Journal of Real-Time Image Processing (2021) 18:2527–2538
https://doi.org/10.1007/s11554-021-01145-4

ORIGINAL RESEARCH PAPER

L-Net: lightweight and fast object detector-based ShuffleNetV2

Jin Han[1] · Yonghao Yang[1]

Received: 22 December 2020 / Accepted: 11 June 2021 / Published online: 19 June 2021

Abstract
Object detection algorithms based on deep learning have made continuous progress in recent years. On the premise of ensuring the accuracy of object detection, reducing model complexity and improving detection speed have always been the goals pursued by current object detection algorithms. A lightweight object detection model its backbone based on ShuffleNetV2 network structure named L-Net is presented in this paper. A suitable backbone network was obtained by changing from 3×3 depth convolution to 5×5 depth convolution and reducing the number of input channels. In order to obtain a more discriminative image feature description, Pyramid Pooling Module and Attention Pyramid Module are added after the backbone network. Experimental results show that the L-Net model only uses 1.54B FLOPs (floating point operations) to achieve 70.2% *mAP* (mean average precision) on PASCAL VOC2007 and 21.8% *mAP* on the MS COCO dataset. The model has achieved competitive results in terms of accuracy and speed while being lightweight.

Keywords Object detection · Lightweight · ShufflenetV2 · Convolutional neural network (CNN)

1 Introduction

Object detection is considered as a critical and challenging task in the field of computer vision. With the development of deep learning algorithms, many convolutional neural network based object detection frameworks have been proposed, such as YOLO [1–3], SSD [4], CornerNet [5], R-CNN [6] and their improved frameworks [7, 8], which greatly improve the accuracy of object detection.

From the point of view of network structure, the object detection model based on convolutional neural networks can be divided into two main parts, one is the backbone part responsible for feature extraction, and the other is the detection part responsible for feature fusion. As far as the backbone part is concerned, current mainstream object detection algorithms tend to make use of complex classification networks (e.g., ResNet-101 [9–11]) and large resolution input images(e.g.,800×1200), which will hinder the direct operation of object detection algorithms on resource-constrained devices. To address this limitation, a large number of lightweight algorithms have emerged. MobileNet-SSD uses MobileNet [12] as the backbone introduced into the SSD framework, and implements a model with only 5.5 M parameters and 1.14B FLOPs on the PASCAL VOC2007 dataset [13]. Pelee [14] uses PeleeNet as the backbone network. The detection part introduces the SSD framework and achieves 70.9% mAP with only 1.21B FLOPs on the PASCAL VOC2007 dataset. Tiny-YOLO which compresses the YOLO parameter size to 15 M and achieves a detection speed of over 200 fps on the PASCAL VOC2007 dataset. These network models largely reduce the computational resource requirements, but differ from the previous networks in terms of accuracy.

In terms of feature fusion, it is mainly to obtain useful feature information for specific tasks (semantic segmentation, object detection, object tracking, etc.) according to some rules, so that features have higher recognition degree and lower correlation degree, to reduce feature dimension, information redundancy and improve performance. For example, in [15], the verification performance of 2D cancelable palmprint code is improved using multi-orientation score level fusion. Zhang et al. [16] proposed a feature pyramid network structure to improve the accuracy of instance segmentation. Chu et al. [17] proposed an effective object detection algorithm based on MCFF (multi-layer convolution feature fusion) and OHEM (online hard

✉ Jin Han
shnk123@163.com

Yonghao Yang
goodlc_yyh@163.com

[1] College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266000, China

example mining), which greatly improves the detection accuracy of small objects and obscured objects. Besides, [18, 19] also prove the effectiveness of feature fusion in related tasks.

This paper gives the L-Net model based on ShuffleNetv2 [20] network. The model gives a backbone based on ShuffleNetV2 of a lightweight convolutional neural network. Pyramid Pooling module and Attention Pyramid module are added after the backbone network to remove redundant information in the feature map and reduce the computation amount of the model on the one hand, and multi-scale feature fusion is performed to improve the accuracy of the model on the other. The primary contributions of the paper are as follows:

- A lightweight backbone network is given based on the ShufflenetV2 network in this paper.
- To obtain a more discriminative image feature description, Attention Pyramid Module and Pyramid Pooling Module are added after the backbone network.
- To provide a solution for the implementation of object detection algorithms on resource-constrained devices.

2 Related work

2.1 Mainstream object detection algorithms

With the development of deep learning technology, many detection frameworks based on convolutional neural networks have emerged, which are mainly classified into one-stage object detectors and two-stage object detectors. The two-stage object detectors mainly include R-CNN, Fast R-CNN, Faster R-CNN, FPN, etc. The two-stage object detection model is mainly divided into two steps: the first step generates candidate regions that may contain objects, and the second step further classifies and calibrates the candidate regions to obtain the final detection results. In 2014, Girshick et al. were to apply a convolutional neural network to the classification operation in the R-CNN model, making great progress in accuracy, followed by continuous improvements [21, 22]. The R-CNN has improved accordingly in speed and candidate region selection, until Region Proposal Networks (RPN) combined candidate region selection and classification operations into a single convolutional network to form the Faster R-CNN framework, with a significant improvement in comprehensive performance, especially in detection speed. One-stage object detectors mainly include YOLO series [1–3], SSD [4], FCOS [23], RetinaNet [11], etc. They detect objects by regular and intensive sampling in position, scale, aspect ratio, which eliminates the regional proposal of the two-stage method, that is, there is no pre-classification and regression process, and directly identifies specific categories and regresses the borders. As an early classic single-stage object detection algorithm, OverFeat [24] realizes the same network framework for recognition, positioning, and detection. With the continuous progress of deep learning, single-stage object detection algorithms such as YOLO and SDD have made significant progress in detection speed. In view of the fact that the object detection algorithm usually has the characteristics of a large number of parameters and a large model volume, the research of its algorithm lightweight has always been the direction of the industry's attention.

2.2 Main methods of light-weighting

There are roughly the following methods to simplify the model: network pruning, low-rank decomposition, network quantification, knowledge distillation and lightweight network design.

Network pruning: Network pruning refers to the deletion of model parameters in the network to achieve the purpose of compressing the network [25]. This has two advantages: first, the reduction of parameters will bring about a corresponding reduction in storage space, and second, the reduction of parameters will lead to a reduction in the amount of calculations during network operation, which can reduce the demand for computing power and increase the operating speed.

Low-rank decomposition: The method of low-rank decomposition is mainly to decompose the convolution kernel to reduce redundancy. For example, transform a four-dimensional convolution kernel shape into $(w \times h \times c) \times n$, and then divide it into two parts. The first part is d convolution kernels of size $w \times h \times c$, and the other part is $n 1 \times 1 \times d$ convolution kernel, where $d < n$, using this method to experiment on VGG [26], theoretically can get three times the speed increase.

Network quantification: Network quantization is to reduce the storage space of weights by sacrificing parameter accuracy to reduce the size of the model. Matthieu Courbariaux et al. proposed a two-value weight network [27], that is, for the weight of the network, there are only two values of 1 or -1. Subsequently, Matthieu Courbariaux et al. again proposed a binary neural network [28], which further increased the degree of quantification and compressed the required calculation to a very low level, making the model easier to transplant to devices with limited computing resources.

