced, and the object detection loss is easily affected by many negative samples. This issue is addressed in RetinaNet [6], where the focal loss is proposed by RetinaNet, the problem is also addressed in RefineDet [18] as it proposed starting from the network structure and combines the advantages of one-stage object detection algorithm and two-stage object detection algorithm to redesign an object detection network with both accuracy and speed of SOTA. This method filters out the negative anchors first to reduce the classifier search space, and then roughly adjusts the position and size of the anchors to provide better an initialization for subsequent regressions.

Subsequent modules utilize the revised Anchors as the input to further improve the regression and enable predicting multi-level labels. The RFBNet [19] model designs the RFB block, which enhances the distinguishability of the features...
as well as the robustness of the model by simulating the relationship between the size of the receptive field (RF) structure and eccentricity in the human visual system, and further adds the RFB block to the SSD Method. The GBFPNSSD [20] adds the SE module as a gate module to the top-down and bottom-up feature pyramid networks. This dynamically readjusts the weight of features to achieve the purpose of transmitting only information features, GBFPNSSD then combines the two feature pyramids to improve the detection performance. Another technique, FFBNET [21] builds a dense feature pyramid based on FSSD which improves the accuracy of the small object recognition. The dense feature pyramid only slightly increases the number of the model’s parameters. The pre-trained model is generally trained on a classification image dataset such as ImageNet dataset [22], and may not necessarily be transferred to the detection model data set. Besides, its structure is fixed and not easy to modify. The training target of the pre-trained classification network is generally inconsistent with object detection, so the pre-trained model may be a suboptimal choice for the detection algorithm.

An object detector trained from the scratch is proposed in DSOD [23], where it uses the Proposal-free method to ensure convergence of the network, and draws on the design principle of DenseNet [13], and uses Dense Block to avoid the disappearance of the gradient. The proposed Dense Prediction Structure in DenseNet greatly reduces the number of model’s parameters, and the extracted features contain a lot of information. The DSOD uses a stem structure, which has the advantage of reducing the loss of input picture information. ScratchDet [24] introduces a new Root-ResNet backbone, which greatly improves the detection accuracy, especially for small objects, and also illustrates the importance of BatchNorm in the network structure. Our focus is on general object detection, mainly detecting more common objects, such as people, animals, etc. The input is taken from RGB images taken by a camera, with only 3 channels; the images with hyperspectral information are mainly remote sensing, aviation, etc. Domain, an image composed of an array of multiple channels (tens or even hundreds), each pixel has a lot of numbers to describe, the “grey value” on a single channel reflects the subject’s reflection of light. Since the hyperspectral image contains more information about the captured target, the recognition and classification of targets such as face recognition and object detection [25], [26] using hyperspectral images has higher accuracy than using RGB images. Rotation is a key and important issue in object detection. In our article, we use common rotation methods, and the rotation methods proposed in hyperspectral images [27], [28], the performance of these methods has been improved to a certain extent. We use the Mixup method [29] to expand the data. Essentially, Mixup trains a neural network on the convex combinations of paired samples and their labels. In Mixup technique two random samples are mixed in proportion, and the results of classification are distributed in proportion. Mixup is to fuse positive and negative samples into a new set of samples, doubling the sample size. Cutout [30] randomly cuts out part of the sample area and fills it with 0-pixel values, and the result of the classification is not changed. CutMix [31] cuts part of the area but not fill it with 0 pixels but randomly fill in the area pixel values of other data in the training set. The classification results are then distributed in a certain proportion. The difference between the above three data enhancement techniques are as follows: cutout and Cutmix differentiate between the pixel value of the filled area. Mixup is to interpolate the two images in proportion to mix the samples, and Cutmix is to use the cut part of the area and then patch the form to mix the image, there is no image mixing after unnatural situation. The main idea of AutoAugment [32] is to create a search space for data enhancement strategies and directly evaluate the quality of specific strategies on some data sets.

The scratch network used in this paper has a small number of convolution operations. Therefore, the extracted features have richer location information. The pre-training model generally contains a deep convolutional network structure, so the extracted features are relatively abstract and have rich semantic information. In our proposed technique, the FSSD method is used as a pre-training model, and then use the proposed Concatenated Feature Pyramid (CFP) to combine FSSD with the scratch network, so that the high-level semantic information of the deep feature map is extended to the shallow layer of the neural network. Therefore, we proposed an object detector that combines the scratch network and pre-training model to enrich the semantic information in the middle and shallow layers of the neural network. This improves the detection performance of small objects.

