*Yiqing Sun, et. al. International Journal of Engineering Research and Applications*
*www.ijera.com*
*ISSN: 2248-9622, Vol. 11, Issue 4, (Series-I) April 2021, pp. 05-17*

RESEARCH ARTICLE OPEN ACCESS

# AnimprovedSSDbasedonfeature fusion and attention

Yiqing Sun, Zhiying Yang
*CollegeofInformationEngineering, Shanghai Maritime University, Shanghai 201306*

**ABSTRACT**
SSD (Single Shot Multibox Detector) is one of the most popular object detection detectors with high precision and fast speed. However, SSD's feature pyramid detection method makes it hard to detects small objects. In this paper, we proposed an improved SSD named FFASSD, which includes a lightweight feature fusion module andanefficientchannelattentionmodule. FFASSD can especially improve the performance of SSDin small object detection. In the feature fusion module, features from different layers with different scales are concatenated together, the features of shallow layers are replaced by the generated new features to predict the final detection results. In the channel attention block, capturing local cross-channel interaction before the multi-scale feature maps being classified and regression. On the Pascal VOC 2007 test, FFASSD can achieve 79.8mAP (mean average precision) at the speed of45.5 FPS (frame per second) with the input size 300×300. In addition, the experimental result on COCO is also better than the conventional SSD with a large margin. FFASSDoutperforms a lot of state-of-the-art object detection detectors in both aspects of accuracy and speed.
***Keywords*–**Object detection, SSD, feature mapfusion, attention mechanism

---

Date of Submission: 26-03-2021 Date of Acceptance: 09-04-2021

---

## I. INTRODUCTION

Objectdetectionisoneofthemost fundamentaltasksincomputervision. In recent years, plentyof object detectors based on CNN withbetterperformance havereplacedtraditionones inobjectdetection field.Theobject detection detectorsbasedonCNN can be roughly divided into two categories: one-stage detectors and two-stage detectors. The two-stage detectors'detection task was composedof two stages. In the first stage, heuristicsalgorithm(selective search[8]) or CNN (RPN[2]) is used to generate candidate bounding boxes (Region Proposal). In the second stage,the selected candidate regionsare used to perform classification and location regression. Although the detection accuracy is higher than the one-stage detector, two-stage detectorneeds much more time cost. The representative ones are R-CNN[1], Fast-RCNN [2]and Faster-RCNN[3] proposed by Girshicket al.The one-stage detectorsplitsinputimageintolotsofcells, and only usesoneCNNto locate and classify the objects existed in each cell on the image. Therefore, the speed of detection is usually faster, but the precision of one-stage detectoris lower than that of the two-stage detector.The representative one-stagedetectors includeYOLO[4], SSD[5], etc.

However, itishardtofindabalancefromtheobjectdetectorswhichisbasedonCNN betweenrecognition and locationowningtothe contradiction betweendeepConvlayersand shallow Convlayers.The feature maps producedby shallow network havemore detailed information to locate objects. The feature mapsgenerated from deep network obtain more semantic information which is morehelpful to classification of targets but lose too much location information. In order to solve this problem, SSD detectoradopts multi-scale feature maps to detect objects. It makes VGG16[6] as the backbone network, the shallow layer Conv4_3 is usedto predict small objects, and the deep layer Conv8_2 is responsible for detecting large targets. This strategy seemslogicalbecausetheshallowfeaturemapcanprovidemorelocation tohelp objects be well located and well recognized. Butthefeaturemapsofshallowlayerstill are lack of semantic information, which leads to poor performance onthedetectionofsmall objects. Besides, small objects also rely on the context information heavily [7]. Although many modificationalgorithmshave been proposed to improve the detection accuracy of SSD in small objects, the running speed ofdetection is greatly slowed down. It is difficult to find a balance between precision and speed for the detector.

In order to improve the detection accuracy for small objectsand the speed is as fast aspossible. In this paper we proposed an improved SSD with feature fusionandchannelattention,named FFASSD

(Feature Fusion and Attention SSD).Firstly, to effectively combine the location information with the semantic information, we construct two feature fusion modules, which concatenated the feature maps from shallow layers and deep layers to form new feature maps. Secondly, we designed an attention module based on the local channels to catch the interaction information inthechanneldimensionofthefeaturemapsand learn the importance of thefeatures among each channel by assigning weight to the channels of the feature map. The experimental results show that FFASSD improves the detection ability of small objects greatly.Besides, itachieves a large performance improvement with sacrificing a small part of the speed.

## II. RELATED WORK

2.1. Objectdetection with deeplearning

The object detection is not only to locate each object in the image, but also to classify the recognized target. With the development of deep learning, object detector based on CNNhas begun to show the dramatic improvements in efficiency. R-CNN [1] appliesselective search [8] or Edge boxes [9] to generate the region proposals, which are used to generate the region-based feature from a pretrained CNN and SVMs are adopted to do classification. SPPNet [10] uses a spatial pyramid pooling layer which allows the classification module to reuse the ConvNetfeature regardless of the input image resolutions. Fast R-CNN [2] introduces to train the ConvNet with both the classification and location regression loss end to end. Faster R-CNN [3] suggests replacing selective search with a region proposal network (RPN).RPN is used to generate the candidate bounding boxes (anchor boxes) and filter out the background regions at the same time. Then another small network is used to do classification and bounding box location regression based on these proposals. R-FCN [11] replaces ROI pooling in the Faster RCNN with a position sensitive ROI pooling (PSROI) to improve the detector's quality with both aspects of accuracy and speed. Recently, Deformable Convolutional Network [12] proposes deformable convolution and deformable PSROI to enhance the RFCN further with better accuracy.

Except for the two-stage detectors, there are also some efficient one-stage object detectors. YOLO (you only look once) [4] divides the input image into several grids and performs localization and classification on each part of the image. Benefited from this method, YOLO can run object detection at a very highspeed, but the accuracy is not satisfactory enough. YOLOv2 [13]and YOLOv3[14] areboth enhanced versions of YOLO.

