LDPNet: A Lightweight Densely Connected Pyramid Network for Real-Time Semantic Segmentation

XUEGANG HU\textsuperscript{1} AND LIYUAN JING\textsuperscript{2}

\textsuperscript{1}School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
\textsuperscript{2}Multimedia Communications Research Laboratory, Chongqing University of Posts and Telecommunications, Chongqing, 400065, China
Corresponding author: Liyuan Jing (jly1084152915@163.com)

This work was supported by the key project of the Natural Science Foundation of Chongqing, China, under Grant cstc2017jcyjBX0037.

Abstract A deep convolutional neural network has been widely used in image semantic segmentation in recent years, its deployment on mobile terminals, however is limited by its high computational costs. Given the slow inference speed and large memory usage of deep convolutional neural networks, we propose a lightweight and densely connected pyramid network (LDPNet) for real-time semantic segmentation. Firstly, a densely connected atrous pyramid (DCAP) module is constructed in the encoding process to extract multi-scale context information for forwarding propagation, strengthen the reuse of features, and offset the spatial information lost in the down-sampling process of the feature map. Secondly, a cross-fusion (CF) module embedded in each other during the decoding process is proposed, which uses high-level semantic features to effectively guide the fusion of low-level spatial details while strengthening context information. Our network is tested on two complex urban road scene data sets. Among them, experiments on the Cityscapes data set show that our structure has 87 frames per second (FPS) on a single NVIDIA GTX1080Ti GPU. The Mean Intersection over Union (mIoU) reaches 71.1%, and the parameter is only 0.8M. Compared with the existing similar networks, the new system achieves a state-of-the-art trade-off between efficiency and accuracy.

Index Terms Deep convolutional neural network, lightweight network, encoder-decoder network, atrous pyramid module, real-time semantic segmentation.

I. INTRODUCTION

The transformation from experience-driven artificial feature paradigm to a data-driven representation learning paradigm is realized by means of the deep learning with strong nonlinear modeling capability. Many models based on deep learning have been used to make breakthroughs in computer vision, speech recognition, natural language processing, and bioinformatics. As an example, image semantic segmentation aims to classify every pixel in a picture at the semantic level, which is a scorching research topic in computer vision, and widely used in augmented reality, automatic driving, video surveillance, and other scenarios [1]. However, many semantic segmentation technologies can achieve higher accuracy by building more in-depth and broader networks at the cost of sacrificing forward inference speed and consuming many computing costs. For example, PSPNet [2] and DeepLab [3] have a total of 250.8M parameters and 262.1M parameters respectively, besides, both of them are over 100 layers, therefore their inference speed is far below the minimum frames required for a video (24 frames).

Moreover, these large-scale and high-precision models still require long processing time, even when running on the most advanced modern GPUs [4]. However, in practice, many terminals have far less computing power and storage capacity than the advanced GPUs, which makes it challenging to deploy large-precision models on terminals. Therefore, small segmentation networks which are low in computing cost, fast in inference speed, and memory-friendly are often desired.

In response to these problems, researchers have proposed many real-time semantic segmentation networks based on deep learning in recent years. The methods adopted by...
these networks can be roughly divided into three categories: (1) Change the input size: in the methods proposed in the literature [5]–[7], cropping or resizing is used to change the input image, thus reducing the network’s parameters and computational complexity. Although these networks are easy to operate, cropping or resizing operations will decrease the image resolution, which causes great loss of spatial image details, especially for edge information and small objects. The segmentation precision of the network will thus be significantly reduced. (2) Convolution Factorization [8]–[10]: The convolution operation of convolution neural network is mainly accomplished by matrix multiplication. However, the weight matrix is often dense and massive under normal circumstances, thereby bringing enormous computing and storage costs to real-time segmentation tasks with limited resources. In view of this, the direct idea to deal with this situation is to take a standard 2D convolution which is then factorized into two 1D convolutions. Specifically, the original convolution kernel \( K \times K \) is factorized into two convolution kernels, i.e., \( 1 \times K \) and \( K \times 1 \). This method is available for significant reduction of the parameters of the segmentation network without too much precision loss. (3) Lightweight backbone network: In the method given in [11]–[14], the last classification layer of the lightweight network used for image classification is removed, the decoder is then combined to compromise between segmentation efficiency and segmentation accuracy. In the meantime, the appearance of depthwise separable convolution and group convolution dramatically reduces the network parameters.

The existing semantic image segmentation networks based on deep learning is successful in certain degree, which is characterized by the advantages in terms of segmentation precision or efficiency. However, the fundamental objective of the current research is still to obtain the highest accuracy in devices with limited computing resources and the least computational cost.

Based on the above analysis, a lightweight, densely connected pyramid network (LDPNet) is proposed for real-time semantic segmentation tasks. While in the encoder part, convolution kernels with different atrous rates are used to construct a densely connected atrous pyramid (DCAP) module, in this manner to extract multi-scale information. Unlike the existing atrous pyramid network, instead of placing the atrous pyramid at the end of the system, the DCAP module, as a feature extraction module, builds a sampling coding network, and then uses dense connections to strengthen feature reuse. In the decoder part, we design a cross-fusion (CF) module to efficiently aggregate low-level spatial details and high-level semantic features.

