IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 7, 2020, accepted January 20, 2020, date of publication February 3, 2020, date of current version February 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2971026

DF-SSD: An Improved SSD Object Detection Algorithm Based on DenseNet and Feature Fusion

SHEPING ZHAI[1,2], DINGRONG SHANG[1], SHUHUAN WANG[1], AND SUSU DONG[1]
[1]School of Computer Science and Technology, Xi'an University of Posts and Telecommunications, Xi'an 710121, China
[2]Shaanxi Key Laboratory of Network Data Analysis and Intelligent Processing, Xi'an 710121, China

Corresponding author: Dingrong Shang (sdr_person@163.com) and Shuhuan Wang (17835422718@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61373116, in part by the Ministry of Industry and Information Technology Communication Soft Science Project under Grant 2018-R-26, in part by the Shaanxi Provincial Social Science Foundation under Grant 2016N008, in part by the Shaanxi Provincial Department of Education Science Research Program Foundation under Grant 18JK0697, in part by the Xi'an Social Science Planning Foundation under Grant 17X63, in part by the Xi'an University of Posts and Telecommunications Graduate Innovation Foundation under Grant CXJJLY2018066, and in part by the Xi'an University of Posts and Telecommunications Science and Technology Project under Grant 19-B-281.

ABSTRACT In view of the lack of feature complementarity between the feature layers of Single Shot MultiBox Detector (SSD) and the weak detection ability of SSD for small objects, we propose an improved SSD object detection algorithm based on Dense Convolutional Network (DenseNet) and feature fusion, which is called DF-SSD. On the basis of SSD, we design the feature extraction network DenseNet-S-32-1 with reference to the dense connection of DenseNet, and replace the original backbone network VGG-16 of SSD with DenseNet-S-32-1 to enhance the feature extraction ability of the model. In the part of multi-scale detection, a fusion mechanism of multi-scale feature layers is introduced to organically combine low-level visual features and high-level semantic features in the network structure. Finally, a residual block is established before the object prediction to further improve the model performance. We train the DF-SSD model from scratch. The experimental results show that our model DF-SSD with 300×300 input achieves 81.4% mAP, 79.0% mAP, and 29.5% mAP on PASCAL VOC 2007, VOC 2012, and MS COCO datasets, respectively. Compared with SSD, the detection accuracy of DF-SSD on VOC 2007 is improved by 3.1% mAP. DF-SSD requires only 1/2 parameters to SSD and 1/9 parameters to Faster RCNN. We inject more semantic information into DF-SSD, which makes it have advanced detection effect on small objects and objects with specific relationships.

INDEX TERMS DenseNet, feature fusion, multi-scale object detection, SSD.

I. INTRODUCTION

Object detection is one of the important research topics in the field of computer vision. The main task is to locate the object of interest from the image, and it is necessary to accurately judge the specific category and location information of each object. In recent years, object detection has been widely used in intelligent video monitoring [1], [2], fault detection [3], [4], medical treatment [5] and other fields. Since AlexNet [6] proposed by Krizhevsky *et al*. made a significant improvement on ImageNet [7] in 2012, various deep learning methods represented by convolutional neural networks (CNNs) have been widely applied in many visual tasks, including object detection.

As an object detection algorithm based on deep learning, Single Shot MultiBox Detector (SSD) [8] has high performance in both detection accuracy and detection speed. SSD algorithm is proposed by Liu W *et al*. in 2016 to solve the problem of insufficient detection accuracy of YOLO [9] series algorithm in object positioning. Its main idea is to sample densely and evenly at different locations of the image. SSD draws on the concept of anchor in Faster R-CNN [10]. At the time of sampling, the object bounding box is predicted by priori boxes of different scales and aspect ratios, then the feature is extracted by CNN and then classified and regressed directly. The whole process only needs one step.

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

SSD adopts the feature layer group of the pyramid structure for object detection. The high-level feature maps with large receptive field are used to predict large targets, and the low-level feature maps with small receptive field are used to predict small targets. However, the feature layers of different scales are independent, lacking the complementarity of features between feature layers, which makes SSD less effective in both general object detection and small object detection tasks. The idea to solve this problem is to fuse the high-level semantic information with the low-level detail information. Therefore, in order to enhance the representation ability of low-level feature maps, we map high-level feature maps rich in semantic information in SSD network structure to low-level networks by deconvolution. In addition, atrous convolution is used to make the low-level feature maps have high enough resolution to obtain the location information of objects. Experimental results show that our detector improves the small objects detection capability of SSD. DF-SSD has advanced detection effects for small objects and objects with specific relationships.

SSD adds a convolutional predictor after each feature map used for detection, which is composed of one 3×3 bounding box regression convolution layer and one 3×3 class label prediction convolution layer. We design the residual prediction module following the design idea proposed by Lee *et al.* [11], which is to encourage features to be passed along the feature extraction network. We use 1×1 small convolution filter to predict object categories and offsets in bounding box locations, thus reducing the calculation cost while maintaining the detection accuracy.

In order to obtain higher accuracy, most of the current object detection models fine-tune networks pre-trained on ImageNet. But there are two problems with using a pre-trained network model. First, the pre-training will limit the network structure design space, and the number of parameters and calculation of the pre-training network model is large, so the network structure cannot be adjusted to adapt to more computationally limited scenarios. Second, the loss function and category distribution of classification and detection tasks are different, so the optimization space is also different [12]. DSOD [13] is the first object detection model to realize the training from scratch. The author's experiments show that the two-stage detection models based on region proposals will not converge if it is trained from scratch. However, the author uses the proposal-free model SSD to obtain advanced accuracy and speed. So we train our models from scratch.

One of the design principles of DSOD is deep supervision, whose fundamental idea is to provide an integrated objective function, that is, the loss function should provide supervisory signals to both the output layer and the non-output layer. Both ResNet [14] and DenseNet [15] have skip-connection, which satisfy this requirement. We design the feature extraction network DenseNet-S-32-1 based on the idea of DenseNet. On the one hand, it satisfies the condition of depth monitoring. On the other hand, it greatly reduces the number of model parameters and improves the feature extraction capability of the backbone network.

In summary, the main contributions of this paper are listed as follows:

1) We propose an improved SSD object detection algorithm DF-SSD. Feature extraction network DenseNet-S-32-1 is designed to replace the original backbone network VGG-16 of SSD. We propose a novel feature fusion module to inject more context information into the pyramid feature layers. A residual prediction module is added for each feature layer used for detection.
2) We train our models from scratch. Using less training data, our detector DF-SSD achieves competitive or even better performance on common datasets (PASCAL VOC and MS COCO) than other state-of-the-art pre-training models.
3) DF-SSD has advanced detection effects for small objects and objects with specific relationships. It requires only 1/2 parameters to SSD and 1/9 parameters to Faster RCNN, which shows great application potential for resource-limited scenarios.

II. RELATED WORK

A. OBJECT DETECTION METHOD BASED ON DEEP LEARNING

Currently, object detection methods based on deep learning are mainly divided into two categories: one is based on region proposal, also known as two-stage algorithms. The other is the one-stage algorithms based on regression. The object detection algorithm based on region proposal divides the detection problem into two stages. First, region proposals are extracted from the input images according to the region selection algorithm (such as Selective Search [16], Edge Box [17], etc.). Second, region proposals are classified and position adjusted to output the target detection results. Although this kind of algorithm has high accuracy, its detection speed is slow, which is difficult to meet the real-time requirements of some scenes. Typical representatives include R-CNN [18], Faster R-CNN, and R-FCN [19].

However, the regression-based object detection algorithm has more advantages in terms of speed. For a given input image, it needs to be processed only once, and the target border and category of this position can be regressed in multiple positions of the image. Typical representatives are YOLO and SSD. YOLO mainly classifies and locates objects of different scales on the single scale feature layer, which makes the burden of the feature layer too large. SSD extracts default boxes on multiple feature layers of different scales to complete detection and location of targets of different sizes. It can achieve 74.3% mAP on VOC 2007 datasets with a processing speed of 46 FPS. Both the mAP indicator and the detection speed have higher performance.

