k-means Mask Transformer

Qihang Yu[1(✉)], Huiyu Wang[1], Siyuan Qiao[2], Maxwell Collins[2], Yukun Zhu[2],
Hartwig Adam[2], Alan Yuille[1], and Liang-Chieh Chen[2]

[1] Johns Hopkins University, Baltimore, USA
`yucornetto@gmail.com`
[2] Google Research, Mountain View, USA

Abstract. The rise of transformers in vision tasks not only advances
network backbone designs, but also starts a brand-new page to achieve
end-to-end image recognition (*e.g.*, object detection and panoptic seg-
mentation). Originated from Natural Language Processing (NLP), trans-
former architectures, consisting of self-attention and cross-attention,
effectively learn long-range interactions between elements in a sequence.
However, we observe that most existing transformer-based vision mod-
els simply borrow the idea from NLP, neglecting the crucial difference
between languages and images, particularly the extremely large sequence
length of spatially flattened pixel features. This subsequently impedes the
learning in cross-attention between pixel features and object queries. In
this paper, we rethink the relationship between pixels and object queries,
and propose to reformulate the cross-attention learning as a clustering
process. Inspired by the traditional k-means clustering algorithm, we
develop a **k**-means **Ma**sk **X**former (kMaX-DeepLab) for segmentation
tasks, which not only improves the state-of-the-art, but also enjoys a
simple and elegant design. As a result, our kMaX-DeepLab achieves a
new state-of-the-art performance on COCO *val* set with 58.0% PQ, and
Cityscapes *val* set with 68.4% PQ, 44.0% AP, and 83.5% mIoU without
test-time augmentation or external dataset. We hope our work can shed
some light on designing transformers tailored for vision tasks. Code and
models are available at https://github.com/google-research/deeplab2.

Keywords: Segmentation · Transformer · k-means Clustering

1 Introduction

Transformers [89] are receiving a growing attention in the computer vision com-
munity. On the one hand, the transformer encoder, with multi-head self-attention
as the central component, demonstrates a great potential for building powerful

---

Work done during an internship at Google.

---

network architectures in various visual recognition tasks [32,70,93]. On the other hand, the transformer decoder, with multi-head cross-attention at its core, provides a brand-new approach to tackling complex visual recognition problems in an end-to-end manner, dispensing with hand-designed heuristics.

Recently, the pioneering work DETR [10] introduces the first end-to-end object detection system with transformers. In this framework, the pixel features are firstly extracted by a convolutional neural network [58], followed by the deployment of several transformer encoders for feature enhancement to capture long-range interactions between pixels. Afterwards, a set of learnable positional embeddings, named object queries, is responsible for interacting with pixel features and aggregating information through several interleaved cross-attention and self-attention modules. In the end, the object queries, decoded by a Feed-Forward Network (FFN), directly correspond to the final bounding box predictions. Along the same direction, MaX-DeepLab [92] proves the success of transformers in the challenging panoptic segmentation task [55], where the prior arts [21,54,100] usually adopt complicated pipelines involving hand-designed heuristics. The essence of this framework lies in converting the object queries to mask embedding vectors [49,87,97], which are employed to yield a set of mask predictions by multiplying with the pixel features.

The end-to-end transformer-based frameworks have been successfully applied to multiple computer vision tasks with the help of transformer decoders, especially the cross-attention modules. However, the working mechanism behind the scenes remains unclear. The cross-attention, which arises from the Natural Language Processing (NLP) community, is originally designed for language problems, such as neural machine translation [4,86], where both the input sequence and output sequence share a similar short length. This implicit assumption becomes problematic when it comes to certain vision problems, where the cross-attention is performed between object queries and spatially flattened pixel features with an exorbitantly large length. Concretely, usually a small number of object queries is employed (*e.g.*, 128 queries), while the input images can contain thousands of pixels for the vision tasks of detection and segmentation. Each object query needs to learn to highlight the most distinguishable features among the abundant pixels in the cross-attention learning process, which subsequently leads to slow training convergence and thus inferior performance [37,112].

In this work, we make a crucial observation that the cross-attention scheme actually bears a strong similarity to the traditional *k*-means clustering [72] by regarding the object queries as cluster centers with learnable embedding vectors. Our examination of the similarity inspires us to propose the novel **k**-means **Ma**sk **X**former (*k*MaX-DeepLab), which rethinks the relationship between pixel features and object queries, and redesigns the cross-attention from the perspective of *k*-means clustering. Specifically, when updating the cluster centers (*i.e.*, object queries), our *k*MaX-DeepLab performs a different operation. Instead of performing *softmax* on the large spatial dimension (image height times width) as in the original Mask Transformer's cross-attention [92], our *k*MaX-DeepLab performs *argmax* along the cluster center dimension, similar to the *k*-means pixel-cluster

assignment step (with a *hard* assignment). We then update cluster centers by aggregating the pixel features based on the pixel-cluster assignment (computed by their feature affinity), similar to the k-means center-update step. In spite of being conceptually simple, the modification has a striking impact: on COCO *val* set [66], using the standard ResNet-50 [41] as backbone, our kMaX-DeepLab demonstrates a significant improvement of **5.2%** PQ over the original cross-attention scheme at a negligible cost of extra parameters and FLOPs. When comparing to state-of-the-art methods, our kMaX-DeepLab with the simple ResNet-50 backbone already outperforms MaX-DeepLab [92] with MaX-L [92] backbone by **1.9%** PQ, while requiring **7.9** and **22.0** times fewer parameters and FLOPs, respectively. Our kMaX-DeepLab with ResNet-50 also outperforms MaskFormer [24] with the strong ImageNet-22K pretrained Swin-L [70] backbone, and runs **4.4** times faster. Finally, our kMaX-DeepLab, using the modern ConvNeXt-L [71] as backbone, sets a new state-of-the-art performance on the COCO *val* set [66] with 58.0% PQ. It also outperforms other state-of-the-art methods on the Cityscapes *val* set [28], achieving 68.4% PQ, 83.5% mIoU, 44.0% AP, without using any test-time augmentation or extra dataset pretraining [66,75].

2   Related Works

Transformers.   Transformer [89] and its variants [2,8,26,39,57,74,94,106] have advanced the state-of-the-art in natural language processing tasks [30,31, 82] by capturing relations across modalities [4] or in a single context [25,89]. In computer vision, transformer encoders or self-attention modules are either combined with Convolutional Neural Networks (CNNs) [9,96] or used as standalone backbones [32,44,70,80,93]. Both approaches have boosted various vision tasks, such as image classification [7,19,32,44,64,70,80,93,101,105], image generation [42,77], object detection [10,43,80,83,96,112], video recognition [3,19,33,96], semantic segmentation [11,17,35,46,99,108,109,111,113], and panoptic segmentation [93].