Knowledge distillation: Knowledge distillation is to use the output of a previously trained complex model as a supervisory signal to train a simple network so that the simple network has similar accuracy to the complex model. Hinton first proposed the concept of knowledge distillation, allowing small networks to try to fit the soft classification results of different categories of large networks when learning classification objects [29]. Based on this concept, even better classification results than the original network can be obtained.

The above methods are all carried out based on the original network. Through operations such as pruning and quantification, the network can be lightweight and achieves ideal results. ShuffleNetv2, as a lightweight convolutional neural network proposed for practical applications, has shown outstanding performance in general tasks such as classification [20] and object detection [30]. This paper gives a lightweight backbone based on the ShuffleNetV2 network, the detection part consists of the Pyramid Pooling module and Attention Pyramid module. The network compresses the input channels in ShuffleNetV2 to reduce the computational volume and partially replaces the 3×3 depth convolution with a 5×5 depth convolution to expand the receptive field. While ensuring accuracy the lightweight of the network has been realized.

3 Network architecture

The lightweight object detection model L-Net is given in the paper, and the network structure is shown in Fig. 1. L-Net consists of three main parts, the first is the backbone network, which is an improvement on ShuffleNetV2, and the second is the detection part. The Pyramid Pooling Module takes the three feature maps obtained on the backbone network as separate inputs and then each feature mapping will select three suitable pooling kernels from the four optional pooling kernels to obtain feature information while removing some redundant information to generate a more discriminative feature map. The Attention Pyramid Module takes advantage of the high resolution of lower-level features, which contain more location and detail information, and the higher-level features, which have stronger semantic information, to fuse the two efficiently and enhance the learning of important channels by introducing an attention mechanism to output the final feature map. The third is the detector head, we refer to the detection head of the YOLOV3 algorithm and design it accordingly.

3.1 Backbone part and improvement

3.1.1 ShuffleNet

ShuffleNetv2 was proposed as an effective classification network in 2018. At the same time, its team combined theory and experiment, put forward four practical guidelines: (1) When the number of input and output feature channels of the convolutional layer is equal, the MAC is the smallest and the model speed is the fastest. (2) The group operation of convolution will increase the MAC, which will slow down the model. (3) The fewer the number of branches in the model, the faster the model speed. (4) The time consumption of element-wise operations is much greater than the value reflected in FLOPs, so element-wise operations should be minimized. Based on the above four criteria, the authors analyzed the shortcomings of the ShufflenetV1 design and improved it to produce ShuffleNetv2. A comparison of the two modules is shown in Fig. 2a, b is the module structure of ShufflenetV1, Fig. 2c, d is the module structure of Shufflenetv2, on which the network structure of Shufflenetv2 was built, as shown in Fig. 3a.

3.1.2 Improvements to shufflenet

The purpose of most convolution operations in CNN is to extract the features of the image. The size of the receptive field is very important because the image information is obtained from the receptive field. A large receptive field contains more contextual information and inter-pixels, which is very important for the localization of large objects. Previous papers [30, 31] also proved the effectiveness of the receptive field for semantic segmentation and object detection tasks, and through the learning of YOLO, SSD, RetinaNet, and other models, it was found that the backbone network, the positioning task is more difficult than the classification task. This indicates that the low-level features should be more adequate in the later feature fusion stage, but this does not mean that the high-level features are not important. Only when the two are fully combined can the ideal results be achieved in the object detection task. At the same time, it is found from the current lightweight backbone network that the input resolution of ShuffleNet [20, 32] MobileNet [12, 33, 34] limits the receptive field, and the lack of low-level features is not conducive to positioning in object detection tasks. Xception [35] under the premise of ensuring smaller FLOPs, the lack of high-level features affects the correct classification of objects.

Based on the solution to the above problems, some improvements are made to the lightweight convolutional neural network

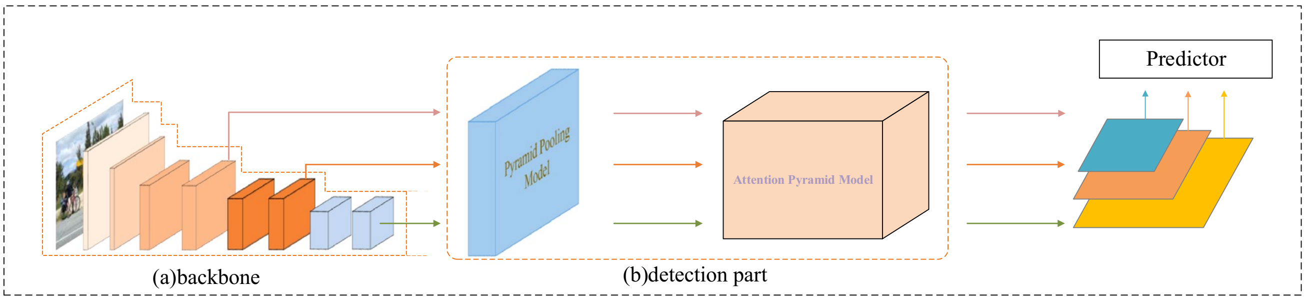

Fig. 1 L-Net network structure

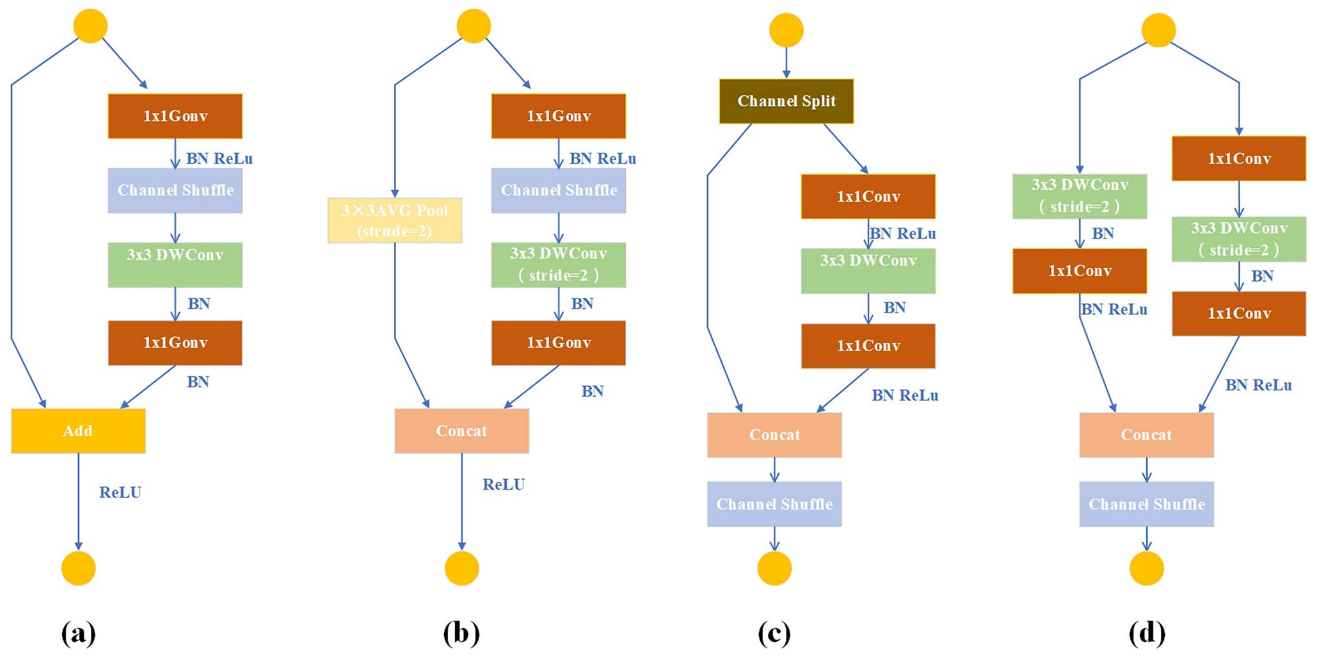

Fig. 2 Building blocks of ShuffleNet. **a** the basic ShuffleNetV1 unit; **b** the ShuffleNetV1 unit for spatial down sampling(2×); **c** the basic ShuffleNetV2; **d** the ShuffleNetv2 unit for spatial down sampling(2×). *DWConv* depthwise convolution. *GConv* group convolution