B. FEATURE EXTRACTION NETWORK

Recently, with the rapid development of CNN, a large number of highly efficient models have emerged in academic circles, such as VGGNet [20], GoogleNet [21], and ResNet. However, with the deepening of the number of network layers, the pretransmit signals and gradient signals in the training process of the network may gradually disappear after passing through many layers [22]. Many excellent papers have focused on solving this problem. For example, ResNets and Highway Networks [23] all propose a skip-layer technique to enable the high-speed flow of signals between the input and output layers.

DenseNet maximizes the information flow between all layers in the network through a new connection manner, that is, each layer in the network is directly connected to all the layers in front of it, so that the features of all the previous layers are used as the input of this layer to realize feature reuse. In addition, DenseNet also has the advantages of effectively solving gradient disappearance, enhancing feature propagation and substantially reducing the number of parameters, which can be well applied to various computer vision tasks based on convolution features, such as [24], [25].

Shen *et al.* [13] design DS/64-192-48-1 to replace the backbone network VGG-16 by referring to DenseNet's dense connection on the basis of SSD, and adopt the training method from scratch to improve the situation that pre-training weights are often required to be loaded during model training. This paper mainly discusses the model design principle of training from scratch, without paying attention to the detection performance of SSD on small objects.

C. MULTI-SCALE OBJECT DETECTION

Feature fusion of different scales is a very effective strategy to realize feature complementarity between feature layers of SSD, which is also confirmed by Feature-Fused SSD [26] and FSSD [27]. SharpeMask [28] and FPN [29] fuse feature layers of different scales through top-down structures to generate new feature layers for classification and position regression.

The low-level feature maps of SSD lacks sufficient context information, which makes it less effective in small object detection. In 2017, Rainbow SSD [30] proposed by Jeong fuses feature maps with simple concatenation and deconvolution, making full use of the direction information of the feature maps. These improvements make Rainbow SSD suitable for small targets detection. The authors of DSSD [31] point out that an effective strategy to improve the accuracy of object detection algorithm is to design a reasonable network structure and introduce more semantic information into the feature layer. Therefore, DSSD replaces the SSD backbone network VGG-16 with ResNet-101 [14], and uses deconvolution structure to introduce stronger semantic information for feature layers. It achieves 81.5% mAP test results on VOC 2007 datasets, but seriously sacrifice detection speed.

Recently, some studies have shown that receptive field plays an important role in improving the performance of multi-scale object detection algorithms [32]–[35]. For example, inspired by the Atrous Spatial Pyramid Pooling (ASPP) [36], literature [32] and [33] introduce the method of gathering features through atrous convolution for object detection. While traditional CNN method increases semantic information or receptive field through a series of convolution filters and pooling layers, but it will reduce the resolution of image features, which is not conducive to accurate object location and detection.

III. DF-SSD

The network structure of DF-SSD can be divided into two parts: the backbone network DenseNet-S-32-1 (a variant of DenseNet) for feature extraction and the front-end network (including feature fusion module and residual prediction module) for object detection on multi-scale feature maps. The structure of the entire model is shown in Fig. 1.

Our model input size is 300×300. The purple module in Fig. 1 is the residual prediction block. In the original SSD, the author select Conv4_3, Conv7 of VGG-16 and the newly added layers Conv8_2, Conv9_2, Conv10_2, and Conv11_2 for object classification and location regression. We construct multi-scale feature layers according to SSD. Under the premise of ensuring accuracy, for the detection speed, we use our own feature fusion modules Conv4_Fu, Conv7_Fu, Conv8_Fu, Conv9_Fu and Conv10_2 and Conv11_2 in the original SSD for target detection. The feature sizes of the feature layer group composed of these six scales are 38×38, 19×19, 10×10, 5×5, 3×3, and 1×1, which is the same as original SSD. The two parts of DF-SSD detector are described in detail below.

A. BACKBONE NETWORK REPLACEMENT

The backbone network of DF-SSD is DenseNet-S-32-1, which is obtained by certain modification of DenseNet. Where S denotes the stem block, 32 denotes the growth rate $k = 32$ in dense blocks, and 1 denotes the compression factor $\theta = 1$ in transition layers. The network structure of DenseNet-S-32-1 is shown in TABLE 1.

The whole backbone network consists of a stem block and a four-stage feature extractor. Both stage (1) and stage (2) use a 2×2 max pooling layer with stride $= 2$. The stage (3) and stage (4) use a transition w/o pooling layer. A four-stage structure is a commonly used structure in the large model design. ShuffleNet [37] uses a three stage structure and shrinks the feature map size at the beginning of each stage. Although this can effectively reduce computational cost, we argue that early stage features are very important for vision tasks, and that premature reducing the feature map size can impair representational abilities. Therefore, we still maintain a four-stage structure.

1) STEM BLOCK

Motivated by Inception-v4 [38] and DSOD, we design an efficient stem block structure before the first dense block, as shown in Fig. 2.

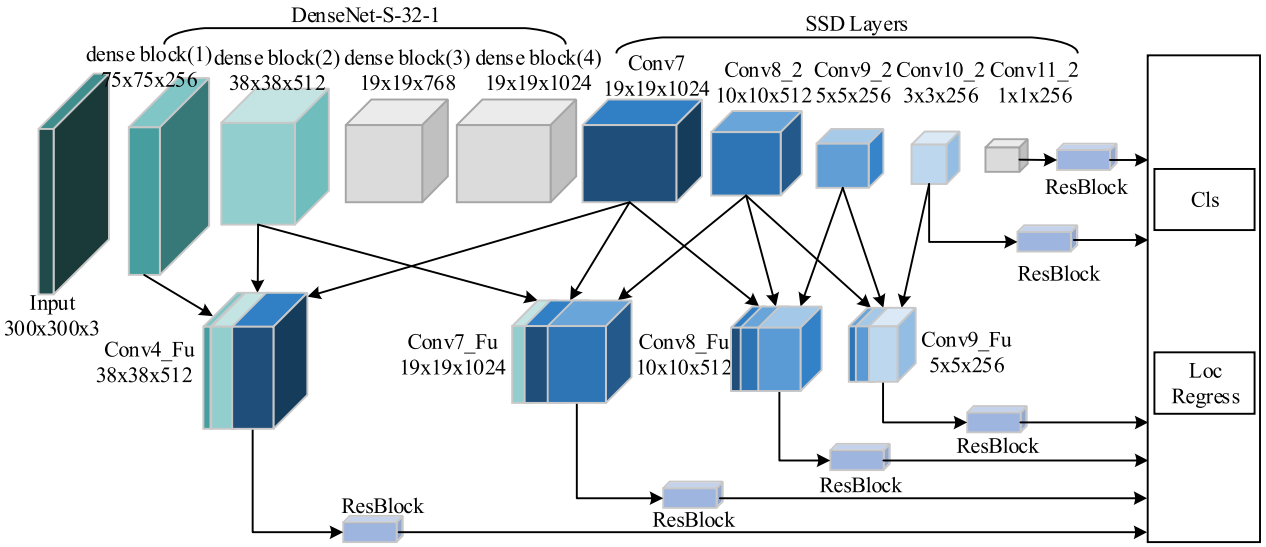


FIGURE 1. Network structure of DF-SSD. We design the DenseNet-S-32-1 and use it to replace the SSD backbone network VGG-16. ConvX_Fu is the feature fusion module designed by us, and the specific fusion method is shown in Fig. 5. The purple module is the residual prediction block that we added, namely ResBlock.

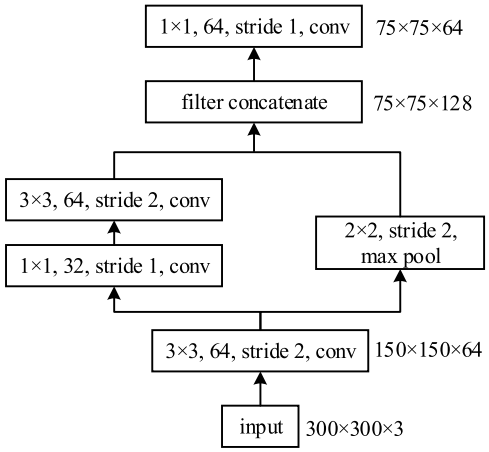


FIGURE 2. Structure of stem block.

