CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Qihang Yu[1*] Huiyu Wang[1] Dahun Kim[2] Siyuan Qiao[3] Maxwell Collins[3] Yukun Zhu[3]
Hartwig Adam[3] Alan Yuille[1] Liang-Chieh Chen[3]
[1]Johns Hopkins University [2]KAIST [3]Google Research

Abstract

We propose Clustering Mask Transformer (CMT-DeepLab), a transformer-based framework for panoptic segmentation designed around clustering. It rethinks the existing transformer architectures used in segmentation and detection; CMT-DeepLab considers the object queries as cluster centers, which fill the role of grouping the pixels when applied to segmentation. The clustering is computed with an alternating procedure, by first assigning pixels to the clusters by their feature affinity, and then updating the cluster centers and pixel features. Together, these operations comprise the Clustering Mask Transformer (CMT) layer, which produces cross-attention that is denser and more consistent with the final segmentation task. CMT-DeepLab improves the performance over prior art significantly by 4.4% PQ, achieving a new state-of-the-art of 55.7% PQ on the COCO test-dev set.

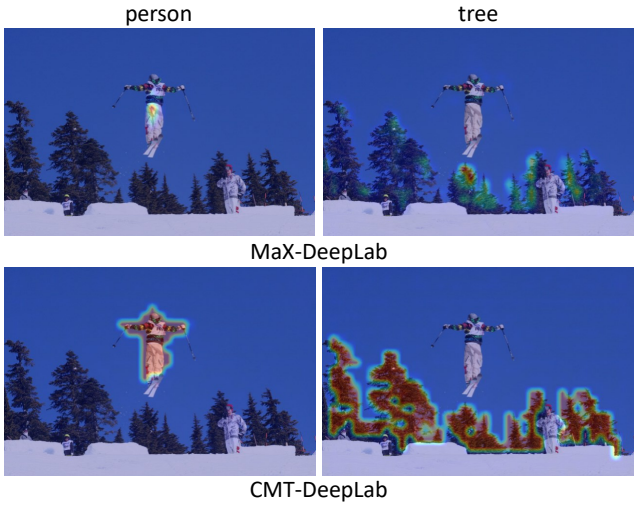

Figure 1. Our CMT-DeepLab generates denser cross-attention maps than MaX-DeepLab [83]. The visualization is based on the last transformer layer with averaged multi-head attentions.

1. Introduction

Panoptic segmentation [47], a recently proposed challenging segmentation task, aims to unify semantic segmentation [34] and instance segmentation [31]. Due to its complicated nature, most panoptic segmentation frameworks [18,47,90] decompose the problem into several manageable proxy tasks, such as box detection [73], box-based segmentation [32], and semantic segmentation [65].

Recently, the paradigm has shifted from the proxy-based approaches to end-to-end systems, since the pioneering work DETR [10], which introduces the first end-to-end object detection method with transformers [81]. In their framework, the image features, extracted by a convolutional network [50], are enhanced by transformer encoders. Afterwards, a set of fixed size of positional embeddings, named object queries, interact with the extracted image features through several transformer decoders, consisting of cross-attention and self-attention modules [3]. The object queries, transformed into output embeddings by the decoders, are then *directly* used for bounding box predictions.

Along the same direction, end-to-end panoptic segmentation framework [83] has been proposed to simplify the panoptic segmentation procedure, avoiding manually designed modules. The core idea is to exploit a set of object queries conditioned on the inputs to predict a set of pairs, each containing a class prediction and a mask embedding vector. The mask embedding vector, multiplied by the image features, yields a binary mask prediction. Notably, unlike the box detection task, where the prediction is based on object queries themselves, segmentation mask prediction requires both object queries and pixel features to interact with each other to obtain the results, which consequently incurs different needs when updating the object queries. To have a deeper understanding towards the role that object queries play, we particularly look into the cross-attention module in the mask transformer decoder, where object queries interact with image features.