In conclusion, our main contributions are as follows:

- The DCAP module is proposed to extract multi-scale features and enhance feature reuse in the encoder stage.
- The CF module is proposed to efficiently aggregate low-level spatial details and high-level semantic features in the decoder stage. Compared with the Pyramid Pooling Module (PPM) [2] in terms of segmentation performance, at similar accuracy, the CF module’s parameter is 47.8 times lower than that of the PPM, and the computational complexity is 11 times lower.
- Based on the DCAP module and the CF module, a lightweight network called LDPNet is proposed. As shown in Fig. 1, compared with the most advanced real-time semantic segmentation methods, LDPNet achieves the best trade-offs in terms of accuracy and inference speed.

![FIGURE 1. Inference speed and accuracy on the Cityscapes data set. Speed is measured on a GTX 1080 Ti GPU. The red dot represent our method, the black dots represent other methods, and the red dotted lines represent the lowest real-time speed.](image.png)

**II. RELATED WORK**

In recent years, real-time semantic segmentation has made significant progress. In this section, current new methods in real-time semantic segmentation are summarized. After that, two ways that are most relevant to our work are discussed, i.e., attention mechanism and dilated/atrous convolution.

**A. REAL-TIME SEMANTIC SEGMENTATION**

Due to the fact that the trained network has to be deployed to terminal devices with limited computing resources in practical applications, lightweight real-time semantic segmentation networks have drawn people’s more attention. A multi-scale image was used by ICNet [15] as the input of the cascade network to extract features. BiSeNet [16] and BiSeNet V2 [17] put forward new thinking on real-time semantic segmentation, which divided the segmentation network into two branches. Spatial path extracted the spatial information of the original image, and the context path obtained the high-level semantic information. Finally, a feature fusion module was designed to merge the two features effectively. A three-level feature extraction network was designed by DFANet [18], which fully promoted the interaction and aggregation of feature information at different levels while making sure that the computational burden was small. On the other hand, a novel multi-feature fusion module was used by MSFNet [19] to strengthen the information flow.
among layers. At the same time, it also enhanced the sensitivity of high-level semantic information to spatial information. Besides, a class boundary supervision was designed by the upsampling process, which further improved the network’s segmentation effect on the edge part.

B. ATTENTION MECHANISMS

The focus of the standard convolution is only placed on the local receptive field instead of the dependence between pixels. In the $K \times K$ convolution kernel, the target pixel’s value is calculated completely by itself and $K \times K - 1$ pixels around it, which may lead to the lack of perception of global information. The attention mechanism is to capture the interrelationship among pixels, realize the global receptive field of each pixel, and obtain the information correlation between different time, space, and channels through various operations eventually. In essence, the attention mechanism refers to the weighted processing of image areas, which is used to highlight the significant regions and weaken the irrelevant regions.

A positional attention mechanism and a channel attention mechanism were proposed by DANet [12] to capture the global feature dependence in space and channel dimensions. While CCNet [20] used the criss-cross attention module to calculate each pixel’s similarity and its pixels in the same row and the same column. The similarity between pixels was calculated indirectly through two cycles of operation, thus reducing the model’s space complexity. SANet [21] decomposed the semantic segmentation task into two sub-tasks, i.e., pixel-group attention and pixel-wise prediction, and proposed the squeeze-and-attention (SA) module with full consideration given to the space-channel interdependence. CPNet [22] obtained a good feature representation by selectively captured contextual dependencies within and between classes.

C. DILATED/ATROUS CONVOLUTION

The receptive field of the traditional convolution neural network amplification model is mainly obtained through stacking convolution layers and pooling layers. However, these operations will cause the image to lose some detailed information after multiple down-sampling while increasing the model’s parameters and computational complexity. The dilated convolution [23] (also known as atrous convolution) is available to increase the model’s receptive field without increasing the parameters or the loss of spatial information by pooling. In the meantime, the output of each dilated convolution contains a massive range of information. Assuming that the standard convolution kernel size is $K \times K$, while the convolution kernel’s size with the atrous rate $d$ is $K + (K - 1) \times (d - 1)$, although the dilated convolution contributes to the increase of the receptive field, stacking multiple convolutions with the same dilation rate will produce a grid effect, leading to unsatisfactory segmentation results. Therefore, Wang et al. [24] proposed hybrid dilated convolution which is able to effectively avoid the gridding effect’s influence.

An Atrous Spatial Pyramid Pooling (ASPP) module was constructed by the DeepLab [3], [25], [26] series as the network’s decoder, which used multiple convolutions with different atrous rates to capture multi-scale context information.

III. THE NETWORK STRUCTURE

In this section, the DCAP module is introduced, which captures multi-scale features, and enhances information transfer among layers. After that, the CF module is introduced, which makes use of the characteristics of the complementary information among different levels of the convolution neural network. Finally, we propose the overall architecture of the system.