The experimental results show that, compared with the original design in DenseNet (a 7 × 7 convolutional layer and a 3 × 3 max pooling before the first dense block), the stem block we designed can reduce the information loss from raw input images, effectively improve the ability to express features without increasing the computational cost, which is also confirmed by DSOD. However, unlike DSOD, we designed a two-way stem block structure that borrows from the parallel structure and asymmetric convolution kernel structure used in Inception-v4 to reduce the calculation amount while keeping the loss of feature information small enough.

2) COMPOSITE FUNCTION

We use the composite function of the original DenseNet, which consists of three consecutive operations: Batch Normalization (BN), followed by a Rectified Linear Nnit (ReLU) and a convolution (Conv).

TABLE 1. DenseNet-S-32-1 architecture.

Stage	Layers	Output Size (Input 300×300×3)	Specification
	stem block	75×75×64	
Stage (1)	dense block (1)	75×75×256	$\begin{pmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{pmatrix}\times6$
	transition layer (1)	75×75×256	1×1 conv
		38×38×256	2×2 max pool, stride 2
Stage (2)	dense block (2)	38×38×512	$\begin{pmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{pmatrix}\times8$
	transition layer (2)	38×38×512	1×1 conv
		19×19×512	2×2 max pool, stride 2
Stage (3)	dense block (3)	19×19×768	$\begin{pmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{pmatrix}\times8$
	transition w/o pooling layer (1)	19×19×768	1×1 conv
Stage (4)	dense block (4)	19×19×1024	$\begin{pmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{pmatrix}\times8$
	transition w/o pooling layer (2)	19×19×1024	1×1 conv

Growth rate k = 32 in each dense block. Note that each "conv" layer shown in the table corresponds the composite function sequence BN-ReLU-Conv.

3) TRANSITION W/O POOLING LAYER

In the original design of DenseNet, each transition layer contains an average pooling operation to down-sample the feature maps. The number of dense blocks is fixed (4 dense blocks in all DenseNet architectures) if one wants to maintain the same scale of outputs. The only way to increase network depth is to add layers to each of the dense blocks of the

original DenseNet [13]. Therefore, we also use the transition w/o pooling layer to eliminate the layer limitation of each of the dense blocks of the DF-SSD network structure, which makes the network deeper and has stronger feature extraction capability, and the final feature map resolution is not reduced.

4) GROWTH RATE

The growth rate refers to the number of 3×3 convolution kernels in the last dense block, denoted as k. Since each dense block is finally connected in a concatenation manner, the feature dimension of the next layer will increase by k after each dense block. The larger its value means the greater the amount of information circulating in the network, and the stronger the network performance, but the size and computation of the entire model will also increase. We use two settings, $k = 16$ and $k = 32$.

5) BOTTLENECK LAYER

We also introduce 1×1 convolution as the bottleneck layer before each 3×3 convolution, so as to reduce the number of input feature maps and improve the calculation efficiency.

6) COMPRESSION

To further improve model compactness, DenseNet use compression factor $\theta(0 < \theta \leqslant 1)$ to reduce the number of feature maps at transition layers. If a dense block contains m feature maps, the following transition layer will generate $\lfloor \theta m \rfloor$ output feature-maps. Experiment in DenseNet shows that the compression factor θ can impair feature expression. Therefore, we set the compression factor $\theta = 1$, that is, keep the number of feature maps across transition layers constant.

B. FRONT-END PREDICTION NETWORK

In this paper, feature fusion and reuse between different layers are enhanced by replacing feature extraction structure and redesigning the front-end prediction network, which includes multi-scale fusion module and residual prediction module.

1) MULTI-SCALE FEATURE FUSION

SSD adopts multi-scale feature layers for object classification and location. The low-level feature maps have the characteristics of high resolution and small receptive field, which can well represent the detail information such as the texture and edge of the image. This is beneficial to object positioning, but its weak global semantic features are not conducive to object classification. On the contrary, the high-level feature maps can provide rich semantic information, which is beneficial to object classification, but the low resolution of the high-level feature maps is not good for the object location task [39].

Our multi-scale fusion module takes full advantage of the relationship between different feature layers. On the one hand, atrous convolution is used to fuse the low-level feature maps and the high-level feature maps. This can significantly improve the receptive field range of the classification network, which is beneficial to the model to learn more global information. On the other hand, the semantic information of the high-level feature maps is integrated into the low-level feature maps by using deconvolution, which is conducive to the small target detection of the low-level feature maps and enhances the semantic representation ability of the model. This multi-scale fusion module enables the front-end prediction network to take into account the different scales of objects and enhance the generalization ability of the model.

a: ATROUS CONVOLUTION

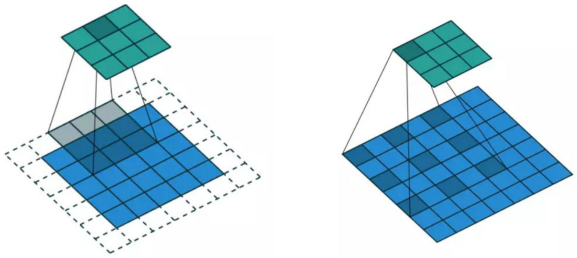
The atrous convolution operation is equivalent to inserting spaces between the convolution kernel elements, where a new super parameter *dilation* is introduced, and the value of $(dilation - 1)$ is the number of spaces inserted. Assuming that the original convolution kernel size is k, then the new kernel size n after inserting $(dilation - 1)$ spaces is:

$$n = k + (k - 1) \times (dilation - 1) \tag{1}$$

After performing the atrous convolution, the receptive field r is:

$$r = \left[2^{(dilation/2)+2} - 1\right] \times \left[2^{(dilation/2)+2} - 1\right] \tag{2}$$

Fig. 3 (a) is an atrous convolution operation with $k = 3$ and $dilation = 1$, which is equivalent to a convolution operation. The 3×3 region is the receptive field range of the current convolution. Fig. 3 (b) is an atrous convolution operation with $k = 3$ and $dilation = 2$. According to (1) and (2), the new convolution kernel size after the atrous convolution operation is 5×5. Compared with Fig. 3 (a), the receptive field r is expanded to 7×7 without loss of feature information.



(a) (b)

FIGURE 3. Atrous convolution operation.

b: DECONVOLUTION

Fig. 4 (a) is the convolution calculation process with input size of 5×5, filter size of 3×3, stride of 2, padding of 1, and output size of 3×3. Fig. 4 (b) is the deconvolution process corresponding to the convolution operation in Fig. 4 (a), whose input size is 3×3 and output is 5×5. The relation between input and output of deconvolution is as follows:

$$d = [s\,(i - 1) + k - 2p] \times [s\,(i - 1) + k - 2p] \tag{3}$$

where,

s : Moving step or stride.
i : The feature size of the input.
k : The size of the convolution kernel.

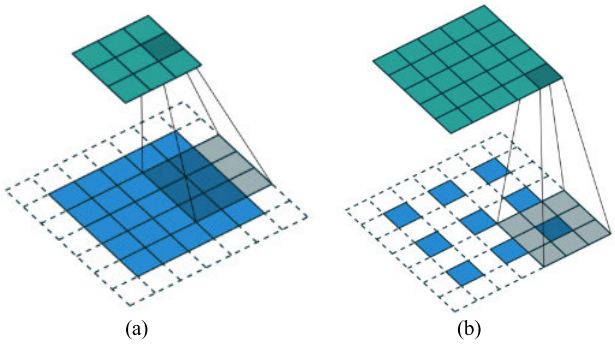


FIGURE 4. Convolution and deconvolution operation.

p : Padding.

d : The feature size of the output.

For example, in Fig. 4 (b), $i = 3, s = 1, k = 3$, and $p = 0$, so the output d $= 5 \times 5$.

We consider that the fusion of feature maps with too small resolution has basically no improvement in detection accuracy, but will slow down detection speed. And SSD generally uses low-level feature map to detect small targets. Therefore, only the feature maps of the five scales of dense block (1), dense block (2), Conv7, Conv8_2, and Conv9_2 are selected for feature fusion (as shown in Fig. 1). There are two main methods to fuse different feature maps together: element-wise summation and concatenation. Element-wise summation requires that the feature maps be the same size and must be converted to the same channels, thus limiting the flexibility of fusing feature maps [40]. In addition, according to ablation studies on PASCAL VOC 2007, concatenation can achieve better results than element-wise summation. Therefore, we use concatenation to perform feature map fusion.