SSD [5] is another efficient one stage object detector. As illustrated in Fig.1(a), SSD predicts the class scores and location offsets for the default bounding boxes by two $3\times3$ convolutional layers. In order to detect objects with different scales, SSD adds a series of progressively smaller convolutional layers to generate pyramid feature maps and sets corresponding anchor size according to the receptive field size of the layers. Then NMS (non-maximum suppression) is used to post-process the final detection results. Because SSD detects objects directly from the plane ConvNet feature maps, it can achieve real-time object detection and process faster than most of the other state-of-the-art object detectors. In order to improve the accuracy, DSSD [15] suggests augmenting SSD+ResNet-101 with deconvolution layers to introduce additional large-scale context. However, the speed is slow because of the model complexity. RSSD [16] uses rainbow concatenation through both pooling and concatenation to fully utilize the relationship between the layers in the feature pyramid to enhance the accuracy with a little speed lost. DSOD [17] investigates how to train an object detector from scratch and designs a DenseNet[41] architecture to improve the parameter efficiency. FSSD[18] was proposed to improve the accuracy of small object, features from different layers with different scales are concatenated together, followed by some down-sampling blocks or bilinear interpolationblocks to generate new feature pyramid, which will be fed to multi-box detectors to predict the final detection results.In order to generate features with strong representational power for small object instances, MDSSD[19] add the high-level features with rich semantic information to the low-level features via deconvolution Fusion Block.

2.2. Feature fusion module

There are a lot of approaches which attempt to use multiple layers' features to improve the performance of computer vision tasks. HyperNet [20], Parsenet [21], ION and FSSD [18]concatenate features from multiple layers before predicting the result. MDSSD [19] use element-wisesum to combine deep layers whichusing deconvolution withshallow layers. FCN [22], U-Net [23] and Stacked Hourglass networks [24] also use skip connections to associate low-level and high-level feature maps to fully utilize the synthetic information. SharpMask [25] and FPN [26] introduce top-down structure to combine the different level features together to enhance the performance.

2.3. Attention mechanism

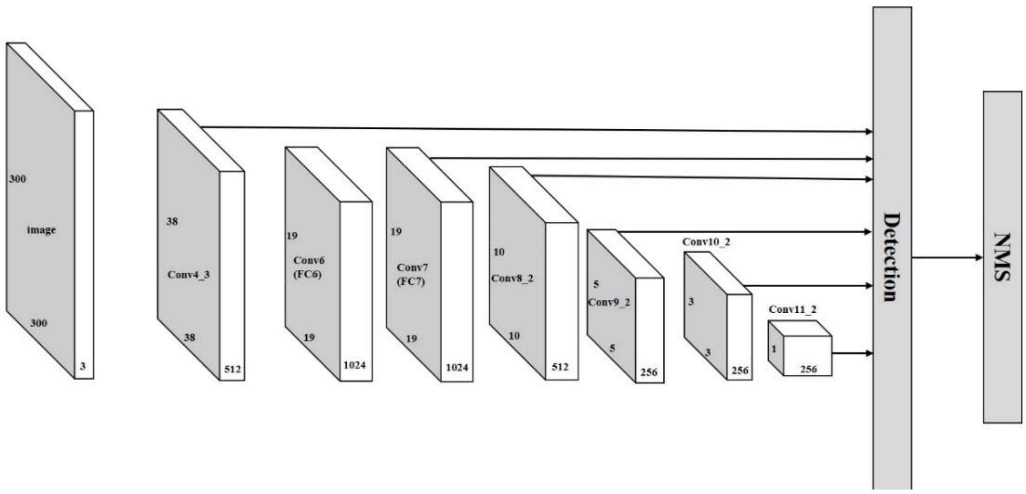
Attention mechanism has proven to be a potential means to enhance deep CNNs. SE-Net [27] presents for the first time an effective mechanism to learn channel attention and achieves promising performance. GSoP [28] introduces a second-order pooling for more effective feature aggregation. GE [29] explores spatial extension using a depth-wise convolution to aggregate features. CBAM [30] and scSE [31] compute spatial attention using a 2D convolution, then combine it with channel attention. Sharing similar philosophy with Non-Local (NL) neural networks [32], GC-Net [33] develops a simplified NL network and integrates with the SE block, resulting in a lightweight module to model long-range dependency. Double Attention Networks (A2-Nets) [34] introduces a novel relation function for NL blocks for image or video recognition. Dual Attention Network (DAN) [35] simultaneously considers NL-based channel and spatial attentions for semantic segmentation. ECA [36] aims at learning effective channel attention with low model complexity.All the above methods focus on developing sophisticated attention modules for better performance.

## III. ARCHITECTURE OF FFASSD

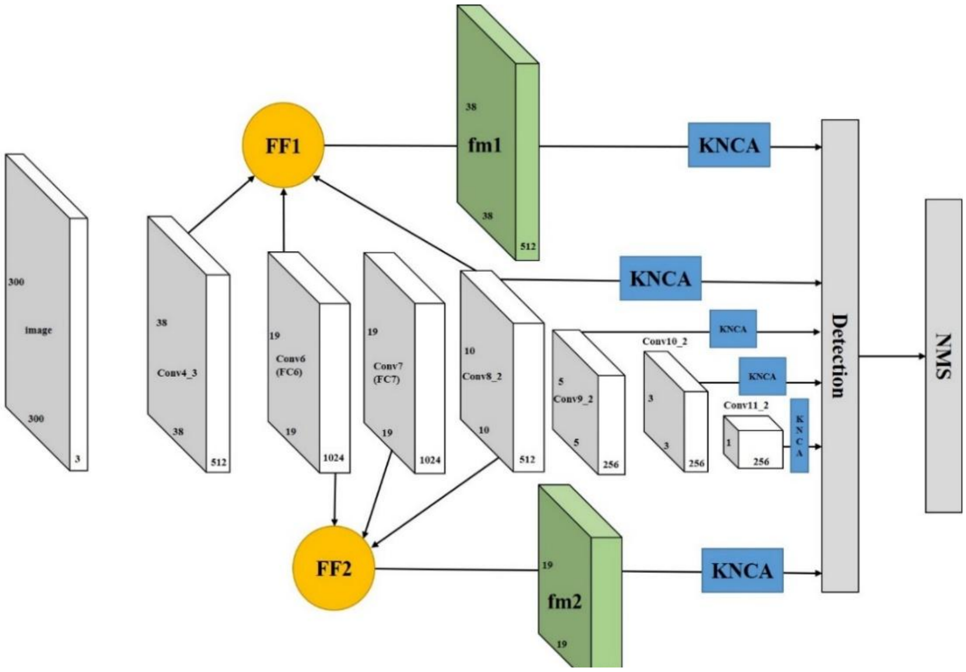
ThearchitectureofFFASSDisshownin Fig. 1 (b). Based on SSD, there are two modulesbeing added as follow:

Wedesigntwolightweightfeaturefusionmodules. The first Feature Fusion module (FF1) generate new feature map (fm1)by combiningconv4_3,conv6 and conv8_2. The second Feature Fusion module(FF2) produce new feature map (fm2) by merging conv6, conv7 and conv8_2.The new fusion features are rich in semantic information with relatively high resolution, providing a significant improvement on detection of small objects.