Mask Transformers for Segmentation.   Besides the usage as backbones, transformers are also adopted as task decoders for image segmentation. MaX-DeepLab [92] proposed **Ma**sk **X**formers (MaX) for end-to-end panoptic segmentation. Mask transformers predict class-labeled object masks and are trained by Hungarian matching the predicted masks with ground truth masks. The essential component of mask transformers is the conversion of object queries to mask embedding vectors [49,87,97], which are employed to generate predicted masks. Both Segmenter [85] and MaskFormer [24] applied mask transformers to semantic segmentation. K-Net [107] proposed dynamic kernels for generating the masks. CMT-DeepLab [104] proposed to improve the cross-attention with an additional clustering update term. Panoptic Segformer [65] strengthened mask transformer with deformable attention [112], while Mask2Former [23] further boosted the performance with masked cross-attention along with a series of technical improvements including cascaded transformer decoder, deformable

attention [112], uncertainty-based pointly supervision [56], *etc.*. These mask transformer methods generally outperform box-based methods [54] that decompose panoptic segmentation into multiple surrogate tasks (*e.g.*, predicting masks for each detected object bounding box [40], followed by fusing the instance segments ('thing') and semantic segments ('stuff') [14] with merging modules [60,62,67,78,100,103]). Moreover, mask transformers showed great success in the video segmentation problems [20,52,61].

Clustering Methods for Segmentation. Traditional image segmentation methods [1,72,110] typically cluster image intensities into a set of masks or superpixels with gradual growing or refinement. However, it is challenging for these traditional methods to capture high-level semantics. Modern clustering-based methods usually operate on semantic segments [13,15,18] and group 'thing' pixels into instance segments with various representations, such as instance center regression [22,50,63,76,88,93,102], Watershed transform [5,90], Hough-voting [6,59,91], or pixel affinity [36,47,51,69,84].

Recently, CMT-DeepLab [104] discussed the similarity between mask transformers and clustering algorithms. However, they only used the clustering update as a complementary term in the cross-attention. In this work, we further discover the underlying similarity between mask transformers and the *k*-means clustering algorithm, resulting in a simple yet effective *k*-means mask transformer.

3   Method

In this section, we first overview the mask-transformer-based segmentation framework presented by MaX-DeepLab [92]. We then revisit the transformer cross-attention [89] and the *k*-means clustering algorithm [72], and reveal their underlying similarity. Afterwards, we introduce the proposed **k**-means **Ma**sk **X**former (*k*MaX-DeepLab), which redesigns the cross-attention from a clustering perspective. Even though simple, *k*MaX-DeepLab effectively and significantly improves the segmentation performance.

3.1   Mask-Transformer-Based Segmentation Framework

Transformers [89] have been effectively deployed to segmentation tasks. Without loss of generality, we consider panoptic segmentation [55] in the following problem formulation, which can be easily generalized to other segmentation tasks.

Problem Statement. Panoptic segmentation aims to segment the image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$ into a set of non-overlapping masks with associated semantic labels:

$$\{y_i\}_{i=1}^{K} = \{(m_i, c_i)\}_{i=1}^{K}. \tag{1}$$

The K ground truth masks $m_i \in \{0,1\}^{H \times W}$ do not overlap with each other, *i.e.*, $\sum_{i=1}^{K} m_i \leq 1^{H \times W}$, and c_i denotes the ground truth class label of mask m_i.

Starting from DETR [10] and MaX-DeepLab [92], approaches to panoptic segmentation shift to a new end-to-end paradigm, where the prediction directly

matches the format of ground-truth with N masks (N is a fixed number and $N \geq K$) and their semantic classes:

$$\{\hat{y}_i\}_{i=1}^{N} = \{(\hat{m}_i, \hat{p}_i(c))\}_{i=1}^{N}, \tag{2}$$

where $\hat{p}_i(c)$ denotes the semantic class prediction confidence for the corresponding mask, which includes 'thing' classes, 'stuff' classes, and the void class $\varnothing$.

The N masks are predicted based on the N object queries, which aggregate information from the pixel features through a transformer decoder, consisting of self-attention and cross-attention modules.

The object queries, updated by multiple transformer decoders, are employed as mask embedding vectors [49,87,97], which will multiply with the pixel features to yield the final prediction $\mathbf{Z} \in \mathbb{R}^{HW \times N}$ that consists of N masks. That is,

$$\mathbf{Z} = \operatorname*{softmax}_{N}(\mathbf{F} \times \mathbf{C}^{\mathrm{T}}), \tag{3}$$

where $\mathbf{F} \in \mathbb{R}^{HW \times D}$ and $\mathbf{C} \in \mathbb{R}^{N \times D}$ refers to the final pixel features and object queries, respectively. D is the channel dimension of pixel features and object queries. We use underscript N to indicate the axis to perform softmax.

3.2 Relationship Between Cross-Attention and k-means Clustering

Although the transformer-based segmentation frameworks successfully connect object queries and mask predictions in an end-to-end manner, the essential problem becomes how to transform the object queries, starting from learnable embeddings (randomly initialized), into meaningful mask embedding vectors.

Cross-Attention. The cross-attention modules are used to aggregate affiliated pixel features to update object queries. Formally, we have

$$\hat{\mathbf{C}} = \mathbf{C} + \operatorname*{softmax}_{HW}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p, \tag{4}$$

where $\mathbf{C} \in \mathbb{R}^{N \times D}$ refers to N object queries with D channels, and $\hat{\mathbf{C}}$ denotes the updated object queries. We use the underscript HW to represent the axis for softmax on spatial dimension, and superscripts p and c to indicate the feature projected from the pixel features and object queries, respectively. $\mathbf{Q}^c \in \mathbb{R}^{N \times D}, \mathbf{K}^p \in \mathbb{R}^{HW \times D}, \mathbf{V}^p \in \mathbb{R}^{HW \times D}$ stand for the linearly projected features for query, key, and value. For simplicity, we ignore the multi-head mechanism and feed-forward network (FFN) in the equation.

As shown in Eq. (4), when updating the object queries, a *softmax* function is applied to the image resolution (HW), which is typically in the range of thousands of pixels for the task of segmentation. Given the huge number of pixels, it can take many training iterations to learn the attention map, which starts from a uniform distribution at the beginning (as the queries are randomly initialized). Each object query has a difficult time to identify the most distinguishable features among the abundant pixels in the early stage of training. This behavior is

very different from the application of transformers to natural language processing tasks, *e.g.*, neural machine translation [4,86], where the input and output sequences share a similar short length. Vision tasks, especially segmentation problems, present another challenge for efficiently learning the cross-attention.

Discussion.   Similar to cross-attention, self-attention needs to perform a *softmax* function operated along the image resolution. Therefore, learning the attention map for self-attention may also take many training iterations. An efficient alternative, such as axial attention [93] or local attention [70] is usually applied on high resolution feature maps, and thus alleviates the problem, while a solution to cross-attention remains an open question for research.

*k***-Means Clustering.**   In Eq. (4), the cross-attention computes the affinity between object queries and pixels (*i.e.*, $\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}$), which is converted to the attention map through the spatial-wise softmax (operated along the image resolution). The attention map is then used to retrieve (and weight accordingly) affiliated pixel features to update the object queries. Surprisingly, we observe that the whole process is actually similar to the classic *k*-means clustering algorithm [72], which works as follows:

$$\mathbf{A} = \underset{N}{\mathrm{argmax}}(\mathbf{C} \times \mathbf{P}^{\mathrm{T}}), \tag{5}$$

$$\hat{\mathbf{C}} = \mathbf{A} \times \mathbf{P}, \tag{6}$$

where $\mathbf{C} \in \mathbb{R}^{N \times D}$, $\mathbf{P} \in \mathbb{R}^{HW \times D}$, and $\mathbf{A} \in \mathbb{R}^{N \times HW}$ stand for cluster centers, pixel features, and clustering assignments, respectively.