As shown in Fig. 5, Conv4_Fu is a feature layer with the size of $38 \times 38 \times 512$ formed by the fusion of dense block (1), dense block (2) and Conv7. Since dense block (1) has a higher resolution and contains more details than dense block(2), we use atrous convolution to down-sample dense block (1). We set the parameters of the atrous convolution operations as: dilation $= 2, k = 3, s = 2$, and $p = 1$. We use deconvolution operation to up-sample Conv7 and set the parameters as: $k = 2, s = 2$, and $p = 0$.

The above operations make the feature map size of the dense block (1) and Conv7 the same as that of the dense block (2). In order to further study the object features, we carry out convolution operation on each of the three modules. The parameters of this convolution operation are: $k = 3, s = 1$, and $p = 1$. Since there is a large gap between the data dimension distribution of the low-dimensional feature layer and the high-dimensional feature layer, the direct fusion effect is not good, so the Batch Normalization layer is added for normalization processing. The three feature maps are activated before concatenation process. Finally, the convolution operation with convolution kernel size of 1×1 is used to reduce the dimension of the fusion module to generate the final feature fusion layer. The other feature fusion layers Conv7_Fu, Conv8_Fu, and Conv9_Fu have similar fusion processes.

2) RESIDUAL PREDICTION MODULE

SSD use a set of convolutional filters at each feature layer for detection to produce a fixed set of predictions. For a feature layer with a size of m × n with p channels, use the convolution kernel of $3 \times 3 \times$ p to perform the convolution operation to obtain a category score or a shape offset relative to the default box coordinates.

MS-CNN [41] points out that improving the subnetwork of each task can improve the accuracy. Following this principle, we improve the convolution predictor of SSD. That is, by adding a residual block, or ResBlock, to each prediction layer, so that the gradient of the loss function does not flow directly into the backbone network. Residual prediction block allows us to apply 1×1 convolution kernel to predict category scores and box offsets. Our experiments show that the use of ResBlock reduces the computational cost while the detection accuracy is improved. The structure of ResBlock is shown in Fig. 6.

C. TRAINING SETTINGS

1) TRAINING STRATEGIES

All our models are train from scratch by the deep learning framework Caffe [42] on NVIDIA Titan X GPU. The advantage of training from scratch is that it is not necessary to rely on the pre-training model on classification dataset to initialize the network as most models do. After all, classification and detection are different visual tasks. In the training process, we can fully consider the characteristics of the detection datasets to avoid the impact of other datasets on the network initialization. Following DSOD, since each scale of DF-SSD feature fusion modules is all concatenated from multiple resolutions, L2 normalization technique [43] is also adopted here to scale the feature norm to 20 on all output.

Most of our training strategies follow SSD, including scale and aspect ratios for default boxes, loss functions, data augmentation (e.g., randomly sample a patch and flip horizontal), and so on. It is worth noting that the latest SSD result, SSD300* [8], includes a random expansion data augmentation strategy that has proven very useful for detecting small targets, and we also adopt this strategy in DF-SSD framework. We adopt the hard negative mining technique of SSD so that the ratio between positive and negative samples is at most 3:1, which lead to faster optimization and more stable training. Other parameter settings such as learning rate and mini-batch size will be specified in the experiment section.

2) LOSS FUNCTION

The Loss function is the weighted sum of the localization loss (loc) and the confidence loss (conf):

$$L(x, c, l, g) = \frac{1}{N}\left(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)\right) \quad (4)$$

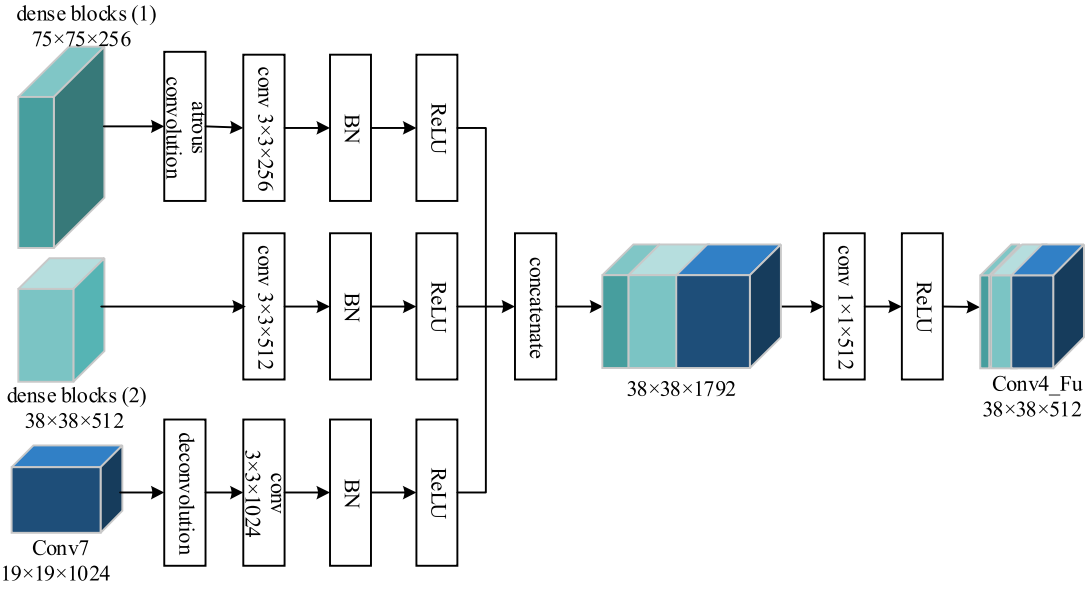


FIGURE 5. Multi-layer fusion of the conv4_Fu layer.

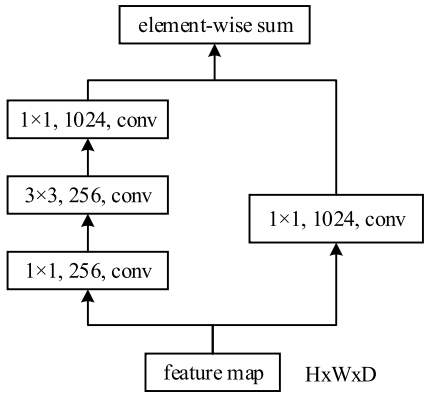


FIGURE 6. ResBlock.

where: N is the number of positive samples of the default boxes, that is, the default boxes that can be successfully matched with the real box. If N = 0, wet set the loss to 0. x_{ij}^{p} is an index parameter. When $x_{ij}^{p} = 1$, it means that the $i-$th default box matches the $j-$th ground truth box of category p.

The localization loss is a Smooth L1 loss between the predicted box (l) and the ground truth box (g) parameters. Similar to Faster R-CNN, we regress to offsets for the center (cx, cy) of the default bounding box (d) and for its width (w) and height (h).

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \sum_{m \in \{cx,cy,w,h\}} x_{ij}^{k} smooth_{L1}\left(l_i^m - \hat{g}_j^m\right) \quad (5)$$

$$\hat{g}_j^{cx} = \left(g_j^{cx} - d_i^{cx}\right) \Big/ d_i^{w} \quad \hat{g}_j^{cy} = \left(g_j^{cy} - d_i^{cy}\right) \Big/ d_i^{h}$$
$$\hat{g}_j^{w} = \log\left(g_j^{w} \Big/ d_i^{w}\right) \quad \hat{g}_j^{h} = \log\left(g_j^{h} \Big/ d_i^{h}\right) \quad (6)$$

$$\text{Where, } smooth_{L1}(x) = \begin{cases} 0.5x^2 & |x| < 1 \\ |x| - 0.5 & otherwise \end{cases} \quad (7)$$

The confidence loss is the softmax loss over multiple classes confidences (c).

$$L_{conf}(x, c) = -\sum_{i \in Pos}^{N} x_{ij}^{p} \log\left(\hat{c}_i^{p}\right) - \sum_{i \in Neg} \log\left(\hat{c}_i^{0}\right) \quad (8)$$

where, $\hat{c}_i^{p} = \exp\left(c_i^{p}\right) \Big/ \sum_{p} \exp\left(c_i^{p}\right)$ and the weight term α is set to 1 by cross validation. We use Stochastic Gradient Descent (SGD) to optimize the loss function to find the optimal solution.

