Segmentation Transformer: Object-Contextual Representation for Semantic Segmentation

——from arxiv2021 By CAS

Speaker: Gong, Qiqi

Overview

- Context aggregation
- Three steps:
  - Learn object regions under the supervision of GT segmentation (a coarse soft segmentation computed from a deep network)
  - Estimate the representation for each object region (object region representation)
  - Augment the erpresentation of each pixel with **object-contextual representation**
- OCR is the weighted aggregation of all the object region representations with the weights calculated according to the relations between pixels and object regions

Motivation

- The label of a pixel is the bategory of the object that pixel belongs to

(a) ASPP (b) OCR

Fig. 2: **Illustrating the multi-scale context with the ASPP as an example and the OCR context for the pixel marked with** . (a) ASPP: The context is a set of sparsely sampled pixels marked with , . The pixels with different colors correspond to different dilation rates. Those pixels are distributed in both the object region and the background region. (b) Our OCR: The context is expected to be a set of pixels lying in the object (marked with color blue). The image is chosen from ADE20K.

Method - Formulation

- Overview
  - Structurizes all the pixels in image into *K* soft object regions
  - Represent each object region by aggregrating all the pixels in the *k*th object region
  - Augments the representation for each pixel by aggregating the *K* object region representations (consider all K regions)

$$\mathbf{y}_i = \rho(\sum_{k=1}^{K} w_{ik}\delta(\mathbf{f}_k)), \tag{3}$$

where $\mathbf{f}_k$ is the representation of the kth object region, w_{ik} is the relation between the ith pixel and the kth object region. $\delta(\cdot)$ and $\rho(\cdot)$ are transformation functions.

Method - Formulation

- Soft object regions (pink box)
  - Compute K object regions from an intermediate representation output from a backbone (ResNet or HRNet)
  - CE Loss
  - Each entry indicates the degree that corresponding pixel belongs to class k

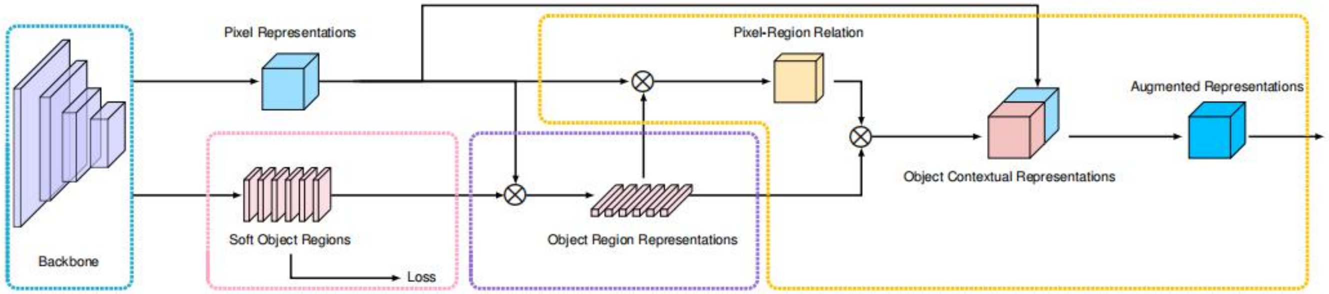


Fig. 3: **Illustrating the pipeline of OCR.** (i) form the soft object regions in the *pink dashed box*. (ii) estimate the object region representations in the *purple dashed box*; (iii) compute the object contextual representations and the augmented representations in the *orange dashed box*. See Section 3.2 and 3.3 for more details.

Method - Formulation

- Object region representations (purple box)

$$\mathbf{f}_k = \sum_{i \in \mathcal{I}} \tilde{m}_{ki} \mathbf{x}_i. \tag{4}$$

Here, $\mathbf{x}_i$ is the representation of pixel p_i. $\tilde{m}_{ki}$ is the normalized degree for pixel p_i belonging to the kth object region. We use spatial softmax to normalize each object region $\mathbf{M}_k$.

