J. Meas. Sci. Instrum., 2024, 15(3): 328-337. http://jmsi.publish.founderss.cn
DOI: 10.62756/jmsi.1674-8042.2024034 jmsi@nuc.edu.cn

Real-time instance segmentation based on contour learning

GE Rui[1,2], LIU Dengfeng[1,2*], ZHOU Haojie[1,2], CHAI Zhilei[1,2], WU Qin[1,2]

1. School of Artificial Intelligence and Computer Science, Jiangnan University, Wuxi 214122, China;
2. Jiangsu Provincial Engineering Laboratory Pattern Recognition and Computational Intelligence, Jiangnan University, Wuxi 214122, China

***Corresponding author:** LIU Dengfeng (liudf@jiangnan.edu.cn)

Received: November 3, 2023 **Revised:** January 14, 2024 **Accepted:** January 21, 2024

Abstract: Instance segmentation plays an important role in image processing. The Deep Snake algorithm based on contour iteration deforms an initial bounding box to an instance contour end-to-end, which can improve the performance of instance segmentation, but has defects such as slow segmentation speed and sub-optimal initial contour. To solve these problems, a real-time instance segmentation algorithm based on contour learning was proposed. Firstly, ShuffleNet V2 was used as backbone network, and the receptive field of the model was expanded by using a 5×5 convolution kernel. Secondly, a lightweight up-sampling module, multi-stage aggregation (MSA), performs residual fusion of multi-layer features, which not only improves segmentation speed, but also extracts effective features more comprehensively. Thirdly, a contour initialization method for network learning was designed, and a global contour feature aggregation mechanism was used to return a coarse contour, which solves the problem of excessive error between manually initialized contour and real contour. Finally, the Snake deformation module was used to iteratively optimize the coarse contour to obtain the final instance contour. The experimental results showed that the proposed method improved the instance segmentation accuracy on semantic boundaries dataset (SBD), Cityscapes and Kins datasets, and the average precision reached 55.8 on the SBD; Compared with Deep Snake, the model parameters were reduced by 87.2%, calculation amount was reduced by 78.3%, and segmentation speed reached 39.8 frame·s^{-1}when instance segmentation was performed on an image with a size of 512×512 pixels on a 2080Ti GPU. The proposed method can reduce resource consumption, realize instance segmentation tasks quickly and accurately, and therefore is more suitable for embedded platforms with limited resources.

Key words: instance segmentation; ShuffleNet V2; lightweight network; contour initialization

0 Introduction

Instance segmentation is a combination task of object detection and semantic segmentation, including location and classification of objects, pixel classification at the semantic level, and distinction between different instances from the same category. Instance segmentation can be widely used in the fields of automatic driving, robot grasping, security monitoring, and remote sensing images. In these fields, accurate and fast object location and contour segmentation of images are the prerequisites for subsequent visual processing.

Instance segmentation was first proposed by Hariharan et al. [1], who designed a simultaneous detection and segmentation (SDS) model to simultaneously complete object detection and segmentation tasks. Based on this, researchers have proposed many algorithms for instance segmentation. In the early stage of research, instance segmentation algorithms mainly focused on two-stage segmentation. Among them, top-down instance segmentation algorithm locates the object, generates a pre-selection box, performs pixel-level segmentation in the pre-selection box, and separates foreground from background. Bottom-up instance segmentation algorithm is based on the idea of semantic segmentation, which clusters the pixels that belong to the same category, and performs category judgment for every category to obtain the instance category. In recent years, with the continuous development of deep learning technology, the accuracy and performance of instance segmentation have also been improved[2-4]. He et al. [5] proposed mask region-based convolutional neural network (Mask R-CNN) that adds a mask branch network on the basis of Faster R-CNN[6] to achieve instance segmentation, which is the baseline algorithm for many derivative applications. Liu et al. [7] proposed path aggregation network (PANet) based on Mask R-CNN,

which introduces a bottom-up path, expands the pyramid feature extraction network, uses adaptive region of interest (ROI) region feature pooling to integrate feature information at different levels and further improves the performance of instance segmentation. Huang et al. [8] proposed a combination of mask scoring R-CNN (MS R-CNN) and mask intersection over union (Mask IoU) branch to predict the mask and score it to improve the instance segmentation performance of the model, and then proposed a new evaluation index for instance segmentation.

To increase the speed of instance segmentation, the single-stage instance segmentation algorithm is proposed. On the basis of fully convolutional networks (FCN), instance FCN[9] introduces instance information into semantic segmentation in the form of a position-sensitive map to achieve translation variability, and translates the original FCN's single output channel into multiple channels that are sensitive to instance positions. The mask of each instance is achieved by aggregating position-sensitive map. SOLOv2[10] dynamically segments every instance in the image, and the mask of every instance is divided into mask kernel prediction and mask feature learning, and convolution generates instance masks to obtain faster segmentation speed. With the introduction of a real-time object detection algorithm, inspired by you only look once (YOLO): unified, real-time object detection, you only look at coefficients (YOLACT) [11] splits the instance segmentation into two parallel subtasks: one part obtains a semantic segmentation prototype map similar to FCN, and the other part predicts mask coefficient of each instance, and finally the two parts are fused to obtain the instance segmentation mask. The speed of YOLACT reaches more than 30 frame per second, and real-time instance segmentation can be realized while generating a high-precision mask. Single shot instance segmentation with polar representation (PolarMask) [12] is a fully convolutional, anchor-free instance segmentation method. Based on fully convolutional one-stage (FCOS) object detection network[13], it expands from 4 rays to 36 rays, and constructs instance contours in polar coordinates. It does not require a detection frame, and converts the instance segmentation problem into an instance center point classification problem and a dense distance regression problem. Although the advantages in accuracy and speed are not great, it has great implications for the research on instance segmentation methods based on anchor-free frames. Deep Snake is an active contour model based on deep learning[14]. The initial instance contour is manually designed, and the circular convolution is designed according to the circular topology of the contour, which is used for the structured feature learning of the instance contour. Based on this, a fast and accurate contour instance segmentation method is proposed.

