The Visual Computer (2023) 39:1137–1148
https://doi.org/10.1007/s00371-021-02393-y

ORIGINAL ARTICLE

Edge-enhanced instance segmentation by grid regions of interest

Ying Gao[1] · Zhiyang Qi[1] · Dexin Zhao[1]

Accepted: 23 December 2021 / Published online: 29 January 2022

Abstract
This paper focuses on the instance segmentation task. The purpose of instance segmentation is to jointly detect, classify and segment individual instances in images, so it is used to solve a large number of industrial tasks such as novel coronavirus diagnosis and autonomous driving. However, it is not easy for instance models to achieve good results in terms of both efficiency of prediction classes and segmentation results of instance edges. We propose a single-stage instance segmentation model EEMask (edge-enhanced mask), which generates grid ROIs (regions of interest) instead of proposal boxes. EEMask divides the image uniformly according to the grid and then calculates the relevance between the grids based on the distance and grayscale values. Finally, EEMask uses the grid relevance to generate grid ROIs and grid classes. In addition, we design an edge-enhanced layer, which enhances the model's ability to perceive instance edges by increasing the number of channels with higher contrast at the instance edges. There is not any additional convolutional layer overhead, so the whole process is efficient. We evaluate EEMask on a public benchmark. On average, EEMask is 17.8% faster than BlendMask with the same training schedule. EEMask achieves a mask AP score of 39.9 on the MS COCO dataset, which outperforms Mask RCNN by 7.5% and BlendMask by 3.9%.

Keywords Instance segmentation · Single-stage · Regions of interest · Edge-enhanced

1 Introduction

Instance segmentation, a downstream task of object detection, is a very difficult task. The first is that segmentation requires predicting the class of each instance. Not only the number of instances in the image is uncertain, but also the pose, angle, and size of each instance is unknown. Meanwhile, the edges of each instance are very complex, which make models hard to achieve a good mask outcome.

Due to the models hard to determine exactly where the instances are located, it becomes difficult to predict the classes of all instances in an image. The current mainstream approaches are divided into two-stage and single-stage. The two-stage models [14] use CNNs (convolutional neural networks) to filter out some proposal boxes first and then classifies and regresses these proposal boxes. The accuracy of the two-stage models is higher, but the speed is slower. The reason is that the two-stage models can only be executed sequentially and require two classifications and regressions. Most single-stage models [15,26] are densely sampled at different locations of the image and use CNN networks to extract features. After that, classification and regression can be performed by using those features. Single-stage models are fast because they only need to perform classification and regression once. However, most of the single-stage models are not as accurate as the two-stage models because of its simple structure.

Edge optimization for instance segmentation is a challenging task. Because the edges of each instance are diverse, and the edges of instances in different poses are complex and varied. Most of the network model predictions are smoothed, i.e., neighboring pixels often use the same label. Therefore, in the edge regions of the instances, the segmentation results are often different from the actual ones.

To address above two difficulties, we propose a single-stage instance segmentation model EEMask. For the first difficulty, we design a SAM (Spatial Attention Module), which establishes the relevance between two regions based on their grayscale values and the distance. There is a high probability that pixels with the same greyscale value and close distance belong to one instance. According to this

✉ Dexin Zhao
zhaodexin@email.tjut.edu.cn

[1] Tianjin Key Laboratory of Intelligence Computing and Novel Software Technology, Tianjin University of Technology, No.391 West Binshui Road, Tianjin 300384, China

rule, we can make the different parts of an instance of an image more relevant. For the second difficulty, we present an edge-enhanced layer which eliminates the redundant channels by adjusting the ratio between channels. Some channels lose basic image information during the convolution process. These channels become redundant channels, which are not beneficial to model learn. Some channels have a relatively large contrast between the instances and the background. These channels allow the model to place an emphasis on the edges of the instances. Our methods do not need to add additional convolutional layers, so the whole process is efficient.

To prove that EEMask is feasible, we do visualization and many ablation experiments, see Sect. 4 for details. In the visualization, we verify that EEMask can capture the ROIs in the image by visualizing the last convolutional layer of the SAM. In the experiments, we do many ablation experiments to verify that all modules in EEMask are useful. Finally, we evaluate EEMask with the state-of-the-art models on the COCO 2017 dataset [22]. EEMask is on average 17.8% faster than BlendMask. EEMask with ResNet-101 [13] as the backbone network obtained 39.9 AP, which is a 3.9% improvement compared to BlendMask and exceeds most of the latest state-of-the-art models.

In summary, the contributions of this paper are:

- A novel, single-stage and end-to-end instance segmentation model, which uses the characteristics of different dimensional feature maps for targeted processing;
- A lightweight SAM that generates accurate class information of instances by computing relevance between different regions in an image.
- A lightweight edge-enhanced layer that improves the model's perception of instance edges by adjusting the ratio between channels.

2 Related work

Object detection. Object detection, the downstream task of image classification(each image has only one class), solves the task that there are multiple objects in an image. Object detection is more challenging than traditional image classification, which consists of two subtasks:object classification and localization. First, objects can appear anywhere in the image with different sizes. Second, there are various poses and angles of those objects, moreover, some objects may be obscured. Traditional object detection methods [5,7–9,32,33] are weak in maintaining instance integrity, and handmade features lack robustness. These methods are gradually being replaced by neural networks. Neural network-based object detection algorithms [11,12,14,15,23,24,26,29] include two categories. One category is two-stage models based on region proposal. Such as R-CNN [11], Fast R-CNN [12], Faster R-CNN [29] and Mask R-CNN [14] etc. They need to first generate the bounding boxes of objects, then classify and regress the bounding boxes. Another category is single-stage models such as YOLO [15] and SSD [26], which use only one convolutional neural network(CNN) to directly predict the class and location of different objects.

Single-stage object detectors are able to speed up existing two-stage detectors by simply removing the second stage and compensating for performance loss in other ways (e.g., strong data augmentation, anchor clustering.). However, the same approach is not easily extend to instance segmentation. Instance segmentation is much more complex than object detection. The state-of-the-art two-stage instance segmentation methods depend heavily on feature localization to generate masks. These methods "re-pool" features in some bounding box region, and then feed these localized features to their mask predictor. This process is inherently sequential and therefore cannot be accelerated. Although the single-stage models predict the class and generate the base masks are run in parallel. The models then fuse the two parts together to generate the final mask. But it is difficult to determine which branch ends first.