A. ATROUS PYRAMID BOTTLENECK BLOCK

Semantic segmentation has to achieve the balance between the local and global information, and also integrate multiple spatial scale information. In that case, the ASPP module is used by many methods [3], [11], [25]–[27] to encode multi-scale information at the end of the network, thus improving the model’s segmentation effect. Due to the fact that multi-scale information and feature fusion is able to make the model produce better segmentation accuracy, we design an atrous pyramid (AP) bottleneck block based on a split-transform-concatenate strategy. This bottleneck block can capture multi-scale information of input features at each encoding stage, as shown in Fig. 2.

To mitigate the model’s computational burden and reduce the parameters, the input feature first passes through a $1 \times 1$ convolution, and then channel $C$ is equally split into four sub-channels. The size of the split feature map is $H \times W \times C_i$, $i \in \{1, 2, 3, 4\}$, where $H$ and $W$ refer to the feature map’s height and width respectively, and $C_i$ represents the number of channels in the feature map. The split feature maps pass through the $3 \times 3$ convolution kernels with different atrous rates in parallel to learn image feature representations. The filters with different atrous rates in each branch enable the AP bottleneck block to learn the local features and surrounding information from a larger receptive field. After that, the channels are concatenated to obtain the $H \times W \times C$ feature map, and perform $1 \times 1$ point convolution. Inspired by ShuffleNet [28], the channel shuffle is eventually used to strengthen the cross-flow of information on the feature channel. The standard unbiased convolution parameter calculation...
formula is \( K_h \times K_w \times C_{in} \times C_{out} \), where \( K_h \) and \( K_w \) refer to the convolution kernel’s height and width respectively, \( C_{in} \) represents the number of input channels, and \( C_{out} \) represents the number of output channels. Because the channel split-transform-concatenate strategy is adopted by the AP bottleneck block, its parameters are \( K_h \times K_w \times \frac{C_{in}}{4} \times \frac{C_{out}}{4} \times 4 \). Compared with the standard unbiased convolution parameters, the AP module is reduced by four times, which dramatically reduces the model’s computational cost.

**B. DENSELY CONNECTED ATROUS PYRAMID MODULES**

As the input feature map undergoes a series of convolution and down-sampling, its resolution is gradually reduced. At present, there are many algorithms [14], [29]–[31] that use either long or short skip connections to fuse the context information of different depth layers, thus making up for the detailed spatial information lost in the encoding process (for example, edges, boundaries, etc.), and refining the segmentation results. This method enables the model to combine both the fine and coarse layers to make local predictions that follow the global structure while ensuring the robustness and accuracy. Besides, each layer using skip connections is able to directly obtain gradients from the loss function and the original input signal, thereby realizing the implicit depth supervision.

Inspired by DenseNet [32], the DCAP module is built, as shown in Fig. 3, which is able to capture multi-scale features in the encoding stage, thus strengthening the reuse of information, and eliminating the gridding artifacts caused by atrous convolution in the AP bottleneck block. In the meantime, the module additionally fuses the feature information learned in the AP bottleneck block hierarchically. Both the shallow and the deep feature representations are combined at each stage of the hierarchy, thus making the model give a smoother decision boundary. Considering that using channel concatenation to achieve dense connections will cause the number of feature maps to increase with the gradual deepening of the network, this will bring about a surge in the model parameters and bring enormous difficulties for real-time semantic segmentation task with limited computing resources. Therefore, we adopt a dense connection operation of element-level addition with low computational cost to encode each network stage.

The output of the traditional convolutional network at the \( p^{th} \) layer is shown as follows:

\[
 x_p = F_p(x_{p-1}).
\]  

(1)

where \( x_p \) refers to an output of the \( p^{th} \) layer, \( x_{p-1} \) represents an input of the \( p^{th} \) layer, and \( F_p \) represents the nonlinear translation function, which is a combination of operations, including Batch Normalization (BN), Parametric Rectified Linear Unit (PReLU), and Convolution (Conv). However, the DCAP module will take the feature map of the sum of the element levels of all the previous layers as input. For example, the output of the DCAP module at the \( p^{th} \) layer is expressed as follows:

\[
 x_p = F_p(x_0 + x_1 + \ldots + x_{p-1}).
\]  

(2)

where, \( x_0 + x_1 + \ldots + x_{p-1} \) denotes that the feature maps of all layers before the \( p^{th} \) layer are the summation at the element level, and then the summation feature maps are used as the input of the \( p^{th} \) layer. Considering that the DCAP module needs to perform element-level addition, it is necessary to keep the feature map size and channel consistent.

**C. CROSS FUSION MODULE**

As per some previous work, only simple bilinear upsampling or transposed convolution is used in most methods [5], [6], [29] to obtain high-resolution segmentation images from low-resolution semantic feature maps. The segmentation results obtained with these two methods are often very rough, indicating that an overly simple decoding structure may lead to sub-optimal results.

Shallow features are characterized by high resolution, less semantic information, rich spatial detail information, and more noise. Deep features are characterized by low resolution, rich semantic information, and insufficient awareness of details. Due to the differences and diversity between the two features, simple pixel summation or channel merging will lower the model’s segmentation performance.
Based on the above observations, the advantages of the attention mechanism are taken to propose the CF module, which is able to effectively integrate the complementary information between the high-level semantics and the low-level details to refine the prediction results, as shown in Fig. 4.