Weinterceptachannelattentionblockbeforethemulti-scalefeaturemapsbeingclassifiedandregression.We propose a local cross-channel interaction strategy without dimensionality reduction, which can be efficiently implemented via 1D convolution with an adaptively select kernel size, determining coverage of local cross-channel interaction.Throughthe proposed attention block, our network can learn to use global information to selectively emphasize informative features and suppress less useful ones, which improve the accuracywithjustalittlespeeddrop.




(a)




(b)

**Figure 1.** (a)is the SSD framework, (b) is our FFASSD framework.

3.1 FeatureFusionModule

As shown in Fig. 2, there are some approaches which have been proposed to solve the multi-scale objects detection problem. A top-down structure like (a) in Fig.2 is popular and has been proved working well in FPN [26], DSSD [15], and SharpMask [25]. But fusing features layer by layer is not efficient enough while there are many layers to be combined together. FSSD [18]proposed a method like(b) in Fig.2 that features from different layers with different scales are concatenated together first and used to generate a series of pyramid features later.We adopted the structure like (c) in Fig.2, feature maps of shallow network were replaced by thefusionresultsofmulti-scale features.However, features of deep network whichtookpartinthe featurefusion still used toclassificationandregressiondirectly.

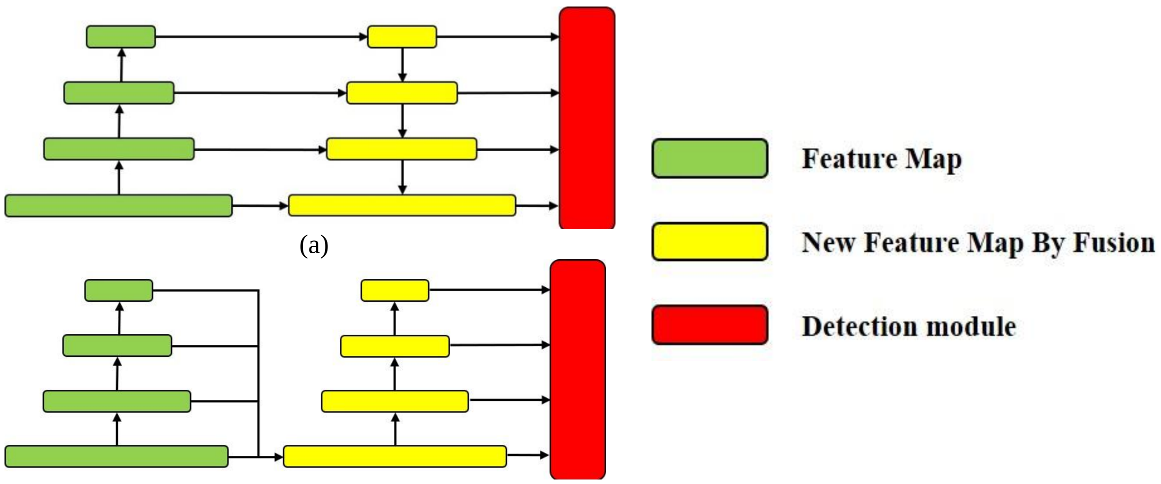
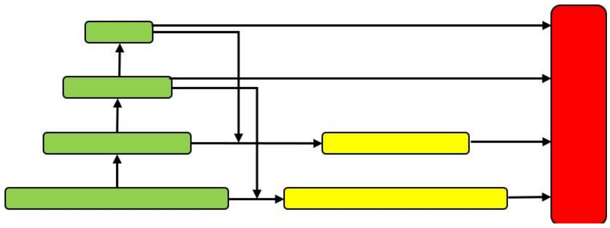



**Figure 2.** (a) Feature fusion method adopted by



(c)

[15, 25, 26], features are fused from top to bottom layer by layer. (b) Method in[18], features from different layers with different scales are concatenated together first and used to generate a series of pyramid features later. (c) Our proposed feature fusion and feature pyramid generation method.

There are mainly two ways to merge different feature maps together: concatenation and element-wise summation. Element-wise summation requires that feature maps should have the same size which means we have to convert the feature maps to the same channels. Because this requirement limits the flexibility of fusing feature maps, we prefer to use concatenation. According to FSSD, concatenation can get a better result than element-wise summation[18]. So, we use concatenation to merge the features.

In order to concatenate the features with different scales in a simple and efficient way, we adopt the following strategy. In the first feature fusion module (FF1) shown in Fig.3 (a), conv$1\times1$ is applied to each of the source layers to reduce the dimension of feature channel firstly. We set the size of conv4_3's feature map as the basic feature map size.As for the feature maps conv6 and conv8_2 whose size is smaller than $38\times38$, we use bilinear interpolation to resize the feature maps to the same size with conv4_3. Then the three feature maps are concatenated together followed by a Batch Normalization layer to normalize the feature values. After Batch Normalization we use the conv $3\times3$ layer to reduce the channel, make the feature dimension is same with the basic feature map.

The second feature fusion module like (b) in Fig.3, is basically the same as FF1, except that the input is changed from conv4_3, conv6 and conv8_2 to conv6, conv7 and conv8_2. The dimension of output is same with feature map conv6.

