International Journal for Housing Science and Its Applications
Publish August 6, 2025. Volume 46, Issue 3 Pages 7744-7754

https://doi.org/10.70517/ijhsa463664

Research on RGB-D urban scene segmentation technique based on full convolutional network

Xiangling Ma[1], Xiangyang Ma[2,*] and Minghui Qiu[1]
[1] School of Information Technology&Engineering, Guangzhou College of Commerce, Guangzhou, Guangdong, 511363, China
[2] Human Resources Office, Shandong Jianzhu University, Jinan, Shandong, 250101, China
Corresponding authors: (e-mail: 18615238728@163.com).

Abstract With the rapid development of urban construction, accurate decoding of urban scenes has become particularly important in the fields of urban infrastructure planning, intelligent transportation management, and environmental monitoring. In this study, a dynamic adaptive convolution-based RGB-D urban scene segmentation algorithm (FastDVFN) is proposed, which aims at solving the limitations of the existing full convolutional neural network (FCN) methods in terms of efficiency and real-time performance. The method combines RGB-D point cloud feature extraction with adaptive convolution mechanism, and optimizes the parameter tuning of the convolutional layer to improve the segmentation accuracy and computational efficiency. On the Cityscapes dataset, the FastDVFN network achieves an mIoU metric of 72.9%, which improves the accuracy by 5.2% compared to the traditional method. In terms of running speed, it reaches 88 frames/s, outperforming other similar lightweight semantic segmentation networks. In the experiment, the number of parameters of FastDVFN network is 0.63m, which is significantly reduced compared to other methods. The effectiveness of the dynamic adaptive convolution and enhanced channel feature normalization (ECFN) module is demonstrated through comparative experiments and analysis of ablation experiments. The results show that the algorithm has strong real-time processing capability while maintaining high accuracy and can meet the practical application requirements of urban scenes.

Index Terms RGB-D, urban scenes, segmentation algorithm, dynamic adaptive convolution, accuracy, real-time

I. Introduction

The urban scene segmentation problem is essentially an image recognition problem, but the requirements for image recognition are higher, recognizing the category and location information of all pixels in the image of the perceived urban scene, including the small targets and target boundaries in the image need to give accurate recognition and localization results [1]-[4]. With the rapid development of cities, high population mobility, and intensive infrastructure construction, efficient and accurate segmentation of images in this scene is of great significance for urban infrastructure construction, urban ecological environment monitoring, urban transportation management, urban natural resources monitoring, and urban security [5]-[8].

With the rapid development of deep learning technology and the successive release of datasets in recent years, the performance of image semantic segmentation technology has been improved by leaps and bounds, but it has not yet met the requirements of urban scenes for high-quality segmentation technology [9]-[11]. In urban scenes, the shooting distance is far and the angle is wide resulting in image data with complex environments, a wide range of small target categories, a large number of them, large scale changes, light changes, and data imbalance [12]-[14]. Small targets are the focus and difficulty of image segmentation in urban scenes, and its efficient and accurate segmentation is of great significance for accurate monitoring and scientific management of smart cities [15], [16]. And with the popularization of full convolutional neural network, it makes great breakthroughs in the field of image semantic segmentation, such as PSPNet, DeepLabv3+, etc., especially the RGB-D urban scene segmentation technology based on full convolutional network, which plays an important role in promoting the development of urban scene segmentation [17]-[20].

With the rapid development of information technology, the interpretation of urban scenes has become the core content of urban planning and management. Semantic segmentation of urban scenes, as an important part of image understanding, aims to accurately delineate different types of urban elements, such as buildings, roads, and green spaces. However, traditional convolutional neural networks (CNNs) face the trade-off between computational efficiency and accuracy when dealing with these complex scenes, which limits their promotion in practical applications. Therefore, how to improve the computational efficiency and real-time performance of the algorithm while ensuring higher segmentation accuracy has become an important direction of current research. In this study, an image segmentation framework based on full convolutional neural network (FCN) is adopted and innovatively improved by introducing dynamic adaptive convolution (DAConv) and enhanced channel feature normalization (ECFN) techniques. By adaptively adjusting the input image features, the DAConv module improves the adaptability of the convolution kernel so that it can capture the detailed information in the image more accurately. The ECFN module, on the other hand, optimizes the feature learning capability of the model by combining the advantages of batch normalization and layer normalization. In addition, the study also combines RGB-D point cloud features to improve the segmentation accuracy of the model for complex urban scenes by fusing depth and color information. The experimental results show that the improved FastDVFN network outperforms multiple existing semantic segmentation methods on the Cityscapes dataset, with high segmentation accuracy and low computational complexity, and can meet the demands of real-time applications.

II. Urban scene segmentation algorithm based on full convolutional neural network

Accurate interpretation of urban scenes is crucial for applications such as urban infrastructure planning, urban intelligent transportation management, and urban environment monitoring and management. Semantic segmentation of urban scenes aims to accurately and efficiently segment the image elements in urban scenes to provide the basis for the interpretation of urban scenes. Convolutional neural network can be said to play a driving role in the field of semantic segmentation of urban scenes, and this chapter will focus on introducing the improved image semantic segmentation method based on convolutional neural network.