EEMask has three main improvements in the efficiency of predicting instance classes. The first is the simplification of the semantic branching operations using a grid layer, i.e., dividing the image uniformly into a fixed number of grids. Second, EEMask generates ROIs instead of bounding boxes and generates grid-ROIs. Finally, EEMask runs in parallel and does not add redundant convolutional layers. In addition to speed improvements, equally important is accuracy. In our previous work, we have proposed DCM [40] (Dual Context aggregation Module), which builds relevance by constructing pairwise relationships between positions of the same row and column to improve feature representation. We continue to improve on our previous work. We redesigned the SAM, which uses grayscale values and distances to establish relevance between different grids in the whole image. Relevance is established not only in the horizontal and vertical directions, but also in the oblique direction. The multi-directional relevance between grids allows EEMask to better identify the position of the instances, thus improving the recognition accuracy.

Instance Segmentation. With the advent of CNNs [16,17,37], many instance segmentation models have been proposed. For example [1,14,25,27,39], the precision of instance segmentation accuracy grew rapidly [6]. Mask R-CNN [14] implements a full convolutional network that is added to the structure of the Faster R-CNN [29] for pixel-level classification. Mask R-CNN [14] is a representative two-stage instance segmentation model, that first scans the image and generates proposals, then generates bounding boxes and masks. FCIS [21] is the first fully convolutional end-to-end

instance-aware semantic segmentation model, which outputs instance masks and classes information by computing location sensitive inside/outside score maps. YOLACT [2] is the first single-stage real-time instance segmentation model that divides the instance segmentation task into two parallel subtasks:generating a set of prototype masks and predicting the per-instance mask coefficients. TensorMask [4] presents the first dense sliding window instance segmentation model that uses a structured 4D tensor to represent the mask in the spatial regions, achieving almost the same results as the Mask R-CNN. PolarMask [34] is an anchor-box free and single shot instance segmentation model, which formulates the instance segmentation problem as predicting the contours of instances by instance center classification and dense distance regression in polar coordinates. Pointins [28] is a novel idea of point-based instance segmentation framework, which is able to maintain performance while improving speed. In recent years, some multi-branch and multi-scale network models [3,38] have become popular, because it exploits the advantages of different scale feature maps. The multi-branch and multi-scale network MMALNet [38] has good classification ability and robustness for images of different scales. BlendMask [3] is an instance segmentation model that combines top-down and bottom-up design strategies, and it utilizes a more reasonable blender module to fuse high-level and bottom-up features. Eventually Blendmask outperforms Mask R-CNN in terms of speed and accuracy.

Recently, single-stage models have surpassed two-stage models in terms of speed and accuracy, such as the BlendMask. Single-stage models often have low segmentation accuracy for instance edges due to their simple structure. We designed an edge-enhanced layer, which removes redundant channels and adds channels that contain richer edge information. In this way, the model allows for better learning of edge details.

In summary, CNN-based models have greatly outperformed traditional models. Even some single-stage models surpass the two-stage models in terms of speed and accuracy, such as BlendMask. However, most current single-stage models suffer from two problems. On the one hand, it is difficult for models to predict the instance class quickly and accurately. On the other hand, the segmentation of instance edges is not accurate enough. For the first one, we replace the bounding boxes with grid-ROIs to improve efficiency and create region relevance to improve the accuracy of class prediction. For the second, we adjust the ratio between channels to enhance the model's perception of instance edges.

3 Our approach

In this section, we first briefly introduce the overall pipeline of the EEMask. Then, we introduce the grid layer and the SAM in the semantic branch. Next, we introduce the edge-enhanced layer and the CAM (Channel Attention Module) in the mask branch, a module that generates the final mask. Finally, we introduce a new L_{EEMask} (see Sect. 3.4) to optimize the quality of the CAM to generate the final mask.

3.1 Overall pipeline

EEMask is based on simultaneous prediction of two branches, which completely eliminates bounding boxes. EEMask uses ResNet101 as the backbone network. The ROIalign layer is used to merge feature maps of different sizes in the FPN to generate higher-level feature maps. The prediction network of EEMask consists of semantic branch and mask branch. The semantic branch contains two parts: grid layer and SAM. The grid layer for simplifying the processing of feature maps. The SAM for predicting the grid-ROIs and the class of each grid. The mask branch also contains two parts: edge-enhanced layer and CAM. The edge-enhanced layer for adjusting the ratio of channels. The CAM for merging grid-ROIs and base-mask features. The whole structure is shown in Fig. 1.

3.2 Semantic branch

The semantic branch is mainly used to predict the classes and ROIs of the instances in images, and it uses the high-level feature maps in the FPN as input. The higher level feature maps contain larger perceptual fields and more semantic information. When predicting ROIs, EEMask needs to focus more on the location information and the affinity between grids. CNNs are insensitive to Cartesian coordinate systems, which cannot be interpreted as One-Hot information. Inspired by the idea of "coordinate transformation" [10], we use fused coordinate features to solve the problem of CNN insensitivity to location. The coordinate features consist of two matrices. The size of the matrix is S*S (S is the number of grids). The first matrix has the same value in each row. The value of each column is from zero to S-1 in integer increments. The second matrix is a symmetric matrix of the first matrix. Then the values of the two matrices are normalized to the range [-1, 1]. Finally, they are added to the last two channels of the feature maps. For the affinity problem between grids, we propose the SAM. See Sect. 3.2.2 for details.

3.2.1 Grid layer

The grid layer divides the higher-level feature maps of dimension (C,H,W) into S^2 grids uniformly in space. The grid feature maps of dimension (C, S, S) are obtained. In the specific process of dividing the grid, we keep the floating-numbers and use bilinear interpolation to generate the values for each grid. We specify that each grid can represent only one class. The right number of grids can accelerate the train-