Our investigation finds that the update and usage of object queries are performed differently in the transformer-

*Work done during an internship at Google.

based method for segmentation tasks [83]. Specifically, when updating the object queries, a softmax operation is applied to the image dimension, allowing each query to identify its most similar pixels. On the other hand, when computing the segmentation output, a softmax is performed among the object queries so that each pixel finds its most similar object queries. The formulation may potentially cause two issues: sparse query updates and infrequent pixel-query communication. First, the object queries are only *sparsely* updated due to the softmax being applied to a large image resolution, so it tends to focus on only a few locations (top row in Fig. 1). Second, the pixels only have *one* chance to communicate with the object queries in the final output. The first issue is particularly undesired, since segmentation tasks require dense predictions, and ideally a query should *densely* activate all the pixels that belong to the same target. This is different from the box detection task, where object extremities are sufficient (see Fig. 6 of DETR paper [10]).

To alleviate the issues, we draw inspiration from the traditional clustering algorithms [1, 64]. In the current end-to-end panoptic segmentation system [83], the final segmentation output is obtained by assigning each pixel to the object queries based on the feature affinity, similar to pixel-cluster assignment step in [1, 64]. The observation motivates us to rethink the transformer-based methods from the clustering perspective by considering the object queries as cluster centers. We therefore propose to additionally perform the cluster-update step, where the centers are updated by pooling pixel features based on the clustering assignment, when updating the cluster centers (*i.e.*, object queries) in the cross-attention module. As a result, our model generates denser attention maps (bottom row in Fig. 1). We also utilize the pixel-cluster assignment to update the pixel features within each transformer decoder, enabling frequent communication between pixel features and cluster centers.

Additionally, we notice that in the cross-attention module, pixel features are treated as in "bag of words" [49], while the location information is not well utilized. To resolve the issue, we propose to adopt a dynamic position encoding conditioned on the inputs for *location-sensitive* clustering. We explicitly predict a reference mask consisting of a few points for each cluster center. The *location-sensitive* clustering is then achieved by adding location information to pixel features and cluster centers via the coordinate convolution [59] at the beginning of each transformer decoder.

Combining all the proposed components results in our CMT-DeepLab, which reformulates and further improves the previous end-to-end panoptic segmentation system [83] from the traditional clustering perspective. The panoptic segmentation result is naturally obtained by assigning each pixel to its most similar cluster center based on the feature affinity (Fig. 2). In the Clustering Mask Transformer (CMT) module, the pixel features, cluster centers, and pixel-cluster assignments are updated in a manner similar to the clustering algorithms [1, 64]. As a result, without bells and whistles, our proposed CMT-DeepLab surpasses its baseline MaX-DeepLab [83] by 4.4% PQ and achieves 55.7% PQ on COCO panoptic *test-dev* set [58].

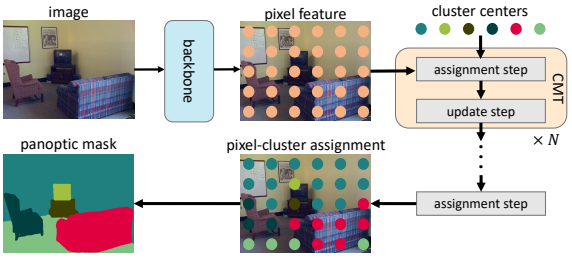

Figure 2. Panoptic segmentation from a clustering perspective. In the proposed Clustering Mask Transformer (CMT) layer, pixels are assigned to cluster centers based on the feature affinity, and the clustering results are used to update both pixel features and cluster centers. After several CMT layers, a refined pixel-cluster assignment is obtained, resulting in the final panoptic mask.

2. Related Works

Transformers. Transformer [81] variants [2, 7, 22, 30, 48, 66, 85, 93] have advanced the state-of-the-art in many natural language processing tasks [25, 26, 75] by capturing relations across modalities [3] or in a single context (self-attention) [21, 81]. In computer vision, transformers are either combined with CNNs [9, 86] or used as standalone models [27, 38, 62, 72, 84]. Both classes of methods have boosted various vision tasks, such as image classification [6, 17, 27, 38, 55, 62, 72, 84], object detection [10, 37, 72, 76, 86, 99], semantic segmentation [15, 28, 40, 95, 98, 100], video recognition [17, 45, 86], image generation [36, 69], and panoptic segmentation [84].