The cross-fusion method guided by high-level semantic features is available to capture feature representations of different scales and enhance information flow among different layers. On the other hand, the CF module strengthens the sensitivity of high-level semantic features to low-level spatial information and expands the perception of low-level detail information to high-level semantic information. Compared with simple operations, such as channel concatenation and element-level addition, the CF module makes full use of context information to make the final result more effective.

D. THE NETWORK ARCHITECTURE

The overall network architecture of LDPNet is shown in Fig. 5, which is a typical encoder-decoder model.

The encoder part of the network consists of three stages. In the first stage (Stage1), three $3 \times 3$ ordinary 2D convolutions are employed to capture the image’s initial features. In the second stage (Stage2), we use four AP bottleneck blocks to build the DCAP module, and the atrous ratios of all AP bottleneck blocks are 1, 2, 5, and 9 respectively. While in the third stage (Stage3), eight AP bottleneck blocks are adopted to construct the DCAP module. The atrous rates of the first four AP bottleneck blocks are all 1, 2, 5, and 9, and those of the last four AP bottleneck blocks are all 2, 5, 9, and 17. The numbers of channels in each stage are 32, 64 and 128 respectively, and the dense connection method of element-level addition is used in each encoding stage. Such a design is available to capture multi-scale contextual information, strengthen the flow of information among layers, and make up for the loss of spatial detail in the encoding process.

The CF module is proposed in the decoder part, which can enhance the sensitivity of high-level semantic features to low-level spatial information and the perception of low-level detail information to high-level semantic information. Finally, two standard convolutions and bilinear upsampling are utilized as segmentation heads (as shown in Fig. 6) to obtain the final segmentation results. The detailed information of the network architecture is shown in Table 1.

![FIGURE 4. The proposed CF module structure. Note: DS-Conv is a $3 \times 3$ depthwise separable convolution, $H$, $W$ refers to the original input image’s height and width respectively, $C_1$, $C_2$, and $C_3$ represent the number of channels of different input branches respectively, AvgPool is average pooling, and $UpNx$ represents Perform $N$ times bilinear upsampling. $σ(·)$ represents the sigmoid activation function, mul means element-level multiplication, and add denotes element-level addition.](image-url)

| Layer | Operator | Output size |
|-------|----------|-------------|
| 1     | Down-Sampling units | $256 \times 512 \times 32$ |
| 2-4 (Stage1) | $3 \times 3$ Conv | $256 \times 512 \times 32$ |
| 5     | Down-Sampling units | $128 \times 256 \times 64$ |
| 6-9 (Stage2) | AP block ($r=1, 2, 5, 9$) | $128 \times 256 \times 64$ |
| 10    | Down-Sampling units | $64 \times 128 \times 128$ |
| 11-14 (Stage3) | AP block ($r=1, 2, 5, 9$) | $64 \times 128 \times 128$ |
| 15-18 (Stage3) | AP block ($r=2, 5, 9, 17$) | $64 \times 128 \times 128$ |
| 19    | CF module | $128 \times 256 \times 64$ |
| 20    | segmentation head | $512 \times 1024 \times C$ |

TABLE 1. The Detailed Structure of LDPNet. The Input Size is $512 \times 1024$ and $C$ Refers to the Number of Output Categories.
Thanks to the ability of the down-sampling operation to expand the network’s receptive field, it is also available for the reduction of the computational cost. The encoder part of our system also uses three down-sampling units, and each down-sampling unit consists of a $3 \times 3$ convolution with a step of 2 and a $2 \times 2$ maximum pooling unit parallel output additively, as shown in Fig. 7.

IV. EXPERIMENTS
In the sections of 4.1-4.3, the Cityscapes data set, Camvid data set, the experiment’s implementation details, and the loss function are introduced. While in Section 4.4, the influence of different parts of LDPNet on the experimental results is analyzed. In Section 4.5, a comparative analysis of LDPNet and other models is given in terms of Mean Intersection over Union metric (mIoU), parameters, computational complexity (FLOPs), and inference speed (FPS).

A. DATASET
1) CITYSCAPES
As a large-scale data set, Cityscapes[33] focus on semantic understanding of urban street scenes, including street scenes of 50 cities in different environments, backgrounds, and seasons. It provides 5,000 finely labeled images, 20,000 coarse labeled images, and 30 types of labeled objects, among which 19 classes are utilized for semantic segmentation. Besides, among the finely labeled images, 2,975 images are used for training, 500 images are used for validation, and 1,525 images are used for testing. The Cityscapes data set does not provide the label of the test set, thus ensuring the fairness of the experiments. In this case, the images predicted by the model have to be converted into 34 labelIDs which will then be uploaded to the official evaluation website to obtain the results. Due to the high resolution of this data set, that is, $1024 \times 2048$, and the existence of similar semantic categories (such as Car and Truck, Person and Rider, Motorcycle and bicycle), it poses a significant challenge to real-time semantic segmentation.

2) CAMVID
As the first video collection with semantic labels for target categories, Cambridge-driving Labeled Video Database (Camvid) contains 701 images with a resolution of $720 \times 960$ extracted from video sequences. Among the 32 candidate categories, 11 categories are employed for semantic segmentation. For a better comparison with the previous work, the same division method is adopted as in [5], [6], with 367 pictures for training, 101 pictures for validation, and 233 pictures for testing. Besides, to make a fair comparison
with other methods, images with the resolution of 360 × 480 are used for training and testing in the experiment.