Although Deep Snake can improve the performance of instance segmentation, the manual octagonal initial contour cannot surround the instance well, nor can it make full use of the information of initial object detection and location. In the contour iteration deformation, the feature vertex change path is complex and changeable, which leads to the degradation of instance segmentation performance. Furthermore, the manual design of the initial contour takes a lot of time, which reduces the speed of model inference. Although the existing Deep Snake can achieve real-time segmentation speed on high-performance GPUs, in practical application scenarios such as autonomous driving and security monitoring, it is often only possible to use devices such as embedded GPUs with limited resources. At this time, Deep Snake algorithm can only reach 5 frame·s^{-1}, which is far from the real-time requirements.

In this study, we proposed a real-time instance segmentation algorithm based on network learning of initial contour. Firstly, we used ShuffleNet V2 instead of deep layer aggregation-34 (DLA-34) as the backbone network to reduce the amount of parameters and calculations in the network and replaced 3×3 depth separable convolution with 5×5 depth separable convolution to expand the receptive field of the model. Secondly, we used multi-stage feature deep aggregation as a lightweight up-sampling module to effectively extract the features of input images. Thirdly, we used direct regression to obtain the initial contour vertex of the instance instead of the manual initial contour method and global contour feature deformation to obtain the coarse contour of the instance, so that the contour input into the snake deformation module was closer to the ground truth of the contour, and the accuracy of segmentation was improved. Finally, the Snake deformation module was used to iteratively optimize coarse contour to obtain the final instance contour.

1 Related work

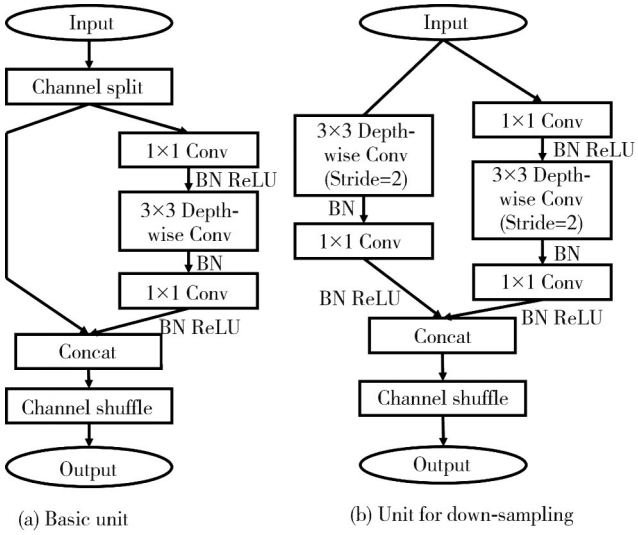
1.1 ShuffleNet V2

In the instance segmentation task, in addition to the accuracy, the computational complexity of a network is also an important factor to be considered, especially in case of limited hardware platform resources and special needs, such as extremely low latency in the field of autonomous drive. In recent years, some excellent

lightweight network structures have emerged, such as MobileNet[15], EfficientNet[16], ShuffleNet[17], etc. These networks maintain a good balance between running speed and accuracy.

ShuffleNet V2[18] was proposed by MEGVII Technology Team in 2018. Compared with ShuffleNet V1, ShuffleNet V2 has new rules considering the factors that affect the speed of the model, and overcomes the shortcomings of the original version. Some original lightweight methods usually use floating point operations (FLOPs) as a main indicator to measure the complexity of the model. However, a series of experiments showed that if ShuffleNet V2 only relies on FLOPs, it is problematic because networks with approximate FLOPs will have different speeds. Therefore, two other important indicators were proposed: memory access cost and network parallelism cost.

ShuffleNet V2 presents a new network basic unit, as shown in Fig. 1. For the basic unit with a step size of 1, an operation method named Channel split is introduced. The channel of the input image is divided into two parts, one part is directly passed down, the other part is convoluted, and the results of two parts are concatenated. ReLU operation is not connected later and Channel shuffle operation is performed before outputting the feature map to solve the problem of poor information exchange of different groups of convolutions. For the down-sampling unit with a step size of 2, the channel is doubled, and the input image is directly transmitted to two parts. Finally, the results are concatenated, and Channel shuffle operation is also performed.



(a) Basic unit (b) Unit for down-sampling

Fig. 1 ShuffleNet V2 basic blocks

1.2 Deep Snake

Deep Snake is an active contour model based on deep learning. The initial instance contour is manually designed, and circular convolution is designed according to the circular topology of the contour, which is used for the structured feature learning of instance contour. Multiple iterations obtain a contour that is more suitable for the contour of the instance. Based on this, a fast and accurate contour instance segmentation method is proposed.

After obtaining the initial contour of the instance through the object detection method, the Snake deformation module is used to deform the contour to obtain the final contour. The feature points of the instance contour are different from ordinary image pixel features, but similar to a discrete signal. Compared with the traditional 2D convolution, circular convolution is more suitable for the calculation of such discrete features. Therefore, Deep Snake uses 1D circular convolution calculation to calculate the contour vertices shown in Fig. 2, where blue points represent the features on the contour points, yellow points represent 1D convolution kernel, green points represent the volume, dark green dots represent the features obtained by circular convolution of the yellow convolution kernel and the corresponding contour points.