II. A. Convolutional Neural Networks

The computation of the l th convolutional layer in a convolutional neural network is based on the output of the input layer or its previous pooling layer, which can be expressed as Equation (1) [21]:

$$x_j^l = f\left(\sum_{i\in M_j} x_i^{l-1} * k_{ij}^l + b_j^l\right) \tag{1}$$

In the above equation, x_j^l denotes the j th feature map of the l th layer, x_i^{l-1} denotes the i th feature map of the $l-1$ th layer, and the range of values of i, M_j, denotes the set of features that are selected as the inputs, and thus the obtained output feature maps synthesize the values of multiple features. The k_{ij}^l denotes the convolution kernel of the j th feature map in the l th layer, each feature map i corresponds to its own convolution kernel, which is convolved and summed with the corresponding features selected in the $l-1$ th layer respectively, together with the bias parameter b_j^l, and finally the output is calculated by the activation function f.

The pooling layer is after the convolutional layer, so if the l th layer is a pooling layer, then the formula for this pooling layer is as in equation (2):

$$x_j^l = f\left(\beta_j^l down(x_j^{l-1}) + b_j^l\right) \tag{2}$$

where β denotes the weights, which are generally of fixed size for the pooling layer. $down$ is the sampling function.

In summary, the convolutional neural network in order to reduce the network model training parameters to propose a solution - local connection and weight sharing, these two methods greatly reduce the number of parameters to train the model, reducing the complexity of the model.

II. B. Improved Semantic Segmentation Method for Urban Scenes

In order to be able to realize pixel-level semantic classification, the traditional segmentation method based on convolutional neural network, CNN must use a piece of local urban scene image around a certain pixel as an input for training and prediction, which exists the problems of computational inefficiency due to repeated computation, and the size of the image block restricts the size of the perceptual domain, for this reason, in this paper, we introduce a full convolutional neural network (FCN) model for the semantic segmentation of the image, which has no requirement on the input image and can obtain the same size of output image as the input image. There is no requirement on the size of the input image, and it can get the output image with the same size as the input image, which ensures the relative position and spatial information of the objects in the input image. In the following, three features of the full convolutional neural network are introduced: convolutionalization, up-sampling, and layer-hopping connections.

(1) Convolutionalization

FCN convolutionalization simply means replacing the three fully connected layers in a traditional CNN with convolutional layers. The feasibility of the replacement lies in the fact that the implementation functions of the fully connected layers and the convolutional layers are the same, both computing the dot product between neurons, and it can be seen that the fully connected layers can be replaced by the convolutional layers. The reason for removing the fully connected layer is that firstly the fully connected layer needs to be trained with more parameters because the neurons in the convolutional layer are locally connected to each other and share parameters, whereas the neurons in the fully connected layer are fully connected to each other and the parameters cannot be shared. Secondly, the convolutional neural network can achieve the same effect as the fully connected layer by performing only one forward propagation calculation, and the convolutional layer is more practical than the fully connected layer.

(2) Up-sampling

The up-sampling mentioned in the previous paragraph in the full convolutional neural network is the inverse convolution calculation, the inverse convolution is the inverse process of the convolution, but the essence of the vector multiplication and addition operation, so the principle is actually very simple is the convolution of the forward propagation process and the backward propagation process under the position of the swap. After the completion of the forward propagation, the loss function $Loss$ can be obtained from the deeper network, according to the matrix of the differential formula $\frac{\partial Ax+b}{\partial x}=A^T$, can be known:

$$\frac{\partial Loss}{\partial x}=\frac{\partial Loss}{\partial Y}\times\frac{\partial Y}{\partial x}=C^T\frac{\partial Loss}{\partial Y} \tag{3}$$

The derivation of the above equation is the matrix operation of the backpropagation process. So the essential operation of the backpropagation is the left multiplication of the forward process by C^T and the left multiplication of the reverse process by $(C^T)^T$.

(3) Layer skipping connection

Full convolutional neural network after the full convolution of the result is relatively rough, directly after the convolution of the image of the deconvolution operation is only able to restore the features of the fifth layer of convolutional kernel, the semantic segmentation results obtained by the poor accuracy, in order to optimize the accuracy of the problem the authors have proposed a jump structure.

The full convolutional neural network has obvious advantages over the traditional convolutional neural network: there is no limitation on the size of the input image, the computation is more efficient, and repeated storage and computation can be avoided. But there are also shortcomings: firstly, the semantic segmentation results are not accurate and not ideal for detail processing; secondly, it lacks spatial consistency and does not fully consider the relationship between pixels.

III. Dynamic adaptive convolution based RGB-D urban scene segmentation algorithm

In this study, urban scene segmentation algorithms based on full convolutional neural networks were introduced in the previous paper, although they have made remarkable achievements in terms of accuracy, their large number of parameters and high computational complexity make it difficult for these models to meet the real-time processing requirements in the practical application of urban scene segmentation. Based on this situation, this paper combines the principle of RGB-D point cloud feature extraction, constructs a fast deep visual fusion network model (FastDVFN), and proposes an RGB-D urban scene segmentation algorithm based on dynamic adaptive convolution.