### B. THE EXPERIMENTAL DETAILS

All the experiments are performed based on the following specifications, including one NVIDIA GeForce GTX 1080Ti GPU, PyTorch 1.5.0, CUDA 10.1, and cuDNN 7.6.5. The convolution layer of the model adopts the initialization method of "Kaiming normal." We use the Apex mixed-precision acceleration model training based on PyTorch and developed by NVIDIA to make full use of GPU memory. The batch size is 8, and the Adam optimizer is used to train the model with the weight decay of $1 \times 10^{-4}$. Based on some previous work [15]–[18], we also adopt the "poly" learning rate strategy, as shown below:

$$lr \times (1 - \frac{\text{epoch}}{\text{max epoch}})^{\text{power}}$$

where, the $\text{epoch}$ represents the current number of cycles, the $\text{max epoch}$ refers to the maximum number of cycles (here 500), the $lr$ refers to the initial learning rate which is set to $6 \times 10^{-4}$ here, and the $\text{power}$ is 0.9.

Inspired by the literature [5], [6], category weights on the Camvid data set are used to improve the class imbalance problem, which is defined as follows:

$$W_{\text{class}} = \frac{1}{\ln(c + p_{\text{class}})}$$

where, $c$ refers to an additional hyperparameter set to 1.12, and $p_{\text{class}}$ represents each category’s probability.

Data augmentation strategies, such as random horizontal flipping, multi-scale, and random cropping to a fixed size are adopted for input images during training to prevent model overfitting and improve the generalization ability. The random scale contains $\{0.75, 1.0, 1.25, 1.5, 1.75, 2.0\}$. It is worth noting that we don’t use any additional data sets to pre-train our network.

### C. LOSS FUNCTION

In this article, we use an auxiliary loss function to supervise the training of the model, which needs very little computational cost to improve the model’s feature expression in the training stage, and can be removed in the model inference stage. Therefore, in addition to obtaining the primary loss ($\text{loss}_m$) at the end of the entire model, a segmentation head at the end of the encoder is also used to get the auxiliary loss ($\text{loss}_a$). The specific operation is defined as follows:

$$\text{loss} = \text{loss}_m + \lambda \cdot \text{loss}_a$$

where, $\lambda$ refers to the weight of the additional loss in the third stage. As shown in Table 2, the best result is obtained at $\lambda$ of 0.5, which is higher than that without the auxiliary loss function by 0.75%.

We adopt the Online Hard Example Mining algorithm [34] at the time of cityscapes data set training. It selects the part with a large difference between the predicted result and the ground truth (i.e., hard example), and then implements back-propagation and retraining of the selected hard examples.

### D. ABLATION STUDIES

In this section, a series of ablation experiments are performed to demonstrate the effectiveness of our model. In addition, the Cityscapes data set is adopted for the implementation of quantitative and qualitative analyses. All experiments are done on Cityscapes training set, validation set, and testing set.

#### 1) ABLATION STUDY FOR DECODER

Content modules, such as ASPP and PPM are widely used to capture contextual information of different feature scales. ASPP and PPM are used as the decoder and LDPNet encoder to build two LDPNet variant networks, in this way to verify whether our CF module is effective. Table 3 shows that the CF module increases the mIoU of the baseline network by 2.03%, i.e., an increase of 1.16% compared with the ASPP module. By combining Table 4, it can be easily found that using the CF module at the end of the network can achieve the most outstanding performance improvement at the lowest computational cost. Although PPM and CF modules are almost the same in mIoU and FPS, the latter has the parameter

| Loss Weight($\lambda$) | mIoU (%) |
|------------------------|----------|
| 0                      | 69.71    |
| 0.25                   | 70.21    |
| 0.5                    | 70.46    |
| 0.75                   | 70.32    |
| 1                      | 70.16    |

### TABLE 3. Evaluation Results of Different Decoders. We Show the Inference Speed (FPS), Segmentation Accuracy (mIoU), and Parameters Under the Input Size of 512 × 1024. Note: Baseline is the Encoder of the Network.

| Baseline | SegHead | CF | ASPP | PPM | mIoU(%) | FPS | Params(M) |
|----------|---------|----|------|-----|---------|-----|-----------|
| ✓        | ✓       | ✓  | ✓    | ✓   | 68.43   | 102 | 0.71      |
| ✓        | ✓       | ✓  | ✓    | ✓   | 69.01   | 97  | 0.86      |
| ✓        | ✓       | ✓  | ✓    | ✓   | 70.46   | 87  | 0.80      |
| ✓        | ✓       | ✓  | ✓    | ✓   | 69.30   | 80  | 1.93      |
| ✓        | ✓       | ✓  | ✓    | ✓   | 70.40   | 87  | 1.67      |

### TABLE 4. Performance Evaluation of Different Decoder Modules.

| Module | GFLOPs | Params(K) |
|--------|--------|-----------|
| CF     | 0.55   | 16.80     |
| ASPP   | 3.77   | 476.4     |
| PPM    | 6.04   | 804.1     |
which is 47.8 times lower than that of the former, besides, the computational complexity is 11 times lower.