Comparing Eq. (4), Eq. (5), and Eq. (6), we notice that the *k*-means clustering algorithm is parameter-free and thus no linear projection is needed for query, key, and value. The updates on cluster centers are not in a residual manner. Most importantly, *k*-means adopts a *cluster-wise argmax* (*i.e.*, argmax operated along the cluster dimension) instead of the spatial-wise softmax when converting the affinity to the attention map (*i.e.*, weights to retrieve and update features).

This observation motivates us to reformulate the cross-attention in vision problems, especially image segmentation. From a clustering perspective, image segmentation is equivalent to grouping pixels into different clusters, where each cluster corresponds to a predicted mask. However, the cross-attention mechanism, also attempting to group pixels to different object queries, instead employs a different *spatial-wise softmax* operation from the *cluster-wise argmax* as in *k*-means. Given the success of *k*-means, we hypothesize that the cluster-wise argmax is a more suitable operation than the spatial-wise softmax regarding pixel clustering, since the cluster-wise argmax performs the hard assignment and efficiently reduces the operation targets from thousands of pixels (HW) to just a few cluster centers (N), which (we will empirically prove) speeds up the training convergence and leads to a better performance.

3.3 k-means Mask Transformer

Herein, we first introduce the crucial component of the proposed k-means Mask Transformer, *i.e.*, k-means cross-attention. We then present its meta architecture and model instantiation.

k-means Cross-Attention.   The proposed *k-means cross-attention* reformulates the cross-attention in a manner similar to k-means clustering:

$$\hat{\mathbf{C}} = \mathbf{C} + \underset{N}{\mathrm{argmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p. \tag{7}$$

Comparing Eq. (4) and Eq. (7), the spatial-wise softmax is now replaced by the cluster-wise argmax. As shown in Fig. 1, with such a simple yet effective change, a typical transformer decoder could be converted to a kMaX decoder. Unlike the original cross-attention, the proposed k-means cross-attention adopts a different operation (*i.e.*, cluster-wise argmax) to compute the attention map, and does not require the multi-head mechanism [89]. However, the cluster-wise argmax, as a hard assignment to aggregate pixel features for the cluster center update, is not a differentiable operation, posing a challenge during training. We have explored several methods (*e.g.*, Gumbel-Softmax [48]), and discover that a simple deep supervision scheme turns out to be most effective. In particular, in our formulation, the affinity logits between pixel features and cluster centers directly correspond to the softmax logits of segmentation masks (*i.e.*, $\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}$ in Eq. (7) corresponds to $\mathbf{F} \times \mathbf{C}^{\mathrm{T}}$ in Eq. (3)), since the cluster centers aim to group pixels of similar affinity together to form the predicted segmentation masks. This formulation allows us to add deep supervision to every kMaX decoder, in order to train the parameters in the k-means cross-attention module.

Meta Architecture.   Figure 2 shows the meta architecture of our proposed kMaX-DeepLab, which contains three main components: pixel encoder, enhanced pixel decoder, and kMaX decoder. The pixel encoder extracts the pixel features either by a CNN [41] or a transformer [70] backbone, while the enhanced pixel decoder is responsible for recovering the feature map resolution as well as enhancing the pixel features via transformer encoders [89] or axial attention [93]. Finally, the kMaX decoder transforms the object queries (*i.e.*, cluster centers) into mask embedding vectors from the k-means clustering perspective.

Model Instantiation.   We build kMaX based on MaX-DeepLab [92] with the official code-base [98]. We divide the whole model into two paths: the pixel path and the cluster path, which are responsible for extracting pixel features and cluster centers, respectively. Figure 3 details our kMaX-DeepLab instantiation with two example backbones.

Pixel Path.   The pixel path consists of a pixel encoder and an enhanced pixel decoder. The pixel encoder is an ImageNet-pretrained [81] backbone, such as ResNet [41], MaX-S [92] (*i.e.*, ResNet-50 with axial attention [93]), and ConvNeXt [71]. Our enhanced pixel decoder consists of several axial attention blocks [93] and bottleneck blocks [41].
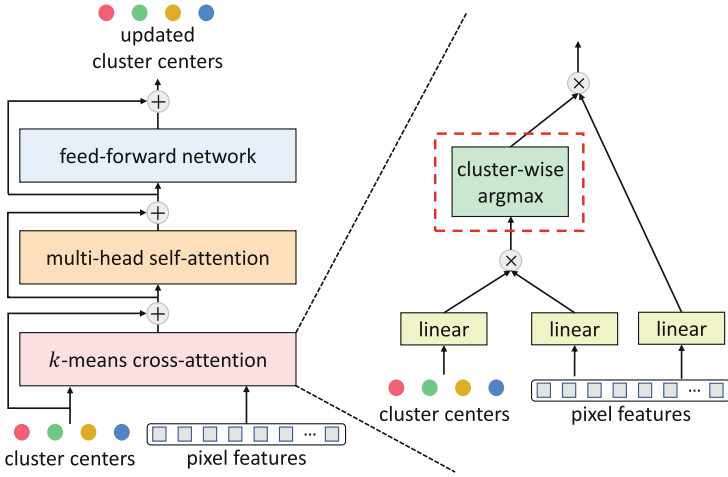
Fig. 1. To convert a typical transformer decoder into our *k*MaX decoder, we simply replace the original cross-attention with our *k*-means cross-attention (*i.e.*, with the only simple change *cluster-wise argmax* high-lighted in red) (Color figure online)
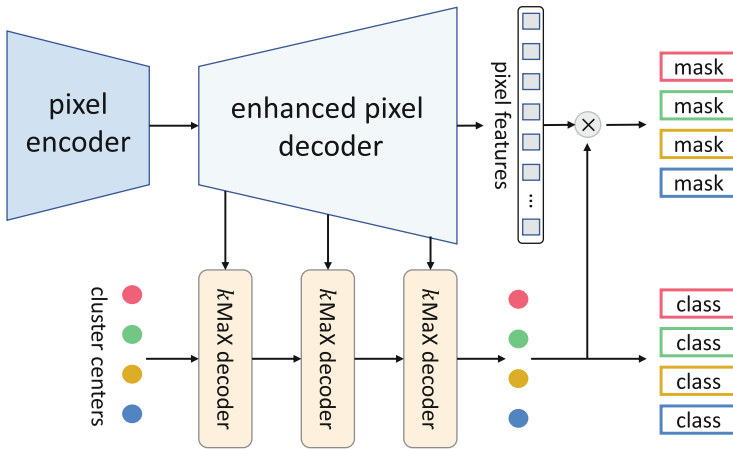

Fig. 2. The meta architecture of *k*-means Mask Transformer consists of three components: pixel encoder, enhanced pixel decoder, and *k*MaX decoder. The pixel encoder is any network backbone. The enhanced pixel decoder includes transformer encoders to enhance the pixel features, and upsampling layers to generate higher resolution features. The series of *k*MaX decoders transform cluster centers into (1) mask embedding vectors, which multiply with the pixel features to generate the predicted masks, and (2) class predictions for each mask.