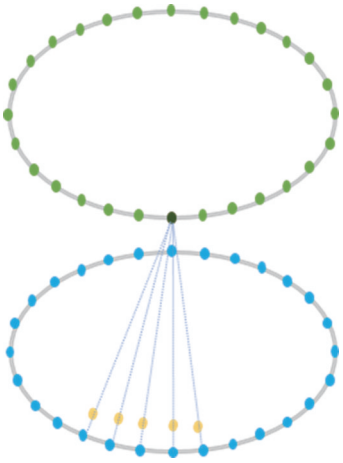


Fig. 2 1D circular convolution[14]

Deep Snake treats the vertices on the contour as a set of discrete signals $(f_N)_i = f_{i \bmod N}$, and computes the features of contour vertex using 1D circular convolution as

$$(f_N * k)_i = \sum_{j=-r}^{r} (f_N)_{i+j} k_j, \tag{1}$$

where k is a 1D convolution kernel, and * is a 1D convolution operation.

The Snake deformation module of Deep Snake algorithm is got based on 1D circular convolution, as shown in Fig. 3, which consists of three parts: backbone network, fusion module and prediction head[14].

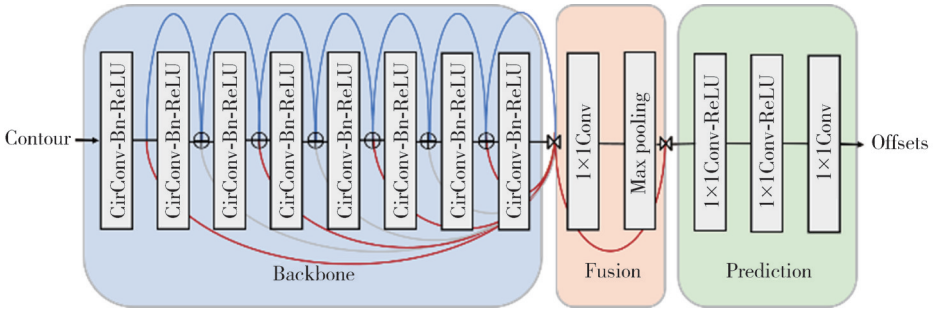


Fig. 3 Snake contour deformation module

In Fig.3, the initial contour is input into the backbone network. There are eight circular convolution layers in the backbone network for multi-scale feature information extraction, and the features of each vertex are output to the fusion module. The fusion module connects the contour vertex features of each scale at each layer, uses a 1×1 convolution kernel to forward, and compresses feature information through maximum pooling to fuse multi-scale contour features. In the prediction head module, three convolutional layers are used to process the fused features to generate offsets for contour deformation of each vertex.

2 Proposed method

This paper presents a lightweight real-time instance segmentation method based on contour learning. The whole network is an easy-training end-to-end structure, as shown in Fig. 4, which consists of three parts: backbone, initial contour module and snake contour deformation module.

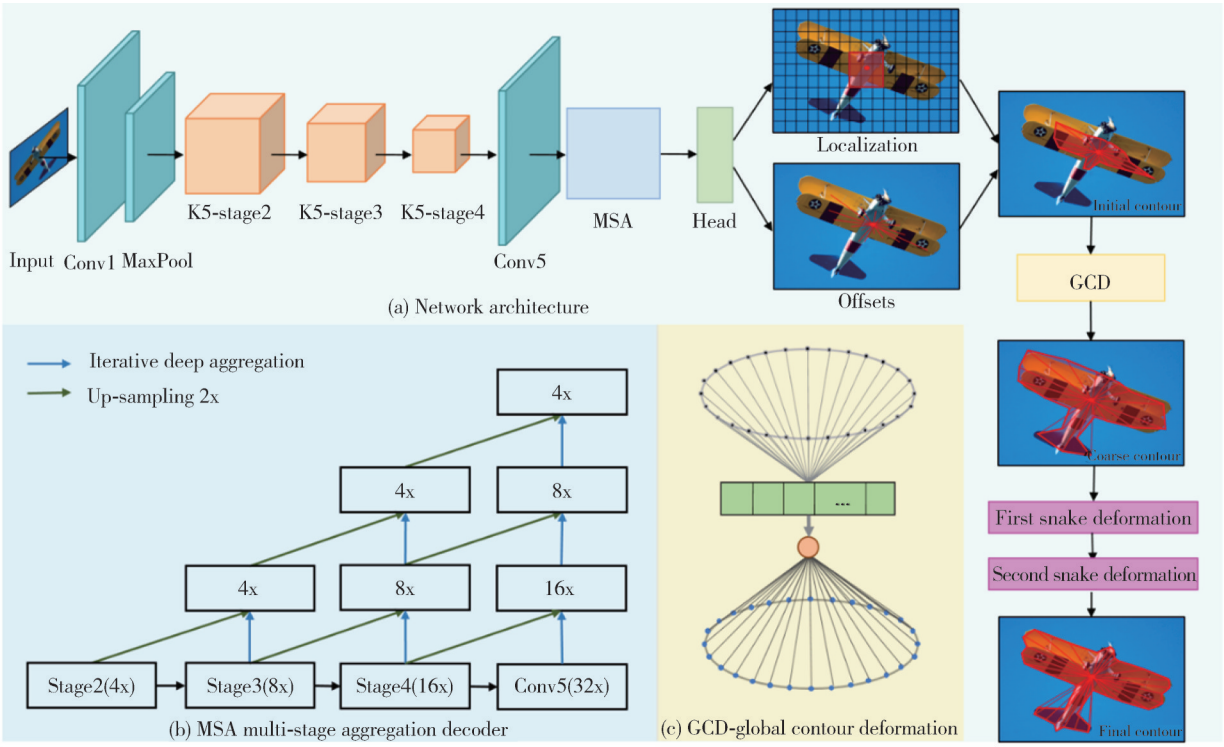


Fig. 4 Lightweight instance segmentation method based on contour learning

Firstly, the improved ShuffleNet V2 named ShuffleNetV2-k5, as backbone network, starts from conv1 and ends with multi-stage aggregation (MSA) to obtain feature maps with different down-sampling ratios at each stage. Secondly, the initial contour module contains a prediction head which regresses the center point of instance and the offsets relative to center point to obtain the initial contour, and then global contour deformation globally fuses the initial contour features to obtain a coarse contour which is closer to the real contour. Finally, the coarse contour is deformed twice by snake contour deformation module which is named the first snake deformation and second snake deformation. The details of snake contour deformation module have been shown in Fig. 3 in Section 1.2.