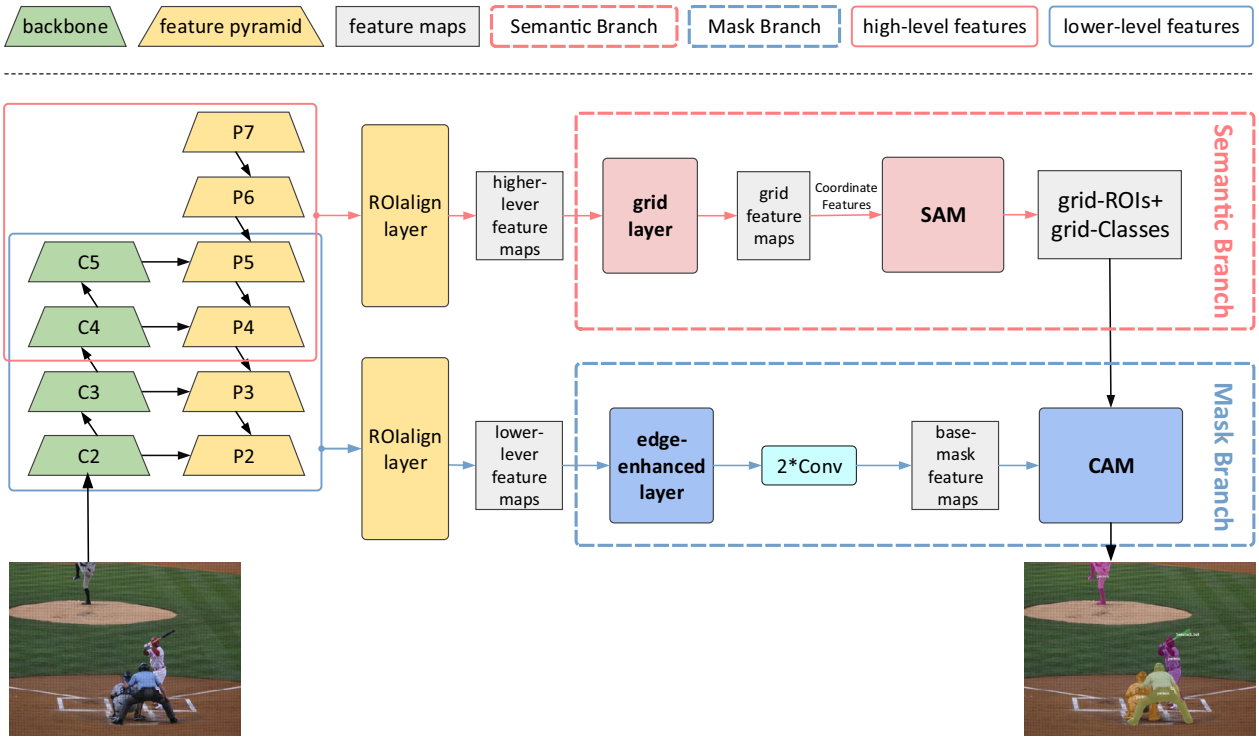

Fig. 1 The boxes of different categories in the figure represent different modules. The specific category of each module is shown at the top of the figure. EEMask uses the fusion of the lower-level features with the high-level features to generate the final mask. The SAM is added to the top of the detection tower to generate grid-ROIs and grid-Classes by calculating the regional relevance of the feature maps. The lower-level features of the backbone and FPN networks are used to generate the base-mask feature maps. The CAM is used to linearly combine the grid-ROIs, grid-Classes and base-mask feature maps to generate the final mask. See Sects. 3.2 and 3.3 for processing details in the semantic branch and the mask branch, respectively

ing while maintaining relatively high accuracy. We compare different numbers of grids in the ablation experiments in Chapter 4.

3.2.2 SAM

The SAM takes the grid feature maps as input, and it improves the feature representation by capturing the global context. Finally, SAM generates the grid ROIs and grid classes. Capturing long-distance-dependent information aims to obtain global contextual information and enhance the discrimination of similar features. Convolutional neural networks obtain image features by stacking network layers, which is relatively inefficient. In this paper, we capture dense spatial contextual information by establishing relevance between different grids. Establishing relevance between grids makes EEMask more accurate in determining the class of each grid.

As shown in Fig. 2, the SAM is first divided into three branches. Each branch contains two 1*1 convolutional layers. The outputs of the 1*1 convolution layers are the feature maps A, B and C respectively. The 1*1 convolution does not change the size S of the grid feature maps. Its main function is to reduce the amount of data by reducing the number of channels. Feature map A is mainly used to calculate grid-ROIs. Feature map B is mainly used to calculate the relevance between grids. Feature map C is mainly used to calculate the classes represented by each grid. We first perform the affinity operation on B. The equation for the affinity operation is as follows.

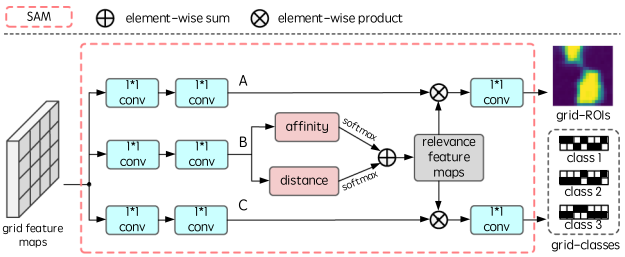

Fig. 2 SAM first establishes the region relevance of the image and then further calculates the grid-ROIs and grid-Classes based on the relevance feature map. The whole SAM is calculating in the size of S*S

$$AF_{ij} = \sqrt{|B_{i_x,i_y} - B_{j_x,j_y}|}$$

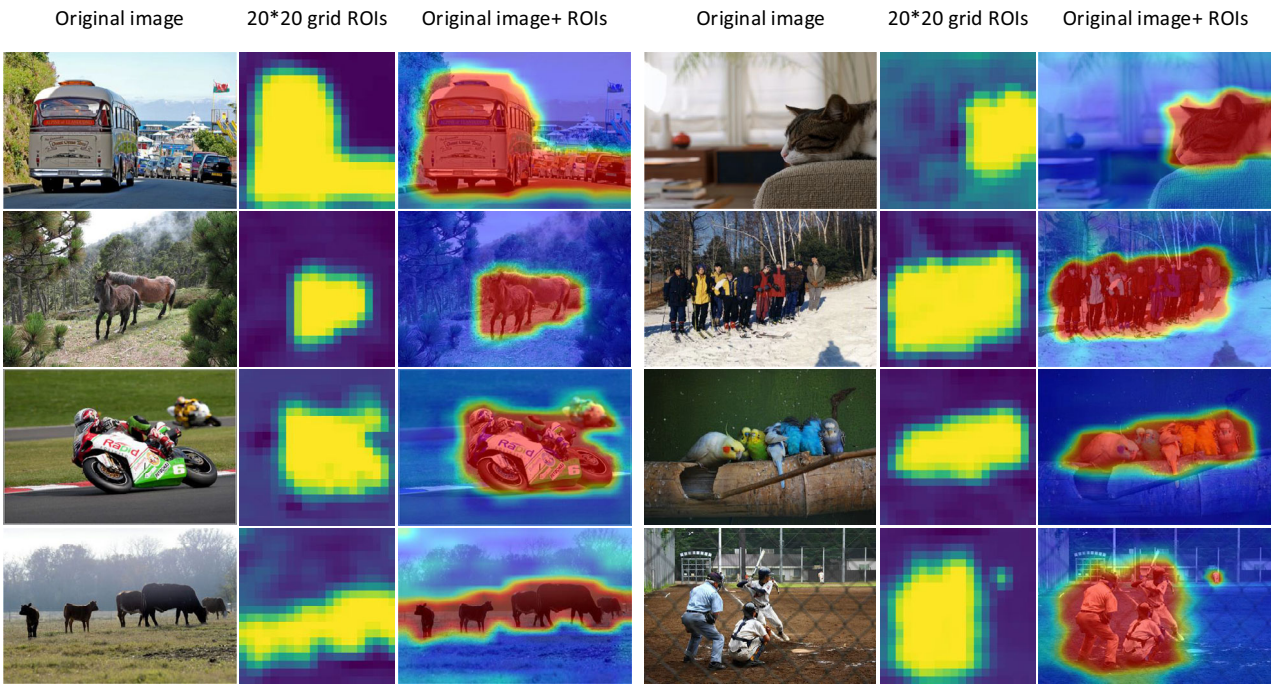

Fig. 3 We visualize the ROIs for different scenes of the images. The number of 20*20 grids is taken as an example. In grid-ROIs, the yellow area represents the model is more attention. In ROIs, the redder color means the model is more attention, and the bluer color means the model is less attention. Due to the multi-directional relevance between the grids, EEMask is able to capture the poses of the instances accurately