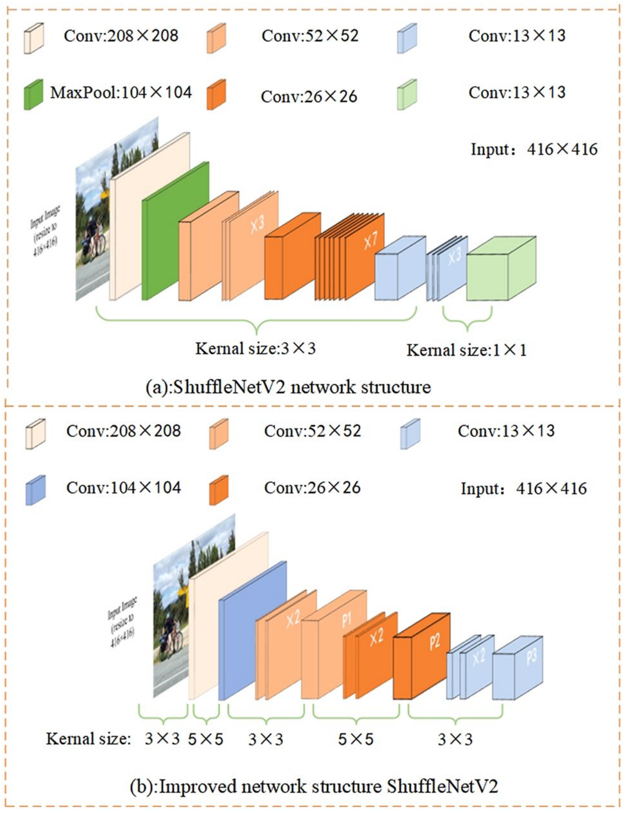

Fig. 3 Network structure comparison chart. **a** Shufflenetv2 network structure diagram; **b** shufflenetv2 improved network structure diagram; conv represents convolution layer; MaxPool represents pooling layer; for example, 13×13 represents feature graph size

ShuffleNet to better meet the feature extraction in the object detection task. The input image resolution is changed to 416×416 to ensure sufficient low-level features. The 3×3 deep convolution part is replaced by the 5×5 deep convolution in the ShuffleNet basic unit to further expand the receptive field of the feature map. The improved module structure is shown in Fig. 4c, d are the module structure originally used by shufflenetv2, and Fig. 4e is the improved module structure in this paper to expand the receptive field. On this basis, an improved backbone network is given, as shown in Fig. 3b.

To ensure the calculation cost of the model and reduce the number of channels of input feature maps, the specific network structure is shown in Table 1.

3.2 Detection part

To make full use of the feature information obtained by feature extraction, the detection part is composed of the Pyramid Pooling module and the Attention Pyramid module. The specific network structure is shown in Fig. 5. The Pyramid Pooling module extracts important information from the three feature maps from the main part to remove redundant information, the Attention Pyramid module performs multi-scale fusion on the processed three feature maps, and obtains the importance of channel information in the fused feature map through learning, and then improves it according to this importance useful features and weakening features that are not important to the current task.

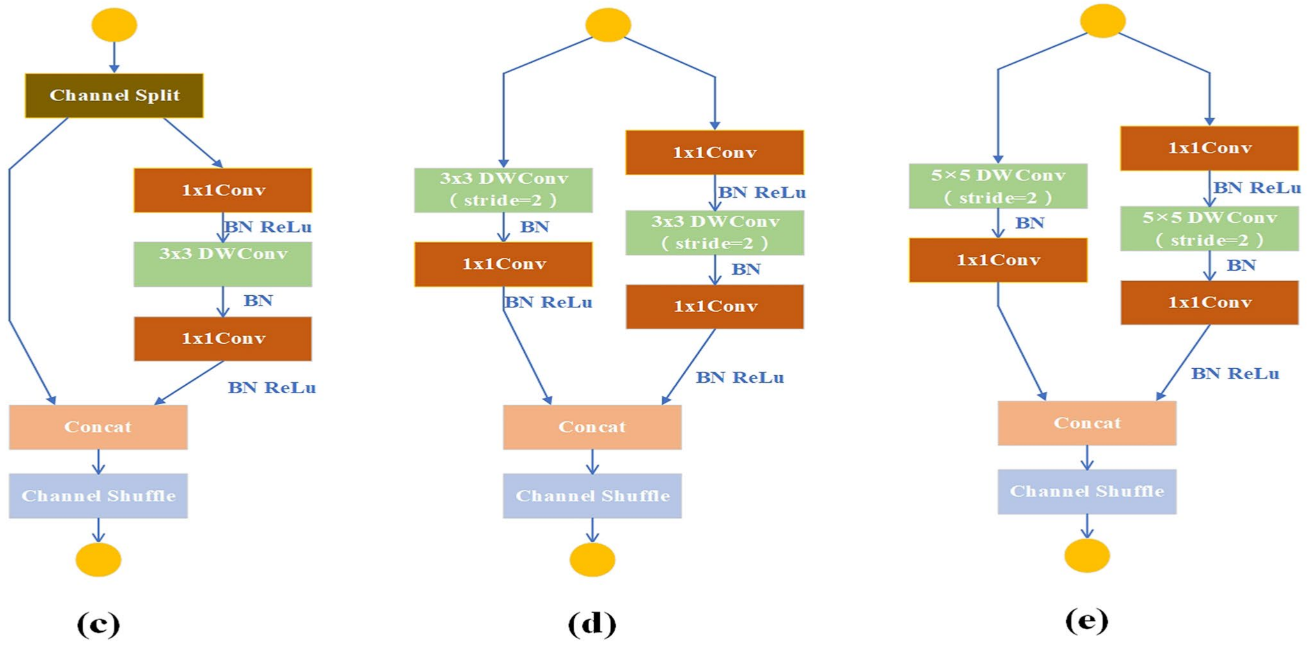

Fig. 4 Module structure used in the backbone of the L-Net model. **c** our basic unit; **d**, **e** our unit for spatial down sampling(2×); *DWConv* depth-wise convolution. *GConv* group convolution

Table 1 Backbone network based on ShuffleNetv2

Layer	Output size	Ksize	Stride	Output channels
Image	416 × 416			3
Conv1	208 × 208	3 × 3	2	24
Stage1	104 × 104	5 × 5	2	16
Stage2	52 × 52	3 × 3	2	32
			1	40
Stage3	26 × 26	5 × 5	2	80
			1	112
Stage4	13 × 13	3 × 3	2	192
			1	320
FLOPs				0.56B