III. A. RGB-D point cloud

III. A. 1) Point cloud data classification

A 3D point cloud is generally a collection of data points containing coordinate depth information and color information acquired using 3D sensors such as Kinect and laser scanners. Due to the different data storage formats acquired by different 3D sensors, the data obtained will have different distribution and sorting structures, as well as different processing methods.

In order to better analyze the point cloud data, we classify the point cloud into four categories based on the spatial layout of the data points [22].

The first category is scattered point clouds, which are obtained by random scanning acquisition, with points arranged in an unorganized manner with no specific topological relationship between them.

The second type is the scanned line point cloud, which is acquired along a straight line using a laser scanner, and the scanned trajectories are linearly arranged. Compared with the scattered point cloud, this type of point cloud data is relatively more regular, and the linear scanned trajectories are retained in the point set, but the point clouds on each line are arranged in a disordered manner.

The third type is gridded point cloud, which is calculated by grid and interpolation on the basis of the original point cloud, a vertex in the matrix grid corresponds to each point in the data point set, which is also called matrix point cloud.

The fourth type is polygonal point cloud, which utilizes the data point set obtained from 3D sensors arranged in a plane parallel to each other, and they are connected to each other with straight lines, obtaining a planar polygonal structure with nesting and topological relationships, which belongs to the category of ordered point cloud.

III. A. 2) Feature extraction for RGB-D point cloud

(1) Normal vector feature

Normal vector features, also called normal vector features, are one of the most commonly used features in 2D image and 3D point cloud processing. There are two main methods for 3D point cloud normal vector estimation, PCA transform based normal vector algorithm and integral map based normal vector algorithm. Since the normal vector features obtained by the PCA transform-based normal vector algorithm are more accurate and convenient for the calculation of the next curvature features, the PCA transform-based normal vector algorithm is used in this paper to calculate the normal vector information of RGB-D point cloud data.

The normal vector algorithm based on PCA transform is also known as principal component analysis or microtangent plane method. The main idea of estimating normal vector based on PCA transform is:

If the surface of the point cloud is smooth (surface surface is continuous of high order), we can use principal component analysis to compute the plane under the least squares of the K domain of a point to represent the tangent plane of that point. Subsequently, the covariance matrix of that K-field is computed, and the eigenvector corresponding to the smallest eigenvalue in the covariance matrix is the normal vector of that point. Compared with the integral map method of calculating normal vectors, the PCA transform method is more accurate and more widely applicable (the integral map method of calculating normal vectors is only applicable to ordered point clouds, while the PCA transform method of estimating normal vectors of surfaces can be applied to both ordered and unordered point clouds).

In a smooth surface, given a point set $\{p_i, N_k(p_i)\}$, the k-field of an arbitrary point p_i in the point set is $N_k(p_i)$, and the representation of the least squares plane of the k-neighborhood fit to it is:

$$Pl(\vec{n}, d) = \arg\min \sum_{i-1}^{k} \left(\vec{n} \cdot p_i - d\right)^2 \tag{4}$$

$\vec{n}$ represents the normalized normal vector of the plane, i.e., $\|\vec{n}\| = 1$, and d represents the distance from the neighboring points to the plane.

Subsequently, the eigenvalue decomposition of the small squares plane is performed using the covariance matrix C, and the eigenvector corresponding to the smallest eigenvalue of the matrix C is the normal vector of the point p_i:

$$C = \begin{bmatrix} p - p_1 \\ p - p_2 \\ \vdots \\ p - p_n \end{bmatrix}^T \begin{bmatrix} p - p_1 \\ p - p_2 \\ \vdots \\ p - p_n \end{bmatrix} \tag{5}$$

The eigenvectors $\vec{v}_0$, $\vec{v}_1$, $\vec{v}_2$ resulting from the decomposition of the covariance matrix C are the same as the corresponding eigenvalues λ_0, λ_1, and λ_2 if the condition $\lambda_0 > \lambda_1 > \lambda_2$. Then the feature vector $\vec{n}$ corresponding to point p_i is denoted as:

$$\vec{n} = \vec{v}_0 \tag{6}$$

(2) Curvature feature

Curvature feature is an important feature in 2D image and 3D point cloud processing, which indicates the curvature of the surface, the normal vector change of the point is small when the curvature is small, indicating that the change of the region is relatively smooth; on the contrary, the normal vector change of the point is large when the curvature is large, indicating that the change of the surface of the region is large, so in the process of detecting the edge contour of the cloud, the curvature feature can be used as an important reference factor for judging the feature point. Therefore, the curvature feature can be used as an important reference factor in the judgment of feature points in the process of edge profile detection of point clouds.