Proxy-based Panoptic Segmentation. Most panoptic segmentation methods rely on proxy tasks, such as object bounding box detection. For example, Panoptic FPN [47] follows a box-based approach that detects object bounding boxes and predicts a mask for each box, usually with a Mask R-CNN [32] and FPN [57]. Then, the instance segments ('thing') and semantic segments ('stuff') [12] are fused by merging modules [52–54, 63, 70, 90, 92] to generate panoptic segmentation. Other proxy-based methods typically start with semantic segments [11, 13, 16] and group 'thing' pixels into instance segments with various proxy tasks, such as instance center regression [19, 42, 56, 67, 80, 84, 91], Watershed transform [4, 8, 82], Hough-voting [5, 8, 51], or pixel affinity [8, 29, 43, 61, 77]. DetectoRS [71] achieved the state-of-the-art in this category with recursive feature pyramid and switchable atrous convolution. Recently, DETR [10] extended the proxy-based methods with its transformer-based end-to-end detector.

End-to-end Panoptic Segmentation. Along the same direction, MaX-DeepLab [83] proposed an end-to-end strategy, in which class-labeled object masks are directly predicted and are trained by Hungarian matching the predicted masks with ground truth masks. In this work, we improve over MaX-DeepLab by approaching the pixel assignment task from a clustering perspective. Concurrent with our work, Segmenter [78] and MaskFormer [20] formulated an end-to-end strategy from a mask classification perspective, same as MaX-DeepLab [83], but extends from panoptic segmentation to semantic segmentation.

3. Method

Herein, we firstly introduce recent transformer-based methods [83] for end-to-end panoptic segmentation. Our observation reveals a difference between the cross-attention and final segmentation output regarding the way that they utilize object queries. We then propose to resolve it with a clustering approach, resulting in our proposed Clustering Mask Transformer (CMT-DeepLab), as shown in Fig. 3 and Fig. 4. In the following parts, object queries and cluster centers refer to the same learnable embedding vectors and we use them interchangeably for clearer representation.

3.1. Transformers for Panoptic Segmentation

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

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

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.

Inspired by DETR [10], several transformer-based end-to-end panoptic segmentation methods [83] have been proposed recently, which directly predict N masks and their semantic classes. N is a fixed number and $N \geq K$.

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

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

To predict these N masks, N object queries are utilized to aggregate information from the image features through a transformer decoder, which consists of self-attention and cross-attention modules. The object queries and image features interact with each other in the cross-attention module:

$$\hat{\mathbf{C}} = \mathbf{C} + \underset{HW}{\text{softmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p, \tag{3}$$

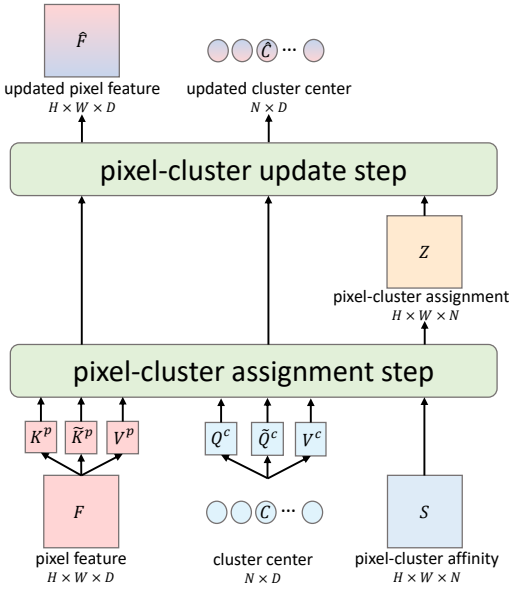

Figure 3. A visual illustration of Clustering Mask Transformer layer, where three variables are updated in a dynamic manner based on the clustering results: pixel features, cluster centers, and pixel-cluster affinity. Details of assignment and update steps are illustrated in Fig. 4.

where $\mathbf{C} \in \mathbb{R}^{N \times D}$ refers to object queries with D channels, and $\hat{\mathbf{C}}$ denotes the updated object queries. We use the underscript to represent the axis for softmax, and superscripts p and c to indicate the feature projected from the image 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 multi-head attention and feed-forward network (FFN) in the equation.