We conducted experiments on two data sets, MS COCO [33] and PASCAL VOC [34], and compared the performance of our proposed object detector with several existing object detectors. The experiments indicate that our proposed method overperforms the existing methods. Compared with the benchmark on the MS COCO data set, the accuracy of detection has greatly improved using our proposed method especially for small targets. For an input size of $512 \times 512$, our method achieves 33.1% AP, within 42 ms, exceeding the performance of object detectors such as YOLOv3 and RefineDet (see, Fig. 1 and Table 1).

II. BASELINE DETECTION FRAMEWORK

In our work, we use FSSD as the baseline structure of the object detector. FSSD [17] is a fast and accurate single-stage object detector. FSSD is an improved version of the SSD algorithm. To extract features, FSSD utilizes VGG-16 [14] as the backbone network, followed by using feature maps of different resolutions at different stages for prediction. Compared to the SSD, FSSD is based on the FPN concept as it constructs a feature fusion method which introduces the lower layer features to the upper layer of the network.

Limitation: Based on the above, although FSSD adds feature fusion to SSD, the feature semantic information on the shallow feature map is too little, and the recall rate for small
FIGURE 1. On the MS COCO dataset, the accuracy (AP) and speed (ms) are compared with the existing single-stage method. We also show the overall accuracy (AP) and the performance of small objects. Here, except for YOLOv3 (608 x 608), DSOD (300 x 300), and ScratchDet (300 x 300), the input image size used by the detector here is about 512 x 512. Our method, like other methods is based on VGG-16 [14] backbone. For a fair comparison, the speed is measured on a single 1080Ti GPU.

TABLE 1. The results of different detectors on the MS COCO dataset.

| Method      | Time | AP  | APS |
|-------------|------|-----|-----|
| SSD512      | 28   | 28.8| 10.9|
| FSSD512     | 35   | 31.8| 14.2|
| DSOD500     | 57.5 | 29.3| 9.4 |
| YOLO v3-608 | 51   | 33.0| 18.3|
| DSSD513[5]  | 156  | 33.2| 13.0|
| RefineDet512| 42   | 33.0| 16.3|
| RetinaNet-500| 73   | 32.5| 13.9|
| ScratchDet300| 25   | 32.7| 13.0|
| RFBNet512   | 30   | 33.8| 16.2|
| Our300      | 28   | 28.0| 11.1|
| Our512      | 42   | 33.1| 18.2|

A. OVERALL ARCHITECTURE
The overall architecture of the proposed technique is presented in Fig. 2 including three main modules: FSSD as the Baseline Detection Framework, the scratch network (SN), and the Concatenated Feature Pyramid. As the backbone network, FSSD uses the image classification network, VGG-16, which is pre-trained by the ImageNet dataset. The feature maps extracted by the FSSD and scratch network are then combined into the Concatenated Feature Pyramid through the sum and batch norm operations for direct prediction.

B. FSSD
The basic network architecture, FSSD, is an algorithm for object detection using a pre-trained VGG16 backbone network. FSSD fuses the feature maps including Conv4_3, FC7, Conv_8 extracted from the original SSD backbone network. We examined two fusion methods including concatenated and ele-sum, where ele-sum represents the pixel-by-pixel addition of the feature maps. Our experiments indicate that concatenated fusion overperforms ele-sum. We then add Convolutional and ReLU layers to get multiple-scale feature maps like the SSD with different channels. Finally, feature maps are input to the prediction layer to generate the results. The shallow features are fused. This is beneficial to object detection and results in improving the accuracy of small object detection.

C. THE SCRATCH NETWORK
The scratch network is similar to the image pyramid network. It saves most of the original image information after a few convolution operations, although there is a less deep high-level semantic information. The scratch network uses a small number of convolutional layers to ensure that the targets location information in the feature maps is rich enough. As it is seen in Fig 2, this architecture directly generates fixed-size feature maps by max-pooling the image, and then generates the feature maps of different scales through the convolution network. The feature maps obtained by the combination of FSSD and the scratch network in shallow/intermediate feature maps have therefore rich location information. This is conducive to the detection of small targets and appropriately improves the shallow semantic information.