The AF_{ij} represents the affinity between grid i and grid j. The equations i, j represent the ordinal numbers in the grid feature maps, from left to right and from top to bottom. i_x represents the row number of grid i and i_y represents the column number of grid i, j similarly. The specific formulas for i_x,i_y are: $i_x = i/S$,$i_y = i\%S$. B_{i_x,i_y} represents the value of grid i in feature map B. The smaller difference between the values of two grids in feature map B indicates the stronger affinity. And the distance between two grids is calculated by the following equation.

$$D_{ij} = \sqrt{(i_x - j_x)^2 + (i_y - j_y)^2}$$

D_{ij} represents the distance between grid i and grid j. Some recent multi-feature fusion architectures [18,19,39] give us some reference. The features with high discrimination and low correlation should be selected and provided with high weights in fusion. In the process of AF_{ij} and D_{ij} fusion, AF_{ij} is with high discrimination, because AF_{ij} represents the affinity between two grids. D_{ij} represents the distance between two grids, which is lower discrimination. Therefore, we give higher weight to AF_{ij}, converge faster and gain more accurate results. After obtaining AF_{ij} and D_{ij}, the final relevance feature maps are obtained by the following equation.

$$R_{ij} = \alpha * softmax(AF_{ij}) + softmax(D_{ij})$$

R_{ij} represents the relevance between grid i and grid j, which is obtained by summing AF_{ij} and D_{ij}, and $R_{ij} = R_{ji}$. α is the dynamic weight of $softmax(AF_{ij})$, which is trained by the network. We specify alpha$\in$ [1, 10] and set alpha to the threshold value when it exceeds the threshold value. Smaller R_{ij} means stronger relevance between grid i and grid j. The specific time complexity of generating the relevance feature maps is $\frac{1}{2}(S^2 - 1)S^2$, where S is the number of grids. The feature map A is combined with the relevance feature maps to generate the grid-ROIs of the image. Feature map C is combined with the relevance feature maps and then passed through the softmax function to generate the classes of each grid.

To demonstrate the ability of EEMask to capture ROIs in images, we visualized the final ROIs generated by the SAM. We draw on recent work in CNN visualization and model explanations. Visualizing CNNs, a number of previous works [30,35,36,41] have visualized CNN predictions by highlighting "significant" pixels (i.e., those pixels whose changes have the greatest impact on prediction scores). The darker the image color, the higher the value of interest. High interest values indicate that the model is more concerned about the region. We tested this with images of different scenes. The results are shown in Fig. 3.

3.3 Mask branch

The mask branch is mainly used to generate the base-mask feature maps and the final mask. The mask branch takes as input the lower-level features in the backbone network and the FPN. Because individual pixels of the lower-level features have a smaller receptive field and more detailed information, e.g., texture, edges.

3.3.1 Edge-enhanced layer

The edge-enhanced layer is the module that we propose to perform data enhancement on the feature maps. The specific implementation is to adjust the ratio between the channels. We propose an average grayscale distant value to determine whether the channels are favorable or not. The specific calculation is as follows:

$$AG = \left(\sum_{l=1}^{H*W} p_l\right) / H * W$$

We first use the above formula to get the average grayscale value AG, where p_l denotes the grayscale value of pixel l. Then we solve for the number of pixels that are far from the AG among all pixels. We judge the grayscale value of each pixel. If the pixel value is greater than $AG + \beta$, or less than $AG - \beta$, the pixel is counted as a distance pixel. β is a parameter for the dynamic training of the neural network and is mainly used to control the distance away from the average grayscale value. We count the number of distance pixels in channel k and write it as n_k. We calculate the distance pixels as a percentage of the total pixels in the channel, $P_k = n_k/(H * W)$. Finally, we calculate the number of channel k using the following equation:

$$N_k = \left\lceil \frac{P_k}{sum(P)} * TNC \right\rceil$$

N_k represents the new number of the channel k. $sum(P)$ represents the cumulative sum of the distance pixel ratios in all channels. TNC represents the original total number of channels. Rounding upward ensures that each channel is trained. After getting N_k, we make a copy of the channels whose scaling needs to be scaled up.

The simple flow is shown in Fig. 4. The original image is passed through the backbone network+FPN module, and the ROIalign layer to obtain the lower-level feature maps. There are 64 channels in the lower-level feature maps. In Fig. 4, we give an example with some of the channels. The N_k of each channel is calculated by our proposed formula. Finally, we change the number of channels k to N_k. By this way, we increase the proportion of channels that are easily distinguishable at the edge.

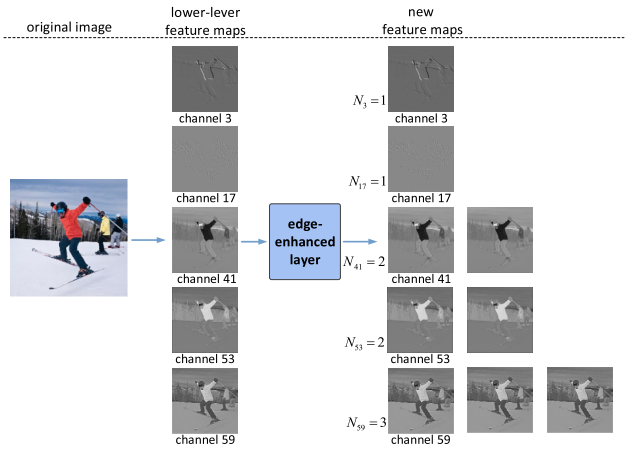

Fig. 4 Overall framework of the control channels. Channel {3,17,41,53,59} was selected from the 64 channels

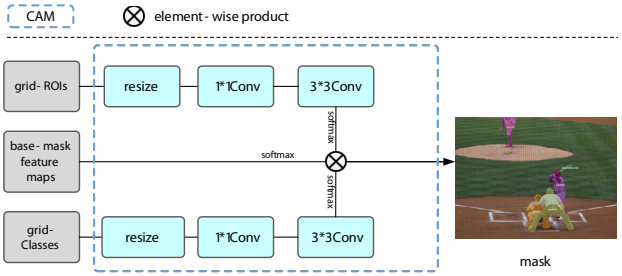

Fig. 5 CAM generate masks by combining grid-ROIs, grid-Classes and base-mask feature maps

In order to fix the number of channels in the new feature maps, we set the threshold value is 90. When the number of channels is greater than 90, we scale down the channels with larger N_k values accordingly. When the number of channels is less than 90, we increase the channels with larger N_k values accordingly. In this way, we ensure that the number of channels output by the edge-enhanced layer is 90.