2) ABLATION STUDY FOR ENCODER DEPTH
A series of experiments are conducted to explore the influence of the number of AP bottleneck blocks in the DCAP module on the segmentation effect, utilizing DCAP modules with different numbers of AP bottleneck blocks in the second and third stages of LDPNet. Furthermore, the mIoU, FPS, and model parameters under different conditions are illustrated in Table 5, from which it can be seen that the number of AP bottleneck blocks has a more significant impact on the model in the third stage than in the second stage. This is because that the AP bottleneck block in the third stage has a larger receptive field and more semantic features. It can also be seen from Table 5 that when the number of AP bottlenecks in the third stage increases to 9, mIoU starts to decrease instead. Therefore, to achieve a better balance among speed, parameters, and accuracy, four AP bottleneck blocks are used in the second stage and eight are used in the third stage.

TABLE 5. Evaluation Results of Different Encoder Depth. FPS are Estimated on 512 × 1024 Input.

| Stage2 | Stage3 | mIoU(%) | Params(M) | FPS |
|--------|--------|---------|-----------|-----|
| 3      | 7      | 69.35   | 0.72      | 100 |
| 3      | 8      | 69.16   | 0.79      | 94  |
| 3      | 9      | 69.39   | 0.86      | 87  |
| 4      | 7      | 69.23   | 0.73      | 93  |
| 4      | 8      | 70.46   | 0.80      | 87  |
| 4      | 9      | 70.24   | 0.87      | 84  |

3) ABLATION STUDY FOR ATROUS RATES
Parallel convolutions with different atrous rates are used in the AP bottleneck blocks, as shown in Table 6. Due to the fact that the atrous convolution obtains a larger receptive field without adding additional parameters, the large receptive field is able to capture more surrounding feature information and learn more multi-scale features, therefore, our network is available to achieve good results. Inspired by the literature [24], multiple convolution kernels with the same atrous rate are left unused by us because this will produce a grid effect, and further lead to the unsatisfactory final segmentation result. As shown in Table 6, increasing the atrous rate in the third stage is better than in the second stage. When the atrous rate in the third stage is increased from 1, 2, 4, 8 to 1, 2, 5, 9, and 2, 5, 9, 17 respectively, the model’s mIoU is increased by 1.59%.

E. COMPARISON WITH STATE-OF-THE-ARTS
In this section, LDPNet is compared with the most advanced real-time semantic segmentation model in terms of inference speed, parameters, computational complexity, and accuracy. As per the results given in Table 7, the advancement and effectiveness of the LDPNet model are proved. As one of the most advanced models at present that balance speed, accuracy, and parameters, our model uses only 0.8M parameters to achieve 71.1% class mIoU, 87.2 category mIoU, and 87 FPS.

TABLE 6. Evaluation Results of Different Atrous Rates on Cityscapes Test Set.

| Dilation rates (stage2) | Dilation rates (stage3) | mIoU(%) |
|-------------------------|-------------------------|---------|
| 1,2,4,8                 | 1,2,4,8                 | 68.87   |
| 1,2,4,8                 | 1,2,5,9                 | 69.41   |
| 1,2,5,9                 | 1,2,5,9                 | 69.88   |
| 1,2,5,9                 | 2,5,9,17               | 70.46   |

TABLE 7. Comprehensive Evaluation Results on the Cityscapes Test Set. "-" Indicates That the Corresponding Data are not Given in the Original Paper. Pretrain Denotes Whether the Model Has Been Trained on the ImageNet Data Set.

| Method      | Input Size | Pretrain | mIoU(%) | FPS | GFLOPs | Params(M) |
|-------------|------------|----------|---------|-----|--------|-----------|
| SegNet[6]   | 360 × 640  | Yes      | 56.1    | 15  | 286    | 29.5      |
| ENet[5]     | 360 × 640  | No       | 58.3    | 135.4 | 3.8   | 0.36      |
| SQ[35]      | 1024 × 2048| Yes      | 59.8    | 16.7 | 270    | -         |
| ESPNet[7]   | 512 × 1024 | No       | 60.3    | 112.9 | -0.36 | -         |
| ThunderNet[36]| 512 × 1024 | No       | 64.0    | 96.2 | -4.7   | -         |
| CENet[37]   | 1024 × 2048| No       | 64.8    | 17.6 | -0.49  | -         |
| ContextNet[38]| 1024 × 2048| No       | 66.1    | 24   | 48.3   | 0.85      |
| EDANet[39]  | 512 × 1024 | No       | 67.3    | 108  | -0.68  | -         |
| ADSNet[31]  | 512 × 1024 | No       | 67.5    | 76.9 | 8.2    | -         |
| LiteSeg[11] | 360 × 640  | Yes      | 67.8    | 161  | 4.9    | 4.38      |
| PFPNet[4]   | 512 × 1024 | No       | 68.0    | 102  | 5.7    | 0.4       |
| ERPNNet[40] | 512 × 1024 | No       | 68.0    | 41.7 | -      | 2.1       |
| Fast-SCNN[41]| 1024 × 2048| No       | 68.0    | 123.5| -1.11  | -         |
| BiSeNet[16] | 768 × 1536 | Yes      | 68.4    | 105.8 | 14.8  | 5.8       |
| DSNet[42]   | 512 × 1024 | Yes      | 69.1    | 68   | -      | 6.5       |
| LEDNet[8]   | 512 × 1024 | No       | 69.2    | 71   | -0.94  | -         |
| ICNet[15]   | 1024 × 2048| Yes      | 69.5    | 30.3 | 28.3   | 2.65      |
| FarSee-Net[43]| 512 × 1024 | Yes      | 70.2    | 68   | -      | -         |
| GUNet[44]   | 512 × 1024 | Yes      | 70.4    | 33   | -      | -         |
| Ours        | 512 × 1024 | No       | 71.1    | 87   | 14.4   | 0.8       |