The calculation of curvature feature in RGB-D point cloud is defined as [23]:

$$|k| = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2} \tag{7}$$

The eigenvalues λ_0, λ_1, and λ_2 have been found during normal vector computation, are the three eigenvectors corresponding to this covariance matrix, and satisfy the condition $\lambda_0 > \lambda_1 > \lambda_2$. And the center of mass of the point set $\{p_i, N_k(p_i)\}$ is $\vec{p} = \sum \frac{p_{ik}}{k}$, and if the direction of the normal vector of the point is set to be positive, the direction of the curvature of the point is determined by whether the point p_i is in the center of mass point $\vec{p}$ in front of or behind the center of mass point $\vec{p}$, if $p_i - \vec{p} > 0$, the curvature is positive, if $p_i - \vec{p} < 0$, the curvature is negative, the curvature sign $sign(k)$ is defined as:

$$sign(k) = sign\left(\left(p_i - \vec{p}\right)^T \cdot \vec{n}\right) \tag{8}$$

(3) Extraction of neighborhood information

When using LightGBM classifier training to predict the edge points of RGB-D point cloud data, the first step is to get the input data, and in this paper, we hope to get the data that contains the features of the point cloud's neighborhood, such as concavity and convexity, and Euclidean distance, in addition to the color, depth, normal vector, and curvature features. In this paper, we hope to use the image sliding window algorithm to obtain the information of the domain points of the point cloud and let the LightGBM classifier adaptively learn the neighborhood features of the point cloud.

III. B. Improved RGB-D urban scene segmentation algorithm

III. B. 1) Dynamic Adaptation Convolution

Convolutional neural networks are a core building block in deep learning techniques, they efficiently extract features from images of urban scenes by performing convolutional operations. These networks utilize a series of convolutional layers to process image data, with each layer capturing local dependencies and features by performing filtering operations on the input image. In traditional convolutional neural network architectures, the convolutional layers typically employ regular sampling grids to perform these operations, e.g., a typical 3×3 convolutional kernel would be applied in the form of a fixed-spaced grid over the input feature map to localize and extract features at the corresponding locations.

Let R denote the sampling grid, and instead of being a fixed structure, it can be adapted to the specific features of the input data. For example, in the 3×3 convolution operation, the definition of the regular sampling grid R is extended to accommodate features of different shapes and scales in the image, allowing the network to capture complex patterns and variations in the image more efficiently:

$$R = (-1,-1), (-1,0), \ldots, (0,1), (1,1) \tag{9}$$

The Dynamic Adaptive Convolution (DAConv) proposed in this chapter is able to automatically adjust its sampling shape by introducing a new coordinate generation algorithm. The algorithm is able to generate initial sampling coordinates for convolution kernels of different sizes and shapes.

However, the sampling grid is regular, whereas dynamically adapted convolution targets irregularly shaped convolution kernels. In order to allow irregular convolution kernels to have sampling grids, an algorithm is created for convolutions of arbitrary size that generates the initial sampling coordinates for the convolution P_n, the flow of the algorithm is shown in Algorithm 1. The sampling grids are first generated as regular sampling grids, then irregular grids are created for the remaining sampling points, and finally they are connected to generate the overall sampling grid. It is shown to generate initial sampling coordinates for convolution of arbitrary size. The sampling grid for the regular convolution is centered at the point (0, 0), and the point (0, 0) in the upper left corner is set as the sampling origin in the algorithm in order to accommodate the size of the convolution used.

After defining the initial coordinates P_n for the irregular convolution, the convolution operation corresponding to position P_0 can be defined as follows:

$$Conv(P_0) = \sum w \times (P_0 + P_n) \tag{10}$$

In Eq. (10), w denotes the convolution parameter. Irregular convolution operation is impossible to realize, because irregular sampling coordinates cannot be matched with the corresponding size convolution operation.

Dynamically adapted convolution breaks through the limitations of traditional convolution kernels in terms of number and shape, and is able to realize the operation using an arbitrary number of convolution kernels and arbitrary shapes. Unlike the depth-separable convolution designed only for specific object shapes, dynamic adaptive convolution does not stop at the exploration of fixed size and shape. The core of its design lies in solving the related problems by introducing the offset mechanism, which enables the convolution operation to effectively capture the feature information of irregular samples and supports an arbitrary number of convolution parameters.

III. B. 2) Enhanced Channel Feature Normalization

Batch normalization (BN) is widely used as a standard normalization method in various convolutional neural networks, which effectively speeds up the model training process and mitigates the overfitting problem, but exhibits a high sensitivity to the batch size and is susceptible to changes in the sample appearance. Although batch normalization reduces image-level variance, it fails to normalize channel feature vectors at the pixel level. In semantic segmentation tasks, accurate utilization of channel features is essential to label each pixel with an accurate category.