2.1 Improved ShuffleNet V2

2.1.1 ShuffleNetV2-k5

The lightweight network ShuffleNet V2 instead of

DLA as the backbone network will affect the depth and receptive field of the model. Heat maps are used to return the center point, height and width of the object, and therefore the receptive field is of great significance. The size of the receptive field is affected by the convolution kernel. The larger the size of convolution kernel, the larger the receptive field. However, as the size of the convolution kernel increases, the parameter amount of the network will increase greatly. Under the same receptive field, the parameter amount and calculation amount of two 3×3 convolution kernels are only 18/25 of one 5×5 convolution kernel.

However, depth convolution is often used instead of traditional convolution in lightweight networks, and stacking multiple small-scale convolution kernels into a large-scale convolution is not applicable in the era of depth-wise convolution. The parameters involved in depthwise separable convolution calculation are mainly composed of two parts: depthwise convolution and pointwise convolution. Although expanding the convolution kernel size increases the amount of parameters and calculations in the depthwise convolution part, it obtains a larger receptive field. Therefore, we replace the 3×3 depthwise separable convolution in ShuffleNet V2 with a 5×5 depthwise separable convolution to obtain a receptive field twice that of the original algorithm and named it ShffleNetV2-k5.

The existing ShuffleNet V2 pre-training model based on ImageNet uses 3×3 depth convolution, and lacks a pre-training model based on 5×5 depth convolution. For faster convergence of the model, we perform a convolution kernel expansion operation on the 3×3 convolution kernel ShuffleNet V2 pre-training model to obtain a 5×5 convolution kernel ShuffleNet V2 pre-trained model. As shown in Fig. 5, the green area represents the existing 3×3 depth convolution kernel parameters, and the blue part is filled with 0 to obtain a 5×5 depth convolution kernel.

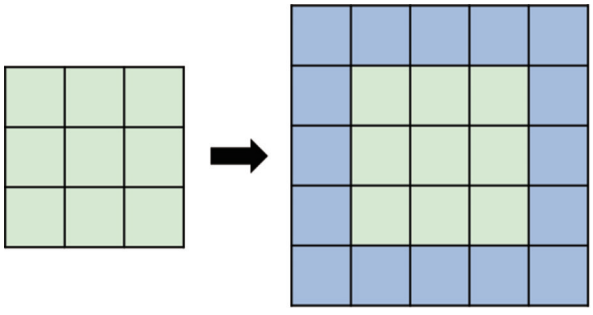


Fig. 5 Convolution kernel edge expansion operation

2.1.2 MSA

ShuffleNet V2, as the backbone network for feature extraction, down samples the input image by 2 times (2x), 4 times (4x), 8 times (8x), and 16 times (16x) respectively to obtain the corresponding feature maps. To integrate the features of different down-sampling layers, we perform iterative deep feature aggregation on the feature maps at different stages, and finally obtain the output that is 4 times the input image, which is used to detect the coordinates of the center point, width and height of the object. As shown in Fig.4 (b), each round of deep feature aggregation calculates the lower right triangle from small to large, and respectively fuses to obtain 16 times, 8 times, and 4 times down-sampling feature maps, of which 4 times down-sampling feature maps are used as output. Inspired by CenterNet algorithm[19], MSA uses iterative deep aggregation to symmetrically increase feature map resolution. Deformable convolution[20] can be closer to the shape and size of the object when sampling, therefore, we use deformable convolutions instead of traditional convolutions in MSA for up-sampling to increase the perception of small objects during sampling from lower layers to the output layer.

2.2 Initial contour method

The initial contour method based on network learning changes the decoding method for object detection. Complex deformation calculation on the bounding box is not necessary and the initial contour can be directly obtained from the center point of the network learning instance and the offsets of the contour vertex relative to the center point. The features of the initial contour are fused to obtain the coarse contour closer to ground truth for the snake deformation.

2.2.1 Initial contour regression

Most of the existing contour initialization methods are manually designed initialization, and the initial contour method based on network learning does not need to specify the specific shape of the initial contour. In Fig.6, Deep Snake adopts a method of manually designing the initial octagonal contour. It first calculates four poles on the four sides of the object detection frame, and then calculates four poles along these two sides of the line on these four poles. Each direction is extended by 1/8 of the length of the line, so that 8 contour vertices are connected to obtain the initial octagonal contour. It is believed that the octagonal shape can better represent the contour characteristics of the instance. In fact, this contour initialization method is very dependent on the accuracy of the object detection frame, and the length of the line has been determined as a hyperparameter. Since

the length of the border applicable to different instances is different, it may affect the accuracy of the initial contour. Additionally, this initial contour method has complex calculations, which may affect the speed of model reasoning. CurveGCN uses a manually designed ellipse as the initial contour, and then deforms it using graph convolution to obtain the final contour.