3.3.2 CAM

The CAM module's main function is combining the feature maps of the different parts to generate the final mask.

As shown in Fig. 5, the input of the CAM is divided into three parts, which are the base-mask feature maps generated by the lower-level features, the grid-ROIs and the grid-Classes generated by the semantic branch. The grid-ROIs are treated the same way as the grid-Classes, but not the same network. Both resize the grid first so that the feature map becomes the original size. Then a 1*1 convolutional layer and a 3*3 convolutional layer are processed. Finally, the K dimension is normalized with the softmax function. The processing of the base-mask feature maps is normalized with the softmax function along the K dimension. The specific processing of each stage is shown in the table.

Table 1 Transformation process of feature map in CAM. F_r represents gird-ROIs, F_b represents base-mask feature maps, and F_c represents grid-Classes

Value	Pre-dimension	Process	Dimension
F_r	[1, S, S]	Resize	[1, H, W]
	[1, H, W]	1*1Conv	[K, H, W]
	[K, H, W]	3*3Conv	[K, H, W]
	[K, H, W]	Softmax	[K, H, W]
F_b	[K, H, W]	Softmax	[K, H, W]
F_c	[1, S, S]	Resize	[1, H, W]
	[1, H, W]	1*1Conv	[K, H, W]
	[K, H, W]	3*3Conv	[K, H, W]
	[K, H, W]	Softmax	[K, H, W]

Finally, we apply the element-wise product to the feature maps F_r, F_b and F_c, and sum along the K dimensions to obtain final mask:

$$mask = \sum_{k=1}^{K} F_r^k \circ F_b^k \circ F_c^k$$

3.4 Loss function

EEMask proposed in this paper adopts a single-shot end-to-end training. In the training process, "poly" strategy is adopted. On the basis of basic learning rate, multiply by $(1 - \frac{iter}{max\_iter})^{power}$, where $power$=0.9. Our network is trained with min-batch stochastic gradient descent (mini-SGD). During the experiment, InPlace-ABN is applied to synchronize the mean and standard deviation of BN on multiple GPUs. On the MS COCO training set, 36 epochs are trained with the initial learning rate of 0.01. During the training, when the 24th epoch is completed, the learning rate is divided by 10. On the 30th epoch, it is divided by 10 again. Set the momentum of the optimizer to 0.9 and the weight decay to 0.001. The weight parameters of the backbone are initialized by the pre-trained network parameters in ImageNet.

The definition of the training loss function is also based on the consideration of two parts. One is the classification loss and the other is the mask prediction loss. The definition is as follows:

$$L_{EEMask} = L_{cl} + \lambda L_{mask}$$

where L_{cl} refers to class loss:

$$L_{cl} = -\{\alpha(1-p)^r log(p) + (1-\alpha)p^r log(1-p)\}$$

Among them, p is the confidence of the sample, and α is used to weigh the class. Here, set α=0.25 and r=2. The loss of mask prediction is calculated with L_{mask}

$$L_{mask} = \frac{1}{NP} \sum_{m} f(c_{i,j}) d_{mask}(p, p_t)$$

Among them, NP represents the number of positive samples, i, j represents the coordinate values of the samples, $m = i \cdot S + j$, $f(c_{i,j})$ is a marking function. When the value of $c_{i,j}$ is greater than 0, its value is 1, otherwise the value is 0. p and p_t represent training value and label value, respectively. d_{mask} is calculated by the Dice Loss function:

$$L_{Dice} = 1 - \frac{2\sum_{i,j}(p_{i,j} \cdot q_{i,j})}{\sum_{i,j} p_{i,j}^2 + \sum_{i,j} q_{i,j}^2}$$

Here, $p_{i,j}$ refers to the predicted value of the grid with coordinates at (i, j), and $q_{i,j}$ represents the target value.

4 Results and discussion

4.1 Dataset and evaluation metrics

The experimental dataset is the MS COCO 2017 dataset with label information comprising classes, location information and text descriptions, and it can be used for various tasks, such as object detection, image caption and instance segmentation. The COCO dataset contains 118,287 training images, 5,000 validation images and 40,670 test images, including 80 target classes and 91 segmentation classes. The dataset is applied in ablation studies, with 5000 images of the validation set for evaluation. The evaluation metrics are COCO mask average precision (AP), AP at IoU 0.5 (AP_{50}), 0.75 (AP_{75}) and AP for objects at different sizes AP_S (AP for small objects: area $< 32^2$), AP_M(medium objects: 32^2 < area < 96^2), and AP_L(large objects: 96^2 <area).

4.2 Ablation studies

We investigate the effectiveness of each module in our semantic branch and mask branch by performing ablation experiments. The performance and time of EEMask is measured by one image per batch on a 1080Ti GPU (11G).

Number of grids The number of grids affects the prediction of different size instances. If the number is too large, small-sized objects are easily ignored in recognition. Conversely, the small size of the grid is beneficial for the segmentation of small-sized instances, but it causes a lot of unnecessary calculations. Therefore, we choose different grid numbers in {10,15,20,25,30} to find the best case. As shown in Table 2, when the grid number is 10*10, the receptive field of a single grid is too large, so it is difficult to identify instances with small size. When the grid number is 30*30, a single grid receptive field is too small, so it is difficult to distinguish

Table 2 Experiments show that when the number of grids is small, the performance is poorer in small objects. When the number of grids is higher, the performance is poorer in large objects

Number of grids	AP	AP_S	AP_M	AP_L
10*10	38.6	16.9	42.4	58.4
15*15	39.6	17.0	43.2	58.6
20*20	39.9	17.2	43.8	58.8
25*25	39.6	17.2	43.4	58.5
30*30	39.5	17.2	42.8	57.9

Table 3 Comparison experiments of different shapes of grid. Numbers represents the number of grids and shape represents the shape of the grids. It can be observed in the table that the rectangular grid does not perform as well as the square grid

Numbers	Shape	AP	AP_S	AP_M	AP_L
10*20	Rectangle	37.8	15.2	40.8	56.5
15*25	Rectangle	38.1	15.8	41.6	56.6
20*30	Rectangle	37.4	15.9	41.1	55.8
20*20	Square	39.9	17.2	43.8	58.8

the instances with large size from the background. When the grid number is set to 20*20, the accuracy reaches a relatively high score and does not generate too many parameters. From the data in the table, we can judge that the optimal value of grid number is between 10 and 30. We believe that a better result can be achieved by refining. Of course, this optimal grid number is only suitable for the MS COCO 2017 dataset.