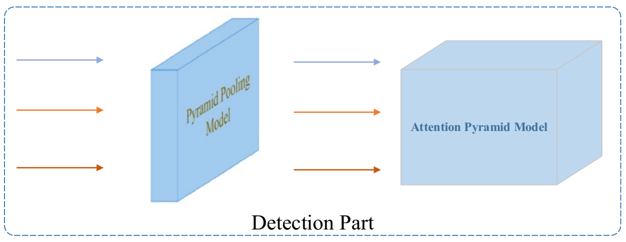

Fig.5 Schematic diagram of model detection part

3.2.1 Pyramid pooling module

The module takes the three feature maps obtained on the backbone network as separate inputs and then uses three suitable pooling cores from the four alternatives pooling cores (2×2, 3×3, 6×6, and 9×9) to obtain feature information while removing some redundant information. The choice of pooling core size is related to the size of the feature map input to this module. For example, if the input feature mapping scale (like P3 in Fig. 3b) is small, three smaller pooling cores (2×2, 3×3, 6×6) will be selected to prevent the loss of useful information. To ensure the weight of the global feature, if there are N (N=3 in the module) levels in the pooling pyramid, a 1×1 convolution is used after each level to reduce the number of channels to 1/N, and then bilinear interpolation is used to obtain the size before pooling. Then the features of three different pooling scales are merged. The structure of the module is shown in Fig. 6.

Specifically, P1, P2, and P3 output from the backbone network are used as the input of this module. When the input is P1, the pooled kernel selects 2/3/6, and the output P1' is obtained by 'Concat' of U1, U2 and U3. When the input is P2, the pooled kernel selects 2/3/9, and the output P2' is obtained by 'Concat' of U1, U2 and U4. Similarly, P3' is obtained by 'Concat' of U2, U3 and U4.

3.2.2 Attention Pyramid module

The traditional feature pyramid is proposed to solve the detection problem of small-scale objects. It mainly uses the feature map output at different stages of the backbone network to construct a top-down feature fusion path. The specific details

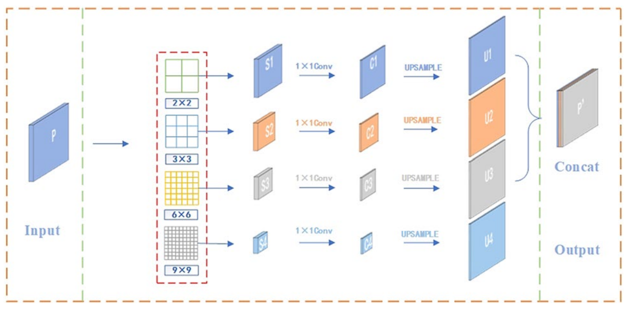

Fig. 6 Structure of Pyramid Pooling Module (PPM). PPM uses three pooling kernels of different sizes to obtain information about the feature map

are shown in Fig. 7. This network structure can fully integrate feature maps with strong low-resolution semantic information and feature maps with rich high-resolution spatial information. However, the spatial information in the high-resolution feature map needs to go through dozens or even hundreds of layers of networks to reach the highest level of the feature pyramid. This process will generate a lot of calculations and lose a lot of spatial information. To solve the above problems, an Attention Pyramid module is given. The specific details of the module are shown in Fig. 8.

As shown in Fig. 8, take the three feature maps of the output of the pyramid pooling module as the input of this module to write {C1, C2, C3}. To improve the accuracy of the model, construct various scales of high-level semantic feature maps for the input feature maps, such as C3, as the highest level feature map of the pyramid, its feature map scale is first used as a benchmark, and let C_1 and C_2 down-sample to get the same size as C_3, and then through the 1 × 1 convolution compression channel, in order to ensure that the overall calculation of the model is set to the default value of 256, and then C_3' is obtained through 'Concat' operation. C_1, C_2 generate high-level semantic feature maps C1', and C_2' does the same. Since the feature maps generated above are all obtained by a single convolution filter, this means that the feature map channels obtained by 'Concat' are uncorrelated and lack channel correlation, so the channel attention mechanism is added to generate the final feature map. Write this part of the input as $\mathbf{U} = [u_1, u_2, \ldots, u_c]$. The first step is to compress all spatial information into a channel descriptor through global average pooling, which has a global receptive field. From a formal point of view, by shrinking the space dimension H × W of U, a statistic z $\in R^C$ is generated, so the calculation method of the c-th element of z is:

$$z_c = F_{sq}\left(u_c\right) = \frac{1}{H \times W}\sum_{i=1}^{H}\sum_{j=1}^{w} u_c(i,j). \tag{1}$$

Then use two 1 × 1 convolution operations to enhance the correlation between the feature map channels, and output the same number of channels as U. The process can be expressed as:

$$z_c' = F_{channel}\left(F_{sq}\left(u_c\right) \cdot W\right) = \delta\left(f\left(z, W_1\right)\right), \tag{2}$$

$$s = F_{channel}\left(F_{EX}(z, W)\right) = \sigma\left(h\left(W_2 \cdot z\right)\right). \tag{3}$$

δ, f in formula (2) refer to the ReLU function and the 1 × 1 dimensionality reduction convolution operation, respectively, $W_1 \in R^{\frac{C}{r}\times C}$, where r = 16. σ, h in formula (3) refers to the Sigmoid function and the 1 × 1 ascending dimension convolution operation, $W_2 \in R^{C\times\frac{C}{r}}$. The final feature map is obtained by rescaling U and activation s:

$$\widetilde{x_c} = F_{scale}\left(u_c, s_c\right) = s_c \cdot u_c. \tag{4}$$

Here $\tilde{X} = \left[\widetilde{x_1}, \widetilde{x_2}, \ldots, \widetilde{x_c}\right]$ and $F_{scale}(u_c, s_c)$ refer to the channel multiplication between the scalar s_c and the feature map u_c. $\tilde{X}$ refers to the final output of the attention pyramid module (Fig. 8).