Inspired by previous ideas, this chapter proposes Enhanced Channel Feature Normalization (ECFN) for training, which combines the advantages of batch normalization and layer normalization to optimize the performance by adaptively exploiting the dependencies of channels and batches, and it performs the normalization along the axes of N, H, and W (batches, heights, and widths) and C, H, and W (channels, heights, and widths) for the inputs Normalization is performed and the normalized output is fused using the adaptive parameter τ. The τ parameter is dynamically adjusted according to the activation features of the network layer to optimize the normalization effect, thus better adapting to the changes during the training process. Enhanced channel feature normalization enhances the model's invariance to changes in image appearance, making the model more able to focus on the essential features of the image, which is very effective in improving the model's generalization ability. The specific algorithmic process of enhancement channel feature normalization is described in detail below.

During the training period of the model, the enhancement channel feature normalization first calculates the mean μ_1 and variance σ_1^2 of the input layer along the (N,H,W) axis. This step targets the feature maps of all samples in the entire batch, thus obtaining statistical information across batches and helping to mitigate the effects of sample variation in a single batch:

$$\mu_1 = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{11}$$

$$\sigma_1^2 = \frac{1}{n}\sum_{i=1}^{n} (x_i - \mu_1)^2 \tag{12}$$

Next, the algorithm computes the mean μ_2 and variance σ_2^2 along (C,H,W). This step focuses on characterizing the features of the individual channels, capturing the data properties within the channels and helping to refine the model's treatment of each channel feature:

$$\mu_2 = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{13}$$

$$\sigma_2^2 = \frac{1}{n}\sum_{i=1}^{n} (x_i - \mu_1)^2 \tag{14}$$

Next, $\overline{x_1}$ and $\overline{x_2}$ are normalized using μ_1, σ_1^2 and μ_2, σ_2^2, respectively, with ò being a constant. This step helps the model process the information more consistently by reducing the variation of the data in the specified dimensions:

$$x_1 = \frac{x_i - \mu_1}{\sqrt{(\sigma_1^2 + \grave{o})}} \tag{15}$$

$$x_2 = \frac{x_i - \mu_2}{\sqrt{(\sigma_2^2 + \grave{o})}} \tag{16}$$

Next, the learnable parameter τ is introduced to adaptively balance the normalized output along the (N,H,W) and (C,H,W) axes, a step that allows for flexibility in controlling the influence of the batch normalization and layer normalization strategies to optimize the overall normalization effect:

$$\overline{y} = \tau\overline{x_1} + (1-\tau)\overline{x_2} \tag{17}$$

Specifically, the update formula for τ is as follows:

$$\tau_{new} = \tau_{old} - \alpha\frac{\partial L}{\partial \tau} \tag{18}$$

In Eq. (18), τ_{new} denotes the updated value, τ_{old} denotes the current value, α denotes the learning rate, and $\frac{\partial L}{\partial \tau}$ denotes the gradient of τ to the loss function.

The final output of the enhanced channel feature normalization is accomplished by combining the above normalization results with the adaptive parameters as shown in Equation (19):

$$Y = \gamma\overline{y} + \beta \tag{19}$$

In Eq. (19), γ and β are learnable parameters.

IV. RGB-D City Scene Segmentation Simulation Experiments

In this chapter, RGB-D urban scene segmentation simulation experiments will be conducted to test the performance of the RGB-D urban scene segmentation algorithm based on dynamic adaptive convolution proposed in this paper through comparison experiments and ablation experiments.

IV. A. Experimental environment configuration

The experiments in this chapter are implemented based on the deep learning framework PyTorch and run on a single RTX 3060Ti GPU. The details of the environment platform are shown in Table 1.

Table 1: Detailed parameters of experimental environment

Parameters	Configuration
CPU	Intel(R) Xeon(R) Gold 5218 CPU
Memory	16G
Hard disk	5T
GPU	NVIDIA RTX3060Ti
display memory	12GB
Operating system	Ubuntu 18.05
CUDA version	10.1.88
cuDNN version	8.2.0
Python version	3.6
Deep learning framework	PyTorch 1.5.0

IV. B. Experimental parameterization

Since this paper is dedicated to the study of semantic segmentation of urban scenes, Cityscapes and CamVid, the most commonly used urban scene datasets in the field of image semantic segmentation, are chosen, while the CamVid dataset is used as the validation set. Deep images are encoded into HHA images using the FastDVFN network proposed in this paper as the backbone network of the encoder after pre-training the improved ResNet-101 and ResNet-50 networks, respectively. In order to increase the diversity of the images, data augmentation is applied to the input images by applying operations such as random horizontal flipping, scaling and cropping to train them in a supervised learning manner, and the loss function chosen during network training is the cross-entropy function, which is computed as shown in Eq. (20):

$$Loss = \frac{1}{N}\sum_{i=1}^{N}(y_i \log(\hat{y_i})) \tag{20}$$

where N denotes the total number of pixels in the input image, y_i denotes the label value corresponding to pixel i in the image, and $\hat{y_i}$ denotes the predicted value of pixel i by the network model. The loss curve during the training process is shown in Fig. 1. It can be seen that the FastDVFN network in this paper achieves convergence and fitting at about 250 training rounds.