The object queries, updated by multiple transformer decoders, are employed as dynamic convolution weights (with kernel size 1×1) [41, 79, 87] to obtain the prediction $\mathbf{Z} \in \mathbb{R}^{HW \times N}$ that consists of N binary masks. That is,

$$\mathbf{Z} = \underset{N}{\text{softmax}}(\mathbf{F} \times \mathbf{C}^{\mathrm{T}}), \tag{4}$$

where $\mathbf{F} \in \mathbb{R}^{HW \times D}$ refers to the extracted image features.

3.2. Current Issues and New Clustering Perspective

Even though effective, the transformer-based architectures were originally designed for object detection [10] and thus they do not naturally deal with segmentation masks. Specifically, they use different formulations for the object query updates and the segmentation specific output head. To be precise, both the update of object queries (Eq. (3)) and final output (Eq. (4)) are based on their corresponding feature affinity (*i.e.*, $\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}$ and $\mathbf{F} \times \mathbf{C}^{\mathrm{T}}$). However, the following softmax operations are applied along different dimensions. To update the object queries, the softmax

is applied to the image spatial dimension (HW) with the goal to identify the most similar pixels for each query. On the other hand, to obtain the final output, the softmax is performed among the object queries (N) so that each pixel finds its most similar object queries. The inconsistency potentially causes two issues. First, the object queries are only *sparsely* updated due to the softmax operated along a large spatial dimension, tending to focus on only a few locations (Fig. 1). Second, the output update is only performed *once* in the end, and therefore the pixels only have one chance to receive the information passed from the object queries.

To alleviate the issues, we take a closer look at Eq. (4), which assigns each pixel to the object queries based on the feature affinity. This is, in fact, very similar to typical clustering methods [1, 64] (particularly, the pixel-cluster assignment step). This observation motivates us to rethink the transformer-based methods from the typical clustering perspective [1, 97] by considering the object queries $\mathbf{C}$ as cluster centers. With the clustering perspective in mind, we re-interpret Eq. (4) as the pixel-cluster assignment. This interpretation naturally inspires us to perform a cluster-update step where the cluster centers are updated by pooling pixel features based on the clustering assignment, *i.e.*, $\mathbf{Z}^{\mathrm{T}} \times \mathbf{F} = (\mathrm{softmax}_N(\mathbf{F} \times \mathbf{C}^{\mathrm{T}}))^{\mathrm{T}} \times \mathbf{F}$.

We propose to extend the formulation to a transformer decoder module, whose query, key, and value are obtained by linearly projecting the image features and cluster centers:

$$\hat{\mathbf{C}} = \mathbf{C} + (\underset{N}{\mathrm{softmax}}(\tilde{\mathbf{K}}^p \times (\tilde{\mathbf{Q}}^c)^{\mathrm{T}}))^{\mathrm{T}} \times \mathbf{V}^p. \tag{5}$$

Comparing Eq. (3) and Eq. (5), we have the query $\tilde{\mathbf{Q}}^c$ and key $\tilde{\mathbf{K}}^p$ coming from another linear projection, and the softmax is performed along the cluster center dimension.

In the following subsection, we detail how the clustering perspective alleviates the issues of current transformer-based methods. In the discussion, we use object queries and cluster centers interchangeably.

3.3. Clustering Mask Transformers

In this subsection, we redesign the cross-attention in the transformer decoder from the clustering perspective, aiming to resolve the issues raised in Sec. 3.2.