Assuming $X_i \quad i\in C$ are the source feature maps we want to fuse the feature fusion module can be described as follows:

$$Y_i = Conv_{1\times1}(X_i) \quad i\in C \quad (1)$$

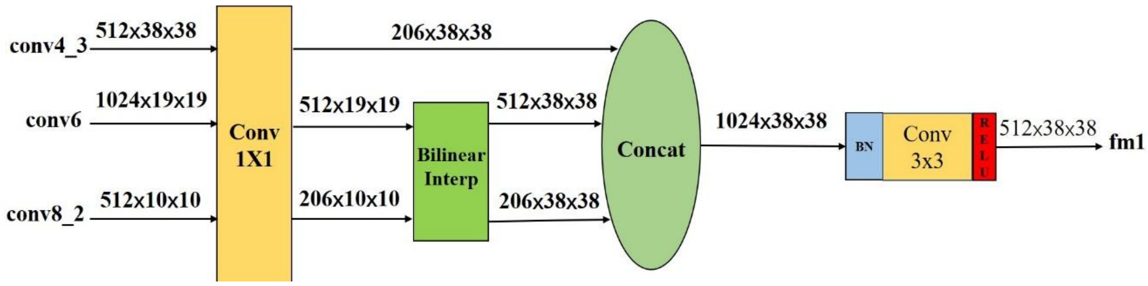
$C$ istherangeofinput, depending on the different feature fusion modules.In the Feature Fusion Module one (FF1), we set the range of input layers as conv4_3, conv6 and conv8_2, In FF2, the collectionofinput is conv6, conv7 and conv8_2. $Conv_{1\times1}$ is a $1\times1$ convolutionlayertoreducethechanneldimension:

$$Z_i = \begin{cases} Y_i & size(i) = basic\ size \\ BI(Y_i) & size(i) < basic\ size \end{cases} \quad i\in C \quad (2)$$

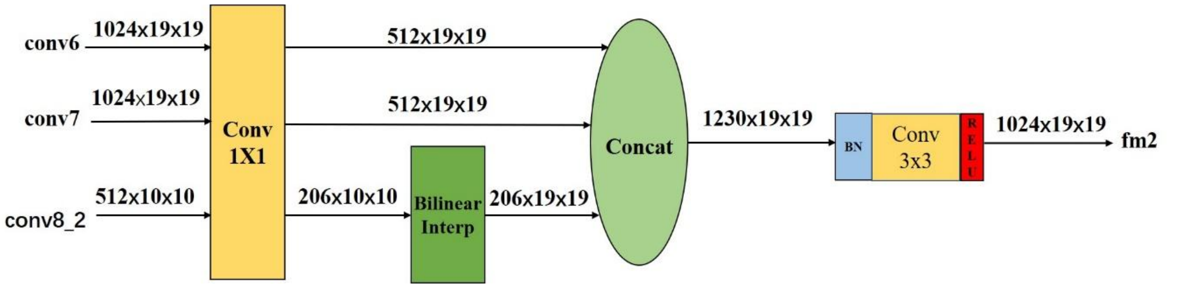
$BI$ means the operation of bilinear interpolation for up-sampling to resize smaller features. $Y_i$ doesn'tneedup-sampling,if $i$ is same size with the feature chosen to be base. Then all the features $Z_i$ have the same size on spatial dimension. Sowe can use them to fusion by concatenation:

$$F_{1,2} = Conv_{3X3}\left(BN\left(Concat\left(\{Z_i\}\right)\right)\right) \quad i\in C \quad (3)$$

$Concat$ is the operation of concatenation. $BN$ is a Batch Normalization layer. The transformed feature maps $Z_i$ are concatenated together followed by a Batch Normalization layer to normalize the feature values. Then we use a $3\times3$ convolution layer as a feature extractor which is same with SSD to generate new feature maps. We use the new feature maps replacing old basic feature maps in SSD to produce object detection results.




(a)




(b)

**Figure 3.** (a) is feature fusion module 1(FF1), (b) is feature fusion module 2 (FF2).

3.2 ChannelAttentionModule

The concatenation introducedin the feature fusion module of the design in section 3.1 causes the feature maps to be combined only in the channel dimension. But the information between cross-channelis still independent of each other. This sectionintroducedthechannel attention mechanism which can learn to use global information to selectively emphasize informative features and suppress less useful ones.

The traditional channel attention mechanism SE Block [27], shown in Fig.4(a),first employs a global average pooling for each channel independently, then two fullyconnected (FC) layers with non-linearity followed by a Sigmoid function are used to generate channel weights. The two FC layers are designed to capture non-linear cross-channel interaction, which involve dimensionality reduction for controlling model complexity. Although this strategy is widely used in subsequent channel attention modules [28, 29, 30], several empirical studies in ECA [36]have shownthat dimensionality reduction brings side effect on channel attention prediction, and it is inefficient and unnecessary to capture dependencies across all channels.

Therefore, this paper proposes a K-neighbor Channel Attention (KNCA) module, which avoids dimensionality reduction and captures cross-channel interaction in an efficient way. After channel-wise global average pooling without dimensionality reduction, KNCA module captures local cross-channel interaction by a 1D conv which considering every channel and its k neighbors.

Let the output of one convolution block be $x \in \mathbb{R}^{W \times H \times C}$, where $W$, $H$ and $C$ are width, height and channel dimension (i.e., number of filters). Accordingly, the weights of channels in KNCA module can be computed as:

$$W = \sigma\left(f_W\left(gap(x)\right)\right) \quad x \in \mathbb{R}^{W \times H \times C} \quad (4)$$

where $\sigma$ is a Sigmoid function, $f_W$ is a function to capture the local interaction information between each channel and $gap(x) = \frac{1}{WH}\sum_{i=1,j=1}^{W,H} X_{ij}$ is channel-wise global average pooling (GAP). Let $y = gap(x)$, $f_W$ takes the form:

$$f_W(y) = Wy \quad (5)$$

We design $W$ as $C \times C$ in equation5 to replace two fully connected (FC) layers in SE Block, which cause huge increase in model complexity and computational burden.

$$\begin{bmatrix} w_1^1 & w_1^2 & \cdots & w_1^C \\ w_2^1 & w_2^2 & \cdots & w_2^C \\ \vdots & \vdots & \ddots & \vdots \\ w_C^1 & w_C^2 & \cdots & w_C^C \end{bmatrix} \quad (6)$$

In order to reduce model complexityfurther avoiding dimensionality reduction.We remake W like equation7. Clearly, W in equation7involves $k \times C$ parameters, which is usually less than those of equation 6.

$$\begin{bmatrix} w_1^1 & \cdots & w_1^k & 0 & 0 & \cdots & 0 \\ 0 & w_2^1 & \cdots & w_2^k & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & \cdots & w_C^{C-k+1} & \cdots & w_C^C \end{bmatrix} \quad (7)$$

As for equation7, the weight of $y_i$ is calculated by only considering interaction between $y_i$ and its $k$ neighbors, i.e.:

$$W_i = \sigma\left(\sum_{j=1}^{k} w_i^j y_i^j\right) \quad y_i^j \in \Omega_i^k \quad (8)$$

where $\Omega_i^k$ indicates the set of $k$ adjacent channels of $y_i$.