D. CONCATENATED FEATURE PYRAMID
The Concatenated Feature Pyramid (CFP) is equivalent to the top-down schemes. The specific operations are as the following:

\[ F_{i+1} = \text{Cat}(U(F_i), \text{Conv}(F_{i+1})) \]

where Cat represents the Concat operation, U(·) represents the up-sampling operation, and Conv(·) represents the convolution of 1 x 1.

There are two fusion methods: Concat or ele-sum. Through experiments, it is found that Concat overperforms ele-sum. As shown in Fig. 2, in the CFP, this step is to increase the size of the deep feature maps to the same size as the shallower layer through Bilinear up-sampling, and the shallower feature maps through the 1 x 1 convolution operation. The two are
FIGURE 2. The overall framework of our proposed object detection method.

then combined using the Concat operation, which extends the high-level semantic information of deep feature maps to shallow features. This improves the detection accuracy of the smaller objects.

The deep network is shown to be effective for classification, and the shallow network is efficient for positioning. The deeper the network, the more abstract is the extracted features and the less location information becomes available for the generated feature maps. A larger number of convolutional layers results in a smaller feature map, and a larger range of the corresponding Receptive Field. This then results in the ability to detect larger objects, and a larger shallow feature map can detect small objects. However, due to the small amount of shallow semantic information, the detection performance for smaller objects is rather low. This paper proposes that based on the combination of pre-training and scratch training, training from scratch through image Maxpooling significantly reduces the number of parameters. This guarantees translation invariance, such as rotation, translation, and expansion, so that the training network from scratch includes more image position information. A small number of convolutional layers is also to ensure that the image has more location information, and the deeper feature map generated by the pre-training model is rich in semantic information. The two methods are combined to improve the effectiveness of small objects. Besides, our top-down concatenated feature pyramid transmits deep semantic information to the shallow network, such that the features at all scales have rich semantic information. The introduced top-down concatenated feature pyramid module offers the advantages of fewer parameters while providing improved performance.

IV. EXPERIMENT

A. DATASETS

In this paper we use two datasets including MS COCO, and PASCAL VOC. PASCAL VOC data set is divided into 4 major categories: vehicle, household, animal, and person, with a total of 20 small categories. This dataset can be used for network evaluation such as target classificaton and detection, and image segmentation. We use VOC2007 and VOC2012 train+val (including 16551 pictures) to train, and VOC2007 (including 4952 pictures) to test the detection performance. The evaluation standard of PASCAL VOC is mAP (mean average precision). MS COCO data set is used for object detection tasks which comprising of a total of 80 classes, and 123287 images including 118287 images in the training set, and 5000 images in the validation data set. For the object detection task, it has an average of 5 targets per image. The target position is constructed according to the key points, and the location annotation is performed. After training these annotations, we perform object detection. The evaluation method in MS COCO data set is different from the one used in the Pascal VOC data set. For MS COCO, a series of IOUs from 0.5 to 0.95 are used with an interval of 0.05 to calculate the AR (average recall) and the AP (average precision), and then calculate the average value as the final AR and AP.

B. PASCAL VOC

In the conducted experiment, Rich feature fusion single-stage detector was trained on the combined data set of Pascal VOC2007 and Pascal VOC2012. We use VOC2007 and VOC2012 train+val (16551 pictures) to train, and then
TABLE 2. Results of different detectors in the PASCAL VOC data set. The data set is a combination of PASCAL VOC2007, and PASCAL VOC2012 data sets.

| Method      | pre-train | Training data | Backbone network | mAP  | fps |
|-------------|-----------|---------------|-------------------|------|-----|
| Faster R-CNN| ✓         | 07 + 12       | VGGNet            | 73.2 | 7   |
| R-FCN       | ✓         | 07 + 12       | ResNet101         | 80.5 | 9   |
| SSD300      | ✓         | 07 + 12       | VGGNet            | 77.5 | 62  |
| DSOD300     | ×         | 07 + 12       | DS/64-192-48-1    | 77.7 | 17.4|
| GRP-DSOD320[36] | × | 07 + 12       | DS/64-192-48-1    | 78.7 | 16.7|
| FSSD300     | ✓         | 07 + 12       | VGGNet            | 78.8 | 65.8|
| DSSD321     | ✓         | 07 + 12       | ResNet101         | 78.6 | 9.5 |
| RefineDet320| ✓         | 07 + 12       | VGGNet            | 80.0 | 40.3|
| ASIF-Det320 | ✓         | 07 + 12       | VGGNet            | 79.2 | 33  |
| RFB Net300  | ✓         | 07 + 12       | VGGNet            | 80.5 | 83  |
| ScratchDet300 | ×      | 07 + 12       | Root-ResNet-34    | 80.4 | 17.8|
| Our300      | ✓         | 07 + 12       | VGGNet            | 79.7 | 83.3|
| SSD512      | ✓         | 07 + 12       | VGGNet            | 79.8 | 26  |
| FSSD512     | ✓         | 07 + 12       | VGGNet            | 80.9 | 35.7|
| DSSD513     | ✓         | 07 + 12       | ResNet101         | 81.5 | 5.5 |
| RefineDet512| ✓         | 07 + 12       | VGGNet            | 81.8 | 24.1|
| RFB Net512  | ✓         | 07 + 12       | VGGNet            | 82.2 | 38  |
| Our512      | ✓         | 07 + 12       | VGGNet            | 81.8 | 66.6|