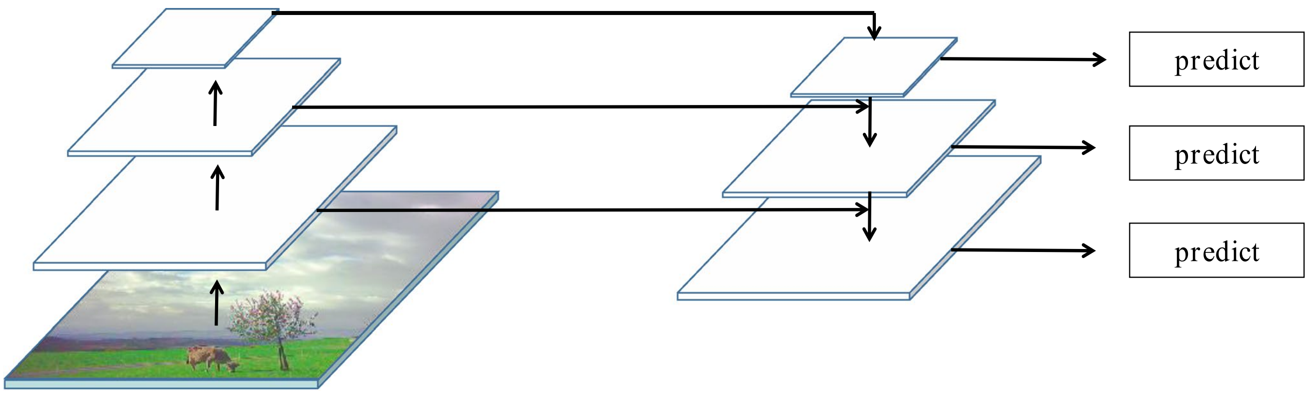

Fig. 7 Feature pyramid network

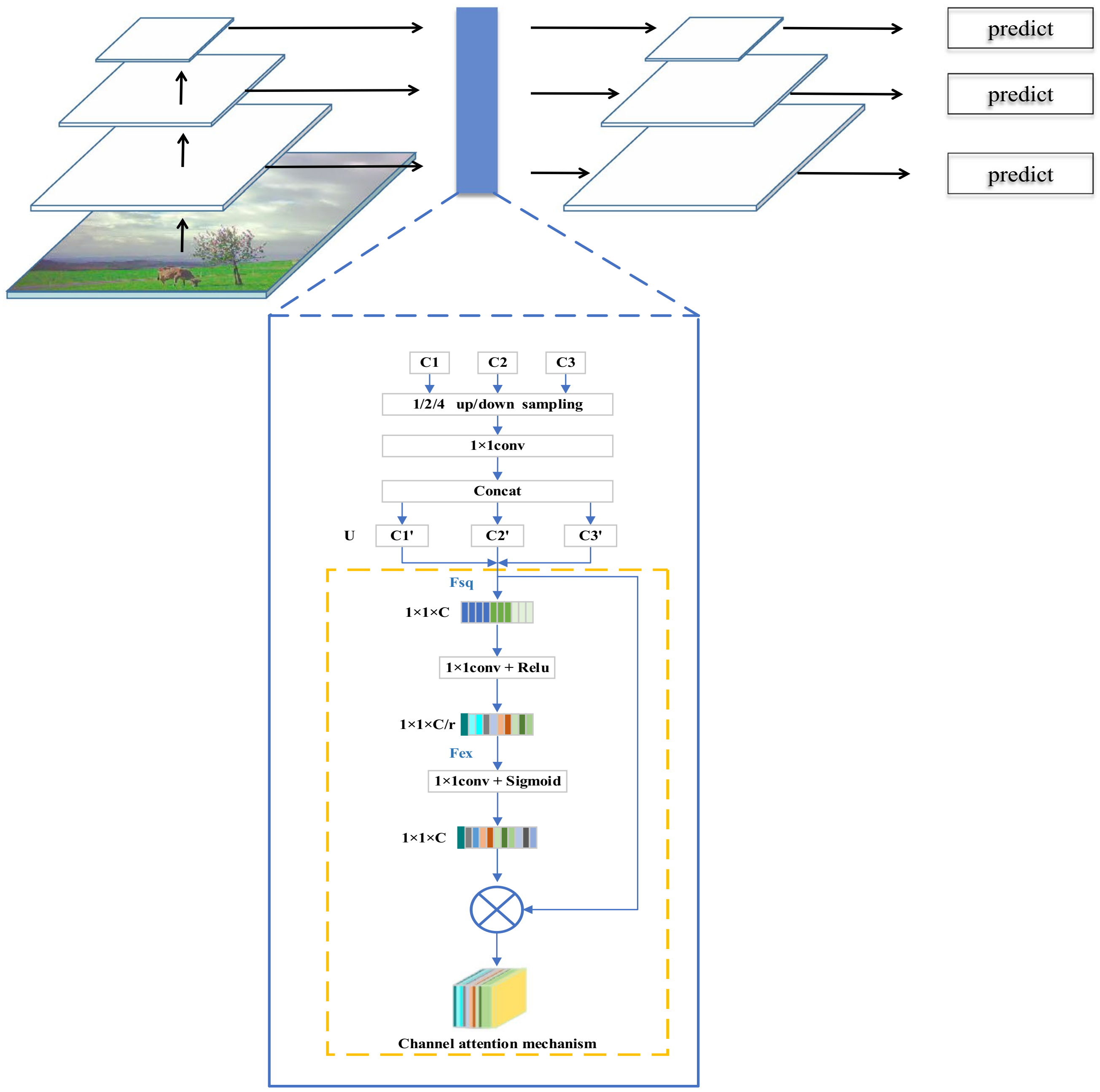

Fig. 8 Structure of Attention Pyramid Module (APM). APM combines feature maps from three scales and encodes more context information

4 Experimental and analysis

In this section, we evaluate the effectiveness of L-Net on PASCAL VOC and COCO benchmarks. And the input resolution and convolutional kernel size suitable for the backbone network were determined through relevant experiments. Then, we conduct ablation studies to evaluate our model design.

4.1 Details and metrics

We evaluate L-Net on the PASCAL VOC and COCO 2017 detection datasets with 118 K training images. The default hyper-parameters are as follows: We set the initial learning rate to 0.01, if the model is trained with val_loss over 2 epochs with no reduction, the learning rate is reduced to 1/10th of the current rate and finally terminated at 180 K iterations. The momentum and weight decay are, respectively, set as 0.9 and 0.0001. For all experiments, we main use one GPU(2080Ti)

Table 2 PASCAL VOC2007 test detection results

Method	Input size	Backbone	FPS	FLOPs	mAP
Faster-R-CNN	600×1000	VGGNet	7	181.12B	73.2
R-FCN	600×1000	ResNet-50	11	–	77.4
SSD	300×300	VGGNet	46	31.75B	77.2
YOLO	448×448	–	45	40.19B	63.4
YOLOv2	416×416	Darknet-19	67	34.90B	76.8
DSOD	300×300	DS/64-192-48-1	17.4	15.07B	77.7
Tiny-YOLO	416×416	–	207	6.97B	57.1
SqueezeNet-SSD	300×300	SqueezeNet	44.7	1.18B	64.3
MobileNet-SSD	300×300	MobileNet	59.3	1.14B	68.0
DSOD small	300×300	DS/64-64-16-1	27.8	5.29B	73.6
Pelee	300×300	PeleeNet	–	1.21B	70.9
L-Net	416×416	ShuffleNetV2*	107	1.54B	70.2

for training. The speed is measured by frame-per-second (fps) on Nvidia TitanX GPU.

We followed the standard evaluation metrics, i.e., IoU, AP,$AP_{0.5}$,$AP_{0.75}$.The following is the definite of IOU:

$$IOU = \frac{area\left(B_p \cap B_{gt}\right)}{area\left(B_p \cup B_{gt}\right)}, \tag{5}$$

where B_p is the prediction borders of the detection network, B_{gt} is the ground-truth body region. We used the mean IoU calculated from all test images.

The AP is definite as follows:

$$AP = \frac{1}{N}\sum_{i=0}^{all}\frac{TP}{(TP+FP)}, \tag{6}$$

where TP means the true positive, FP means the false positive. And N means the numbers of the detection result. AP represents the averaged value of all categories. Traditionally, this is called "mean average precision" (mAP). In the MS COCO dataset, it makes no distinction between AP and mAP. Currently, the MS COCO dataset uses 10 *IOU* thresholds of 0.50:0.05:0.95. $AP_{0.5}$ represents the value of AP when *IOU* is 0.5, and so on.