It can be seen from Table 7 that, for FarSee-Net and GUNet with the accuracy slightly lower than that of LDPNet, the inference speed of LDPNet is significantly higher. Compared with ICNet, the resolution of the input image of our model is just half, and the parameters of the model are ten times less, however, the inference speed is almost three times faster, and the segmentation accuracy is higher by 1.6%. Besides, no additional data set is used for pretraining.
TABLE 8. Class mIoU Scores on Cityscapes Test Set for the Per-Class Category.

| Method   | Roa | Sid | Bui | Wal | Fen | Pol | TLi | TSi | Veg | Ter | SKy | Ped | Rid | Car | Tru | Bus | Tra | Mot | Bic | clo | Cat |
|----------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| SegNet[6]| 96.4| 73.2| 84.0| 28.4| 29.0| 35.7| 39.8| 45.1| 87.0| 63.8| 91.8| 62.8| 42.8| 89.3| 38.1| 43.1| 44.1| 35.8| 41.9| 57.0| 79.1|
| ENet[5]  | 96.3| 74.2| 75.0| 32.2| 33.2| 43.4| 34.1| 44.0| 88.6| 61.4| 90.6| 65.5| 38.4| 90.6| 36.9| 50.5| 48.1| 38.8| 55.4| 58.3| 80.4|
| ESPNet[7]| 97.0| 77.5| 76.2| 35.0| 36.1| 45.0| 35.6| 46.3| 90.8| 63.2| 92.6| 67.0| 40.9| 92.3| 38.1| 52.5| 50.1| 41.8| 57.2| 60.3| 82.2|
| CGNet[37]| 95.5| 78.7| 88.1| 40.0| 43.0| 54.1| 59.8| 63.9| 89.6| 67.6| 92.9| 74.9| 54.9| 90.2| 44.1| 59.5| 25.2| 47.3| 60.2| 64.8| 85.7|
| LRFNet[40]| 97.2| 80.0| 89.5| 41.6| 45.3| 56.4| 60.5| 64.6| 91.4| 68.7| 94.2| 76.1| 56.4| 92.4| 45.7| 60.6| 27.0| 48.7| 61.8| 66.3| 85.2|
| LEDNet[8] | 97.1| 78.6| 90.4| 46.5| 48.1| 60.9| 60.4| 71.1| 91.2| 60.0| 93.2| 74.3| 51.8| 92.3| 61.0| 72.4| 51.0| 43.3| 70.2| 69.2| 86.8|
| ICNet[15] | 97.1| 79.2| 89.7| 43.2| 48.9| 61.5| 60.4| 63.4| 91.5| 68.3| 93.5| 74.6| 56.1| 92.6| 51.3| 72.7| 51.3| 53.6| 70.5| 69.5| 86.4|
| Ours     | 98.0| 82.2| 90.8| 46.9| 46.6| 56.2| 61.3| 66.9| 92.0| 70.2| 94.5| 79.5| 60.3| 93.9| 56.8| 74.0| 62.2| 52.5| 65.8| 71.1| 87.2|

FIGURE 8. Visual comparisons in terms of the Cityscapes validation set. From left to right are input images, ground truth, segmentation outputs from ESPNet, and our LDPNet.

It can be observed from Table 7 that the parameters of LDPNet are significantly smaller than those of most real-time semantic segmentation methods, and the results obtained are much higher than those from other models. This proves that the parameters utilization rate of the LDPNet is rather high without excessive parameter redundancy.
Table 8 lists all the individual category results, that is, LDPNet achieved the best scores in 13 out of the 19 categories, and the scores of the remaining 6 categories are also higher than those of most other networks. Compared with SegNet [6], LDPNet achieves an improvement of more than 15% for small objects (for example, pole, traffic sign, bicycle, etc.), and about 2% for large ones (for example, sky, sidewalk, road, etc.). For example, the traffic light rises from 39.8% to 61.3%. The experimental results show that the reuse of features and the larger receptive field in the coding stage can significantly improve the network’s segmentation performance.

The visualization result of the Cityscapes validation set is shown in Fig. 8. Thanks to the large receptive field of LDPNet, more contextual information is encoded in the downsampling stage. During the up-sampling process, high-level features and low-level details are complementarily fused. Compared with ESPNet, LDPNet presents a good segmentation effect on both large objects and small targets. For example, in the first row of Fig. 8, the segmentation effect of LDPNet on Pole and Car is significantly better than that of ESPNet. While, in the second row, ESPNet segments the Truck incorrectly, however, the segmentation result of LDPNet is perfect. In the third row, LDPNet is also significantly better than ESPNet in terms of boundary segmentation between Vegetation and Road.