Shape of the grid We explore the results obtained by using a rectangular grid shape compared to a square grid shape. Because the size of the image is randomly variable, there are H> W and W> H. If we use rectangle, it will cause some images to be compressed too much in one direction. For example, if the image size is H*W, 640*360, and the grid numbers is 20*30, this will lead to over-compression in height and loss of feature information of the image. Compared to rectangles, squares are a best method for compatibility and robustness. We have also conducted relevant ablation experiments to confirm this. From the results, it can be observed that the rectangular grid shape has a much lower AP and other evaluation values than the square grid shape. But we agree that better results can be achieved by refining the grid number. The results are shown in Table 3.

Coordinate features The disadvantage of traditional CNN is that the effect of coordinates cannot be considered in pixel segmentation. Although convolutional kernels perform well in handling local information, they do not work well in segmenting instance edges. Instead, we enhance the model's ability to perceive location by fusing coordinate features. The fused coordinate features affect the final result by influencing the region relevance.

Table 4 EEMask without fused coordinate features achieved 39.3 AP. The segmentation result of EEMask with added coordinate features improved significantly to 39.9 AP, which is 0.6 higher than the original score. The experimental results show that fusing the coordinate features twice does not continue to improve the scores

Times	AP	AP_{50}	AP_{75}
0	39.3	61.5	42.8
1	39.9	62.5	43.5
2	39.9	62.4	43.4

Table 5 Grid represents the grid layer, Pooling represents the pooling layer, and Channel represents the edge-enhanced layer

Grid/Pooling	Channel	Time(ms)	AP	AP_{50}	AP_{75}
Pooling		86.4	38.3	60.4	41.9
Grid		86.0	38.5	60.6	42.0
Pooling	✓	86.9	39.6	62.1	43.3
Grid	✓	86.4	39.9	62.5	43.5

Table 6 Grid represents the grid layer, Pooling represents the pooling layer, and Coordinate represents the coordinate features

Grid	Pooling	Coordinate	Time(ms)	AP	AP_{50}	AP_{75}
	✓		85.1	39.3	61.5	42.8
✓			84.9	39.3	61.4	42.8
	✓	✓	86.9	39.6	62.1	43.3
✓		✓	86.4	39.9	62.5	43.5

We perform ablation experiments on the number of fused coordinate features to test whether it affects the segmentation performance. The results are shown in Table 4.

Grid layer and edge-enhanced layer In our model, we add a grid layer to reduce the number of parameters and computations, and an edge-enhanced layer to improve the accuracy of EEMask. The results of the ablation experiments are shown in Table 5. The advantage of the grid layer over the pooling layer is that it can be better integrated with the coordinate features. The results of the ablation experiments comparing the grid layer with the pooling layer are shown in Table 6. In addition, we also performed experiments on the method of generating each grid value, and the results are shown in Table 7.

Mask branch We add the ablation experiments with mask branch and without mask branch (using one convolutional layer to produce the final mask by fusing the three feature maps). We compare EEMask with the classical instance segmentation model Mask RCNN. The final experimental results show that our mask branch works well. EEMask

Table 7 Nearest represents nearest neighbor upsampling and bilinear is bilinear interpolation

Nearest	Bilinear	AP	AP_{50}	AP_{75}
✓		39.7	62.3	43.4
	✓	39.9	62.5	43.5

Table 8 The effect of mask branching and convolution layers on the results

Numbers	AP	AP_S	AP_M	AP_L
Mask R-CNN	37.1	16.9	39.9	53.5
EEMask+Conv	39.4	17.0	42.5	57.2
EEMask+Mask branch	39.9	17.2	43.8	58.8

Table 9 The effect of different loss functions on the results

Loss Function	AP	AP_{50}	AP_{75}
BCE	38.8	60.8	42.3
Focal Loss	39.1	61.3	42.6
Dice Loss	39.8	62.5	43.5

without mask branching also achieves a high score because we establish the relevance between the grids. The semantic information in grid-ROIs and grid-Classes is more obvious because we establish relevance between grids. The detail information in the base mask feature maps is more obvious because we add edge enhancement layers. CAM combines the feature maps with different information and finally achieves a high score. The results are shown in Table 8.

Loss functions Different loss functions also have an effect on the results of the experiment. When considering the function to optimize the mask loss, we choose the commonly used BCE(binary cross-entropy loss), FL(Focal Loss), and DL(Dice Loss) for comparison.

In the calculation of the cross-entropy loss, the loss parameter is initially set to 10. In the Focal Loss, it is set to 25. These three loss functions have different emphasis on sample processing. The cross-entropy loss checks each pixel and compares the prediction result of each pixel class with its One-Hot label. For instance, with different numbers of pixels, it does not perform well. Focal Loss divides pixels into easy-to-learn and hard-to-learn, reduces the loss of easy-to-learn samples, and focuses on difficult-to-learn samples. Dice Loss views the problem in a "holistic" way, which measures the loss refers to the Dice coefficient, automatically establish the balance between foreground and background pixels. It can be seen from Table 9 that Dice Loss has obvious advantages compared to the other loss functions. The training effect of Focal Loss is slightly better than BCE.

4.3 Experimental results

In our work, we use ResNet-50 and ResNet-101 as the backbone of the network on the experiments with the COCO dataset and compare our results with state-of-the-art models including two-stage and single-stage. The results demonstrate the superiority of EEMask based on region relevance and edge-enhanced.

We present the benchmark results for EEMask in Table 10. The network based on ResNet-101 achieves a mask AP score of 39.9, which is better than most advanced instance segmentation methods, including BlendMask [3]. When ResNet-50 is adopted as the backbone, EEMask also achieves 38.1 AP, which is comparable to the effect of some networks based on ResNet-101.