Compared with the manual initial contour in Fig.6(a) and (b), the initial contour method of network learning does not return the width and height of object detection. Since it is based on the center point feature, the offset of each initial contour vertex relative to the center point is directly obtained for an ordered point set regression, which is denoted as

$$\left\{\left(\Delta x^i_{\text{init}}, \Delta y^i_{\text{init}}\right) \mid \ i=1,2,\cdots,N\right\}, \tag{2}$$

where N is the number of vertices of the initial contour, N=128 here. The initial contour vertices are obtained by adding the center point and the offset, and denoted as

$$\left\{\left(x^i_{\text{init}}, y^i_{\text{init}}\right) \mid \ i=1,2,\cdots,N\right\}. \tag{3}$$

(a) DeepSnake (b) CurveGCN (c) Ours

Fig. 6 Comparison of different contour initial methods

In this way, the initial contour learned by the network is closer to the real contour, and direct regression to the initial contour vertices avoids complex deformation calculation for the bounding box. Since the direction of deformation path of the initial contour learned by the network is from the center point to the initial contour vertex, the vertex deformation path diverges from inside to outside. Therefore, it is more conducive to the convergence of the contour than the deformation path from outside to inside

2.2.2 Global contour deformation

There is still a large gap between the initial contour and the final contour obtained by only using the feature regression of the center point of instance. It is impossible to directly deform the initial contour to obtain the final instance contour, and it is also difficult to obtain the instance contour only based on a single contour vertex feature or the local features of several adjacent vertices. Deep Snake uses the snake deformation module to aggregate local features, fuse the features of several adjacent vertices, obtain relatively complete contour features, and complement the global contour information to a certain extent. Although the use of the snake deformation module can better aggregate contour features, it increases the complexity of the model and reduces the speed of model inference, but only improves the accuracy to a limited extent.

Therefore, adopting a global contour feature aggregation mechanism can simply and efficiently improve the ability of the network to learn global contour features and deform the initial contour based on all contour vertex features. The specific implementation process is shown in Fig. 4(c). The N initial contour vertex features are first connected into a vector with a length of $N\times C$ (where C is the number of channels of the vertex features), which is the gray rectangle in Fig.4(c). Then, the vector is input to the global contour deformation module to predict the contour vertex offset, where the number of hidden and output layer channels of the global contour deformation module is $N\times 2$, as shown in the black dot in Fig.4(c). The offsets of the silhouette vertices are vectors with a length of $N\times 2$, and expressed as

$$\left\{\left(\Delta x^i_{\text{coarse}}, \Delta y^i_{\text{coarse}}\right) \| \ i=1,2,\cdots,N\right\}, \tag{4}$$

where N is the number of contour vertices. Finally, the offsets are added to the initial contour coordinates to obtain the adjusted coarse instance contour, which is expressed as

$$\left\{\left(x^i_{\text{coarse}}, y^i_{\text{coarse}}\right) \| \ i=1,2,\cdots,N\right\}. \tag{5}$$

3 Experiment

3.1 Datasets and evaluation metrics

The PASCAL SBD[21], Cityscapes[22] and Kins[23] datasets were used in the experiments. PASCAL SBD contains 20 instance classes that consist of 11, 355 re-annotated images from the PASCAL VOC dataset with instance-level boundaries and being divided into 5 623 training images and 5 732 testing images. The Cityscapes dataset has eight instance classes and contains a diverse set of stereo video sequences recorded from street scenes in 50 different cities, with high quality pixel-level annotations of 5 000 frames: 2 975 training images, 500 validation images, and 1 525 testing images.

In this study, average precision (indicated by AP) is used as a measure of the instance segmentation accuracy of the proposed algorithm, and it calculates the intersection over union between the calculation result and the ground-truth result of image segmentation, which ranges from 0.5 to 0.95. Taking 0.05 as a step, the average value of IoU under these 10 different IoU

thresholds $(0.5, 0.55, \cdots, 0.95)$ can be got, i.e., AP_{50} refers to the calculation result when IoU threshold is 0.5. The lightness of the model is measured by the amount of model parameters, calculation amount, and segmentation speed. The segmentation speed is measured by frames·s^{-1}.

3.2 Experimental environment

In this study, hardware experimental environment is Intel i9-9900X processor model, 128 G memory, and RTX 2080Ti GPU; and software environmental environment is Ubuntu 18.04, Python 3.7, torch 1.1.0, cuda10.0, and cudnn7.5.

3.3 Loss function

In this study, loss function mainly includes two parts that are responsible for the prediction of initial object center point and the deformation of instance contour, respectively. Among them, confidence loss function for the prediction of object center point refers to CenterNet[19], which uses FocalLoss as loss function, denoted as L_{ct}; *smoothl*1 is used to supervise the deformation of instance contour, including initial contour loss, coarse contour loss, and iterative deformation contour loss, which are defined as

$$L_{init} = \frac{1}{N}\sum_{i=1}^{N} smoothl1(\hat{x}_i^{init} - x_i^{gt}), \tag{6}$$

$$L_{coarse} = \frac{1}{N}\sum_{i=1}^{N} smoothl1(\hat{x}_i^{coarse} - x_i^{gt}), \tag{7}$$

$$L_{iter} = \frac{1}{N}\sum_{i=1}^{N} smoothl1(\hat{x}_i^{iter} - x_i^{gt}), \tag{8}$$

where N is the number of contour vertices, x_i^{gt} is the ground truth of label contour, $\hat{x}_i^{init}$ is the predicted initial contour vertices, $\hat{x}_i^{coarse}$ is the predicted coarse contour vertices, and $\hat{x}_i^{iter}$ is the predicted iterative deformation contour vertices, including two iterative optimizations. Therefore, the overall loss function is

$$L_{overall} = L_{ct} + \alpha L_{init} + \beta L_{coarse} + L_{iter1} + L_{iter2}. \tag{9}$$

In Deep Snake, α and β are set to be 0.1. In this study, α and β were set to be 0.05, 0.1 and 0.2 respectively when the model was trained on SBD, and the results are shown in Fig. 7. It can be seen that the average precision on condition that α and β are 0.1 is higher than that under the other two settings in the second half of training. At the end of training, the highest average precision is achieved when α and β are 0.1. As a result, in the following experiments in this study, α and β were all set to be 0.1.