A more efficient way is to make all channels share the same learning parameters, i.e.:

$$W_i = \sigma\left(\sum_{j=1}^{k} w^j y_i^j\right) \quad y_i^j \in \Omega_i^k \quad (9)$$

Note that such strategy can be readily implemented by a fast 1D convolution with kernel size of k, i.e.:

$$W = \sigma\left(Conv1d_k\left(y\right)\right) \quad (10)$$

where $Conv1d_k$ indicates 1D convolution with kernel size of $k$. Here, the method in equation10 is called by K-neighbor Channel Attention (KNCA) module, which only involves $k$ parameters. As presented in table 1ofsection5.1, theSSD with KNCA moduleachieves similar results compared withthe SSDappendingSE block while having much lower model complexity, which guarantees both efficiency and effectiveness by appropriately capturing local cross-channel interaction.

Fig. 4(b) illustrates the overview of KNCA module. After using GAP (global average pooling layer) to aggregate convolution features, KNCAmodule adaptively determines kernel size k and perform 1D convolution replacingthe FC layers in SE Block, which not required dimensionality reduction. Then we gain the collection of per-channel modulation weights by a Sigmoid function. These weights are applied to the feature maps to generate the output of the KNCA module which can be fed directly into subsequent layers of the network.

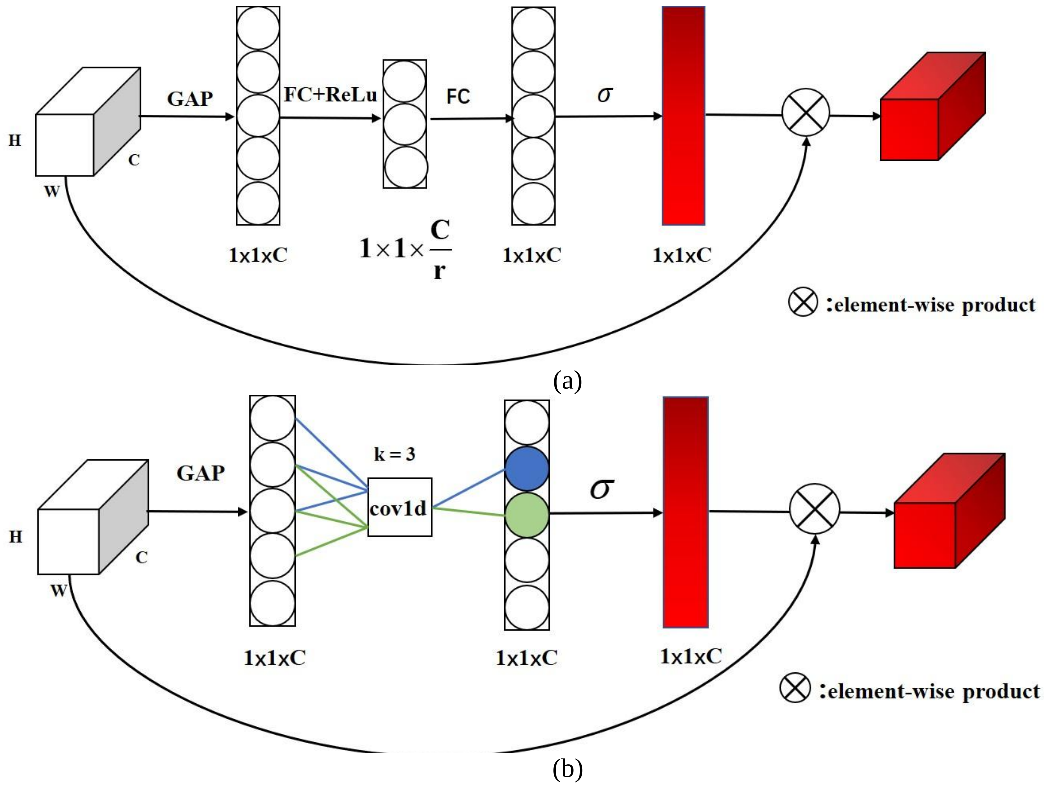



**Figure 4.** (a) is SE block proposed in [27], (b) is our KNCA block

Since KNCA module aims at appropriately capturing local cross-channel interaction, so the coverage of interaction (i.e., kernel size $k$ of 1D convolution) needs to be determined. The optimized coverage of interaction could be tuned manually for convolution blocks with different channel numbers in various CNN architectures. However, manual tuning via cross-validation will cost a lot of computing resources. Group convolutions have been successfully adopted to improve CNN architectures [37], where high-dimensional (low-dimensional) channels involve long range (short range) convolutions given the fixed number of groups. Sharing the similar philosophy, it is reasonable that the coverage of interaction (i.e., kernel size $k$ of 1D convolution) is proportional to channel dimension C. In other words, there may exist a mapping $\phi$ between $k$ and $C$:

$$C = \phi\left(k\right) \quad (11)$$

The simplest mapping is a linear function i.e.: $C = r * k + b$ However,therelations characterized by linear function are too limited. On the other hand, it is well known that channel dimension $C$ (i.e., number of filters) usually is set to power of 2. Therefore, we introduce a possible solution by extending the linear function $f\left(k\right) = r * k + b$ to a non-linear one, i.e.:

$$C = f\left(k\right) = 2^{\left(r*k+b\right)} \quad (12)$$

Then, given channel dimension $C$, kernel size $k$ can be adaptively determined by

$$k = \psi(C) = \left| \frac{log_2 C - b}{r} \right|_{odd} \quad (13)$$

where $|n|_{odd}$ indicates the nearest odd number of $n$. In this paper, we set $r$ and $b$ to2and 1 throughout all the experiments, respectively. Clearly, through the mapping $\psi$, high-dimensional channels have longer range interaction while low-dimensional ones undergo shorter range interaction by using a non-linear mapping.

## IV. TRAINING

We use two size of images( $300\times300$, $512\times512$) as inputandadopt the well-trained SSD model as our pre-trained model. The training objective is the same as SSD. We use the center code type to encode the bounding boxes and have the same matching strategy, hard negative mining strategy and data augmentation with SSD. A predicted bounding box is correct if its intersection over union (IOU) with the ground truth is higher than 0.5. The batch size is 32 for 120k iterations. The initial learning rate is set to 0.001 and then divided by 10 at step 80k and 100k. All the experiments are implemented in Pytorch 1.0 on the machine with two 1080Ti GPUs.