Cluster Path.   The cluster path contains totally six kMaX decoders, which are evenly distributed among features maps of different spatial resolutions. Specifically, we deploy two kMaX decoders each for pixel features at output stride 32, 16, and 8, respectively.

Loss Functions.   Our training loss functions mostly follow the setting of MaX-DeepLab [92]. We adopt the same PQ-style loss, auxiliary semantic loss, mask-id cross-entropy loss, and pixel-wise instance discrimination loss [104].
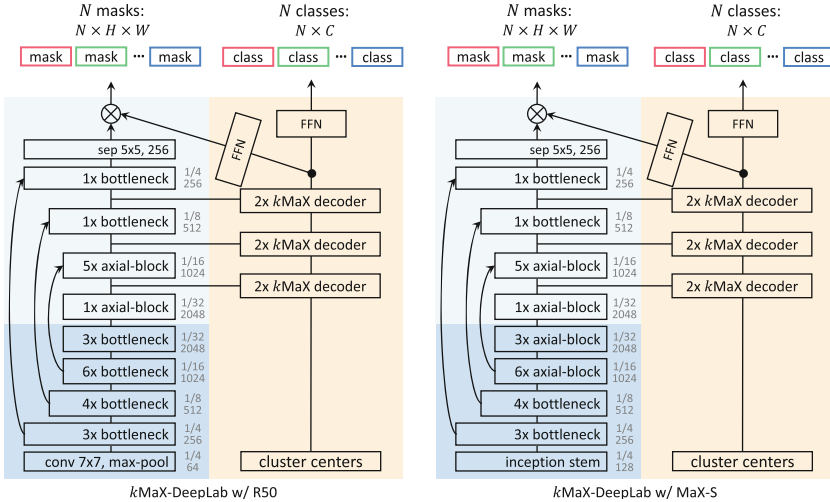

Fig. 3. An illustration of kMaX-DeepLab with ResNet-50 and MaX-S as backbones. The hidden dimension of FFN is 256. The design of kMaX-DeepLab is general to different backbones by simply updating the pixel encoder (marked in dark-blue). The enhanced pixel decoder and kMaX decoder are colored in light-blue and yellow, respectively (Color figure online)

4   Experimental Results

In this section, we first provide our implementation details. We report our main results on COCO [66] and Cityscapes [28]. We also provide visualizations to better understand the clustering process of the proposed kMaX-DeepLab. The ablation studies are provided in the appendix.

4.1   Implementation Details

The meta architecture of the proposed kMaX-DeepLab contains three main components: the pixel encoder, enhanced pixel decoder, and kMaX decoder, as shown in Fig. 2. We provide the implementation details of each component below.

Pixel Encoder.   The pixel encoder extracts pixel features given an image. To verify the generality of *k*MaX-DeepLab across different pixel encoders, we experiment with ResNet-50 [41], MaX-S [92] (*i.e.*, ResNet-50 with axial attention [93] in the 3rd and 4th stages), and ConvNeXt [71].

Enhanced Pixel Decoder.   The enhanced pixel decoder recovers the feature map resolution and enriches pixel features via self-attention. As shown in Fig. 3, we adopt one axial block with channels 2048 at output stride 32, and five axial blocks with channels 1024 at output stride 16. The axial block is a bottleneck block [41], but the 3×3 convolution is replaced by the axial attention [93]. We use one bottleneck block at output stride 8 and 4, respectively. We note that the axial blocks play the same role (*i.e.*, feature enhancement) as the transformer encoders in other works [10, 24, 104], where we ensure that the total number of axial blocks is six for a fair comparison to previous works [10, 24, 104].

Cluster Path.   As shown in Fig. 3, we deploy six *k*MaX decoders, where each two are placed for pixel features (enhanced by the pixel decoders) with output stride 32, 16, 8, respectively. Our design uses six transformer decoders, aligning with the previous works [10, 24, 104], though some recent works [23, 65] adopt more transformer decoders to achieve a stronger performance.

Training and Testing.   We mainly follow MaX-DeepLab [92] for training settings. The ImageNet-pretrained [81] backbone has a learning rate multiplier 0.1. For regularization and augmentations, we adopt drop path [45], random color jittering [29], and panoptic copy-paste augmentation, which is an extension from instance copy-paste augmentation [34, 38] by augmenting both 'thing' and 'stuff' classes. AdamW [53, 73] optimizer is used with weight decay 0.05. The *k*-means cross-attention adopts cluster-wise argmax, which aligns the formulation of attention map to segmentation result. It therefore allows us to directly apply deep supervision on the attention maps. These auxiliary losses attached to each *k*MaX decoder have the same loss weight of 1.0 as the final prediction, and Hungarian matching result based on the final prediction is used to assign supervisions for all auxiliary outputs. During inference, we adopt the same mask-wise merging scheme used in [24, 65, 104, 107] to obtain the final segmentation results.

COCO Dataset.   If not specified, we train all models with batch size 64 on 32 TPU cores with 150k iterations (around 81 epochs). The first 5k steps serve as the warm-up stage, where the learning rate linearly increases from 0 to 5×10^{-4}. The input images are resized and padded to 1281×1281. Following MaX-DeepLab [92], the loss weights for PQ-style loss, auxiliary semantic loss, mask-id cross-entropy loss, instance discrimination loss are 3.0, 1.0, 0.3, and 1.0, respectively. The number of cluster centers (*i.e.*, object queries) is 128, and the final feature map resolution has output stride 4 as in MaX-DeepLab [92].

We have also experimented with doubling the number of object queries to 256 for *k*MaX-DeepLab with ConvNeXt-L, which however leads to a performance loss. Empirically, we adopt a **drop query** regularization, where we randomly drop half of the object queries (*i.e.*, 128) during each training iteration, and all queries (*i.e.*, 256) are used during inference. With the proposed drop query

regularization, doubling the number of object queries to 256 consistently brings 0.1% PQ improvement under the large model regime.

Cityscapes Dataset.   We train all models with batch size 32 on 32 TPU cores with 60k iterations. The first 5k steps serve as the warm-up stage, where learning rate linearly increases from 0 to 3×10^{-4}. The inputs are padded to 1025×2049. The loss weights for PQ-style loss, auxiliary semantic loss, mask-id cross-entropy loss, and instance discrimination loss are 3.0, 1.0, 0.3, and 1.0, respectively. We use 256 cluster centers, and add an additional bottleneck block in the pixel decoder to produce features with output stride 2.

4.2   Main Results

Our main results on the COCO [66] and Cityscapes [28] *val* set are summarized in Table 1 and Table 2, respectively.