VOC2007 (4952 pictures) to test the performance. For input size of 300 × 300, we set the batch size to 32 during training, and we set the total epoch to 250. We also set the initial learning rate to 0.003. For stable training, we use a warm-up strategy that gradually increases the learning rate from $1 \times 10^{-6}$ to $3 \times 10^{-3}$ within the first 6 epochs. Subsequently, it goes back to the original learning rate schedule divided by 10 at 150, 200 and 240 epochs. In our experiments, the weight decay is set to 0.0005 and the momentum to 0.9. For an input size of 512 × 512, compared with input size of 300 × 300, the total epoch is set to 200, the batch size is set to 16, and other settings are the same.

The ablation experiment is shown in Table 2. For the input size of 300 × 300, the value of the mAP in our method reaches 79.7, and for the input size of 512 × 512, the mAP value for our method reaches 81.8. Note that the inference speed of our proposed method is the fastest among the comparison methods.

C. MS COCO

In this experiment, the rich feature fusion single-stage detector was trained based on MS COCO2017 data set, which contains 118287 pictures in the training set and 5000 pictures in the verification set. Multiple GPUs are used for training. For an input size of 300 × 300, we set the batch size to 31 for each GPU during the training process, the total batch size is 93, and the total epoch is set to 150. At the beginning of the training, we apply the warm-up technique that gradually increases the learning rate from $1 \times 10^{-6}$ to $3 \times 10^{-3}$ during the first five epochs and then decrease it after 80 and 100 epochs by a factor of 10, ending up at 140. In our experiments, the weight decay is set to 0.0005, and the momentum to 0.9. For an input size of 512 × 512 we use 4 GPUs, and the total batch size is set to 38, and the other settings are the same as 300 × 300.

The ablation experiment is shown in Table 3. For the input size of 300 × 300, the AP of our method reaches 28, and for the input size of 512 × 512, the AP of our method reaches 33.1.

D. COMPARATIVE ANALYSIS OF EXPERIMENTS

Our method and FSSD, YOLOv3, RefineDet and RFBNet are all object detectors that use multi-scale prediction. FSSD is an improved version of the SSD detector to improve the detection effect of small targets. Small targets are usually predicted by shallow networks, with insufficient feature abstraction capabilities and lack of semantic information. Secondly, small object detection usually relies heavily on context information. Therefore, FPN is proposed to fuse shallow and deep features to better assist shallow features for object detection, thereby improving the detection effect of small targets. The scratch network used by our network takes the image through simple maxpooling and a small number of convolutional layers to obtain the shallow feature map with rich location information. It then uses a top-down concatenated feature pyramid to improve the detection effect of small targets.

YOLOv3 is a multi-scale object detector based on the network structure Darknet-53. The network structure draws on the practice of the residual network. Shortcut connections are set between some layers. The accuracy is high, and the speed is fast. It has a good effect on image feature extraction and further uses multi-scale prediction (similar to FPN).