## V. EXPERIMENTS AND COMPARISON

In order to compare FFASSD with the conventional SSD fairly, our experiments are all based on VGG16[6] which is preprocessed like SSD [5]. We conduct experiments on PASCAL VOC 2007, 2012 [38] and MS COCO dataset [39]. The performance is measured by mean average precision (mAP) on VOC2007 test and COCO test-dev2015 datasets. We compare the results with state-of-the-art deep convolutional networks about the mAP and inference speed.

### 5.1 Ablation Study on PASCAL VOC2007

In this Section, we investigatetheinfluenceofFeatureFusionmoduleand ChannelAttentionblockonSSD. We compare the results on PASCAL VOC 2007 with input size 300×300. In these experiments, the models are trained with the combined dataset from 2007 *trainval* and 2012 *trainval* (VOC07+12) and tested on VOC 2007 *test* set. The results are summarized in Table 1.

In Table 1, we compare the SSD with different module. While we only use SE block, the mAP on VOC2007 test set (row 5) is 78.2%. It is interesting that if we only replace the SE block by KNCA block, the mAP is 78.1% (row 6), but the fps is fasterthan SE block with a large margin of 41 points. which means that the SSD with KNCA module achieves similar results compared with it appending SE Block while having much lower model complexity, which guarantees both efficiency and effectiveness by appropriately capturing local cross-channel interaction.The row 5 shown that adopting Feature Fusion module 1(FF1) andFeature Fusion module 2 (FF2) both can get 78.8%, which is higher than the results of SSD appending only FF1 or FF2.SSD with two Feature Fusion modules and KNCA block achieves the best performance, improving mAP to 79.8% (row 8).

**Table 1.**Ablation Study on PASCAL VOC2007.

| Feature Fusion Module 1 (FF1) | Feature Fusion Module 2 (FF2) | KNCA Block | SE Block | mAP(%) | speed(fps) |
|---|---|---|---|---|---|
| × | × | × | × | 77.5 | 120 |
| ✓ | × | × | × | 77.9 | 76 |
| × | ✓ | × | × | 78.4 | 76 |
| ✓ | ✓ | × | × | 78.8 | 65 |
| × | × | ✓ | × | 78.1 | 84 |
| × | × | × | ✓ | 78.2 | 43 |

| | | | | | |
|---|---|---|---|---|---|
| ✓ | ✓ | ✓ | × | **79.8** | 59.5 |

5.2 ExperimentalResults on VOC2007

Our results on VOC2007 *test* set are shown in Table 2. FFASSD300 can achieve 79.8% mAP, which improves 2.6 points compared with the conventional SSD300. In addition, the result of FFASSD300 is higher than DSSD321 and MDSSD300, itisnotedthat DSSD321 uses ResNet-101 [40] as the backbone network, which has better performance compared with VGG16. Comparing with FSSD300, FFASSD300 improves the performance with 1.2 pointswith just a little speed drop.FFASSD512 also improves the SSD512 from 78.5% to 81.5% with the aspect of accuracy, which is similar with DSSD512. FFASSD512 also exceeds RSSD512 by 1.2%, FSSD512 by 1.1% and MDSSD512 by 1.2%.

BesidesFFASSD300 is faster (59.5 fps) than most of the object detection algorithms except for FSSD300, because of FSSD300 only joinedone feature fusion module onthebaseofSSDwithout any attention block. FFASSDfounda justice balance between precision and speed.

**Table 2.**PASCAL VOC 2007 *test* detection results.

| method | train data | backbone network | GPU | speed(fps) | mAP(%) |
|---|---|---|---|---|---|
| Faster RCNN[3] | 07+12 | ResNet-101 | K40 | 2.4 | 76.4 |
| R-FCN[11] | 07+12 | ResNet-101 | K40 | 5.8 | 79.5 |
| YOLOv2[13] | 07+12 | Darknet-19 | Titan X | 81 | 73.7 |
| SSD300[5] | 07+12 | VGG16 | 1080Ti | 85 | 77.2 |
| SSD512[5] | 07+12 | VGG16 | 1080Ti | 19 | 78.5 |
| DSSD321[15] | 07+12 | ResNet-101 | Titan X | 9.5 | 78.6 |
| DSSD513[15] | 07+12 | ResNet-101 | Titan X | 5.5 | 81.5 |
| RSSD300[16] | 07+12 | VGG16 | Titan X | 35 | 78.5 |
| RSSD512[16] | 07+12 | VGG16 | Titan X | 16.6 | 80.8 |
| FSSD300[18] | 07+12 | VGG16 | 1080Ti | **65.8** | 78.8 |
| FSSD512[18] | 07+12 | VGG16 | 1080Ti | 35.7 | 80.9 |
| MDSSD300 [19] | 07+12 | VGG16 | 1080Ti | 38.5 | 78.6 |
| MDSSD512 [19] | 07+12 | VGG16 | 1080Ti | 17.3 | 80.3 |
| FFASSD300 | 07+12 | VGG16 | 1080Ti | 59.5 | 79.8 |
| FFASSD512 | 07+12 | VGG16 | 1080Ti | 26.4 | **81.5** |

5.3 Experimental Results on MS COCO

We use the MSCOCO data to prepare our dataset. The training set is the original trainval35k. We test FFASSDon the 2017*test-dev* set. The COCO test results are shown in Table 3.

FFASSD300 achieves 28.0% mAPon the *test-dev* set, which is higher than the SSD300(25.1%)andFSSD300 (27.1%).BesidesFFASSDperforms as well as DSSD300,it should be noted that FFASSD still takes VGG as the base network while DSSD swaps the backbone network for ResNet-101 which has a better performance than VGG.FFASSD512(32.8%)outperforms conventional SSD(28.8%) by 3 pointsandexceedsFSSD(31.8%) by 1 point. Even though FFASSD512 is slightly lower than DSSD513. But FFASSD's AP on small objects is still higher than among all of the object detectors in table 3 with a large margin. This performance proves the effectiveness of FFASSD for small objects.