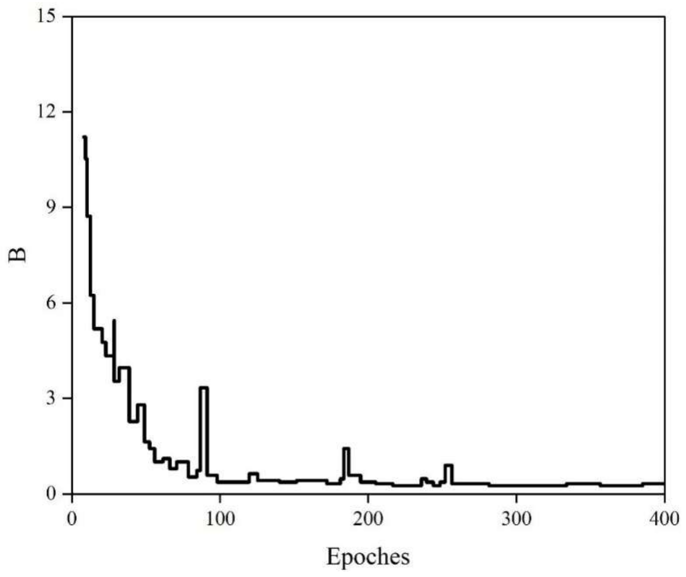

Figure 1: Loss function curve

IV. C. Comparative Experimental Analysis

In this section, the proposed FastDVFN network is compared with other urban scene semantic segmentation networks on the Cityscapes dataset, and the experiment compares not only advanced large-scale semantic segmentation networks, but also more advanced lightweight real-time semantic segmentation networks. The evaluation metrics used include mIoU, the running speed of the network and the number of parameters of the network.

IV. C. 1) Analysis of Cityscapes Test Set Comparison Results

The FastDVFN network of this paper is trained on the Cityscapes training and validation sets, and then the prediction results are obtained on the test set and uploaded to the Cityscapes online server to get the final test set experimental results as shown in Table 2. This paper FastDVFN network achieves 72.9% in mIoU evaluation metrics and runs at 88 frames/s without using additional data and pre-trained models for training. This paper FastDVFN network significantly outperforms all the compared lightweight semantic segmentation networks in mIoU evaluation metrics, and even outperforms several large-scale semantic segmentation networks. The Pyramid Scene Parsing Network (PSPNet) and SFNet achieve 77.7% and 79.4% in the mIoU evaluation metrics, but the FastDVFN network in this paper obtains better segmentation accuracy with only 1% and 5% of its parameter number, respectively. In terms of network efficiency, the FastDVFN network operates 76% and 29.41% faster than FPENet and LEDNet, respectively. Although both the two-way segmentation network (BiseNet) and DABNet are faster than the FastDVFN network, their mIoU evaluation metrics are decreased by 4.94% and 5.08%, respectively, compared to the FastDVFN network in this paper. In terms of network size, FastDVFN network in this paper has only 0.63m parameters, which is at a lower level among all compared semantic segmentation networks, and it can be seen that FastDVFN network in this paper has fewer redundant parameters, and the network structure is streamlined and efficient.

Table 2: Comparative experimental results on Cityscapes dataset

Network	Network name	MioU (%)	Operating speed (frame·s^{-1})	Number of parameters (m)
Large-scale network	SegNet	57.6	18	29.6
	Deeplab	63.2	<1	-
	FCN-8S	66	4	134.44
	Dilation10	66.9	<1	-
	Deeplabv2	70.3	<1	44.1
	PSPNet	77.7	<1	65.67
	SFNet	79.4	29	13.45
Lightweight network	ENet	57.6	78	0.33
	CGNet	64.6	50	0.43
	EDANet	67.8	84	0.77
	ERFNet	68.5	44	2.12
	BiSeNet	69.3	111	5.77
	FPENet	69.7	50	0.48
	DABNet	69.2	101	0.81
	ICNet	70.8	33	26.47
	LEDNet	71.4	68	0.86
	FDDWNet	70.5	55	0.88
	LRNNet	72.4	76	0.73
	FastDVFN	72.9	88	0.63

The scatterplot of the lightweight networks on the Cityscapes test set compared in terms of segmentation accuracy and running speed is specifically shown in Figure 2. It can be seen that the FastDVFN network of this paper is located in the upper right corner of the image, and its segmentation accuracy surpasses all other lightweight semantic segmentation networks, while the running speed also maintains a high level. Therefore, the FastDVFN network in this paper is able to better balance the accuracy and efficiency with a smaller number of parameters, and achieves a better balance between segmentation accuracy, network size and running speed.