Table 1. COCO *val* set results. Our FLOPs and FPS are evaluated with the input size 1200×800 and a Tesla V100-SXM2 GPU. †: ImageNet-22K pretraining. ⋆: Using 256 object queries with drop query regularization. ‡: Using COCO *unlabeled* set

method	backbone	params	FLOPs	FPS	PQ	PQ$^{\text{Th}}$	PQ$^{\text{St}}$
MaskFormer [24]	ResNet-50 [41]	45M	181G	17.6	46.5	51.0	39.8
K-Net [107]	ResNet-50 [41]	–	–	–	47.1	51.7	40.3
CMT-DeepLab [104]	ResNet-50 [41]	–	–	–	48.5	–	–
Panoptic SegFormer [65]	ResNet-50 [41]	51M	214G	7.8	49.6	54.4	42.4
Mask2Former [23]	ResNet-50 [41]	44M	226G	8.6	51.9	57.7	43.0
kMaX-DeepLab	ResNet-50 [41]	57M	168G	22.8	**53.0**	**58.3**	**44.9**
MaX-DeepLab [92]	MaX-S [92]	62M	324G	-	48.4	53.0	41.5
CMT-DeepLab	MaX-S† [92]	95M	396G	8.1	53.0	57.7	45.9
kMaX-DeepLab	MaX-S† [92]	74M	240G	16.9	**56.2**	**62.2**	**47.1**
MaskFormer [24]	Swin-B (W12)† [70]	102M	411G	8.4	51.8	56.9	44.1
CMT-DeepLab [104]	Axial-R104† [104]	135M	553G	6.0	54.1	58.8	47.1
Panoptic SegFormer [65]	PVTv2-B5† [95]	105M	349G	–	55.4	61.2	46.6
Mask2Former [23]	Swin-B (W12)† [70]	107M	466G	–	56.4	62.4	47.3
kMaX-DeepLab	ConvNeXt-B† [71]	122M	380G	11.6	**57.2**	**63.4**	**47.8**
MaX-DeepLab [92]	MaX-L [92]	451M	3692G	–	51.1	57.0	42.2
MaskFormer [24]	Swin-L (W12)† [70]	212M	792G	5.2	52.7	58.5	44.0
K-Net [107]	Swin-L (W7)† [70]	–	–	–	54.6	60.2	46.0
CMT-DeepLab [104]	Axial-R104-RFN† [79]	270M	1114G	3.2	55.3	61.0	46.6
Panoptic SegFormer [65]	Swin-L (W7)† [70]	221M	816G	-	55.8	61.7	46.9
Mask2Former [23]	Swin-L (W12)† [70]	216M	868G	4.0	57.8	64.2	48.1
kMaX-DeepLab	ConvNeXt-L† [71]	232M	744G	6.7	57.9	64.0	48.6
kMaX-DeepLab⋆	ConvNeXt-L† [71]	232M	749G	6.6	58.0	64.2	48.6
kMaX-DeepLab‡	ConvNeXt-L† [71]	232M	744G	6.7	**58.1**	**64.3**	**48.8**

COCO *val* Set. In Table 1, we compare our *k*MaX-DeepLab with other transformer-based panoptic segmentation methods on COCO *val* set. Notably, with a simple ResNet-50 backbone, *k*MaX-DeepLab already achieves 53.0% PQ, surpassing *most* prior arts with stronger backbones. Specifically, *k*MaX-DeepLab outperforms MaskFormer [24] and K-Net [107], all with the ResNet-50 backbone as well, by a large margin of **6.5%** and **5.9%**, while maintaining a similar level of computational costs. Our *k*MaX-DeepLab with ResNet-50 even surpasses the largest variants of MaX-DeepLab [92] by **1.9%** PQ (while using **7.9×** fewer parameters and **22.0×** fewer FLOPs), and MaskFormer (while using **3.7×** fewer parameters and **4.7×** fewer FLOPs) by 0.3% PQ, respectively. With a stronger backbone MaX-S [92], *k*MaX-DeepLab boosts the performance to 56.2% PQ, outperforming MaX-DeepLab with the same backbone by **7.8%** PQ. Our *k*MaX-DeepLab with MaX-S backbone also improves over the previous state-of-art K-Net with Swin-L [70] by **1.6%** PQ. To further push the envelope, we adopt the modern CNN backbone ConvNeXt [71] and set new state-of-the-art results of 57.2% PQ with ConvNeXt-B and 58.0% PQ with ConvNeXt-L, outperforming K-Net with Swin-L by a significant margin of **3.4%** PQ.

When compared to more recent works (CMT-DeepLab [104], Panoptic Seg-Former [65], and Mask2Former [23]), *k*MaX-DeepLab still shows great performances without the advanced modules, such as deformable attention [112], cascaded transformer decoder [23], and uncertainty-based pointly supervision [56]. As different backbones are utilized for each method (*e.g.*, PVTv2 [95], Swin [70], and ConvNeXt [71]), we start with a fair comparison using the ResNet-50 backbone. Our *k*MaX-DeepLab with ResNet-50 achieves a significant better performance compared to CMT-DeepLab, Panoptic SegFormer and Mask2Former by a large margin of **4.5%**, **3.4%**, and **1.1%** PQ, respectively. Additionally, our model runs almost **3×** faster than them (since *k*MaX-DeepLab enjoys a simple design without deformable attention). When employing stronger backbones, *k*MaX-DeepLab with ConvNeXt-B outperforms CMT-DeepLab with Axial-R104, Panoptic SegFormer with PVTv2-B5, and Mask2Former with Swin-B (window size 12) by **3.1%**, **1.8%**, and **0.8%** PQ, respectively, while all models have a similar level of cost (parameters and FLOPs). When scaling up to the largest backbone for each method, *k*MaX-DeepLab outperforms CMT-DeepLab, and Panoptic SegFormer significantly by **2.7%** and **2.2%** PQ. Although we already perform better than Mask2Former with Swin-L (window size 12), we notice that *k*MaX-DeepLab benefits much less than Mask2Former when scaling up from base model to large model (+0.7% for *k*MaX-DeepLab but +1.4% for Mask2Former), indicating *k*MaX-DeepLab's strong representation ability and that it may overfit on COCO *train* set with the largest backbone. Therefore, we additionally perform a simple experiment to alleviate the over-fitting issue by generating pseudo labels [12] on COCO *unlabeled* set. Adding pseudo labels to the training data slightly improves *k*MaX-DeepLab, yielding a PQ score of **58.1%** (the drop query regularization is not used here and the number of object query remains 128).