**Table 3.**MS COCO *test-dev* 2017 detection results.

| Method | Train data | Backbone network | 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 RCNN[3] | trainval35k | ResNet-101 | 21.9 | 42.7 | - | - | - | - | - | - | - | - | - | - |
| R-FCN[11] | trainval35k | ResNet-101 | 29.9 | 51.9 | - | 10.8 | 32.8 | 45.0 | - | - | - | - | - | - |
| YOLOv2[13] | trainval35k | Darknet-19 | 21.6 | 44.0 | 19.2 | 5.0 | 22.4 | 35.5 | 20.7 | 31.6 | 33.3 | 9.8 | 36.5 | 54.4 |
| SSD300[5] | trainval35k | VGG16 | 25.1 | 43.1 | 25.8 | 6.6 | 25.9 | 41.4 | 23.7 | 35.1 | 37.2 | 11.2 | 40.4 | 58.4 |
| SSD512[5] | trainval35k | VGG16 | 28.8 | 48.5 | 30.3 | 10.9 | 31.8 | 43.5 | 26.1 | 39.5 | 42.0 | 16.5 | 46.6 | 60.8 |
| DSSD321[15] | 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 |
| DSSD513[15] | trainval35k | ResNet-101 | **33.2** | 53.3 | 35.2 | 13.0 | 35.4 | 51.1 | 28.9 | 43.5 | 46.2 | 21.8 | 49.1 | 66.4 |
| FSSD300[18] | trainval35k | VGG16 | 27.1 | 47.7 | 27.8 | 8.7 | 29.2 | 42.2 | 24.6 | 37.4 | 40.0 | 15.9 | 44.2 | 58.6 |
| FSSD512[18] | trainval35k | VGG16 | 31.8 | 52.8 | 33.5 | 14.2 | 35.1 | 45.0 | 27.6 | 42.4 | 45.0 | 22.3 | 49.9 | 62.0 |
| FFASSD300 | trainval35k | VGG16 | **28.0** | 48.0 | 28.8 | **10.8** | - | - | 26.3 | 38.6 | 40.8 | 16.8 | - | - |
| FFASSD512 | trainval35k | VGG16 | 32.8 | 53.5 | 34.4 | **15.9** | - | - | 28.3 | 43.3 | 45.9 | 24.4 | - | - |

## VI. PERFORMANCE IMPROVEMENT IN FFASSD

Our improved detector FFASSDperforms better than conventional SSD mainly in two aspects. Firstly, FFASSDperforms better on small objects. On the one hand, small objects can only activate smaller regions in the network compared with large objects and the location information is easy to be lost in the detection process. On the other hand, small object’s recognition relies more on the context around it. Because SSD only detects small objects from the shallow layers such as conv4_3, whose receptive field is too small to observe the morecontextinformationcompared with the deep layers, which leads to the SSD’s bad performance on small objects. FFASSD can observe all the objects synthetically benefited from the feature fusion module. As shown in Fig. 5 column 1and column 2. FFASSDdetects more small objects than SSD successfully. Secondly,FFASSDgains higher precision than SSD. For example, as illustrated in Fig.5 column 3, the SSD only detects twopersonsofthe three inthepicture. But FFASSDdetectsthemallatonce.

## VII. CONCLUSION AND FUTURE WORK

In this paper, we proposed an enhanced SSD by applying two lightweight feature fusion modules and an efficient channel attention module on it. Firstly, we replace outputs of conv4_3 and conv7 by two new feature maps generated by twolightweightfeaturefusionmoduleswhich combine three different scale features.Secondly, we intercept a channel attention block by capturing local cross-channel interaction before the multi-scale feature maps being classified and regression. Experiments on VOC PASCAL and MS COCOprove that FFASSDimproves the traditionalSSD a lot and outperforms several other state-of-the-art object detectors both in accuracy and efficiency with a simpleimprovement.

In order to improve the detection performance, it is imperative to replace VGG [6] by more effective networks, such as ResNet [40] and DenseNet [41]. But how to improve the inference speed of these deep backbones will be our future work. In addition, there are still some false detections in our visualized results. Some examples are given in Fig. 6. We will investigate these issues in our future work as well.

**Figure 5.** SSD300 vs FFASSD300. Both models are trained with VOC2007*test* set. The top row isthe results from the conventional SSD300 and the bottom row is from FFASSD300.