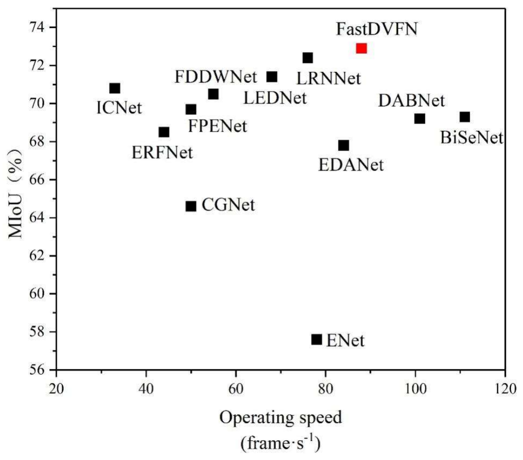

Figure 2: Comparison of precision and speed of lightweight network

IV. C. 2) Analysis of CamVid Validation Set Comparison Results

Experiments were conducted on CamVid training and validation sets to train this paper FastDVFN network and validation results were obtained on the test set as shown in Table 3. Without additional data and pre-trained models for training, the FastDVFN network achieves 68.1% on the mIoU evaluation metrics and runs at 108 frames/s. The FastDVFN network significantly outperforms all compared lightweight real-time semantic segmentation networks on the mIoU evaluation metrics. Despite the higher accuracy of PSPNet, the FastDVFN network is smaller and runs faster. Compared with ICNet and FDDWNet, FastDVFN network slightly improves in mIoU evaluation metrics, while the operation speed is improved by 4.7 times and 1.4 times, respectively.

Table 3: Comparative experimental results on the CamVid dataset

Network	Network name	MIoU (%)	Operating speed(frame·s^{-1})
Large-scale network	SegNet	47.4	7
	FCN-8S	57.8	-
	Deeplab	61.8	6
	PSPNet	70	8
Lightweight network	ENet	50.8	58
	FPENet	65.6	-
	BiSeNet	66.6	-
	CGNet	65	-
	EDANet	65.7	-
	FDDWNet	65.9	77
	ICNet	66.3	23
	FastDVFN	68.1	108

IV. D. Analysis of ablation experiments

The purpose of the ablation experiment is to verify the usefulness of this paper's dynamic adaptive convolution-based RGB-D urban scene segmentation method at each stage, and the results of the ablation experiment are shown in Table 4. From the experimental data in the table, it can be seen that compared with the baseline model, the MioU evaluation index decreases by 5.6% after adding the DAConv module alone, and the segmentation accuracy decreases, while the number of parameters decreases by 1.1, and the processing speed of the model improves. Comparing the effectiveness of the DFConv and DAConv modules, the DAConv module proposed in this paper has some improvement in the number of parameters and frame rate, while the DFConv module has a lower decrease in the MIoU evaluation index than the DAConv module. Adding ECFN module alone, the MIoU decreased by 6.2%, and when the combined approach of two modules, DAConv+ECFN, was used, the decrease of MIoU increased further, but the number of parameters decreased by 5.6m compared to the baseline model, and the running speed increased by 39 frames/second, which is a significant improvement of the model's performance, and is crucial for modeling the performance of real-time or need for fast response applications. Although the incompatibility between the modules resulted in a decrease in the MioU evaluation metrics, the significant optimization of the number of parameters and the running speed effectively improved the efficiency of the model. When choosing to use these modules, it is necessary to make a balanced consideration between the efficiency and accuracy of the model.

Table 4: Ablation experiment

-	Module	MioU (%)	Number of parameters r(m)	Operating speed (frame·s^{-1})
DVFN (baseline model)	-	59.6	18.2	43.6
FastDVFN	DFConv	57.6	17.1	45.8
	DAConv	54	16.2	46.5
	ECFN	53.4	13.3	70.4
	DAConv+ECFN	50	12.6	82.6

V. Conclusion

With the dynamic adaptive convolution-based RGB-D urban scene segmentation algorithm (FastDVFN) proposed in this paper, an important breakthrough in segmentation accuracy and computational efficiency is realized. On the Cityscapes dataset, the FastDVFN network achieves an mIoU index of 72.9%, which significantly outperforms other similar networks, and has a computational speed of 88 frames/s. Compared with existing large-scale semantic segmentation networks, FastDVFN only lags behind PSPNet and SFNet in terms of accuracy, but its number of parameters is only 1% and 5% of theirs, which demonstrates an excellent lightweight design. Especially in the experiments on the CamVid dataset, the mIoU of FastDVFN is 68.1%, and the running speed is 108 frames/s, which is far beyond the similar networks. The experimental results show that by introducing Dynamic Adaptive Convolution (DAConv) with Enhanced Channel Feature Normalization (ECFN) module, the FastDVFN network not only obtains an improvement in accuracy, but also exhibits a better balance between the number of parameters and the running speed. In addition, the ablation experiments further validate the enhancement effect of each module on the performance of the algorithm. Therefore, the application of FastDVFN in urban scene segmentation has wide prospects.

Acknowledgements

The 14th-Five Year Plan Projects of Guangdong Association of Higher Education (22GYB159).

References

[1] Jung, S., Lee, J., Gwak, D., Choi, S., & Choo, J. (2021). Standardized max logits: A simple yet effective approach for identifying unexpected road obstacles in urban-scene segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 15425-15434).

[2] Choi, S., Kim, J. T., & Choo, J. (2020). Cars can't fly up in the sky: Improving urban-scene segmentation via height-driven attention networks. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 9373-9383).