4.2 Results on PASCAL VOC

As shown in Table 2, we compared the detection speed of L-Net to other lightweight object detectors on the PASCAL VOC 2007 dataset. The speed is measured by frame-per-second (fps) on Nvidia TitanX GPU. With 416 × 416 input, L-Net can process images at a speed of 9.35 ms (107 fps). Compared to SqueezeNet-SSD, L-Net saves 7.55 ms in processing time per image and has better performance in terms of accuracy, L-Net saves nearly 70% of the computation and more than twice the speed compared to DSOD small while sacrificing a slight accuracy. However, L-Net is still slower than Tiny-YOLO, the reason is that Tiny-YOLO is based on a simple convolutional structure (no residuals or tandem) and the authors have optimized a tailored GPU implementation. We argue that when the ShuffleNet unit or backbone network structure is well implemented, our L-Net should run at a faster speed (Fig. 9) .

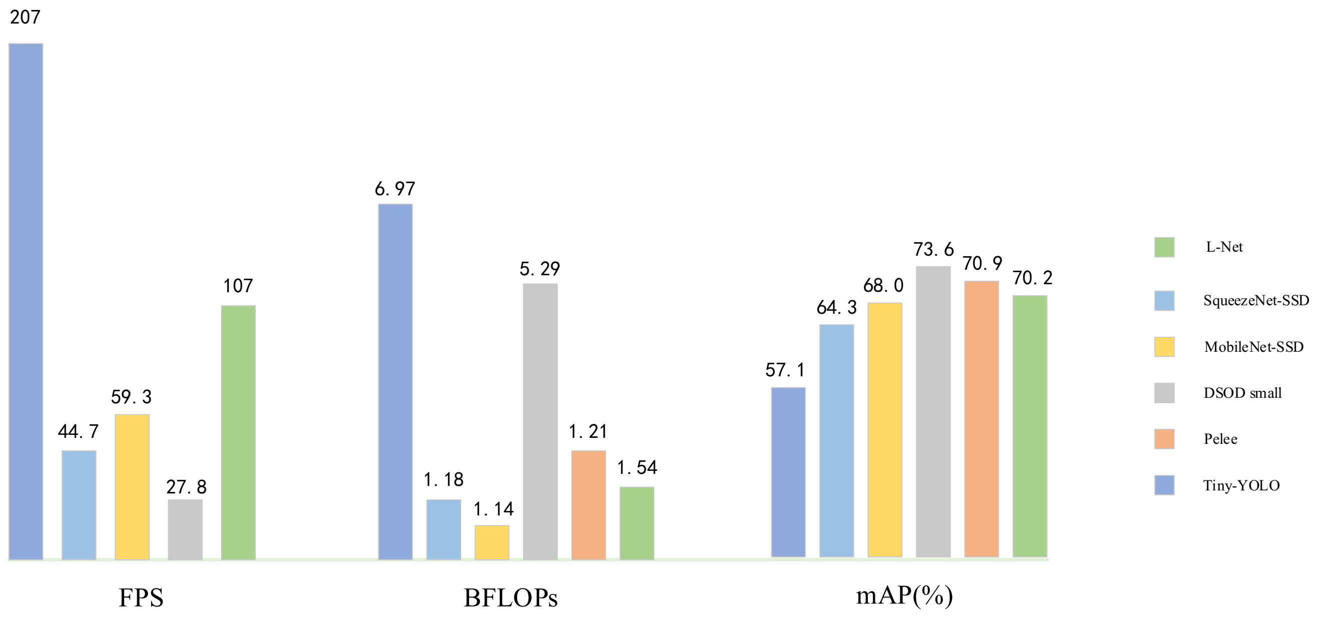

Fig. 9 Experimental results on PASCAL VOC2007

Table 3 Evaluation results on COCO test-dev

Model	Backbone	Input	MFLOPs	AP	AP_{50}	AP_{75}
SSD300*	VGG-16	300 ×300	35,200	25.1	43.1	25.8
SSD321	ResNet-101	321 ×321	16,700	28.0	45.4	29.3
DSSD321	ResNet-101 + FPN	321 ×321	22,300	28.0	46.1	29.2
YOLOv2	Darknet-19	416 × 416	17,500	21.6	44.0	19.2
Light-Head R-CNN	ShuffleNet v2*	800 × 1200	5650	23.7	–	–
MobileNet-SSD	MobileNet	300 × 300	1200	19.3	–	–
MobileNet-SSDLite	MobileNet	320 × 320	1300	22.2	–	–
MobileNetV2-SSDLite	MobileNetV2	320 × 320	800	22.1	–	–
Pelee	PeleeNet	304 × 304	1290	22.4	38.3	22.9
Tiny-DSOD	DDB-Net + D-FPN	300 × 300	1120	23.2	40.4	22.8
ThunderNet	SNet49	320 × 320	264	19.1	33.7	19.6
L-Net	ShuffleNetv2*	416 × 416	1540	21.8	36.5	22.1

4.3 Results on MS COCO

As shown in Table 3, by comparing the L-Net model with other lightweight models, the L-Net model gets a better AP than the MobileNet-SSD when inputting 416×416, but also adds a small number of FLOPs, which is mainly due to the larger input resolution. However, compared with the MobileNet-SSDLite, MobileNetV2-SSDLite, and Pelee models, L-Net does not get outstanding AP with the addition of a slight number of FLOPs. We analyze that there may be two reasons for this, the first being that the shufflenet unit or backbone network has not yet reached its optimum, and the second being the large increase in FLOPs due to the large number of ordinary convolution operations used in PPM and APM. This will also be part of our future work. Compared with the proposed Light-Head R-CNN, which is also based on the Shufflenetv2 model, L-Net saves about 72% of FLOPs at the expense of 1.9mAP, which shows that the L-Net model is effective in object detection tasks (Figs. 10 and 11).

4.4 Ablation experiments

To demonstrate the validity of the backbone network and PPM, APM, we performed comparison experiments on PASCAL VOC and MS COCO.

4.4.1 Input resolution

To find the input resolution that best suits the backbone network, we performed simple comparison experiments on the PASCAL VOC dataset, each combining the Feature Fusion Network to achieve the object detection task, while only changing the resolution. The reason we do not directly classify the backbone network on ImageNet for the classification