Cityscapes *val* Set.   In Table 2, we compare our *k*MaX-DeepLab with other state-of-art methods on Cityscapes *val* set. Our reported PQ, AP, and mIoU results use the same panoptic model to provide a comprehensive comparison. Notably, *k*MaX-DeepLab with ResNet-50 backbone already surpasses most baselines, while being more efficient. For example, *k*MaX-DeepLab with ResNet-50 achieves **1.3%** PQ higher performance compared to Panoptic-DeepLab [22] (Xception-71 [27] backbone) with **20%** computational cost (FLOPs) reduced. Moreover, it achieves a similar performance to Axial-DeepLab-XL [93], while using **3.1×** fewer parameters and **5.6×** fewer FLOPs. *k*MaX-DeepLab achieves even higher performances with stronger backbones. Specifically, with MaX-S backbone, it performs on par with previous state-of-the-art Panoptic-DeepLab with SWideRNet [16] backbone, while using **7.2×** fewer parameters and **17.2×** fewer FLOPs. Additionally, even only trained with panoptic annotations, our *k*MaX-DeepLab also shows superior performance in instance segmentation (AP) and semantic segmentation (mIoU). Finally, we provide a comparison with the recent work Mask2Former [23], where the advantage of our *k*MaX-DeepLab becomes even more significant. Using the ResNet-50 backbone for a fair comparison, *k*MaX-DeepLab achieves **2.2%** PQ, **1.2%** AP, and **2.2%** mIoU higher performance than Mask2Former. For other backbone variants with a similar size, *k*MaX-DeepLab with ConvNeXt-B is **1.9%** PQ higher than Mask2Former with Swin-B (window size 12). Notably, *k*MaX-DeepLab with ConvNeXt-B already obtains a PQ score that is **1.4%** higher than Mask2Former with their best backbone. With ConvNeXt-L as backbone, *k*MaX-DeepLab sets a new state-of-the-art record of 68.4% PQ without any test-time augmentation or COCO [66]/Mapillary Vistas [75] pretraining.
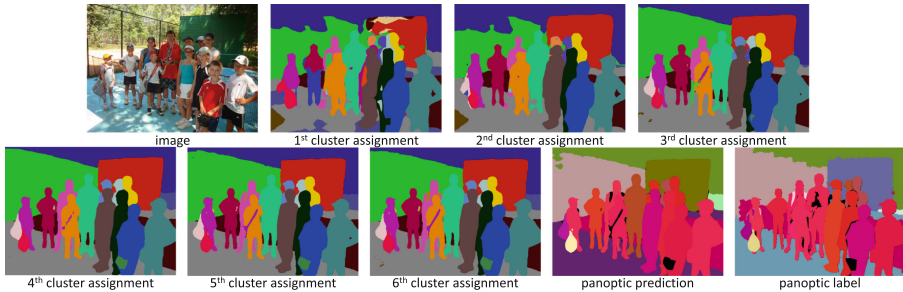

Fig. 4. Visualization of *k*MaX-DeepLab (ResNet-50) pixel-cluster assignments at each *k*MaX decoder stage, along with the final panoptic prediction. In the cluster assignment visualization, pixels with same color are assigned to the same cluster and their features will be aggregated for updating corresponding cluster centers

Visualizations.   In Fig. 4, we provide a visualization of pixel-cluster assignments at each *k*MaX decoder and final prediction, to better understand the working mechanism behind *k*MaX-DeepLab. Another benefit of *k*MaX-DeepLab

Table 2. Cityscapes *val* set results. We only consider methods without extra data [66, 75] and test-time augmentation for a fair comparison. We evaluate FLOPs and FPS with the input size 1025×2049 and a Tesla V100-SXM2 GPU. Our instance (AP) and semantic (mIoU) results are based on the same panoptic model (*i.e.*, no task-specific fine-tuning). †: ImageNet-22K pretraining

method	backbone	params	FLOPs	FPS	PQ	AP	mIoU
Panoptic-DeepLab [22]	Xception-71 [27]	47M	548G	5.7	63.0	35.3	80.5
Axial-DeepLab [93]	Axial-ResNet-L [93]	45M	687G	–	63.9	35.8	81.0
Axial-DeepLab [93]	Axial-ResNet-XL [93]	173M	2447G	–	64.4	36.7	80.6
CMT-DeepLab [104]	MaX-S [92]	–	–	–	64.6	–	81.4
Panoptic-DeepLab [22]	SWideRNet-(1,1,4.5) [16]	536M	10365G	1.0	66.4	40.1	82.2
Mask2Former [23]	ResNet-50 [41]	–	–	–	62.1	37.3	77.5
Mask2Former [23]	Swin-B (W12)† [70]	–	–	–	66.1	42.8	82.7
Mask2Former [23]	Swin-L (W12)† [70]	–	–	–	66.6	43.6	82.9
SETR [109]	ViT-L† [32]	–	–	–	–	–	79.3
SegFormer [99]	MiT-B5 [99]	85M	1460G	2.5	–	–	82.4
Mask R-CNN [40]	ResNet-50 [41]	–	–	–	–	31.5	-
PANet [68]	ResNet-50 [41]	–	–	–	–	36.5	–
kMaX-DeepLab	ResNet-50 [41]	56M	434G	9.0	64.3	38.5	79.7
kMaX-DeepLab	MaX-S† [92]	74M	602G	6.5	66.4	41.6	82.1
kMaX-DeepLab	ConvNeXt-B† [71]	121M	858G	5.2	68.0	43.0	83.1
kMaX-DeepLab	ConvNeXt-L† [71]	232M	1673G	3.1	**68.4**	**44.0**	**83.5**

is that with the cluster-wise argmax, visualizations can be directly drawn as segmentation masks, as the pixel-cluster assignments are exclusive to each other with cluster-wise argmax. Noticeably, the major clustering update happens in the first three stages, which already updates cluster centers well and generates reasonable clustering results, while the following stages mainly focus on refining details. This coincides with our observation that 3 *k*MaX decoders are sufficient to produce good results. Besides, we observe that 1st clustering assignment tends to produce over-segmentation effects, where many clusters are activated and then combined or pruned in the later stages. Moreover, though there exist many fragments in the first round of clustering, it already surprisingly distinguishes different semantics, especially some persons are already well clustered, which indicates that the initial clustering is not only based on texture or location, but also depends on the underlying semantics. Another visualization is shown in Fig. 5, where we observe that *k*MaX-DeepLab behaves in a part-to-whole manner to capture an instance. More experimental results (*e.g.*, ablation studies, test set results) and visualizations are available in the appendix.
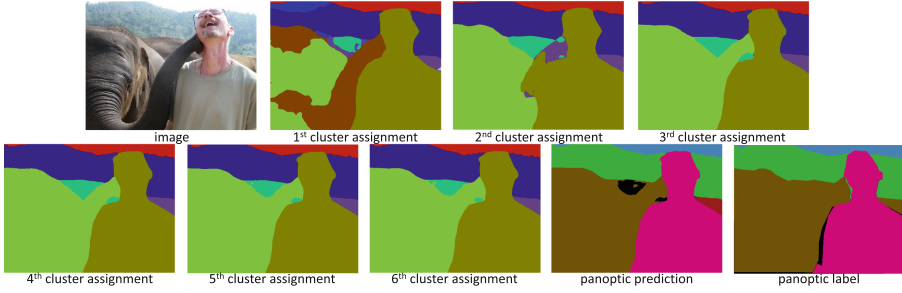
Fig. 5. Visualization of kMaX-DeepLab (ResNet-50) pixel-cluster assignments at each kMaX decoder stage, along with the final panoptic prediction. kMaX-DeepLab shows a behavior of recognizing objects starting from their parts to their the whole shape in the clustering process. For example, the elephant's top head, body, and nose are separately clustered at the beginning, and they are gradually merged in the following stages