[3] Yi, S., Liu, X., Li, J., & Chen, L. (2023). UAVformer: a composite transformer network for urban scene segmentation of UAV images. Pattern Recognition, 133, 109019.

[4] Zhong, Z., Zhao, Y., Lee, G. H., & Sebe, N. (2022). Adversarial style augmentation for domain generalized urban-scene segmentation. Advances in neural information processing systems, 35, 338-350.

[5] Wang, L., Li, R., Zhang, C., Fang, S., Duan, C., Meng, X., & Atkinson, P. M. (2022). UNetFormer: A UNet-like transformer for efficient semantic segmentation of remote sensing urban scene imagery. ISPRS Journal of Photogrammetry and Remote Sensing, 190, 196-214.

[6] Zhang, X., Chen, Z., Wu, Q. J., Cai, L., Lu, D., & Li, X. (2018). Fast semantic segmentation for scene perception. IEEE Transactions on Industrial Informatics, 15(2), 1183-1192

[7] Sheng, H., Cong, R., Yang, D., Chen, R., Wang, S., & Cui, Z. (2022). UrbanLF: A comprehensive light field dataset for semantic segmentation of urban scenes. IEEE Transactions on Circuits and Systems for Video Technology, 32(11), 7880-7893.

[8] Gao, L., Zhang, Y., Zou, F., Shao, J., & Lai, J. (2020). Unsupervised urban scene segmentation via domain adaptation. Neurocomputing, 406, 295-301.

[9] Luo, Z., Yang, W., Yuan, Y., Gou, R., & Li, X. (2024). Semantic segmentation of agricultural images: A survey. Information Processing in Agriculture, 11(2), 172-186.

[10] Liu, X., Deng, Z., & Yang, Y. (2019). Recent progress in semantic image segmentation. Artificial Intelligence Review, 52, 1089-1106.

[11] Jiang, B., An, X., Xu, S., & Chen, Z. (2023). Intelligent image semantic segmentation: a review through deep learning techniques for remote sensing image analysis. Journal of the Indian society of remote sensing, 51(9), 1865-1878.

[12] Yang, Z., Yu, H., Feng, M., Sun, W., Lin, X., Sun, M., ... & Mian, A. (2020). Small object augmentation of urban scenes for real-time semantic segmentation. IEEE Transactions on Image Processing, 29, 5175-5190.

[13] Yue, Z., Lo, C. Y., Wu, R., Ma, L., & Sham, C. W. (2024). Urban aquatic scene expansion for semantic segmentation in cityscapes. Urban Science, 8(2), 23.

[14] Tang, Q., Liu, F., Zhang, T., Jiang, J., Zhang, Y., Zhu, B., & Tang, X. (2022). Compensating for local ambiguity with encoder-decoder in urban scene segmentation. IEEE Transactions on Intelligent Transportation Systems, 23(10), 19224-19235.

[15] Xue, Z., Mao, W., & Zheng, L. (2021). Learning to simulate complex scenes for street scene segmentation. IEEE Transactions on Multimedia, 24, 1253-1265.

[16] Han, X., Liu, C., Zhou, Y., Tan, K., Dong, Z., & Yang, B. (2024). WHU-Urban3D: An urban scene LiDAR point cloud dataset for semantic instance segmentation. ISPRS Journal of Photogrammetry and Remote Sensing, 209, 500-513.

[17] Shelhamer, E., Long, J., & Darrell, T. (2016). Fully convolutional networks for semantic segmentation. IEEE transactions on pattern analysis and machine intelligence, 39(4), 640-651.

[18] Wurm, M., Stark, T., Zhu, X. X., Weigand, M., & Taubenböck, H. (2019). Semantic segmentation of slums in satellite images using transfer learning on fully convolutional neural networks. ISPRS journal of photogrammetry and remote sensing, 150, 59-69.

[19] Ji, J., Lu, X., Luo, M., Yin, M., Miao, Q., & Liu, X. (2020). Parallel fully convolutional network for semantic segmentation. Ieee Access, 9, 673-682.

[20] Pastorino, M., Moser, G., Serpico, S. B., & Zerubia, J. (2022). Semantic segmentation of remote-sensing images through fully convolutional neural networks and hierarchical probabilistic graphical models. IEEE Transactions on Geoscience and Remote Sensing, 60, 1-16.

[21] Yue Zhong,Jun Li,Hong Hao,Ling Li & Ruhua Wang. (2025). Structural damage identification using phase space matrix with convolutional neural networks. Structures,76,108885-108885.

[22] He Zhang,Mindong Wu,Tengxin Lin,Jiangpeng Shu & Zhicheng Zhang. (2025). A NURBS-based approach to the generation of geometric models for complex-shaped bridge using point clouds. Engineering Structures,335,120324-120324.

[23] Jiacun Qiu,Zhenqi Han,Lizhaung Liu & Jialu Zhang. (2025). FFMT: Unsupervised RGB-D Point Cloud Registration via Fusion Feature Matching with Transformer. Applied Sciences,15(5),2472-2472.