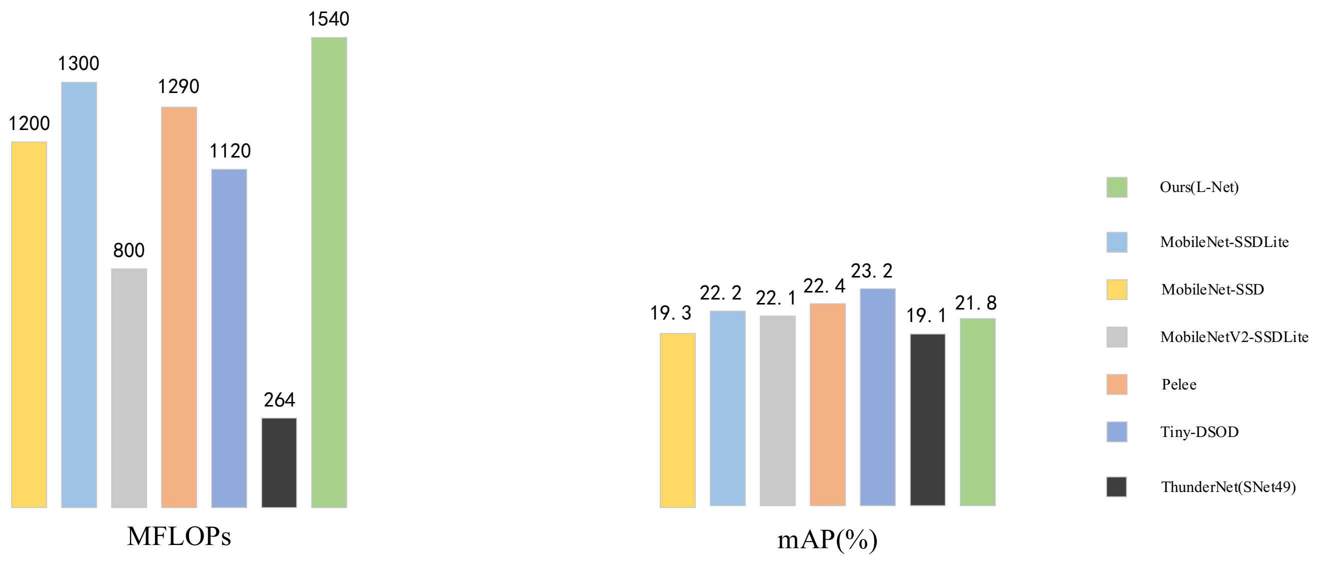

Fig. 10 Experimental results on COCO test-dev

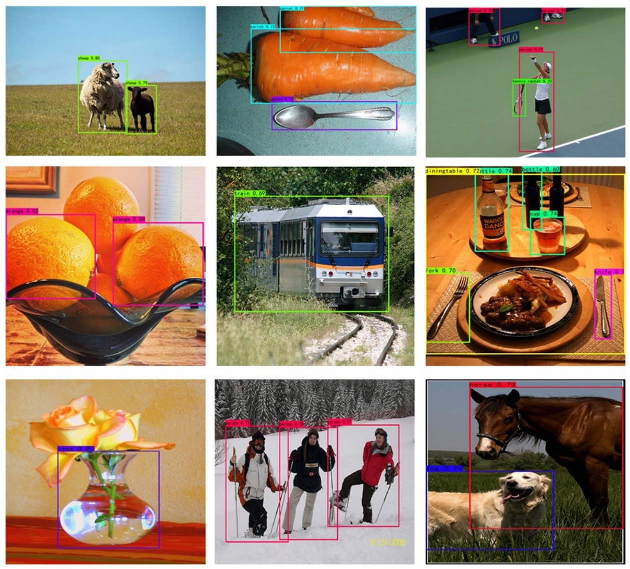

Fig. 11 Visualizes several examples on COCO val-dev

task is that the results it yields do not indicate the effectiveness of the backbone network for the object detection task.

Table 4 Evaluation of different input resolutions on PASCAL VOC2007 test detection results (FFN refers to PPM + APM)

Backbone	Input	MFLOPs	AP
ShuffleNetv2* + FFN	480 × 480	1760	69.4
ShuffleNetv2* + FFN	416 × 416	1540	70.2
ShuffleNetv2* + FFN	320 × 320	1246	65.9

Table 5 The influence of kernel size for backbone network

Backbone	Input	MFLOPs	AP
ShuffleNetv2* (all 3 × 3)	416 × 416	1486	65.3
ShuffleNetv2* (all 5 × 5)	416 × 416	1623	67.7
ShuffleNetv2* (ours)	416 × 416	1540	70.2

The result shows that the input resolution 416 is more suitable for the backbone network, and further confirms that the input resolution should match the size of the backbone network (Table 4).

4.4.2 5×5 Depthwise convolutions

We change the size of the kernel to get the optimal structure for the backbone network, as shown in the Table 5. If all the kernels with 3 are used, the computation amount is smaller but the result is not as good as ours. If all the kernels with 5 are used, the computation amount will be higher but the result is not satisfactory. We speculate that the reason is that although more information may be obtained with larger convolution kernels, their excessive use causes information redundancy, which is detrimental to the object detection task.

4.4.3 Backbone network

We evaluate the design of the backbones, ShuffleNetv2 and ShuffleNetv2* are used as the baseline. When setting the feature map size of the network input to a uniform 416×416, we performed a small number of convolution operations after the respective backbone networks to convenient prediction on the PASCAL VOC dataset. The experimental results are shown in Table 6, where the improved network structure based on ShuffleNetV2 showed no significant decrease in accuracy compared to ShuffleNetV2, but the FLOPs were reduced by approximately 79%. This is probably due to the larger receptive field of shufflenetv2* building blocks than the shufflenetv2(5 vs. 3). This result also validates the effectiveness of the receptive field in object detection tasks.

Table 6 Evaluation of lightweight backbones on PASCAL VOC

Backbone	Input	FLOPs	AP
ShuffleNetV2	416×416	2.94B	45.2
ShuffleNetV2*	416×416	0.59B	43.6

4.4.4 Detection part

To find a suitable feature fusion module for the backbone network, we combined the traditional Feature Pyramid Network (FPN) and the proposed Feature Fusion Network (FFN) after the ShuffleNetV2* backbone, respectively, and conducted comparison experiments on the PASCAL VOC dataset when inputting the same size feature maps. The experimental results are shown in Table 7, we get a good result that the AP is 69.7 with only 1.72B FLOPs when the FPN is applied to the ShuffleNetV2*. However, the combination of backbone and FFN achieves better results, saving 0.18B FLOPs compared to FPN and increasing the AP value by 0.5. In addition, we plotted the performance comparison curves for the FFN and the FPN. In Fig. 12, the abscissa is the threshold of IOU, and the ordinate is the AP value of the FFN and the FPN. We use different color polylines to distinguish different methods. Obviously, we can see that the FFN proposed by us has higher AP values than the FPN in any threshold range.

Table 7 Evaluation of different feature fusion modules on PASCAL VOC dataset (FFN refers to PPM + APM)

Model	Input	MFLOPs	AP
ShuffleNetV2* + FPN	416×416	1720	69.7
ShuffleNetV2* + FFN	416×416	1540	70.2

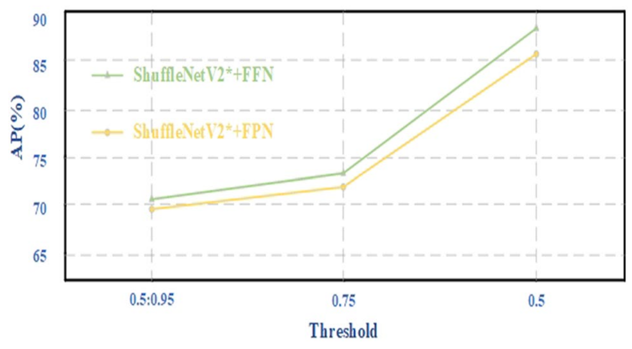

Fig. 12 Means Average Precision of the proposed FPN-based model and FFN-based model in different thresholds

4.4.5 PPM

If the PPM is added directly after the backbone network for the object detection task, there will be a slight increase in accuracy at a small computational cost. By comparing the information in Table 8, it can be concluded that PPM will obtain more useful information for the object detection task after fusing multiple feature information in different regions of the feature map.

4.4.6 APM

From Table 8, we can obtain that the computational cost of adding the APM model directly after the backbone network improves the 315 M FLOPs, while the AP improves the 1.3, indicating that feature fusion of feature maps of different resolutions followed by channel feature learning will improve the object detection task. When we combined the PPM and APM, we obtained the desired results using repeated feature extraction.