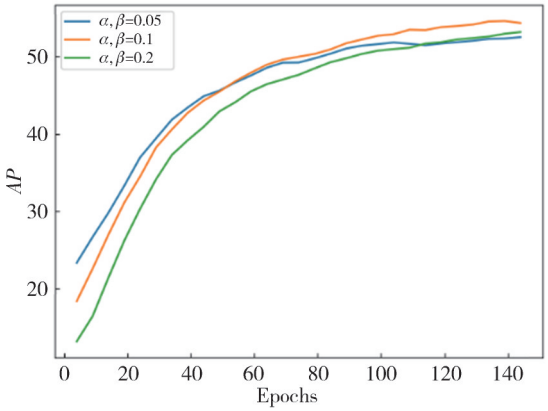


Fig. 7 Average precision under different α and β

3.4 Ablation experiment

The ablation experiment was conducted on the SBD. The model was trained for a total of 150 epochs, and the learning rate was reduced by half in the 80th and 120th cycles. The results of ablation experiment are shown in Table 1.

Table 1 Results of ablation experiment on SBD

Method	Backbone	AP_{vol}	FLOPs	*Params*	Segmentation speed/(frame·s^{-1})
Deep Snake	DLA-34	54.4	4.98×10^{10}	2.31×10^{7}	32.3
+Initial contour	DLA-34	**56.2**	5.23×10^{10}	2.49×10^{7}	31.2
+ShuffleNet V2	ShuffleNet V2	52.3	**1.02×10^{10}**	**2.65×10^{6}**	36.3
+ShuffleNetV2-k5	ShuffleNetV2-k5	55.8	1.08×10^{10}	2.95×10^{6}	**39.8**

The baseline is Deep Snake and the second model replaces manually designed initial contour with initial contour regression which is introduced in Section 2.2. The average precision of the second model is improved by 1.8 while the calculation amount and parameters of the second model also increase. The third model replaces the backbone DLA-34 with original ShuffleNet V2 on the basis of the second model, and the average precision of instance segmentation drops a lot while the calculation amount and parameters significantly decrease. The fourth model matches the proposed method, whose convolution kernel in ShuffleNet V2 is replaced with 5×5 convolution kernel. Compared with Deep Snake, the average precision of instance segmentation is improved by 1.4, the number of parameters is reduced by ~87.2%, the calculation amount is reduced by ~78.3%, and the segmentation speed is improved by 7.5 frame·s^{-1}, achieving a good balance between

accuracy and speed.

3.5 Comparison experiments

3.5.1 SBD

On the PASCAL SBD, we performed instance segmentation task on an image of 512×512 pixels using fully convolutional instance-aware semantic segmentation (FCIS)[24], explicit shape encoding for real-time instance segmentation (ESE-20)[25], Deep Snake, and the proposed algorithm, respectively, and the results are shown in Table 2. Compared with Deep Snake, the proposed algorithm largely improves the segmentation speed of instance by 7.5 frame·s^{-1} while improving the segmentation accuracy. Therefore, compared with other algorithms, the proposed algorithm has high competitiveness in terms of the accuracy and the speed of instance segmentation.

Table 2 Performance comparison on SBD

Method	Segmentation speed/(frame·s^{-1})	AP_{vol}	AP_{50}	AP_{70}
FCIS[24]	6.3	—	**63.7**	52.1
ESE-20[25]	38.5	35.3	38.2	12.1
Deep Snake	32.3	54.4	62.1	48.3
Proposed algorithm	**39.8**	**55.8**	62.8	**52.9**

It can be seen from Fig.8 that FCIS can roughly give the contour of instance while has some false detection such as the fourth image in the first row. ESE-20 can identify different instances and their categories while the contour of instance segmentation is not close enough to the real contour, especially in the fifth image, ESE-20 is not able to give the instance contour probably because the train occupies most of the area of the image. Deep Snake has false detection, but the proposed algorithm can greatly reduce false detection and missed detection, for example, it accurately identify the three girls in the third column and clearly segments the contour.

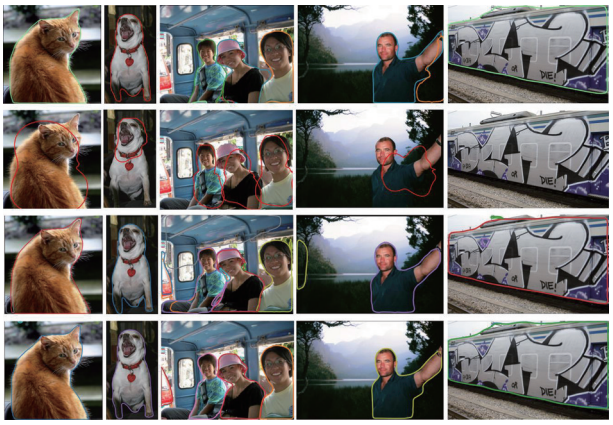


Fig. 8 Visualization results on SBD (from top to bottom are FCIS, ESE-20, Deep Snake and Ours)

3.5.2 Cityscapes dataset

The test set contains 1 525 images provided by the Cityscapes dataset. We performed the instance segmentation based on the proposed algorithm on an image of 1 024×2 048 pixels, and obtained a segmentation speed of 6.5 frame·s^{-1}, as shown in Table 3. Compared with Deep Snake, the average precision of the proposed algorithm increases by 0.4, and the accuracy is significantly improved when segmenting three types of instances: person, truck, and bus.