IV. EXPERIMENTS

We conduct experiments on PASCAL VOC 2007, PASCAL VOC 2012 and MS COCO datasets and evaluate state-of-the-art architectures. These datasets have 20, 20, and 80 target categories, respectively.

We train our models from scratch on these common datasets with 0.9 momentum, 0.0005 weight decay, and initial learning rate 0.1. The learning rate decay policy is slightly different for each dataset, and we will describe details later. The batch size we used is 128, which beyond the GPU memory capacity. Therefore, we refer to the training trick of DSOD implemented on Caffe platform to overcome GPU memory constraints by accumulating gradients over two training iterations. All conv-layers are initialized with the "xavier" method [44]. Model detection performance is mainly evaluated with mean Average Precision (mAP). Other indicators such as Frame Per Second (FPS) and parameters will also help us further evaluate model performance.

TABLE 2. Ablation study on PASCAL VOC 2007 test set.

Method	Data	Pre-train	Transition W/O Pooling	K=16	K=32	Stem Block	Eltw-sum	Concat	Residual Prediction Module	#Parameters	mAP
DF-SSD300	07+12	×	×	√	×	×	×	×	×	7.7M	64.3
DF-SSD300	07+12	×	√	√	×	×	×	×	×	8.1M	67.2
DF-SSD300	07+12	×	√	×	√	×	×	×	×	11.5M	71.5
DF-SSD300	07+12	×	√	√	×	√	×	×	×	8.3M	68.1
DF-SSD300	07+12	×	√	×	√	√	×	×	×	11.7M	73.4
DF-SSD300	07+12	×	√	×	√	√	√	×	×	13.5M	75.7
DF-SSD300	07+12	×	√	×	√	√	×	√	×	14.9M	76.6
DF-SSD300	07+12	×	√	×	√	√	×	√	√	**15.2M**	**78.9**
DF-SSD300	07+12+COCO	×	√	×	√	√	×	√	√	**15.2M**	**81.4**

A. ABLATION STUDY ON PASCAL VOC 2007

We first conduct ablation study on the PASCAL VOC 2007 datasets and analyze the experimental results of each component and design module of our DF-SSD detector. The experimental results are shown in TABLE 2. All experiments are performed in a consistent setting, except for the inspection of certain components or structures. We train the models on VOC 2007 trainval and 2012 trainval ("07 + 12") joint training set (16,551 images), and test on the VOC 2007 test set (4,952 images).

1) TRANSITION W/O POOLING LAYER

From TABLE 2 (rows 2 and 3), we can see that compared with the case without transition w/o pooling layer (64.3% mAP), the case with this design layer (67.2% mAP) brings about a performance improvement of 2.9% mAP, so as to verify the effectiveness of this layer. That is, increasing the number of dense blocks without decreasing the resolution of the final feature maps.

2) GROWTH RATE

When k = 16, the detection accuracy of the model is 67.2% mAP. While k = 32, the detection accuracy is improved to 71.5% mAP. On the one hand, DenseNet shows that smaller growth rate can also produce better results. On the other hand, according to DSOD, we have reason to believe that a larger growth rate (like 48) can produce better model performance. But in order to avoid the network becoming very wide and for the sake of computing costs, we do not set a higher growth rate. So we set the growth rate of DF-SSD at 32.

3) STEM BLOCK

As can be seen from TABLE 2, the stem block improves the performance of the model by 0.9% mAP (67.2% vs. 68.1%) at k = 16, and by 1.9% mAP (71.5% vs. 73.4%) at k = 32. This proves the usefulness of this component, that is, using the stem block can reduce the loss of information in the original input image.

4) FEATURE FUSION MODULE

The element-wise summation and concatenation fusion methods increase the model performance by 2.3% mAP and 3.2% mAP, respectively (rows 6, 7, and 8), and concatenation is better than element-wise summation with a margin of 0.9 points. Therefore, we chose concatenation as the feature fusion method of DF-SSD.

5) RESIDUAL PREDICTION MODULE

By adding the residual prediction module, the model performance increase from 76.6% mAP to 78.9% mAP, which is an improvement of 2.3%. After adding COCO datasets for training, the module performance is further improved.

B. RESULTS ON PASCAL VOC 2007

We train the models on VOC 2007 trainval and 2012 trainval ("07 + 12") joint training set, and test on VOC 2007 test set. We set the initial learning rate to 0.1 and divide it by 10 after each 20k iterations until the number of training iterations reaches 80k. In order to keep the gradient size of each layer almost the same, we use "Xavier" method to initialize all convolution layers. By using this initialization method, the gradient explosion or dispersion of the last convolution layer can be avoided. The results of some of state-of-the-art detectors and our detectors on VOC 2007 test set are shown in TABLE 3.

1) ANALYSIS OF EXPERIMENTAL RESULTS

By replacing the backbone network VGGNet with DenseNet-S-32-1, DF-SSD improve detection accuracy by 3.8% compared with SSD300S† [13] (73.4% vs. 69.6%), while the detection speed decrease by half. SSD300S†is a model that is trained from scratch using VGGNet. It can be seen that DenseNet has strong feature reuse and extraction ability.

It is worth noting that by adding feature fusion modules using concatenation, DF-SSD is 3.2% higher than DF-SSD without feature fusion modules (76.6% vs. 73.4%). The effectiveness of the feature fusion method propose in this paper

TABLE 3. PASCAL VOC 2007 test detection results.

Method	Data	Backbone Network	Input Size	Pre-train	Feature Fusion Module	Residual Prediction Module	GPU	#Parameters	mAP	FPS
Faster R-CNN[10]	07+12	VGGNet	~600×1000	√	-	-	Titan X	134.7M	73.2	7
R-FCN[19]	07+12	ResNet-101	~600×1000	√	-	-	K40	50.9M	79.5	5.8
YOLOv2[9]	07+12	Darknet-19	352×352	√	-	-	Titan X	-	73.7	81
SSD300S†[13]	07+12	VGGNet	300×300	×	-	-	Titan X	26.3M	69.6	46
SSD300*[8]	07+12+COCO	VGGNet	300×300	√	-	-	Titan X	26.3M	81.2	46
SSD300[8]	07+12	VGGNet	300×300	√	-	-	Titan X	26.3M	75.8	46
FSSD300S†[27]	07+12	VGGNet	300×300	×	-	-	1080Ti	-	72.7	65.8
DSSD321[31]	07+12	ResNet-101	321×321	√	-	-	Titan X	-	78.6	9.5
DSSD513[31]	07+12	ResNet-101	513×513	√	-	-	Titan X	-	81.5	5.5
DSOD300[13]	07+12	DS/64-192-48-1	300×300	×	-	-	Titan X	14.8M	77.7	17.4
DSOD300*[13]	07+12+COCO	DS/64-192-48-1	300×300	×	-	-	Titan X	14.8M	81.7	17.4
DF-SSD300	07+12	DenseNet-S-32-1	300×300	×	×	×	Titan X	11.7M	73.4	20.5
DF-SSD300	07+12	DenseNet-S-32-1	300×300	×	√	×	Titan X	14.9M	76.6	14.7
DF-SSD300	07+12	DenseNet-S-32-1	300×300	×	√	√	Titan X	**15.2M**	**78.9**	**11.6**
DF-SSD300	07+12+COCO	DenseNet-S-32-1	300×300	×	√	√	Titan X	**15.2M**	**81.4**	**11.6**

is proved. Compare with FSSD300S† [27] (72.7% mAP) with VGGNet as the backbone network, the detection accuracy of DF-SSD is 3.9% higher. FSSD300S† refer to the idea of FPN and adopt concatenation for lightweight fusion of multi-scale feature maps, while DF-SSD is based on DenseNet-S-32-1 network and integrate more context information, so the detection speed decrease.

After adding the residual prediction module, the model detection performance of DF-SSD is improved from 76.6% mAP to 78.9% mAP, which verifies the validity of the residual prediction module. Our overall model improved by 3.1% (78.9% vs. 75.8%) compared to SSD300 [8], and by 1.2% and 0.3% compared to DSOD300 [13] (77.7% mAP) and DSSD321 [31] (78.6% mAP), respectively. The feature fusion module of DSSD321 only uses deconvolution to integrate the semantic information of high-level feature maps into low-level feature maps, which proves that the bi-directional fusion method of DF-SSD is more effective.