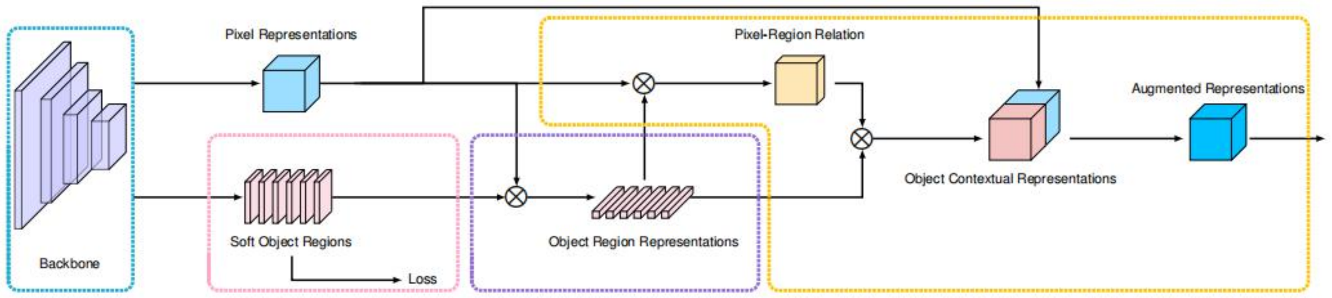


Fig. 3: **Illustrating the pipeline of OCR.** (i) form the soft object regions in the *pink dashed box*. (ii) estimate the object region representations in the *purple dashed box*; (iii) compute the object contextual representations and the augmented representations in the *orange dashed box*. See Section 3.2 and 3.3 for more details.

Method - Formulation

- Object contextual representations (orange box)
  - Compute relation between each pixel and each object region (yellow cube)

$$w_{ik} = \frac{e^{\kappa(\mathbf{x}_i, \mathbf{f}_k)}}{\sum_{j=1}^{K} e^{\kappa(\mathbf{x}_i, \mathbf{f}_j)}}. \quad (5)$$

Here, $\kappa(\mathbf{x}, \mathbf{f}) = \phi(\mathbf{x})^\top \psi(\mathbf{f})$ is the unnormalized relation function, $\phi(\cdot)$ and $\psi(\cdot)$ are two transformation functions implemented by 1×1 conv $\rightarrow$ BN $\rightarrow$ ReLU. This is inspired by self-attention [61] for a better relation estimation.

  - Compute OCR according to Eq (3) (in slide 4)

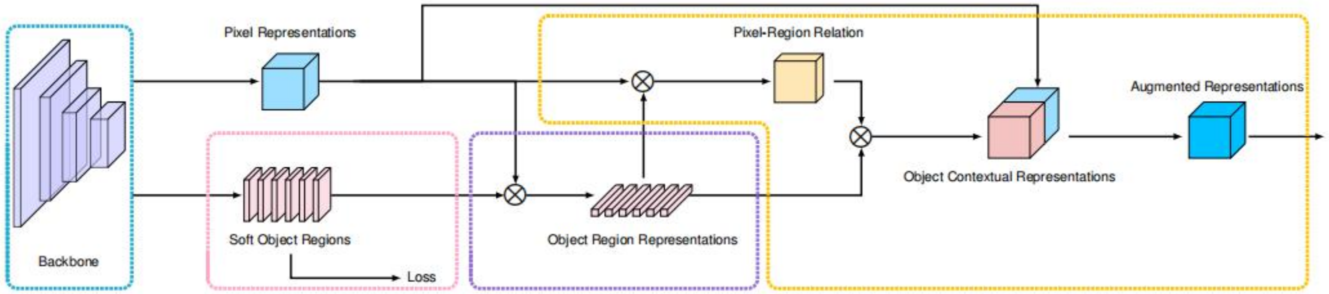


Fig. 3: **Illustrating the pipeline of OCR.** (i) form the soft object regions in the *pink dashed box*. (ii) estimate the object region representations in the *purple dashed box*; (iii) compute the object contextual representations and the augmented representations in the *orange dashed box*. See Section 3.2 and 3.3 for more details.