Table 3 Performance comparison on Cityscapes test dataset

Method	Segmentation speed/(frame·s^{-1})	AP	AP_{50}	Person	Rider	Car	Truck	Bus	Train	Mcycle	Bicycle
Mask R-CNN[4]	2.2	26.2	49.9	30.5	23.7	46.9	22.8	32.2	18.6	19.1	16.0
PANet[6]	<1	31.8	57.1	36.8	**30.4**	**54.8**	27.0	36.3	25.5	**22.6**	**20.8**
Deep Snake	4.6	31.7	58.4	37.2	27.0	56.0	29.5	40.5	**28.2**	19.0	16.4
Proposed algorithm	**6.5**	**32.1**	**58.9**	**38.7**	27.8	54.1	**32.1**	**45.0**	17.2	18.9	16.9

As shown in Fig.9, on the Cityscapes test dataset, the proposed algorithm achieves good effect of segmentation. In complex street scenes, such as in the first four images, the algorithm can accurately identify whether cars are near or far away and segment them. For the dense crowd in the fifth image, the segmentation of the instance contour can be also achieved with almost no error.

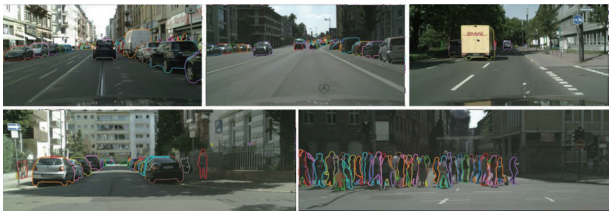


Fig. 9 Visualization results on Cityscapes test dataset

3.5.3 Kins dataset

The proposed method was trained on Kins dataset. In the experiment, the batchsize was set to be 8. The experimental results are shown in Table 4.

Table 4 Performance comparison on Kins dataset

Method	Amodal segmentation
Mask R-CNN[4]	29.2
FCIS[25]	23.5
ESE-20[22]	35.3
Deep Snake	31.3
Proposed algorithm	31.6

Compared with Deep Snake, the average precision of instance segmentation using the proposed method is improved by 0.3. In Fig.10 (a), it can identify and clearly

segment people and bicycles in the image even if objects are overlapped. In Fig. 10(b), for vehicles on the street from near to far, it can segment the contour of the instance more accurately.

(a) Person and bicycle (b) Vehicle

Fig. 10 Visualization results on Kins dataset

4 Conclusions

In this study, contrary to Deep Snake, the lightweight backbone network ShuffleNet V2 was used to replace DLA-34, and methods such as enlarged convolution kernel, depth-separable convolution to replace ordinary convolution. Lightweight up-sampling modules were used to compress the model and increase segmentation speed while maintaining accuracy. In addition, the global contour feature fusion method was used to make full use of the overall information of the contour to obtain an initial deformed contour that is closer to the real value and finally improve the accuracy of contour deformation. The proposed method was tested on the datasets of PASCAL SBD, Cityscapes, and Kins datasets. The results show that the lightweight contour learning instance segmentation algorithm proposed in this paper has an average precision of 55.8 on SBD, which is improved by 1.4 compared with Deep Snake. The model parameters are reduced by 87.2% compared with Deep Snake, and the calculation amount is reduced by 78.3%, while the segmentation speed is increased by about 23%.

The lightweight and fast segmentation algorithm proposed makes it possible to deploy instance segmentation methods on embedded platforms with limited resources. In the next step, the study will focus on how to deploy this algorithm on edge platforms such as Nvidia Xavier and achieve fast segmentation in practical application scenarios.

Acknowledgement

This work was supported by National Key Research and Development Program (No. 2022YFE0112400); National Natural Science Foundation of China (No. 21706096); Natural Science Foundation of Jiangsu Province (No.BK20160162).

Declaration of conflicting interests

The authors have no conflict of interests related to this publication.

References

[1] HARIHARAN B, ARBELÁEZ P, GIRSHICK R, et al. Simultaneous detection and segmentation//European Conference on Computer Vision, September 5-12, 2014, Zurich, Switzerland. Cham: Springer, 2014: 297-312.

[2] SU L, SUN Y X, YUAN S Z. A Survey of instance segmentation based on deep learning. Journal of Intelligent Systems, 2022, 17(1):16-31.

[3] LI X X, HU X G, QUAN Y C, et al. Research progress of instance segmentation based on deep learning. Computer Engineering and Applications, 2021, 57(9): 60-67.

[4] HAFIZ A M, BHAT G M. A survey on instance segmentation: state of the art. International Journal of Multimedia Information Retrieval, 2020, 9(3): 171-189.

[5] HE K, GKIOXARI G, DOLLÁR P, et al. Mask r-cnn// International Conference on Computer Vision, October 22-29, 2017, Venice, Italy. Cham: IEEE, 2017: 2980-2988.

[6] REN S Q, HE K M, GIRSHICK R, et al. Faster R-CNN: towards real-time object detection with region proposal networkrs. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39(6): 1137-1149.

[7] WANG K, LIEW J H, ZOU Y, et al. PANet: tew-shot image semantic segmentation with prototype alignment. International Conference on Computer Vision, October 27-November 2, 2019, Seoul, Korea. Cham: IEEE, 2019: 9196-9205.

[8] HUANG Z, HUANG L, GONG Y, et al. Mask scoring R-CNN//IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 16-20, 2019, Long Beach, CA, USA. Cham: IEEE, 2019: 6402-6411.

[9] DAI J, HE K, LI Y, et al. Instance-sensitive fully convolutional networks. European Conference on Computer Vision, October 10-16, 2016, Amsterdam, Netherlands. Berlin: Springer, 2016: 534-549.

[10] WANG X, ZHANG R, KONG T, et al. Solov2: Dynamic and fast instance segmentation. Advances in Neural Information Processing Systems, 2020, 33: 17721-17732.