After adding COCO datasets for training, the model performance of DF-SSD is further improved to 81.4% mAP. Slightly better than SSD300* [8] (81.2% mAP). SSD300* is the latest SSD results with the new expansion data augmentation trick, which are already better than many other state-of-the-art detectors. DF-SSD decreased by 0.3% compared to DSOD300* [13] (81.7% mAP), due to the growth rate k adopted by DSOD300* is 48, and we set the growth rate of DF-SSD to 32 for consideration of computing costs and to avoid the network becoming too wide.

2) RUNTIME ANALYSIS

Column 11 in TABLE 3 shows the detection rates of different models. With 300 × 300 input, the processing speed of our model DF-SSD (rows 13) is 20.5 FPS on a single Titan X GPU, 14.7 FPS with the feature fusion module, and 11.6 FPS with the residual prediction structure. As a comparison, R-FCN [19] runs at 5.8 FPS with the backbone network ResNet-101. The processing speeds of DSSD321 and SSD300 are 9.5 FPS and 46 FPS, respectively. DF-SSD (15.2M) uses about only 1/2 parameters to SSD300 (26.3M) with VGGNet, 1/9 to Faster R-CNN [10] (134.7M) with VGGNet and 1/4 to R-FCN (50.9M) with ResNet-101.

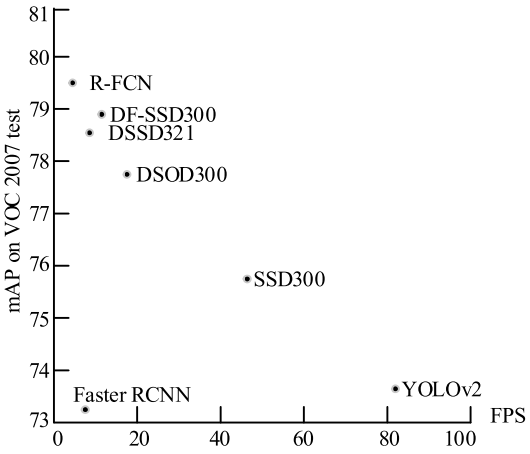


FIGURE 7. Speed and accuracy distribution with different object detection algorithms.

The distribution of detection accuracy and speed of different object detection algorithms on VOC 2007 test set is shown in Fig. 7. It can be seen intuitively that DF-SSD is slightly better than DSSD321 in both detection accuracy and speed. Compared with SSD300, the detection accuracy is improved

TABLE 4. (a) PASCAL VOC 2012 test detection results. (b) PASCAL VOC 2012 test detection results.

(a)

Method	Data	Backbone Network	Pre-train	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow
ION[45]	07+12+S	VGGNet	√	76.4	87.5	84.7	76.8	63.8	58.3	82.6	79.0	90.9	57.8	82.0
Faster R-CNN[10]	07++12	ResNet-101	√	73.8	86.5	81.6	77.2	58.0	51.0	78.6	76.6	**93.2**	48.6	80.4
R-FCN[19]	07++12	ResNet-101	√	**77.6**	86.9	83.4	**81.5**	63.8	**62.4**	81.6	**81.1**	93.1	58.0	**83.8**
SSD300*[8]	07++12	VGGNet	√	75.8	88.1	82.9	74.4	61.9	47.6	82.7	78.8	91.5	58.1	80.0
SSD321[8]	07++12	ResNet-101	√	75.4	87.9	82.9	73.7	61.5	45.3	81.4	75.6	92.6	57.4	78.3
DSSD321[31]	07++12	ResNet-101	√	76.3	87.3	83.3	75.4	64.6	46.8	82.7	76.5	92.9	59.5	78.3
DSOD300[13]	07++12	DS/64-192-48-1	×	76.3	89.4	85.3	72.9	62.7	49.5	**83.6**	80.6	92.1	60.8	77.9
DF-SSD300	07++12	DenseNet-S-32-1	×	76.5	**89.5**	**85.6**	72.6	**65.8**	51.3	82.9	79.9	92.2	**62.4**	77.5
DF-SSD300	07++12+COCO	DenseNet-S-32-1	×	**79.0**	**90.4**	**86.9**	76.3	**68.2**	59.2	**84.2**	81.0	92.6	**64.3**	81.4

(b)

Method	Data	Backbone Network	Pre-train	mAP	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
ION[45]	07+12+S	VGGNet	√	76.4	64.7	88.9	86.5	84.7	82.3	51.4	78.2	69.2	85.2	73.5
Faster R-CNN[10]	07++12	ResNet-101	√	73.8	59.0	92.1	85.3	84.8	80.7	48.1	77.3	66.5	84.7	65.6
R-FCN[19]	07++12	ResNet-101	√	**77.6**	60.8	**92.7**	86.0	84.6	84.4	**59.0**	**80.8**	68.6	86.1	72.9
SSD300*[8]	07++12	VGGNet	√	75.8	64.1	89.4	85.7	85.5	82.6	50.2	79.8	73.6	**86.6**	72.1
SSD321[8]	07++12	ResNet-101	√	75.4	65.0	90.8	**86.8**	85.8	81.5	50.3	78.1	75.3	85.2	72.5
DSSD321[31]	07++12	ResNet-101	√	76.3	64.3	91.5	86.6	86.6	82.1	53.3	79.6	**75.7**	85.2	**73.9**
DSOD300[13]	07++12	DS/64-192-48-1	×	76.3	**65.6**	88.9	85.5	**86.8**	84.6	51.1	77.7	72.3	86.0	72.2
DF-SSD300	07++12	DenseNet-S-32-1	×	76.5	64.5	89.5	85.4	86.4	**85.7**	51.9	77.8	72.6	85.1	71.6
DF-SSD300	07++12+COCO	DenseNet-S-32-1	×	**79.0**	65.1	**92.9**	86.3	**88.2**	**87.9**	56.8	80.2	74.2	**88.0**	**75.9**

07+12+S: 07+12 plus segmentation labels

while the speed is greatly decreased. The detection speed of DF-SSD is faster than that of R-FCN, which has the highest detection accuracy.

C. RESULTS ON PASCAL VOC 2012

For the VOC 2012 dataset, we use VOC 2012 trainval and VOC 2007 trainval + test (''07 + +12'') (21,503 images) for training, and test on VOC 2012 test set (10,991 images). For the first 40k iterations, the initial learning rate is set to 0.1 and then divided by 10 after every 20k iteration. The total number of iterations is 100k. The experimental results of some of state-of-the-art detectors and our detectors on VOC 2012 test set are shown in TABLE 4.

DF-SSD achieves 76.5% mAP, which is better than SSD300∗ (75.8% mAP). After adding COCO datasets for training, our model performance is improved to 79.0% mAP, 1.4% higher than R-FCN (77.6% mAP). DF-SSD shows a large improvement in testing tasks with specific backgrounds and small targets. For example, airplane (89.5% mAP), boat (65.8% mAP), chair (62.4% mAP), person (85.7% mAP), etc. This shows that DF-SSD improves the weakness of SSD for small target detection to some extent, and achieves better performance for classes with specific context semantic relationship.

D. RESULTS ON MS COCO

Finally, our DF-SSD model is evaluated on the MS COCO datasets. MS COCO detection task contains a total of 80 categories, and the data distribution includes 80k training images, 40k validation images and 20k testing images (test-dev set). We use the trainval set for training and evaluate the results from test-dev 2015 evaluation server. Compared with PASCAL VOC, there are more small targets in COCO datasets, more objects in a single image, and most objects are not center-distributed, which is more consistent with the daily environment. Therefore, COCO detection task is more difficult. For the first 60k iterations, we set the initial learning rate to 0.1 and divide it by 10 after each 40k iterations. The training is completed when the number of iterations reaches 300k. The test results of DF-SSD on MS COCO test-dev 2015 are shown in TABLE 5.

Although the input image size of SSD300* is smaller than that of Faster R-CNN and ION [45], the detection performance of SSD300* is better. After a series of improvements to the original SSD model, DF-SSD achieves 29.5/50.7%, superior to SSD300* and DSOD300, and close to R-FCN. We observe that our [0.5:0.95] result is 0.3% higher than R-FCN, but our result (50.7% mAP) with 0.5 IoU is lower

TABLE 5. MS COCO test-DEV 2015 detection results.