Method - Formulation

- Augmented Representation (red box)

$$\mathbf{z}_i = g([\mathbf{x}_i^\top \ \mathbf{y}_i^\top]^\top). \tag{6}$$

where $g(\cdot)$ is a transform function used to fuse the original representation and the object contextual representation, implemented by 1×1 conv $\rightarrow$ BN $\rightarrow$ ReLU.

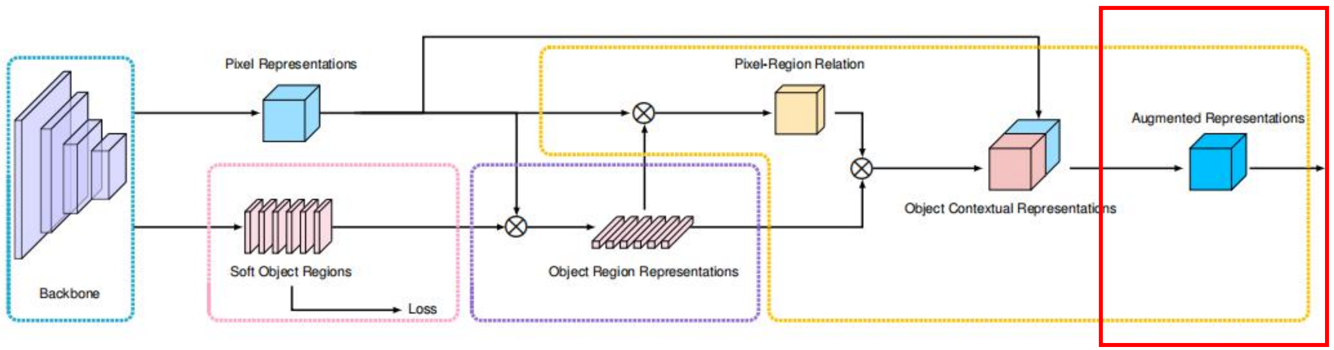
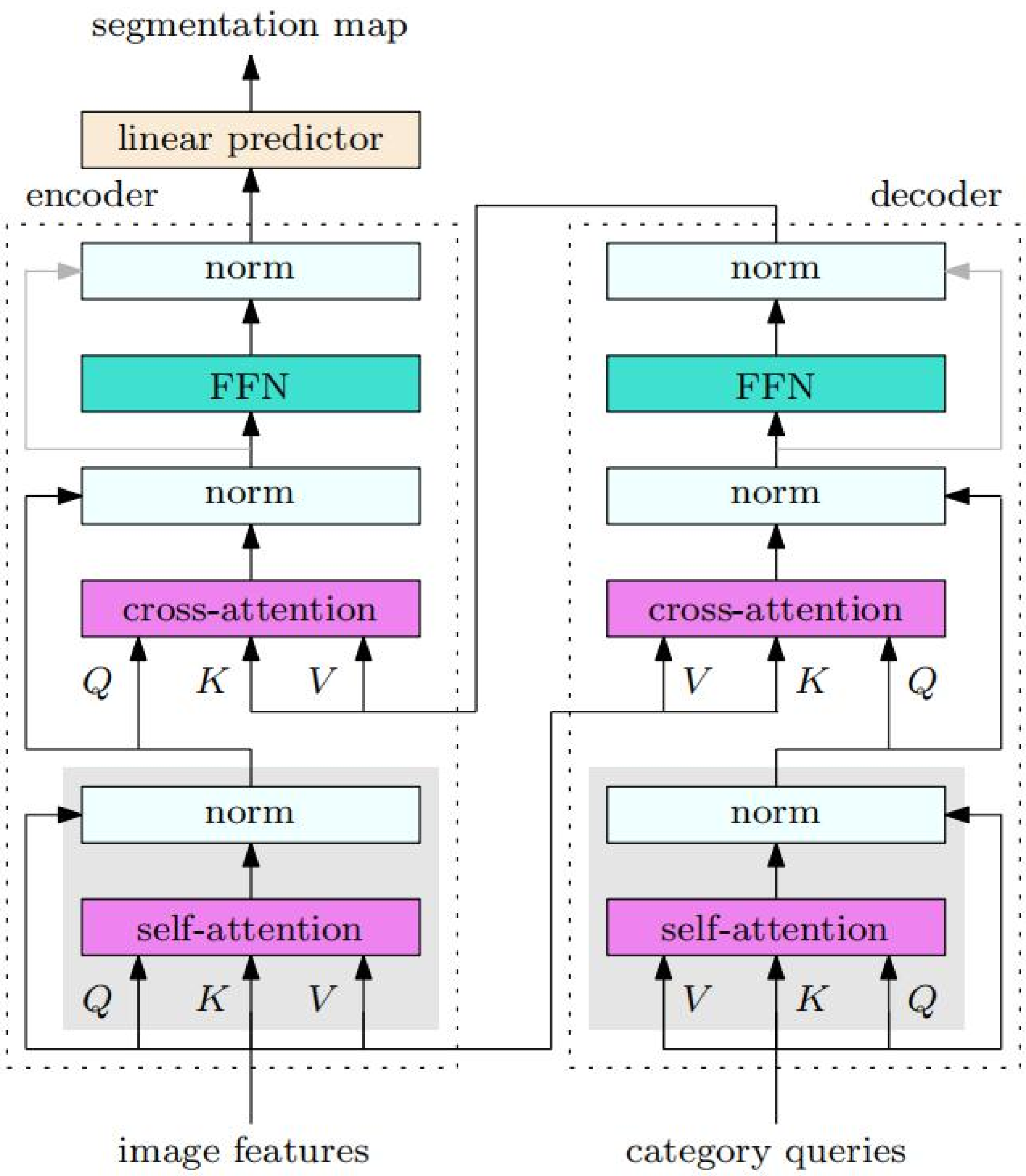