5   Conclusion

In this work, we have presented a novel end-to-end framework, called k-means Mask Transformer (kMaX-DeepLab), for segmentation tasks. kMaX-DeepLab rethinks the relationship between pixel features and object queries from the clustering perspective. Consequently, it simplifies the mask-transformer model by replacing the multi-head cross attention with the proposed single-head k-means clustering. We have tailored the transformer-based model for segmentation tasks by establishing the link between the traditional k-means clustering algorithm and cross-attention. We hope our work will inspire the community to develop more vision-specific transformer models.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: Slic super-pixels compared to state-of-the-art superpixel methods. In: IEEE TPAMI (2012)
2. Ainslie, J., Ontanon, S., Alberti, C., Pham, P., Ravula, A., Sanghai, S.: Etc: Encoding long and structured data in transformers. In: EMNLP (2020)
3. Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. In: ICCV (2021)
4. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: ICLR (2015)
5. Bai, M., Urtasun, R.: Deep watershed transform for instance segmentation. In: CVPR (2017)
6. Ballard, D.H.: Generalizing the hough transform to detect arbitrary shapes. In: Pattern Recognition (1981)

7. Bello, I., Zoph, B., Vaswani, A., Shlens, J., Le, Q.V.: Attention augmented convolutional networks. In: ICCV (2019)
8. Beltagy, I., Peters, M.E., Cohan, A.: Longformer: The long-document transformer. arXiv:2004.05150 (2020)
9. Buades, A., Coll, B., Morel, J.M.: A non-local algorithm for image denoising. In: CVPR (2005)
10. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12346, pp. 213–229. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58452-8_13
11. Chen, J., et al.: Transunet: Transformers make strong encoders for medical image segmentation. arXiv:2102.04306 (2021)
12. Chen, L.-C., et al.: Naive-student: leveraging semi-supervised learning in video sequences for urban scene segmentation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12354, pp. 695–714. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58545-7_40
13. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Semantic image segmentation with deep convolutional nets and fully connected crfs. In: ICLR (2015)
14. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. In: IEEE TPAMI (2017)
15. Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv:1706.05587 (2017)
16. Chen, L.C., Wang, H., Qiao, S.: Scaling wide residual networks for panoptic segmentation. arXiv:2011.11675 (2020)
17. Chen, L.C., Yang, Y., Wang, J., Xu, W., Yuille, A.L.: Attention to scale: Scale-aware semantic image segmentation. In: CVPR (2016)
18. Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11211, pp. 833–851. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_49
19. Chen, Y., Kalantidis, Y., Li, J., Yan, S., Feng, J.: Aˆ 2-nets: Double attention networks. In: NeurIPS (2018)
20. Cheng, B., Choudhuri, A., Misra, I., Kirillov, A., Girdhar, R., Schwing, A.G.: Mask2former for video instance segmentation. arXiv:2112.10764 (2021)
21. Cheng, B., et al.: Panoptic-DeepLab. In: ICCV COCO + Mapillary Joint Recognition Challenge Workshop (2019)
22. Cheng, B., et al.: Panoptic-DeepLab: A Simple, Strong, and Fast Baseline for Bottom-Up Panoptic Segmentation. In: CVPR (2020)
23. Cheng, B., Misra, I., Schwing, A.G., Kirillov, A., Girdhar, R.: Masked-attention mask transformer for universal image segmentation. In: CVPR (2022)
24. Cheng, B., Schwing, A.G., Kirillov, A.: Per-pixel classification is not all you need for semantic segmentation. In: NeurIPS (2021)
25. Cheng, J., Dong, L., Lapata, M.: Long short-term memory-networks for machine reading. In: EMNLP (2016)
26. Child, R., Gray, S., Radford, A., Sutskever, I.: Generating long sequences with sparse transformers. arXiv:1904.10509 (2019)
27. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: CVPR (2017)

28. Cordts, M., et al.: The cityscapes dataset for semantic urban scene understanding. In: CVPR (2016)
29. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: Autoaugment: Learning augmentation policies from data. In: CVPR (2019)
30. Dai, Z., Yang, Z., Yang, Y., Carbonell, J.G., Le, Q., Salakhutdinov, R.: Transformer-xl: Attentive language models beyond a fixed-length context. In: ACL (2019)
31. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: NAACL (2019)
32. Dosovitskiy, A., et al.: An image is worth 16×16 words: Transformers for image recognition at scale. In: ICLR (2021)
33. Fan, H., et al.: Multiscale vision transformers. In: ICCV (2021)
34. Fang, H.S., Sun, J., Wang, R., Gou, M., Li, Y.L., Lu, C.: Instaboost: Boosting instance segmentation via probability map guided copy-pasting. In: ICCV (2019)
35. Fu, J., et al.: Dual attention network for scene segmentation. In: CVPR (2019)
36. Gao, N., et al.: Ssap: Single-shot instance segmentation with affinity pyramid. In: ICCV (2019)
37. Gao, P., Zheng, M., Wang, X., Dai, J., Li, H.: Fast convergence of detr with spatially modulated co-attention. In: ICCV (2021)
38. Ghiasi, G., et al.: Simple copy-paste is a strong data augmentation method for instance segmentation. In: CVPR (2021)
39. Gupta, A., Berant, J.: Gmat: Global memory augmentation for transformers. arXiv:2006.03274 (2020)
40. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV (2017)
41. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
42. Ho, J., Kalchbrenner, N., Weissenborn, D., Salimans, T.: Axial attention in multidimensional transformers. arXiv:1912.12180 (2019)
43. Hu, H., Gu, J., Zhang, Z., Dai, J., Wei, Y.: Relation networks for object detection. In: CVPR (2018)
44. Hu, H., Zhang, Z., Xie, Z., Lin, S.: Local relation networks for image recognition. In: ICCV (2019)
45. Huang, G., Sun, Yu., Liu, Z., Sedra, D., Weinberger, K.Q.: Deep networks with stochastic depth. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 646–661. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_39
46. Huang, Z., Wang, X., Huang, L., Huang, C., Wei, Y., Liu, W.: Ccnet: Criss-cross attention for semantic segmentation. In: ICCV (2019)
47. Hwang, J.J., et al.: SegSort: Segmentation by discriminative sorting of segments. In: ICCV (2019)
48. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. In: ICLR (2017)
49. Jia, X., De Brabandere, B., Tuytelaars, T., Gool, L.V.: Dynamic filter networks. In: NeurIPS (2016)
50. Kendall, A., Gal, Y., Cipolla, R.: Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: CVPR (2018)
51. Keuper, M., Levinkov, E., Bonneel, N., Lavoué, G., Brox, T., Andres, B.: Efficient decomposition of image and mesh graphs by lifted multicuts. In: ICCV (2015)
52. Kim, D., et al.: TubeFormer-DeepLab: Video Mask Transformer. In: CVPR (2022)
53. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)