RefineDet is an object detector that combines the advantages of one-stage and two-stage object detectors. It refers to the coarse-to-fine regression idea of the two-stage type box (first obtain the coarse-grained box information through the RPN network, and then pass The conventional regression branch performs further regression to obtain more accurate frame information), and refers to the feature fusion idea of...
TABLE 3. The results of different detectors on MS COCO dataset. The trainval dataset includes 5000 pictures more than that of trainval35k (123287 vs 118287).

| Method        | pre-train | Training data | Backbone network | Time(ms) | AP   | AP$_{50}$ | AP$_{75}$ | AP$_{M}$ | AP$_{L}$ |
|---------------|-----------|---------------|------------------|----------|------|----------|----------|----------|----------|
| Faster-RCNN   | ✓         | trainval35k   | VGGNet           | 147      | 24.2 | 45.3     | 23.5     | 7.7      | 26.4     | 37.1     |
| Faster-FPN    | ✓         | trainval35k   | ResNet-101-FPN   | 240      | 36.2 | 59.1     | 39.0     | 18.2     | 39.0     | 48.2     |
| R-FCN         | ✓         | trainval      | ResNet-101      | 110      | 29.9 | 51.9     | -        | 10.8     | 32.8     | 45.0     |

| RetinaNet-500 | ✓         | trainval35k   | ResNet-101      | 90       | 34.4 | 53.1     | 36.8     | 14.7     | 38.5     | 49.1     |
| SSD300        | ✓         | trainval35k   | VGGNet          | 12       | 25.3 | 42.0     | 26.5     | 6.2      | 28.0     | 43.3     |
| DSOD300       | ✗         | trainval      | DS/64-192-48-1  | -        | 29.3 | 47.3     | 30.6     | 9.4      | 31.5     | 47.0     |
| GRP-DSOD320   | ✗         | trainval      | DS/64-192-48-1  | -        | 30.0 | 47.9     | 31.8     | 10.9     | 33.6     | 46.3     |
| FSSD300       | ✓         | trainval35k   | VGGNet          | 16       | 27.1 | 47.7     | 27.8     | 8.7      | 29.2     | 42.2     |
| DSSD321       | ✓         | trainval35k   | ResNet-101      | -        | 28.0 | 46.1     | 29.2     | 7.4      | 28.1     | 47.6     |
| RefineDet320  | ✓         | trainval35k   | VGGNet          | 26       | 29.4 | 49.2     | 31.3     | 10.0     | 32.0     | 44.4     |
| RFBNet500     | ✓         | trainval35k   | VGGNet          | 15       | 30.3 | 49.3     | 31.8     | 11.8     | 31.9     | 45.9     |
| ScratchDet300 | ✗         | trainval35k   | Root-ResNet-34  | 25       | 32.7 | 52.0     | 34.9     | 13.0     | 35.6     | 49.0     |
| Our300        | ✓         | trainval35k   | VGGNet          | 28       | 28.0 | 48.4     | 28.3     | 11.1     | 31.4     | 42.9     |
| SSD512        | ✓         | trainval35k   | VGGNet          | 28       | 28.8 | 48.5     | 30.3     | 10.9     | 31.8     | 43.5     |
| FSSD512       | ✓         | trainval35k   | VGGNet          | 36       | 31.8 | 52.8     | 33.5     | 14.2     | 35.1     | 45.0     |
| DSSD513       | ✓         | trainval35k   | ResNet-101      | 156      | 33.2 | 53.3     | 35.2     | 13.0     | 25.4     | 51.1     |
| RefineDet512  | ✓         | trainval35k   | VGGNet          | 45       | 33.0 | 54.5     | 35.5     | 16.3     | 36.3     | 44.3     |
| RFBNet512     | ✓         | trainval35k   | VGGNet          | 30       | 33.8 | 54.2     | 35.9     | 16.2     | 37.1     | 47.4     |
| YOLO v3-608   | ✓         | trainval35k   | Darknet-53      | 51       | 33.0 | 57.9     | 34.4     | 18.3     | 35.4     | 41.9     |
| Our512        | ✓         | trainval35k   | VGGNet          | 42       | 33.1 | 54.4     | 35.1     | 18.2     | 36.5     | 46.8     |

TABLE 4. On the MS COCO dataset, our method is compared with FSSD, RefineDet, YOLOv3 and RFBNet in terms of accuracy (AP) and speed (ms).