Method	Data	Backbone Network	Pre-train	Avg. Precision, IoU:			Avg. Precision, Area:			Avg. Recall, #Dets:			Avg. Recall, Area:		
				0.5:0.95	0.5	0.75	S	M	L	1	10	100	S	M	L
Faster R-CNN[10]	trainval	VGGNet	√	21.9	42.7	-	-	-	-	-	-	-	-	-	-
ION[45]	train	VGGNet	√	23.6	43.2	23.6	6.4	24.1	38.3	23.2	32.7	33.5	10.1	37.7	53.6
R-FCN[19]	trainval	ResNet-101	√	29.2	**51.5**	-	**10.3**	**32.4**	43.3	-	-	-	-	-	-
SSD300*[8]	trainval35k	VGGNet	√	25.1	43.1	25.8	6.6	25.9	41.4	23.7	35.1	37.2	11.2	40.4	58.4
SSD321[8]	trainval35k	ResNet-101	√	28.0	45.4	29.3	6.2	28.3	**49.3**	25.9	37.8	39.9	11.5	43.3	64.9
FSSD300[27]	trainval35k	VGGNet	√	27.1	47.7	27.8	8.7	29.2	42.2	24.6	37.4	40.0	15.9	44.2	58.6
DSSD321[31]	trainval35k	ResNet-101	√	28.0	46.1	29.2	7.4	28.1	47.6	25.5	37.1	39.4	12.7	42.0	62.6
DSOD300[13]	trainval	DS/64-192-48-1	×	29.3	47.3	30.6	9.4	31.5	47.0	**27.3**	40.7	**43.0**	16.7	**47.1**	**65.0**
DF-SSD300	trainval	DenseNet-S-32-1	×	**29.5**	50.7	**31.3**	9.8	31.1	46.5	27.1	**41.5**	42.7	**17.3**	46.8	64.4

than R-FCN (51.5% mAP). This indicates that the object position prediction of DF-SSD is more accurate than that of R-FCN under larger overlap settings.

DF-SSD's small object detection accuracy is slightly lower than R-FCN (9.8% vs. 10.3%), because DF-SSD's input image size (300 × 300) is much smaller than R-FCN's ($\sim 600 \times 1000$). However, DF-SSD is better than other detection models in the table, which proves that the feature fusion module we designed is effective in small object detection. Moreover, the large object detection accuracy of DF-SSD is 3.2% mAP higher than that of R-FCN (46.5% vs. 43.3%), which further demonstrates the effectiveness of the whole DF-SSD model.

E. VISUALIZATION

In Fig. 8, we show some detection examples of the SSD and DF-SSD model on COCO datasets. Compared to SSD, our DF-SSD model improves mainly in two aspects.

The first aspect is in a scenes that contains small or dense objects, as shown in Fig. 8 (a). SSD algorithm does not work well on small targets, but DF-SSD shows obvious improvement. On the one hand, compared with large objects, the position information of small targets is more likely to be lost during the detection process. On the other hand, the recognition of small targets depends more on their surroundings. Since SSD only detects smaller objects from shallow layers such as conv4_3, whose receptive field is too small to observe the object's context information, which results in bad detection performance of the SSD on smaller objects.

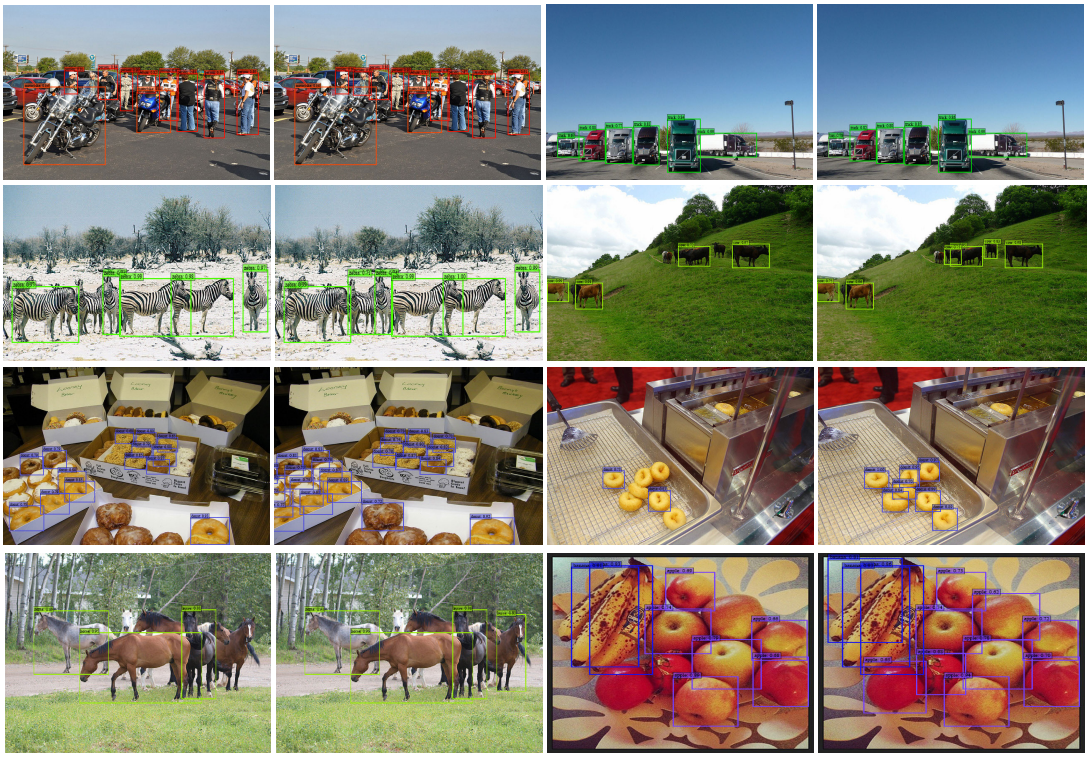
The second aspect is some classes with different contexts. Compared to SSD, DF-SSD can capture scene contexts. In Fig. 8 (b), we can see that the results of classes with specific relationships can be improved: baseball player and baseball bat, skateboard and jumping people, men in suit and tie, and football and football player, etc. In Fig. 8, we can observe all the objects that benefit from the feature fusion module of DF-SSD detector.

V. DISCUSSION

We train our models from scratch. Our DF-SSD detector is only trained with 16,551 images on VOC 2007, but achieves competitive or even better performance than those models trained with 1.2 million + 16,551 images. We also acknowledge that given a modest assumption of unlimited training data and computing power, deep neural networks should perform very well. But as datasets get larger, training deep neural networks will become more expensive. Moreover, most of the pre-trained models may have huge domain differences from the pre-trained mode domain to the target domain. It is very difficult to apply the pre-trained model on ImageNet to medical images, multi-spectral images and other fields.

Model fine-tuning limits the structural design space of the object detection networks. This is critical for deploying deep neural networks models into esource-limited Internet-of-Things scenario. However, our model DF-SSD only uses about only 1/2 parameters to SSD300, 1/9 to Faster R-CNN, and 1/4 to R-FCN, which shows great application potential in low-end devices such as embedded electronic products and mobile phones. All in all, the way of training from scratch is very interesting and deserves our in-depth study and discussion.

We compare the performance of DF-SSD, SSD and DSOD on VOC 2007 from three indicators, including detection accuracy, parameters, and speed. Compared with SSD, the detection accuracy of DF-SSD increased from 75.8% mAP to 78.9% mAP, the parameters decreased from 26.3M to 15.2M, while the processing speed decreased from 46 FPS to 11.6 FPS. Compared with the state-of-the-art model DSOD300, which is also trained from scratch, the mAP of DF-SSD increased from 77.7% to 78.9%, the parameters is basically close (14.8M vs. 15.2M), and the processing speed decreased from 17.4 FPS to 11.6 FPS. The reason is that our model DF-SSD introduce bidirectional fusion module and residual prediction module, which makes the detection accuracy of DF-SSD improved, but the detection speed is not ideal. How to design the structure of the model so that it has higher real-time performance is worth pondering.



(a) Dense Scenes: Compared to SSD, DF-SSD has better detection performance on dense scenes and small objects.

(b) Context Scenes: DF-SSD also performs well in some classes with specific relationships.