54. Kirillov, A., Girshick, R., He, K., Dollár, P.: Panoptic feature pyramid networks. In: CVPR (2019)
55. Kirillov, A., He, K., Girshick, R., Rother, C., Dollár, P.: Panoptic segmentation. In: CVPR (2019)
56. Kirillov, A., Wu, Y., He, K., Girshick, R.: Pointrend: Image segmentation as rendering. In: CVPR (2020)
57. Kitaev, N., Kaiser, Ł., Levskaya, A.: Reformer: The efficient transformer. In: ICLR (2020)
58. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
59. Leibe, B., Leonardis, A., Schiele, B.: Combined object categorization and segmentation with an implicit shape model. In: Workshop on statistical learning in computer vision, ECCV (2004)
60. Li, Q., Qi, X., Torr, P.H.: Unifying training and inference for panoptic segmentation. In: CVPR (2020)
61. Li, X., et al.: Video k-net: A simple, strong, and unified baseline for video segmentation. In: CVPR (2022)
62. Li, Y., et al.: Attention-guided unified network for panoptic segmentation. In: CVPR (2019)
63. Li, Y., et al.: Fully convolutional networks for panoptic segmentation. In: CVPR (2021)
64. Li, Y., et al.: Neural architecture search for lightweight non-local networks. In: CVPR (2020)
65. Li, Z., et al.: Panoptic segformer. In: CVPR (2022)
66. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
67. Liu, H., et al.: An end-to-end network for panoptic segmentation. In: CVPR (2019)
68. Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path aggregation network for instance segmentation. In: CVPR (2018)
69. Liu, Y., Yang, S., Li, B., Zhou, W., Xu, J., Li, H., Lu, Y.: Affinity derivation and graph merge for instance segmentation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11207, pp. 708–724. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01219-9_42
70. Liu, Z., et al.: Swin transformer: Hierarchical vision transformer using shifted windows. In: ICCV (2021)
71. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: CVPR (2022)
72. Lloyd, S.: Least squares quantization in pcm. IEEE Trans. Inf. Theory **28**(2), 129–137 (1982)
73. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR (2019)
74. Luong, M.T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. In: EMNLP (2015)
75. Neuhold, G., Ollmann, T., Rota Bulo, S., Kontschieder, P.: The mapillary vistas dataset for semantic understanding of street scenes. In: ICCV (2017)
76. Neven, D., Brabandere, B.D., Proesmans, M., Gool, L.V.: Instance segmentation by jointly optimizing spatial embeddings and clustering bandwidth. In: CVPR (2019)
77. Parmar, N., et al.: Image transformer. In: ICML (2018)

78. Porzi, L., Bulò, S.R., Colovic, A., Kontschieder, P.: Seamless scene segmentation. In: CVPR (2019)
79. Qiao, S., Chen, L.C., Yuille, A.: Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. In: CVPR (2021)
80. Ramachandran, P., Parmar, N., Vaswani, A., Bello, I., Levskaya, A., Shlens, J.: Stand-alone self-attention in vision models. In: NeurIPS (2019)
81. Russakovsky, O., et al.: Imagenet large scale visual recognition challenge. IJCV **115**, 211–252 (2015)
82. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. In: NAACL (2018)
83. Shen, Z., Zhang, M., Zhao, H., Yi, S., Li, H.: Efficient attention: Attention with linear complexities. In: WACV (2021)
84. Sofiiuk, K., Barinova, O., Konushin, A.: Adaptis: Adaptive instance selection network. In: ICCV (2019)
85. Strudel, R., Garcia, R., Laptev, I., Schmid, C.: Segmenter: Transformer for semantic segmentation. In: ICCV (2021)
86. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: NeurIPS (2014)
87. Tian, Z., Shen, C., Chen, H.: Conditional convolutions for instance segmentation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12346, pp. 282–298. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58452-8_17
88. Uhrig, J., Rehder, E., Fröhlich, B., Franke, U., Brox, T.: Box2pix: Single-shot instance segmentation by assigning pixels to object boxes. In: IEEE Intelligent Vehicles Symposium (IV) (2018)
89. Vaswani, A., et al.: Attention is all you need. In: NeurIPS (2017)
90. Vincent, L., Soille, P.: Watersheds in digital spaces: an efficient algorithm based on immersion simulations. In: IEEE TPAMI (1991)
91. Wang, H., Luo, R., Maire, M., Shakhnarovich, G.: Pixel consensus voting for panoptic segmentation. In: CVPR (2020)
92. Wang, H., Zhu, Y., Adam, H., Yuille, A., Chen, L.C.: Max-deeplab: End-to-end panoptic segmentation with mask transformers. In: CVPR (2021)
93. Wang, H., Zhu, Y., Green, B., Adam, H., Yuille, A., Chen, L.-C.: Axial-DeepLab: stand-alone axial-attention for panoptic segmentation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12349, pp. 108–126. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58548-8_7
94. Wang, S., Li, B., Khabsa, M., Fang, H., Ma, H.: Linformer: Self-attention with linear complexity. arXiv:2006.04768 (2020)
95. Wang, W., et al.: Pvtv 2: Improved baselines with pyramid vision transformer. arXiv:2106.13797 (2021)
96. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: CVPR (2018)
97. Wang, X., Zhang, R., Kong, T., Li, L., Shen, C.: SOLOv2: Dynamic and fast instance segmentation. In: NeurIPS (2020)
98. Weber, M., et al.: DeepLab2: A TensorFlow Library for Deep Labeling. arXiv: 2106.09748 (2021)
99. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: Segformer: Simple and efficient design for semantic segmentation with transformers. In: NeurIPS (2021)
100. Xiong, Y., et al.: Upsnet: A unified panoptic segmentation network. In: CVPR (2019)

101. Yang, C., et al.: Lite vision transformer with enhanced self-attention. In: CVPR (2022)
102. Yang, T.J., et al.: Deeperlab: Single-shot image parser. arXiv:1902.05093 (2019)
103. Yang, Y., Li, H., Li, X., Zhao, Q., Wu, J., Lin, Z.: Sognet: Scene overlap graph network for panoptic segmentation. In: AAAI (2020)
104. Yu, Q., et al.: Cmt-deeplab: Clustering mask transformers for panoptic segmentation. In: CVPR (2022)
105. Yu, Q., Xia, Y., Bai, Y., Lu, Y., Yuille, A.L., Shen, W.: Glance-and-gaze vision transformer. In: NeurIPS (2021)
106. Zaheer, M., et al.: Big bird: Transformers for longer sequences. In: NeurIPS (2020)
107. Zhang, W., Pang, J., Chen, K., Loy, C.C.: K-net: Towards unified image segmentation. In: NeurIPS (2021)
108. Zhao, H., et al.: PSANet: Point-wise spatial attention network for scene parsing. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11213, pp. 270–286. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01240-3_17
109. Zheng, S., et al.: Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In: CVPR (2021)
110. Zhu, S.C., Yuille, A.: Region competition: Unifying snakes, region growing, and bayes/mdl for multiband image segmentation. In: IEEE TPAMI (1996)
111. Zhu, X., Cheng, D., Zhang, Z., Lin, S., Dai, J.: An empirical study of spatial attention mechanisms in deep networks. In: ICCV (2019)
112. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. In: ICLR (2021)
113. Zhu, Z., Xu, M., Bai, S., Huang, T., Bai, X.: Asymmetric non-local neural networks for semantic segmentation. In: CVPR (2019)