[11] BOLYA D, ZHOU C, XIAO F Y, et al. YOLACT: real-time instance segmentation//2019 IEEE/CVF International Conference on Computer Vision, October 27-November 2, 2019, Seoul, Korea (South). New York: IEEE, 2019: 9156-9165.

[12] XIE E Z, SUN P Z, SONG X G, et al. PolarMask: single shot instance segmentation with polar representation// 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 14-19, 2020, Seattle, WA, USA. IEEE, 2020: 12190-12199.

[13] TIAN Z, SHEN C H, CHEN H, et al. FCOS: fully convolutional one-stage object detection//2019 IEEE/CVF International Conference on Computer Vision, October 27-November 2, 2019, Seoul, Korea (South). New York:IEEE, 2019: 9626-9635.
[14] PENG S, JIANG W, PI H, et al. Deep snake for real-time instance segmentation//IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 14-19, 2020, Seattle, WA, USA. IEEE, 2020: 8533-8542.
[15] HOWARD A G, ZHU M L, CHEN B, et al. MobileNets: efficient convolutional neural networks for mobile vision applications. 2017: 1704.04861. http://arxiv.org/abs/1704.04861v1.
[16] TAN M X, LE Q V. Efficientnet: Rethinking model scaling for convolutional neural networks. 2019: 1905.11946. http://arxiv.org/abs/1905.11946v5.
[17] ZHANG X Y, ZHOU X Y, LIN M X, et al. ShuffleNet: an extremely efficient convolutional neural network for mobile devices//2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 18-22, 2018, Salt Lake City, UT, USA. New York: IEEE, 2018: 6848-6856.
[18] MA N N, ZHANG X Y, ZHENG H T, et al. ShuffleNet V2: practical guidelines for efficient CNN architecture design// European Conference on Computer Vision, September 8-14, 2018, Munich, Germany. Berlin: Cham: Springer, 2018: 122-138.
[19] ZHOU X Y, WANG D Q, KRÄHENBÜHL P. Objects as points. 2019: 1904.07850. http://arxiv.org/abs/1904.07850v2.
[20] DAI J F, QI H Z, XIONG Y W, et al. Deformable convolutional networks//2017 IEEE International Conference on Computer Vision, October 22-29, 2017, Venice, Italy. New York: IEEE, 2017: 764-773.
[21] HARIHARAN B, ARBELÁEZ P, BOURDEV L, et al. Semantic contours from inverse detectors//2011 International Conference on Computer Vision, November 6-13, 2011, Barcelona, Spain. New York: IEEE, 2011: 991-998.
[22] CORDTS M, OMRAN M, RAMOS S, et al. The cityscapes dataset for semantic urban scene understanding// 2016 IEEE Conference on Computer Vision and Pattern Recognition, June 26-July 1, 2016, Las Vegas, NV, USA. New York: IEEE, 2016: 3213-3223.
[23] QI L, JIANG L, LIU S, et al. Amodal instance segmentation with KINS dataset//2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 16-20, 2019, Long Beach, CA, USA. New York: IEEE, 2019: 3009-3018.
[24] LI Y, QI H Z, DAI J F, et al. Fully convolutional instance-aware semantic segmentation//2017 IEEE Conference on Computer Vision and Pattern Recognition, July 21-26, 2017, Honolulu, HI, USA. New York: IEEE, 2017: 4438-4446.
[25] XU W Q, WANG H Y, QI F B, et al. Explicit shape encoding for real-time instance segmentation// 2019 IEEE/CVF International Conference on Computer Vision, June 16-20, 2019, Seoul, Korea (South). New York: IEEE, 2019: 5167-5176.

基于轮廓学习的实时实例分割

葛　锐[1,2], 刘登峰[1,2*], 周浩杰[1,2], 柴志雷[1,2], 吴　秦[1,2]

1. 江南大学 人工智能与计算机学院, 江苏 无锡 214122;
2. 江南大学 江苏省模式识别与计算智能工程实验室, 江苏 无锡 214122

摘　要: 实例分割在图像处理中发挥着重要作用, 基于轮廓迭代的Deep Snake算法实现初始目标框到目标轮廓端到端变形, 能提升实例分割的性能, 但存在分割速度较慢、初始轮廓欠优等缺陷。针对这些问题, 提出了一种轻量化的实例轮廓迭代分割算法。首先, 采用ShuffleNet V2作为主干网络, 使用5×5卷积核扩大了模型感受野。其次, 设计了轻量级上采样模块—多阶段聚合(Multi stage aggregation, MSA)进行多层特征的残差融合, 不仅提高了分割速度, 而且更加全面地提取了有效特征。再次, 设计了一种网络学习的轮廓初始化方法, 采用全局轮廓特征聚合机制回归得到粗糙轮廓, 解决了手工初始化轮廓和真实轮廓误差过大的问题。最后, 使用Snake模型对粗糙轮廓进行迭代优化得到最终的实例轮廓。实验结果表明, 所提出的方法在Semantic boundaries dataset (SBD)、Cityscapes和Kins数据集上提高了实例分割精度, 在SBD数据集上平均分割精度达到55.8, 同时模型参数量较Deep Snake减少了87.2%, 计算量减少了78.3%, 在2080Ti GPU上对512×512像素的图片进行实例分割时速度达到39.8帧·s^{-1}。该方法能够减少资源消耗, 快速准确实现实例分割任务, 更适合应用于资源有限的嵌入式平台。

关键词: 实例分割; ShuffleNet V2; 轻量化网络; 轮廓初始化

引用格式: GE Rui, LIU Dengfeng, ZHOU Haojie, et al. Real-time instance segmentation based on contour learning. Journal of Measurement Science and Instrumentation, 2024, 15(3): 328-337.