| Method       | Backbone network | Time(ms) | AP   | AP$_{50}$ | AP$_{75}$ | AP$_{M}$ | AP$_{L}$ |
|--------------|------------------|----------|------|----------|----------|----------|----------|
| SSD300       | VGGNet           | 12       | 25.3 | 42.0     | 26.5     | 6.2      | 28.0     | 43.3     |
| FSSD300      | VGGNet           | 16       | 27.1 | 47.7     | 27.8     | 8.7      | 29.2     | 42.2     |
| RefineDet320 | VGGNet           | 26       | 29.4 | 49.2     | 31.3     | 10.0     | 32.0     | 44.4     |
| RFBNet300    | VGGNet           | 15       | 30.3 | 49.3     | 31.8     | 11.8     | 31.9     | 45.9     |
| Our300       | VGGNet           | 28       | 28.0 | 48.4     | 28.3     | 11.1     | 31.4     | 42.9     |
| SSD512       | VGGNet           | 28       | 28.8 | 48.5     | 30.3     | 10.9     | 31.8     | 43.5     |
| FSSD512      | VGGNet           | 36       | 31.8 | 52.8     | 33.5     | 14.2     | 35.1     | 45.0     |
| RefineDet512 | VGGNet           | 45       | 33.0 | 54.5     | 35.5     | 16.3     | 36.3     | 44.3     |
| RFBNet512    | VGGNet           | 30       | 33.8 | 54.2     | 35.9     | 16.2     | 37.1     | 47.4     |
| YOLO v3-608  | Darknet-53       | 51       | 33.0 | 57.9     | 34.4     | 18.3     | 35.4     | 41.9     |
| Our512       | VGGNet           | 42       | 33.1 | 54.4     | 35.1     | 18.2     | 36.5     | 46.8     |

FPN (improve the detection effect of small targets). The framework of RefineDet detection is still SSD. The difference with SSD refers to the RPN idea to multi-feature map detection.

The RFBNet detector also uses a multi-scale detection framework SSD. The RFB module is embedded to make the lightweight backbone SSD network faster and more accurate. RFB is a multi-branch convolution module similar to the Inception module. Its internal structure can be divided into two components: a multi-branch convolution layer and a subsequently expanded convolution layer.

As shown in Table 4, on the MS COCO data set, our method is compared with FSSD, RefineDet, YOLOv3 and RFBNet in terms of accuracy (AP) and speed (ms). In this
Table 5. The validity of the proposed module is verified on the PASCAL VOC data set.

|                | fps | mAP |
|----------------|-----|-----|
| FSSD           | 65.8| 78.8|
| FSSD+CFP       | 100 | 79.5|
| Our            | 88.3| 79.7|
| FSSD           | 35.7| 80.9|
| FSSD+CFP       | 77.3| 81.6|
| Our            | 66.6| 81.8|

table, the performance of AP, IOU of 0.5, IOU of 0.75, large objects, medium objects, and small objects are shown, and our results are all sub-optimal. Detailed comparison in the experiment: Our method outperforms FSSD and SSD in all aspects, except for AP_{50} and AP_{75} indicators in which, it is better than RefineDet; it is better than RFBNet in the detection of small and medium targets; except for AP_{50} and AP_{5} indicators, it is better than YOLO v3. Besides, our method is also faster than RefineDet and YOLO v3.

Ablation Study: To evaluate the effectiveness of CFP and the scratch network in our method, we also conducted a series of ablation experiments summarized in Table 5. To be fair, we used the same training strategy and input size (300 × 300 and 512 × 512) in all experiments.

V. CONCLUSION

In this paper we proposed an object detector that combines the scratch network and pre-training model to enrich the semantic information in the middle and shallow layers of the neural network. In our proposed technique, FSSD method was used as the pre-training model, followed by Concatenated Feature Pyramid to extend the high-level semantic information of the deep Feature maps to the neural network shallow layer. This improved the detection performance of small objects. Our experiments on Pascal VOC dataset and MS COCO dataset indicate that the proposed method outperformed single-stage object detectors, including, YOLO v3, RefineDet, ScratchDet.

ACKNOWLEDGMENT

The authors would like to express their gratitude to EditSprings (https://www.editsprings.com/) for the expert linguistic services provided.

REFERENCES

[1] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “SSD: Single shot multibox detector,” in Proc. 14th Eur. Conf. Comput. Vis. Amsterdam, Netherlands: Springer-Verlag, 2016, pp. 21–37.

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proc. 29th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR). Las Vegas, NV, USA: IEEE Computer Society, Jul. 2016, pp. 779–788.

[3] J. Redmon and A. Farhadi, “YOLO9000: Better, faster, stronger,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR). Honolulu, HI, USA: Institute of Electrical and Electronics Engineers Inc, Jul. 2017, pp. 4203–4212.