Fig. 3: **Illustrating the pipeline of OCR.** (i) form the soft object regions in the *pink dashed box*. (ii) estimate the object region representations in the *purple dashed box*; (iii) compute the object contextual representations and the augmented representations in the *orange dashed box*. See Section 3.2 and 3.3 for more details.

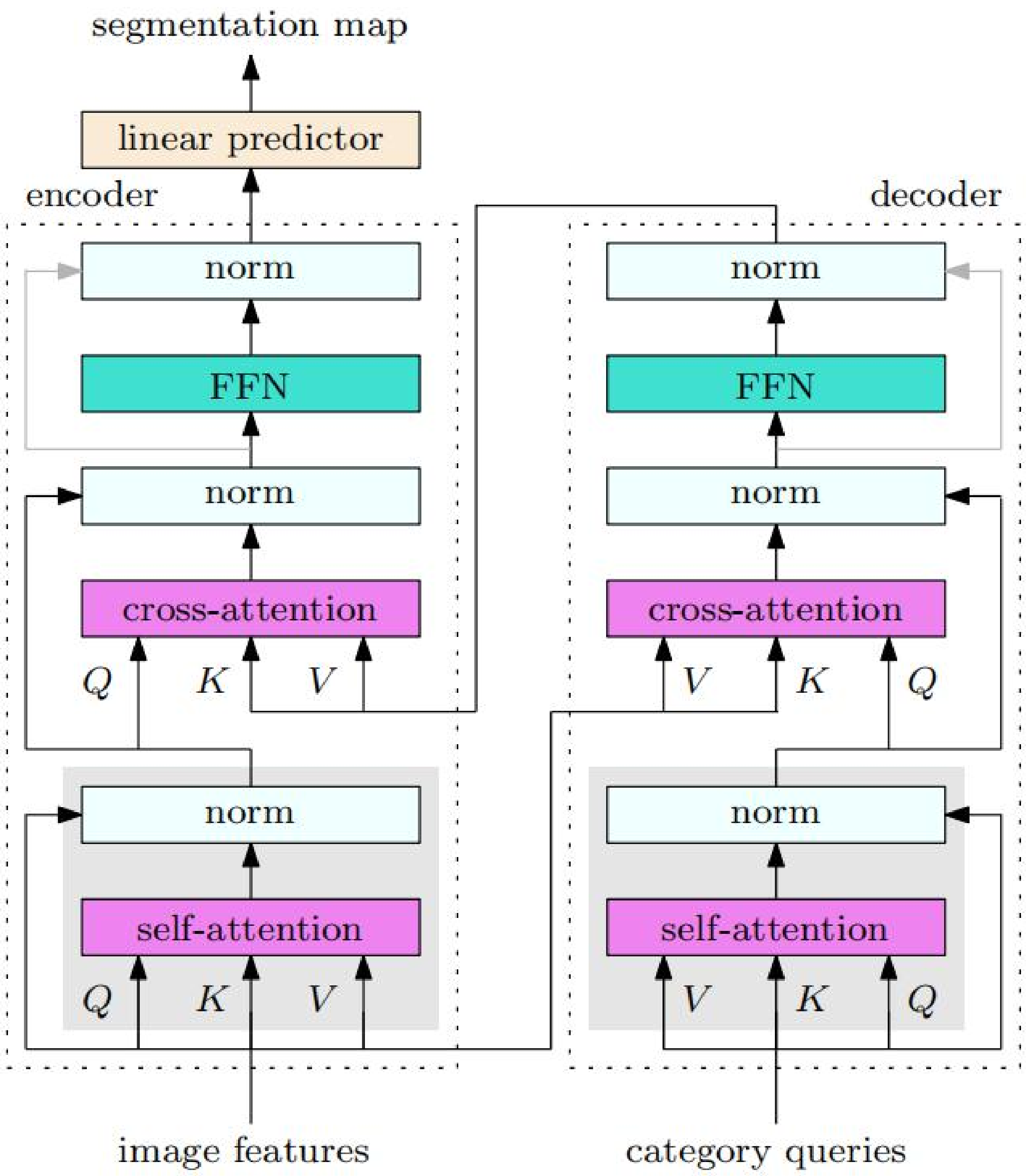
Method - Use Transformer

- Segmentation Transformer
  - Decoder:
    - Extract soft object region
    - Compute object region represent.
    - Keys and values: image features
    - Quries: *K* category quries



Method - Use Transformer

- Segmentation Transformer
    - Encoder:
        - Aggregating object region representations



Experiments

- Cityscapes, ADE20K, LIP, PASCAL-Context, COCO-Stuff
- HRNet + OCR won 1st on Cityscapes before DDL of ECCV2020

Table 1: **Complexity comparison.** We use input feature map of size $[1 \times 2048 \times 128 \times 128]$ to evaluate their complexity during inference. The numbers are obtained on a single P40 GPU with CUDA 10.0. All the numbers are the smaller the better. Our OCR requires the least GPU memory and the least runtime.

Method	Parameters▲	Memory▲	FLOPs ▲	Time▲
PPM	23.1M	792M	619G	99ms
ASPP	15.5M	284M	492G	97ms
OCR	10.5M	**202**M	340G	**45**ms

Experiments

- Comparison with SOTA
  - Table 5 in paper
  - Win first 3 places on all datasets