As shown in Table 9, compared with other models on the Camvid data set, LDPNet once again achieved excellent performance in the case of low-resolution input, and can quickly segment high-precision pictures with an inference speed of 256FPS. The qualitative results of the Camvid test set are shown in Fig. 9.

V. CONCLUSION

The DCAP module and the CF module are proposed for the segmentation of complex urban road scenes. The DCAP module can effectively increase the receptive field of the model and extract contextual features. While the CF module uses rich high-level semantic features to guide the fusion of low-level spatial information, thereby effectively improving the model’s segmentation ability. Based on these two modules, LDPNet is carefully designed, which is available to achieve significant segmentation effects under the premise of ensuring fast inference speed and small parameters. The experimental results on two challenging urban road scene datasets (Cityscapes and Camvid) show that the LDPNet is available to achieve the best trade-off between segmentation accuracy and speed. Specifically, LDPNet achieves 71.1% mIoU and 87 FPS on the Cityscapes dataset with only 0.8M parameters.

REFERENCES

[1] S. Choi, J. T. Kim, and J. Choo, “Cars can’t fly up in the sky: Improving urban-scene segmentation via height-driven attention networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Seattle, WA, USA, Jun. 2020, pp. 9370–9380, doi: 10.1109/CVPR42600.2020.00939.
[2] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Honolulu, HI, USA, Jul. 2017, pp. 6230–6239, doi: 10.1109/CVPR.2017.660.
[3] 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, doi: 10.1109/TPAMI.2017.2699184.
[4] M. Liu and H. Yin, “Feature pyramid encoding network for real-time semantic segmentation,” 2019, arXiv:1909.08599. [Online]. Available: http://arxiv.org/abs/1909.08599
[5] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, “ENet: A deep neural network architecture for real-time semantic segmentation,” 2016, arXiv:1606.02147. [Online]. Available: http://arxiv.org/abs/1606.02147
[6] V. Badrinarayanan, A. Kendall, and R. Cipolla, “SegNet: A deep convolutional encoder-decoder architecture for image segmentation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 12, pp. 2481–2495, Dec. 2017, doi: 10.1109/TPAMI.2016.2644615.
[7] S. Mehta, M. Rastegari, A. Caspi, L. G. Shapiro, and H. Hajishirzi, “ESPNet: Efficient spatial pyramid of dilated convolutions for semantic Segmentation,” in Proc. Eur. Conf. Comput. Vis. (ECCV), Munich, Germany, 2018, pp. 561–580.

## Table 9. Evaluation Results on the Camvid Test Set. “-” Indicates That the Corresponding Data Are Not Given in the Original Paper.

| Method   | Input Size | Pretrain | mIoU (%) | FPS | Params (M) |
|----------|------------|----------|----------|-----|------------|
| ENet[5]  | 360 × 480  | No       | 51.3     | -   | 0.36       |
| SegNet[6]| 360 × 480  | No       | 55.6     | 46  | 29.5       |
| ERPNNet[49]| 360 × 480 | No       | 60.5     | 133 | 2.07       |
| ESPNet[7]| 360 × 480  | No       | 62.6     | 205 | 0.68       |
| DFANet[18]| 720 × 960 | Yes      | 64.7     | 120 | 7.8        |
| BiSENet[16]| 720 × 960 | No       | 65.6     | -   | 5.8        |
| CGNet[37]| 360 × 480  | Yes      | 65.6     | -   | 0.5        |
| SwiftNet[45]| 720 × 960 | Yes      | 65.7     | -   | 12.9       |
| EDANet[39]| 360 × 480  | No       | 66.4     | 163 | 0.68       |
| ICNet[15]| 720 × 960  | Yes      | 67.1     | 27.8| 26.5       |
| Ours     | 360 × 480  | No       | 67.3     | 256 | 0.8        |

FIGURE 9. Qualitative examples of the segmentation on Camvid test set. From left to right are input images, ground truth, segmentation outputs from ENet, and our LDPNet.
XUEGANG HU was born in Chongqing, China, in 1965. He received the B.S. degree in applied mathematics from China West Normal University, China, in 1988, and the M.S. and Ph.D. degrees in applied mathematics from Sichuan University, China, in 1995 and 2006, respectively.

From 2002 to 2008, he was an Associate Professor with the Chongqing University of Posts and Telecommunications, China, where he has been a Professor since 2008. He has authored three books, and more than 80 articles and three inventions. His research interests include digital image processing and analysis and partial differential equations and their applications.

LIYUAN JING was born in Sichuan, China, in 1997. He received the B.S. degree in Internet of Things engineering from the Chengdu College of University of Electronic Science and Technology of China, in 2019. He is currently pursuing the M.S. degree in electronics and communication engineering with the Chongqing University of Posts and Telecommunications. His research interests include computer vision, deep learning, panoramic segmentation, and semantic segmentation.

***