[4] S. Liu, D. Huang, and Y. Wang, “Receptive field block net for accurate and fast object detection,” in Proc. 15th Eur. Conf. Comput. Vis. Munich, Germany: Springer-Verlag, Sep. 2018, pp. 385–400.

[5] B. Fan, Y. Chen, J. Qu, Y. Chai, C. Xiao, and P. Huang, “FFBN: Lightweight backbone for object detection based feature fusion block,” in Proc. IEEE Int. Conf. Image Process. (ICIP). Taipei, Taiwan: IEEE Computer Society, Sep. 2019, pp. 3920–3924.

[6] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 42, no. 2, pp. 318–327, Feb. 2020.

[7] Z. Cai and N. Vasconcelos, “Cascade R-CNN: Delving into high quality object detection,” in Proc. 31st Meeting IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR). Salt Lake City, UT, USA: IEEE Computer Society, Jun. 2018, pp. 6154–6162.

[8] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[9] R. Girshick, “Fast R-CNN,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV). Santiago, Chile: Institute of Electrical and Electronics Engineers Inc., Dec. 2015, pp. 1440–1448.

[10] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 42, no. 2, pp. 386–397, Feb. 2020.

[11] 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). Honolulu, HI, USA: Institute of Electrical and Electronics Engineers Inc, Jul. 2017, pp. 936–944.

[12] J. Dai, Y. Li, K. He, and J. Sun, “R-FCN: Object detection via regionbased fully convolutional networks,” in Proc. 30th Annua. Conf. Neural Inf. Process. Syst. (NIPS). Barcelona, Spain, 2016, pp. 379–387.

[13] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in Proc. 36th IEEE Conf. Comput. Vis. Pattern Recognit. Honolulu, HI, USA: Institute of Electrical and Electronics Engineers Inc, Jul. 2017, pp. 4700–4708.

[14] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in Proc. ICLR, 2015, pp. 1–14.

[15] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR). Las Vegas, NV, USA: IEEE Computer Society, Jun. 2016, pp. 770–778.

[16] J. Redmon and A. Farhadi, “YOLov3: An incremental improvement,” 2018, arXiv:1804.02767. [Online]. Available: http://arxiv.org/abs/1804.02767.

[17] Z. Li and F. Zhou, “FSSD: Feature fusion single shot multibox detector,” 2017, arXiv:1712.00960. [Online]. Available: http://arxiv.org/abs/1712.00960.

[18] S. Zhang, L. Wen, Y. Chai, Z. Li, and S. Z. Li, “One-shot single-shot detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Salt Lake City, UT, USA: IEEE Computer Society, Jun. 2016, pp. 770–778.

[19] J. Dai, D. Huang, and Y. Wang, “Receptive field block net for accurate and fast object detection,” in Proc. 15th Eur. Conf. Comput. Vis. Munich, Germany: Springer-Verlag, Sep. 2018, pp. 385–400.

[20] S. Woo, S. Hwang, H.-D. Jang, and I. S. Kweon, “Gated bidirectional feature pyramid network for accurate one-shot detection,” Mach. Vis. Appl., vol. 30, no. 4, pp. 543–555, Jun. 2019.

[21] B. Fan, Y. Chen, J. Qu, Y. Chai, C. Xiao, and P. Huang, “FFBN: Lightweight backbone for object detection based feature fusion block,” in Proc. IEEE Int. Conf. Image Process. (ICIP). Taipei, Taiwan: IEEE Computer Society, Sep. 2019, pp. 3920–3924.

[22] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, “ImageNet large scale visual recognition challenge,” Int. J. Comput. Vis., vol. 115, no. 3, pp. 211–252, 2015.

[23] Z. Shen, Z. Liu, J. Li, Y.-G. Jiang, Y. Chen, and X. Xue, “DSOD: Learning deeply supervised object detectors from scratch,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV). Venice, Italy: Institute of Electrical and Electronics Engineers Inc, Oct. 2017, pp. 1919–1927.

[24] R. Zhu, S. Zhang, X. Wang, L. Wen, H. Shi, L. Bo, and T. Mei, “ScratchDet: Training single-shot object detectors from scratch,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Long Beach, CA, USA: IEEE Computer Society, Jun. 2019, pp. 2268–2277.

[25] B. Rasti, D. Hong, R. Hang, P. Ghamisi, X. Kang, J. Chauhanus, and J. A. Benediktsson, “Feature extraction for hyperspectral imagery: The evolution from shallow to deep (overview and toolbox),” IEEE Geosci. Remote Sens. Mag., early access, Apr. 29, 2020, doi: 10.1109/MGRS.2020.2979764.