Method	Baseline	Stride	Context schemes	Cityscapes (w/o coarse)	Cityscapes (w/ coarse)	ADE20K	LIP	PASCAL Context	COCO-Stuff
Simple baselines									
PSPNet [80]	ResNet-101	8×	M	$78.4^{\flat}$	81.2	43.29	-	47.8	-
DeepLabv3 [6]	ResNet-101	8×	M	-	81.3	-	-	-	-
PSANet [81]	ResNet-101	8×	R	80.1	81.4	43.77	-	-	-
SAC [79]	ResNet-101	8×	M	78.1	-	44.30	-	-	-
AAF [29]	ResNet-101	8×	R	$79.1^{\flat}$	-	-	-	-	-
DSSPN [41]	ResNet-101	8×	-	77.8	-	43.68	-	-	38.9
DepthSeg [32]	ResNet-101	8×	-	78.2	-	-	-	-	-
MMAN [48]	ResNet-101	8×	-	-	-	-	46.81	-	-
JPPNet [39]	ResNet-101	8×	M	-	-	-	51.37	-	-
EncNet [76]	ResNet-101	8×	-	-	-	44.65	-	51.7	-
GCU [38]	ResNet-101	8×	R	-	-	44.81	-	-	-
APCNet [24]	ResNet-101	8×	M,R	-	-	45.38	-	54.7	-
CFNet [77]	ResNet-101	8×	R	79.6	-	44.89	-	54.0	-
BFP [12]	ResNet-101	8×	R	81.4	-	-	-	53.6	-
CCNet [27]	ResNet-101	8×	R	81.4	-	45.22	-	-	-
ANNet [84]	ResNet-101	8×	M,R	81.3	-	45.24	-	52.8	-
OCR (Seg. transformer)	ResNet-101	8×	R	81.8	82.4	45.28	55.60	54.8	39.5
Advanced baselines									
DenseASPP [68]	DenseNet-161	8×	M	80.6	-	-	-	-	-
DANet [18]	ResNet-101 + MG	8×	R	81.5	-	45.22	-	52.6	39.7
DGCNet [78]	ResNet-101 + MG	8×	R	82.0	-	-	-	53.7	-
EMANet [36]	ResNet-101 + MG	8×	R	-	-	-	-	53.1	39.9
SeENet [51]	ResNet-101 + ASPP	8×	M	81.2	-	-	-	-	-
SGR [40]	ResNet-101 + ASPP	8×	R	-	-	44.32	-	52.5	39.1
OCNet [72]	ResNet-101 + ASPP	8×	M,R	81.7	-	45.45	54.72	-	-
ACFNet [75]	ResNet-101 + ASPP	8×	M,R	81.8	-	-	-	-	-
CNIF [63]	ResNet-101 + ASPP	8×	M	-	-	-	56.93	-	
GALD [37]	ResNet-101 + ASPP	8×	M,R	81.8	82.9	-	-	-	-
GALD† [37]	ResNet-101 + CGNL + MG	8×	M,R	-	83.3	-	-	-	-
Mapillary [52]	WideResNet-38 + ASPP	8×	M	-	82.0	-	-	-	-
GSCNN† [55]	WideResNet-38 + ASPP	8×	M	82.8	-	-	-	-	
SPGNet [10]	2× ResNet-50	4×	-	81.1	-	-	-	-	-
ZigZagNet [42]	ResNet-101	4×	M	-	-	-	-	52.1	-
SVCNet [13]	ResNet-101	4×	R	81.0	-	-	-	53.2	39.6
ACNet [19]	ResNet-101 + MG	4×	M,R	82.3	-	45.90	-	54.1	40.1
CE2P [45]	ResNet-101 + PPM	4×	M	-	-	-	53.10	-	
VPLR†‡ [83]	WideResNet-38 + ASPP	4×	M	-	83.5	-	-	-	
DeepLabv3+ [7]	Xception-71	4×	M	-	82.1	-	-	-	-
DPC [4]	Xception-71	4×	M	82.7	-	-	-	-	
DUpsampling [57]	Xception-71	4×	M	-	-	-	-	52.5	-
HRNet [54]	HRNetV2-W48	4×	-	81.6	-	-	55.90	54.0	
OCR (Seg. transformer)	HRNetV2-W48	4×	R	82.4	83.0	45.66	56.65	56.2	40.5
OCR† (Seg. transformer)	HRNetV2-W48	4×	R	83.6	84.2	-	-	-	-