Through the above experimental results on the VOC and COCO data sets and the model ablation experiments, it can be concluded that the L-Net model has the performance of performing real-time object detection tasks and achieving ideal results under the condition of limited computing resources.

5 Conclusion

In this paper, we have proposed a lightweight network L-Net. The network consists of three parts, the first part is an improved backbone network based on ShufflenetV2, the second part consists of a Pyramid Pooling module and an Attention Pyramid module. By reducing the number of input channels, setting the input resolution to 416×416, and changing the depth convolution from 3×3 to 5×5, a suitable lightweight backbone was obtained. The third is the detector head, we refer to the detection head of the YOLOV3 algorithm and design it accordingly. And the effectiveness of each module in the feature fusion network is verified through ablation experiments. The effectiveness of this lightweight object detection algorithm is illustrated by the fact that L-Net achieved 21.8mAP on the MS COCO dataset using only 1.54B FLOPs and 70.2mAP on the PASCAL VOC dataset at 107fps (Frames Per Second).

In subsequent research, we are going to further exploration on ShuffleNet unit and backbone network to achieve better performance, and carry out lightweight research on PPM and APM, so that they can save more computational resources while maintaining the same detection accuracy.

Acknowledgements This work is supported by Ministry of Education, China (201901055015), Natural Science Foundation of Shandong Province (ZR2020KE023) and Shandong University of Science and Technology (JXTD20170503).

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

Table 8 Ablation studies of L-Net on MS COCO val-set

ShuffleNetv2*	PPM	APM	AP	AP_{50}	AP_{75}	MFLOPs
√			19.6	33.1	19.8	1046
√	√		20.4	33.6	19.7	1205
√		√	20.9	34.2	20.0	1361
√	√	√	21.8	36.5	22.1	1540

References

1. Redmon J, Divvala S, Girshick R, et al.:You only look once: unified, real-time object detection. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788 (2016).
2. Redmon J, Farhadi A.: YOLO9000: better, faster, stronger. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7263–7271 (2017).
3. Redmon, J., & Farhadi, A.: Yolov3: an incremental improvement. arXiv preprint arXiv:1804.02767 (2018).
4. Liu W, Anguelov D, Erhan D, et al.: Ssd: single shot multibox detector. European conference on computer vision. Springer, Cham, pp. 21–37 (2016).
5. Law H, Deng J.: Cornernet: detecting objects as paired keypoints. Proceedings of the European conference on computer vision (ECCV), pp. 734–750 (2018).
6. Girshick R, Donahue J, Darrell T, et al.: Rich feature hierarchies for accurate object detection and semantic segmentation. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580–587 (2014).
7. Girshick R.: Fast r-cnn. Proceedings of the IEEE international conference on computer vision, pp. 1440–1448 (2015).
8. Ren, S., He, K., Girshick, R., & Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. arXiv preprint arXiv:1506.01497 (2015).
9. He K, Zhang X, Ren S, et al.: Deep residual learning for image recognition. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778 (2016).
10. Lin T Y, Dollár P, Girshick R, et al.: Feature pyramid networks for object detection. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2117–2125 (2017).
11. Lin T Y, Goyal P, Girshick R, et al.: Focal loss for dense object detection. Proceedings of the IEEE international conference on computer vision, pp. 2980–2988 (2017).
12. Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017).
13. Li, Y., Li, J., Lin, W., Li, J.: Tiny-DSOD: Lightweight object detection for resource-restricted usages. arXiv preprint arXiv:1807.11013 (2018).
14. Wang, R. J., Li, X., Ling, C. X.: Pelee: A real-time object detection system on mobile devices. arXiv preprint arXiv:1804.06882 (2018).
15. Leng, L., Zhang, J.: Palmhash code vs. palmphasor code. Neurocomputing **108**, 1–12 (2013)
16. Zhang, Y., Chu, J., Leng, L., et al.: Mask-refined R-CNN: a network for refining object details in instance segmentation. Sensors **20**(4), 1010 (2020)
17. Chu, J., Guo, Z., Leng, L.: Object detection based on multi-layer convolution feature fusion and online hard example mining. IEEE Access **6**, 19959–19967 (2018)
18. Liu, S., Huang, D., Wang, Y.: Learning spatial fusion for single-shot object detection. arXiv preprint arXiv:1911.09516 (2019).
19. Leng, L., Li, M., Kim, C., et al.: Dual-source discrimination power analysis for multi-instance contactless palmprint recognition. Multimedia Tools and Applications **76**(1), 333–354 (2017)
20. Ma N, Zhang X, Zheng H T, et al.: Shufflenet v2: Practical guidelines for efficient cnn architecture design. Proceedings of the European conference on computer vision (ECCV), pp. 116–131 (2018).
21. He, K., Zhang, X., Ren, S., et al.: Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans. Pattern Anal. Mach. Intell. **37**(9), 1904–1916 (2015)
22. Erhan D, Szegedy C, Toshev A, et al.: Scalable object detection using deep neural networks. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2147–2154 (2014).
23. Tian Z, Shen C, Chen H, et al.: Fcos: fully convolutional one-stage object detection. Proceedings of the IEEE/CVF international conference on computer vision, pp. 9627–9636 (2019).
24. Sermanet P, Eigen D, Zhang X, et al.: Overfeat: integrated recognition, localization and detection using convolutional networks. arXiv preprint arXiv:1312.6229 (2013).
25. Liu Z, Sun M, Zhou T, et al.: Rethinking the value of network pruning. arXiv preprint arXiv:1810.05270 (2018).
26. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014).
27. Courbariaux M, Bengio Y, David J. P.: Binaryconnect: Training deep neural networks with binary weights during propagations. arXiv preprint arXiv:1511.00363 (2015).
28. Hubara I, Courbariaux M, Soudry D, et al.: Binarized neural networks. Proceedings of the 30th international conference on neural information processing systems, pp. 4114–4122 (2016).
29. Hinton G, Vinyals O, Dean J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015).
30. Li Z, Peng C, Yu G, et al.: Light-head r-cnn: in defense of two-stage object detector. arXiv preprint arXiv:1711.07264 (2017).
31. Peng C, Zhang X, Yu G, et al.: Large kernel matters--improve semantic segmentation by global convolutional network[C]// Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4353–4361 (2017).
32. Zhang X, Zhou X, Lin M, et al.: Shufflenet: an extremely efficient convolutional neural network for mobile devices. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 6848–6856 (2018).
33. Sandler M, Howard A, Zhu M, et al.: Mobilenetv2: inverted residuals and linear bottlenecks. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4510–4520 (2018).
34. Howard A, Sandler M, Chu G, et al.: Searching for mobilenetv3. Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1314–1324 (2019).
35. Chollet F.: Xception: deep learning with depthwise separable convolutions. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1251–1258 (2017).

Jin Han received the degrees of Bachelor of Engineering and Master of Technology from the Shandong University of Science and Technology, China , in1989 and 2003,respectively. She received her Ph.D. degree from Shanghai University, China. Her current research interests include embedded intelligent system and Internet of Things technology.

Yonghao Yang born in Shandong province, China in 1996, studied computer science and Technology at Qilu Normal University from 2015 to June 2019, and obtained a bachelor's degree. He is currently pursuing the M.S. degree with the Shandong University of Science and Technology. His current research interest includes image processing and deep learning.