FIGURE 8. **(a) Dense scenes. (b) Context scenes. Detection examples on COCO test-dev with SSD and DF-SSD. A score threshold of 0.6 is used for displaying. Each color corresponds to an object category. For each pair of images, the left side is the detection result of SSD, and the right side is the detection result of DF-SSD.**

VI. CONCLUSION

The DF-SSD we proposed is an improved SSD object detection algorithm based on DenseNet, feature fusion, and residual prediction module. The way of training from scratch enables us to use less training data to achieve more competitive performance compared to other pre-training models. Our DF-SSD can achieve advanced performance on three common datasets with real-time processing speed and more compact models. Moreover, DF-SSD shows good detection effects for small objects and objects with specific relationships.

REFERENCES

[1] L. Hu and Q. Ni, "IoT-driven automated object detection algorithm for urban surveillance systems in smart cities," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 747–754, Apr. 2018.

[2] A. Mhalla, T. Chateau, S. Gazzah, and N. E. B. Amara, "An embedded computer-vision system for multi-object detection in traffic surveillance," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 11, pp. 4006–4018, Nov. 2019.

[3] Y. Wu, B. Jiang, and N. Lu, "A descriptor system approach for estimation of incipient faults with application to high-speed railway traction devices," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 10, pp. 2108–2118, Oct. 2019.

[4] Y. Wu, B. Jiang, and Y. Wang, "Incipient winding fault detection and diagnosis for squirrel-cage induction motors equipped on CRH trains," *ISA Trans.*, to be published, doi: 10.1016/j.isatra.2019.09.020.

[5] M. Liu, J. Jiang, and Z. Wang, "Colonic polyp detection in endoscopic videos with single shot detection based deep convolutional neural network," *IEEE Access*, vol. 7, pp. 75058–75066, 2019.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Lake Tahoe, NV, USA, Dec. 2012, pp. 1097–1105.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 248–255.

[8] 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. (ECCV)*. Cham, Switzerland: Springer, 2016, pp. 21–37.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 779–788.

[10] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Montreal, QC, Canada, Dec. 2015, pp. 91–99.

[11] K. Lee, J. Choi, J. Jeong, and N. Kwak, "Residual features and unified prediction network for single stage detection," 2017, *arXiv:1707.05031*. [Online]. Available: https://arxiv.org/abs/1707.05031

[12] Z. Wang, B. Du, and Y. Guo, "Domain adaptation with neural embedding matching," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published.

[13] 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, ITA, Oct.*, vol. 2017, pp. 1919–1927.

[14] 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, Jun. 2016, pp. 770–778.

[15] 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)*, Honolulu, HI, USA, Jul. 2017, pp. 4700–4708.

[16] J. R. R. Uijlings, K. E. A. Van De Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, Sep. 2013.

[17] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Zürich, Switzerland: Springer, Sep. 2014, pp. 391–405.

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Columbus, OH, USA, Jun. 2014, pp. 580–587.

[19] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Barcelona, Spain, Dec. 2016, pp. 379–387.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition,"2014, arXiv: 1409.1556. [Online]. Available: https://arxiv.org/abs/1409.1556

[21] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 1–9.

[22] H. Cui and Z. Wei, "Multi-scale receptive field detection network," *IEEE Access*, vol. 7, pp. 138825–138832, 2019.

[23] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Montreal, QC, Canada, Dec. 2015, pp. 2377–2385.

[24] J. R. Gardner, M. J. Kusner, Y. Li, P. Upchurch, K. Q. Weinberger, and J. E. Hopcroft, "Deep manifold traversal: Changing labels with convolutional features," 2015, *arXiv:1511.06421*. [Online]. Available: https://arxiv.org/abs/1511.06421

[25] L. A. Gatys, A. S. Ecker, and M. Bethge, "A neural algorithm of artistic style," 2015, *arXiv:1508.06576*. [Online]. Available: https://arxiv.org/abs/1508.06576

[26] G. Cao, X. Xie, W. Yang, Q. Liao, G. Shi, and J. Wu, "Feature-fused SSD: Fast detection for small objects," *Proc. SPIE*, vol. 10615, Apr. 2018, Art. no. 106151E.

[27] Z. Li and F. Zhou, "FSSD: Feature fusion single shot multibox detector," 2019, *arXiv:1712.00960*. [Online]. Available: https://arxiv.org/abs/1712.00960

[28] P. O. Pinheiro, T. Y. Lin, R. Collobert, and P. Dollár, "Learning to refine object segments," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Amsterdam, The Netherlands, Oct. 2016, pp. 75–91.

[29] 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)*, Las Vegas, NV, USA, Jun. 2016, pp. 2117–2125.

[30] J. Jeong, H. Park, and N. Kwak, "Enhancement of SSD by concatenating feature maps for object detection," May 2017, *arXiv:1705.09587*. [Online]. Available: https://arxiv.org/abs/1705.09587

[31] C. Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "DSSD: Deconvolutional single shot detector," 2017, *arXiv:1701.06659*. [Online]. Available: https://arxiv.org/abs/1701.06659

[32] S. Liu, D. Huang, and Y. Wang, "Receptive field block net for accurate and fast object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Munich, Germany, Sep. 2018, pp. 385–400.

[33] Y. Li, Y. Chen, N. Wang, and Z. Zhang, "Scale-aware trident networks for object detection," Jan. 2019, *arXiv:1901.01892*. [Online]. Available: https://arxiv.org/abs/1901.01892

[34] S.-W. Kim, H.-K. Kook, J.-Y. Sun, M.-C. Kang, and S.-J. Ko, "Parallel feature pyramid network for object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Munich, Germany, Sep. 2018, pp. 234–250.

[35] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[36] H. Pan, G. Chen, and J. Jiang, "Adaptively dense feature pyramid network for object detection," *IEEE Access*, vol. 7, pp. 81132–81144, 2019.

[37] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Salt Lake City, UT, USA, Jun. 2018, pp. 6848–6856.

[38] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-V4, inception-resnet and the impact of residual connections on learning," in *Proc. AAAI*, San Francisco, CA, USA, Feb. 2017, pp. 4278–4284.

[39] X. Ying, Q. Wang, X. Li, M. Yu, H. Jiang, J. Gao, Z. Liu, and R. Yu, "Multi-attention object detection model in remote sensing images based on multi-scale," *IEEE Access*, vol. 7, pp. 94508–94519, 2019.

[40] L. Zhang, Q. Zhang, L. Zhang, D. Tao, X. Huang, and B. Du, "Ensemble manifold regularized sparse low-rank approximation for multiview feature embedding," *Pattern Recognit.*, vol. 48, no. 10, pp. 3102–3112, Oct. 2015.

[41] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Amsterdam, The Netherlands, Oct. 2016, pp. 354–370.

[42] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. ACM*, San Francisco, CA, USA, Nov. 2014, pp. 675–678.
[43] W. Liu, A. Rabinovich, and A. C. Berg, "ParseNet: Looking wider to see better," 2015, *arXiv:1506.04579*. [Online]. Available: https://arxiv.org/abs/1506.04579
[44] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. AISTATS*, Sardinia, Italy, Mar. 2010, pp. 249–256.
[45] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick, "Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 2874–2883.

SHEPING ZHAI was born in 1971. He received the Ph.D. degree in computer systems architecture from Xi'an Jiaotong University, in 2010. He is currently an Associate Professor and the Deputy Dean of the School of Computer Science, Xi'an University of Posts and Telecommunications. His main research interests include semantic web, computer vision, and artificial intelligence. He became a member of China Computer Federation (77328M), in 2016.

DINGRONG SHANG was born in 1994. She received the B.S. degree in computer science and technology from the Baoji University of Arts and Sciences, in 2017. She is currently pursuing the M.S. degree in computer application technology with the Xi'an University of Posts and Telecommunications. Her research interests include machine learning and computer vision.

SHUHUAN WANG was born in 1996. She received the B.S. degree in network engineering from Shanxi Agricultural University, in 2018. She is currently pursuing the M.S. degree in computer application technology with the Xi'an University of Posts and Telecommunications. Her research interests include object detection and video target tracking.

SUSU DONG was born in 1996. He received the B.S. degree in network engineering from the Xi'an University of Posts and Telecommunications, in 2018, where he is currently pursuing the M.S. degree with the major of computer system architecture. His research interests include machine learning and image processing.