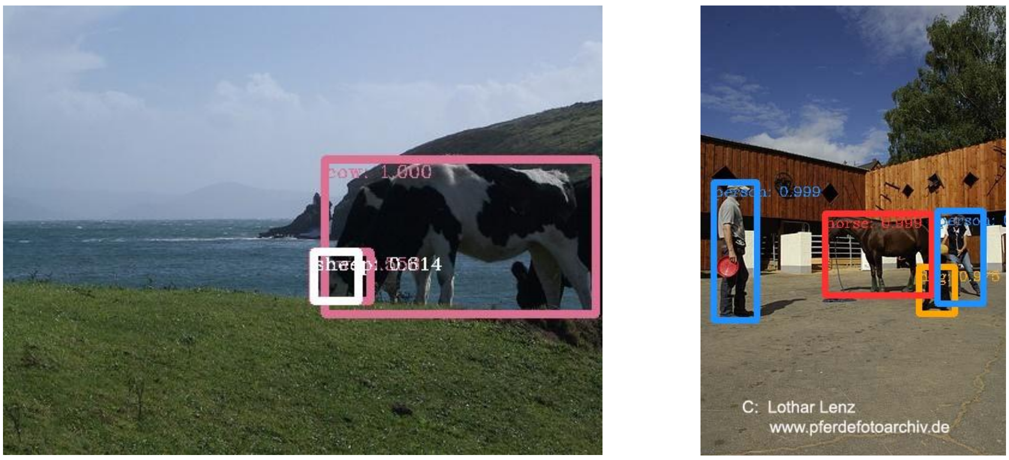



**Figure 6.** The false detections of FFASSD300 on VOC2007 *test* set

## REFERENCE

[1]. Ross G, Jeff D, Trevor D and Jitendra MFeature, hierarchies for accurate object detection and semantic segmentation,*Computer Vision and Pattern Recognition*, 2014.
[2]. Ross G, Fast r-cnn,*The IEEE International Conference on Computer Vision (ICCV), 2015*.
[3]. Shaoqing R, Kaiming H, Ross G, Jian S, Faster r-cnn: Towards real-time object detection with region proposal networks,*Advances in Neural Information Processing Systems*,ed C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, vol 28, 2015,p 91–99.
[4]. Redmon J, Divvala S, Girshick R andFarhadi A,You only look once: Unified, real-time object detection,*The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015,p 779-788.
[5]. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu and A. Berg,SSD: Single shot multibox detector,*LNCS of Lecture Notes in Computer Science*,vol 9905, 2016, p 21–37.
[6]. K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition,2014, *arXiv preprint arXiv:1409.1556*.
[7]. C. Chen, M.-Y. Liu, O. Tuzel and J. Xiao,R-CNN for small object detection,*Springer International*, 2017p 214–230.
[8]. J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers and A. W. M. Smeulders, Selective search for object recognition,*International Journal of Computer Vision*, vol 104(2), 2013, p 154–171.
[9]. L. Zitnick and P. Dollar, Edge boxes: Locating object proposals from edges,*European Conference on Computer Vision (ECCV)*, 2014.
[10]. K.He, X.Zhang, S.Ren and J.Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition,*IEEE transactions on pattern analysis and machine intelligence*, vol 37(9), 2015, p 1904–1916.
[11]. J. Dai, Y. Li, K. He and J. Sun, R-fcn: Object detection via region-based fully convolutional networks,*Advances in Neural Information Processing Systems*, ed D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, vol 29, 2016, p 379–387.
[12]. J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu and Y. Wei,Deformable convolutional networks *The IEEE International Conference on Computer Vision (ICCV), 2017*.
[13]. J. Redmon and A. Farhadi, Yolo9000: Better, faster, stronger *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
[14]. Redmon JandFarhadi A,YOLOv3:an incremental improvement*arXiv preprint arXiv:1804.02767*, 2018.
[15]. C. Fu, W. Liu, A. Ranga, A. Tyagi and A. C. Berg, DSSD: Deconvolutional single shot detector. *CoRR abs/1701.06659*, 2017.
[16]. J. Jeong, H. Park and N. Kwak, Enhancement of SSD by concatenating feature maps for object detection,*CoRR abs/1705.09587*, 2017.
[17]. Z. Shen, Z. Liu, J. Li, Y.-G. Jiang, Y. Chen and X. Xue,Dsod: Learning deeply supervised object detectors from scratch,*The IEEE International Conference on Computer Vision (ICCV)*, 2017.
[18]. ZuoXin L and FuQiang Z, FSSD: Feature fusion single shot multibox detector,*arXiv preprint arXiv:1712.00960v3*,2018.
[19]. Lisha C, Rui M, Pei L, Xiaoheng J, Zhimin G, Bing Z and Mingliang X,*Science China Information Sciences*,vol 63(2), 2020.
[20]. T. Kong, A. Yao, Y. Chen and F. Sun, Hypernet: Towards accurate region proposal generation and joint object detection,*Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, p 845–853.
[21]. W. Liu, A. Rabinovich and A. C. Berg,Parsenet: looking wider to see better,*arXiv preprint arXiv:1506.04579*, 2015.
[22]. J. Long, E. Shelhamer and T. Darrell, Fully convolutional networks for semantic segmentation,*The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
[23]. O. Ronneberger, P. Fischer and T. Brox, U-Net: convolutional networks for biomedical image segmentation,*Springer International Publishing*, 2015, p 234–241.
[24]. A. Newell, K. Yang and J. Deng, Stacked hourglass networks for human pose estimation,*Springer International Publishing*, 2016,p 483–499.
[25]. P. O. Pinheiro, T. Y. Lin, R. Collobert and P. Dollr, Learning to refine object

segments,*European conference on computer vision (ECCV)*, 2016.

[26]. T. Lin, P. Dolla ́r, R. B. Girshick, K. He, B. Hariharan and S. J. Belongie, Feature pyramid networks for object detection,*CoRR abs/1612.03144*, 2016.

[27]. J. Hu, L. Shen and G. Sun, Squeeze-and-excitation networks,*The IEEE Conference on Computer Vision and Pattern Recognition*(*CVPR)*, 2018.

[28]. Z. Gao, J. Xie, Q. Wang and P. Li, Global second-order pooling convolutional networks,*The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[29]. J. Hu, L. Shen, S. Albanie, G. Sun and A. Vedaldi, Gather-excite: Exploiting feature context in convolutional neural networks,*NeurIPS*, 2018.

[30]. S. Woo, J. Park, J. Lee and I. Kweon, CBAM: Convolutional block attention module,*European conference on computer vision*(*ECCV)*, 2018.

[31]. A. Roy, N. Navab and C Wachinger, Recalibrating fully convolutional networks with spatial and channel "squeeze and excitation" blocks,*IEEE Trans. Med. Imaging*, vol 38(2), 2019, p 540–549.

[32]. X. Wang, R. Girshick, A. Gupta and K. He, Non-local neural networks,*The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[33]. Y. Cao, J. Xu, S. Lin, F. Weiand H. Hu,Gcnet: Non-local networks meet squeeze-excitation networks and beyond,*The IEEE International Conference on Computer Vision (ICCV) Workshops*, 2019.

[34]. Y. Chen, Y. Kalantidis, J. Li, S. Yan and J. Feng, A2-Nets: Double attention networks,*NIPS*, 2018.

[35]. Jun F, Jing L, Haijie T, Yong L, Yongjun B, Zhiwei F and Hanqing L, Dual attention network for scene segmentation,*The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[36]. Qilong W, Banggu W, Pengfei Z, Peihua L, Wangmeng Z and Qinghua H,*IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[37]. Ting Z, Guo-Jun Q, Bin X, and Jingdong W, Interleaved group convolutions,*The IEEE International Conference on Computer Vision (ICCV)*, 2017.

[38]. Mark E, Luc Van G and Christopher K. I. W, The pascal visual object classes (voc) challenge,*International Journal of Computer Vision*, vol 88(2), 2010 p 303–338.

[39]. T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollr, and C. L. Zitnick, Microsoft coco: common objects in context,*European conference on computer vision (ECCV)*, 2014,p 740–755.

[40]. K. He, X. Zhang, S. Ren and J. Sun, Deep residual learning for image recognition,*Computer Vision and Pattern Recognition (CVPR)*, 2016.

[41]. G. Huang, Z. Liu, L. V. D. Maaten and K. Q. Weinberger, Densely connected convolutional networks,*CoRR* vol abs/1608.06993, 2016.