Since EEMask does not rely on bounding boxes to detect objects, it is able to fully recognize large-scale instances. Moreover, due to the integration of global context, EEMask

Table 10 Compare the overall performance of mainstream methods on the COCO dataset

Method	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage:							
FCIS [8]	Res-101	29.2	49.5	–	7.1	31.3	50.0
CenterMask [14]	Hourglass-104	34.5	56.1	36.3	16.3	37.4	48.4
Mask R-CNN [20]	Res-101-FPN	37.1	60.0	39.4	16.9	39.9	53.5
Single-stage:							
YOLACT [5]	Res-101-FPN	31.2	50.6	32.8	12.1	33.3	47.1
TensorMask [33]	Res-101-FPN	37.2	59.3	39.2	17.4	39.6	51.2
PolarMask [22]	Res-101-FPN	32.1	53.7	33.1	14.7	33.8	45.3
FCOS [31]+PointINS [28]	Res-101-FPN	38.3	60.3	40.0	18.1	40.3	52.4
BlendMask [23]	Res-101-FPN	38.4	60.7	41.3	**18.2**	41.5	53.3
EEMask(ours)	Res-50-FPN	38.1	59.7	41.3	15.8	40.6	54.6
EEMask(ours)	Res-101-FPN	**39.9**	**62.5**	**43.5**	17.2	**43.8**	**58.8**

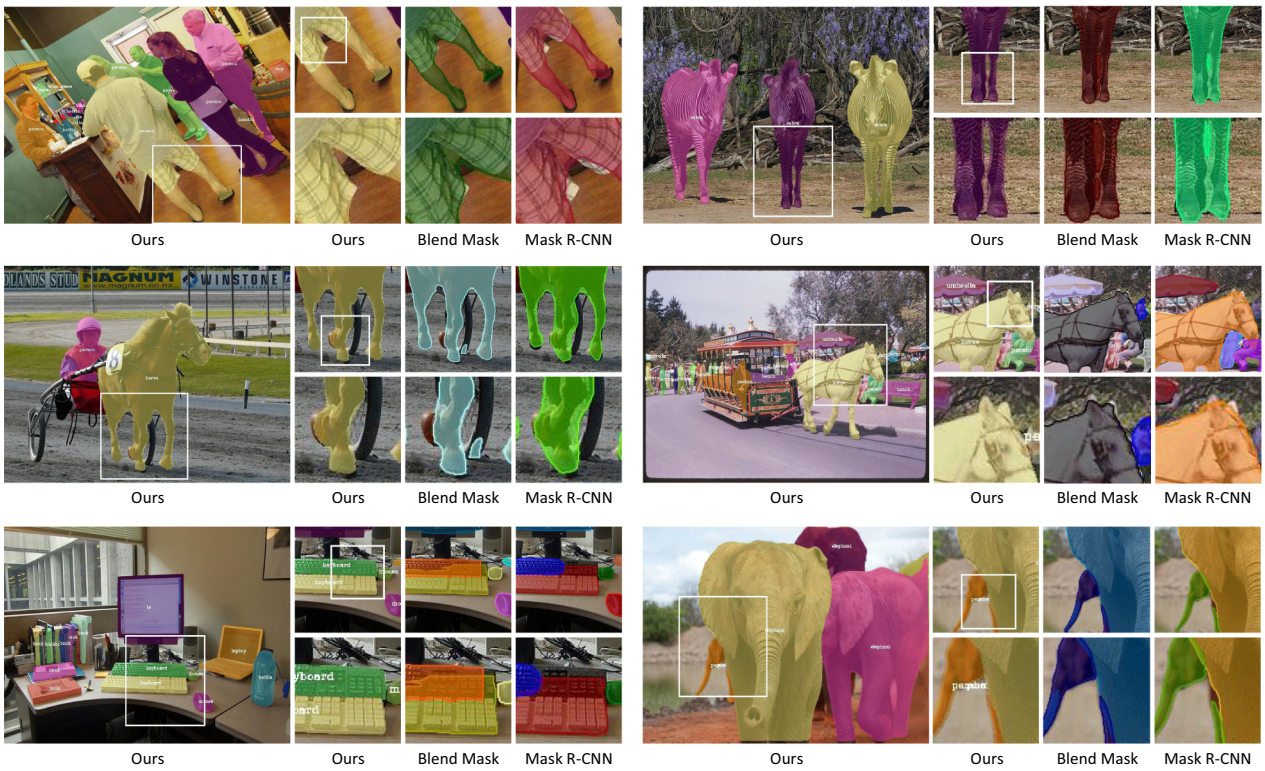

Fig. 6 The large image on the left shows the segmentation results of our method. We further zoom in on our results and compare them with the Blend Mask [3] and Mask R-CNN [14] on the right. The results prove that our masks are of higher quality at the instance edges. Best viewed in digital format with zoom

presents a great advantage for the segmentation of large-scale objects, reaching 58.8 AP. Figure 6 shows part of the visualization results of EEMask on the COCO test-dev. It can be seen that the classes and masks have achieved pretty accurate predictions.

5 Conclusions

In this paper, we propose a new instance segmentation model EEMask based on grid ROIs and edge-enhanced, which can quickly and accurately predict the instance classes and segment the edges of the instances precisely. EEMask is very efficient without adding additional convolutional layers, and it is also easy to be ported to other models. And it is so robust that even if the image is distorted, rotated, cropped and scaled, EEMask is still effective. With the evaluation of benchmark metrics, EEMask demonstrates promising results, achieving a mask AP score of 39.9. Finally, we believe that EEMask will contribute to advancing the field of instance segmentation.

Declarations

Conflict of interest The authors certify that there is no conflict of interest with any individual/organization for the present work.

References

1. Bai, M., Urtasun, R.: Deep watershed transform for instance segmentation. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2858–2866. https://doi.org/10.1109/CVPR.2017.305 (2017)
2. Bolya, D., Zhou, C., Xiao, F., Lee, Y.J.: YOLACT: real-time instance segmentation. CoRR abs/1904.02689, arXiv:1904.02689, (2019)
3. Chen, H., Sun, K., Tian, Z., Shen, C., Huang, Y., Yan, Y.: Blendmask: Top-down meets bottom-up for instance segmentation. CoRR abs/2001.00309, arXiv:2001.00309, (2020)
4. Chen, X., Girshick, R., He, K., Dollar, P.: Tensormask: A foundation for dense object segmentation. pp 2061–2069. https://doi.org/10.1109/ICCV.2019.00215 (2019)
5. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol 1, pp 886–893 vol. 1. https://doi.org/10.1109/CVPR.2005.177 (2005)
6. Everingham, M., van Gool, L., Williams, C., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. Int. J. Comput. Vision **88**(2), 303–338 (2010). https://doi.org/10.1007/s11263-009-0275-4
7. Felzenszwalb, P., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: 2008 IEEE Conference on Computer Vision and Pattern Recognition, pp 1–8. https://doi.org/10.1109/CVPR.2008.4587597 (2008)
8. Felzenszwalb, P.F., Girshick, R.B., McAllester, D.: Cascade object detection with deformable part models. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recogni-