[26] X. Wu, D. Hong, J. Tian, J. Chauhanus, W. Li, and R. Tao, “ORSIM detector: A novel object detection framework in optical remote sensing imagery using spatial-frequency channel features,” IEEE Trans. Geosci. Remote Sens., vol. 57, no. 7, pp. 5146–5158, Jul. 2019.
K. Zhang et al.: Rich Feature Fusion Single-Stage Object Detector

[27] X. Wu, D. Hong, J. Chanussot, Y. Xu, R. Tao, and Y. Wang, “Fourier-based rotation-invariant feature boosting: An efficient framework for geospatial object detection,” IEEE Geosci. Remote Sens. Lett., vol. 17, no. 2, pp. 302–306, Feb. 2020.

[28] D. Hong, N. Yokoya, J. Chanussot, and X. X. Zhu, “An augmented linear mixing model to address spectral variability for hyperspectral unmixing,” IEEE Trans. Image Process., vol. 28, no. 4, pp. 1923–1938, Apr. 2019.

[29] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “MixUp: Beyond empirical risk minimization,” in Proc. 6th Int. Conf. Learn. Represent., Vancouver, BC, Canada, 2018, pp. 1–13.

[30] T. DeVries and G. W. Taylor, “Improved regularization of convolutional neural networks with cutout,” 2017, arXiv:1708.04552. [Online]. Available: http://arxiv.org/abs/1708.04552

[31] S. Yun, D. Han, S. Chun, S. J. Oh, Y. Yoo, and J. Choe, “CutMix: Regularization strategy to train strong classifiers with localizable features,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV). Seoul, Republic of Korea: Institute of Electrical and Electronics Engineers Inc, Oct. 2019, pp. 6023–6032.

[32] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, “AutoAugment: Learning augmentation strategies from data,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR). Long Beach, CA, USA: IEEE Computer Society, Jun. 2019, pp. 113–123.

[33] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft COCO: Common objects in context,” in Proc. 13th Eur. Conf. Comput. Vis. Zurich, Switzerland: Springer-Verlag, 2014, pp. 740–755.

[34] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The Pascal visual object classes (VOC) challenge,” Int. J. Comput. Vis., vol. 88, no. 2, pp. 303–338, Jun. 2010.

[35] Y. Pang, T. Wang, R. M. Anwer, F. S. Khan, and L. Shao, “Efficient featurized image pyramid network for single shot detector,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR). Long Beach, CA, USA: IEEE Computer Society, Jun. 2019, pp. 7336–7344.

[36] Z. Shen, H. Shi, J. Yu, H. Phan, R. Feris, L. Cao, D. Liu, X. Wang, T. Huang, and M. Savvides, “Improving object detection from scratch via gated feature reuse,” 2017, arXiv:1712.00886. [Online]. Available: http://arxiv.org/abs/1712.00886

KAI ZHANG was born in Jiaozuo, Henan, China, in 1995. He received the bachelor’s degree in mechanical engineering from Henan Polytechnic University, Jiaozuo, in 2017. He is currently pursuing the master’s degree with Xinjiang University, Urumqi. His research interests is object detection algorithms based on deep learning.

YASENJANG MUSHA was born in Atushi, Suzhou, Kyzyl, Xinjiang Uygur Autonomous Region, in October 1972. He received the bachelor’s degree from Xinjiang University, in 1996, and the master’s and Ph.D. degrees from the Nanjing University of Science and Technology, in 2009 and 2017, respectively. He is currently a Teacher with the College of Mechanical Engineering, Xinjiang University. He has published more than five academic articles at home and abroad, and participated in more than ten provincial and ministerial or national scientific research projects. Based in the field of computer vision, he conducts theoretical research on information processing technologies such as pattern recognition, digital image processing, machine learning, and neural networks in digital image processing, feature extraction, artificial intelligence, and neural networks.

BINGLONG SI was born in Zhoukou, Henan, China, in 1993. He received the bachelor’s degree in mechanical design and manufacturing from the Zhengzhou Aviation Industry Management College, in 2019. He is currently pursuing the master’s degree with Xinjiang University, researching robot vision. During this period, he participated in the school robot society and won the third prize of the Robot Innovation Competition.