tion, pp 2241–2248, https://doi.org/10.1109/CVPR.2010.5539906 (2010)
9. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Trans. Pattern Anal. Mach. Intell. **32**(9), 1627–1645 (2010). https://doi.org/10.1109/TPAMI.2009.167
10. Felzenszwalb, P.F., Girshick, R.B., McAllester, D.A., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Trans. Pattern Anal. Mach. Intell. **32**(9), 1627–1645 (2010). https://doi.org/10.1109/TPAMI.2009.167
11. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp 580–587, https://doi.org/10.1109/CVPR.2014.81 (2014)
12. Girshick, R.B.: Fast R-CNN. CoRR abs/1504.08083, arXiv:1504.08083, (2015)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 770–778. https://doi.org/10.1109/CVPR.2016.90 (2016)
14. He, K., Gkioxari, G., Dollár, P., Girshick, R.B.: Mask R-CNN. CoRR abs/1703.06870, arXiv:1703.06870, (2017)
15. Huang, R., Pedoeem, J., Chen, C.: Yolo-lite: A real-time object detection algorithm optimized for non-gpu computers. In: 2018 IEEE International Conference on Big Data (Big Data), pp 2503–2510, https://doi.org/10.1109/BigData.2018.8621865 (2018)
16. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, Curran Associates Inc., Red Hook, NY, USA, NIPS'12, p 1097-1105 (2012)
17. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998). https://doi.org/10.1109/5.726791
18. Leng, L., Zhang, J.: Palmhash code vs. palmphasor code. Neurocomputing **108**, 1–12 (2013). https://doi.org/10.1016/j.neucom.2012.08.028
19. Leng, L., Li, M., Kim, C., Bi, X.: Dual-source discrimination power analysis for multi-instance contactless palmprint recognition. Multim. Tools Appl. **76**(1), 333–354 (2017). https://doi.org/10.1007/s11042-015-3058-7
20. Li, J., Zhao, X., Li, H.: Method for detecting road pavement damage based on deep learning. (2019). https://doi.org/10.1117/12.2514437
21. Li, Y., Qi, H., Dai, J., Ji, X., Wei, Y.: Fully convolutional instance-aware semantic segmentation. CoRR abs/1611.07709, arXiv:1611.07709, (2016)
22. Lin, T., Maire, M., Belongie, S.J., Bourdev, L.D., Girshick, R.B., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: common objects in context. CoRR abs/1405.0312, arXiv:1405.0312, (2014)
23. Lin, T., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection. CoRR abs/1612.03144, arXiv:1612.03144, (2016)
24. Lin, T., Goyal, P., Girshick, R.B., He, K., Dollár, P.: Focal loss for dense object detection. CoRR abs/1708.02002, arXiv:1708.02002, (2017)
25. Liu, C., Yuen, J., Torralba, A.: Nonparametric scene parsing via label transfer. IEEE Trans. Pattern Anal. Mach. Intell. **33**(12), 2368–2382 (2011). https://doi.org/10.1109/TPAMI.2011.131
26. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S.E., Fu, C., Berg, A.C.: SSD: single shot multibox detector. CoRR abs/1512.02325, arXiv:1512.02325, (2015)
27. Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., Alsaadi, F.E.: A survey of deep neural network architectures and their applications. Neurocomputing **234**(234), 11–26 (2017)
28. Qi, L., Zhang, X., Chen, Y., Chen, Y., Sun, J., Jia, J.: Pointins: Point-based instance segmentation. CoRR abs/2003.06148, arXiv:2003.06148, (2020)
29. Ren, S., He, K., Girshick, R.B., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. CoRR abs/1506.01497, arXiv:1506.01497, (2015)
30. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017, IEEE Computer Society, pp 618–626, https://doi.org/10.1109/ICCV.2017.74, (2017)
31. Tian, Z., Shen, C., Chen, H., He, T.: FCOS: fully convolutional one-stage object detection. In: 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019, IEEE, pp 9626–9635. https://doi.org/10.1109/ICCV.2019.00972, (2019)
32. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, vol 1, pp I–I, https://doi.org/10.1109/CVPR.2001.990517 (2001)
33. Viola, P., Jones, M.J.: Robust real-time face detection. Int. J. Comput. Vision **57**(2), 137–154 (2004). https://doi.org/10.1023/B:VISI.0000013087.49260.fb
34. Xie, E., Sun, P., Song, X., Wang, W., Liu, X., Liang, D., Shen, C., Luo, P.: Polarmask: Single shot instance segmentation with polar representation. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp 12190–12199. https://doi.org/10.1109/CVPR42600.2020.01221 (2020)
35. Xu, W., Wang, J., Wang, Y., Xu, G., Lin, D., Dai, W., Wu, Y.: Where is the model looking at?—concentrate and explain the network attention. IEEE J. Sel. Top Signal Process **14**(3), 506–516 (2020). https://doi.org/10.1109/JSTSP.2020.2987729
36. Xu, W., Xian, Y., Wang, J., Schiele, B., Akata, Z.: Attribute prototype network for zero-shot learning. CoRR abs/2008.08290, arXiv:2008.08290, (2020)
37. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. arXiv:1311.2901, cite arxiv:1311.2901 (2013)
38. Zhang, F., Li, M., Zhai, G., Liu, Y.: Multi-branch and multi-scale attention learning for fine-grained visual categorization. In: Lokoc J, Skopal T, Schoeffmann K, Mezaris V, Li X, Vrochidis S, Patras I (eds) MultiMedia Modeling - 27th International Conference, MMM 2021, Prague, Czech Republic, June 22-24, 2021, Proceedings, Part I, Springer, Lecture Notes in Computer Science, vol 12572, pp 136–147. https://doi.org/10.1007/978-3-030-67832-6_12, (2021)
39. Zhang, Y., Chu, J., Leng, L., Miao, J.: Mask-refined R-CNN: a network for refining object details in instance segmentation. Sensors **20**(4), 1010 (2020). https://doi.org/10.3390/s20041010
40. Zhao, D., Qi, Z., Yang, R., Wang, Z.: Attention-based dual context aggregation for image semantic segmentation. Multim Tools Appl **80**(18), 28201–28216 (2021). https://doi.org/10.1007/s11042-021-11094-6
41. Zhou, B., Khosla, A., Lapedriza, À., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, IEEE Computer Society, pp 2921–2929, https://doi.org/10.1109/CVPR.2016.319, (2016)

Ying Gao Master degree candidate, researching in Tianjin Key Laboratory of Intelligence Computing and Novel Software Technology from Tianjin University of Technology. His research interest includes computer vision, deep learning and instance segmentation.

Zhiyang Qi Master degree candidate, researching in Tianjin Key Laboratory of Intelligence Computing and Novel Software Technology from Tianjin University of Technology. His research interest includes computer vision, deep learning and instance segmentation.

Dexin Zhao is currently a professor at the school of computer science and engineering of Tianjin University of Technology, China. She received her Ph.D. in computer science at Tianjin University in 2008. She worked as a visiting scholar at the State University of New York at Buffalo during 2012 to 2013. Her research interests include multi-media content retrieval, knowledge representation and medical image processing.