Residual Path between Cluster Assignments. Similar to other designs [10], we stack the transformer decoder multiple times. To facilitate the learning of pixel-cluster assignment, we add a residual connection [33] between clustering results including the final segmentation result. That is,

$$\mathbf{Z} = \underset{N}{\mathrm{softmax}}(\mathbf{S} + \tilde{\mathbf{K}}^p \times (\tilde{\mathbf{Q}}^c)^{\mathrm{T}}), \tag{6}$$

where $\mathbf{S} \in \mathbb{R}^{HW \times N}$ is the affinity logits between linearly projected pixel features and cluster centers in the previous

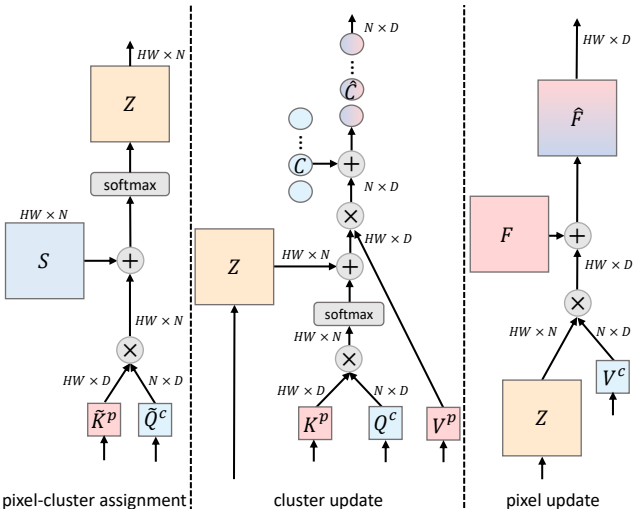

Figure 4. Detailed visual illustration of pixel-cluster assignment (left), cluster centers update (middle), and pixel features update (right). The tensor shapes are specified for illustration.

decoder (left panel of Fig. 4). We emphasize that since our clustering results have the same format as the segmentation output, we are able to add residual connections between them, which is further supervised by the ground-truths.

Solution to Sparse Query Update. We propose a simple and effective solution to avoid the sparse query update by combining the proposed clustering center update (*i.e.*, Eq. (5)) with the original cross-attention (*i.e.*, Eq. (3)), resulting in

$$\begin{aligned}\hat{\mathbf{C}} =& \mathbf{C} + \underset{HW}{\mathrm{softmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p + \mathbf{Z}^{\mathrm{T}} \times \mathbf{V}^p \\ =& \mathbf{C} + (\underset{HW}{\mathrm{softmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) + \mathbf{Z}^{\mathrm{T}}) \times \mathbf{V}^p,\end{aligned} \tag{7}$$

where $\mathbf{Z}$ is obtained from Eq. (6). The update is shown in the center panel of Fig. 4, while the effect of *densified* attention could be found in Fig. 1.

Solution to Infrequent Pixel Updates. We propose to also utilize the clustering result $\mathbf{Z}$ to perform an update on the pixel features using the features of cluster centers, *i.e.*,

$$\hat{\mathbf{F}} = \mathbf{F} + \mathbf{Z} \times \mathbf{V}^c, \tag{8}$$

where $\mathbf{V}^c \in \mathbb{R}^{N \times D}$ is the linearly projected values from the cluster centers. This update is performed within each stacked transformer decoder, enabling frequent communication between pixel features and cluster centers (right panel of Fig. 4).

To this end, we have improved the transformer cross-attention module by simultaneously updating the clustering result (*i.e.*, pixel-cluster assignment), pixel features, and cluster centers. However, we notice that during the interaction between pixel features and cluster centers, pixel fea-

tures are treated as bag of words [49], while the location information is not well utilized. Although learnable positional encodings (*i.e.* object queries [10]) are used for the cluster center embeddings, the positional encodings are fixed for all input images, which is suboptimal when an object query predicts masks at different locations in different input images. To resolve the issue, we propose to adopt a dynamic positional encoding conditioned on the inputs for *location-sensitive* clustering.

Location-Sensitive Clustering. To inject dynamic location information to cluster centers, we explicitly predict a reference mask that consists of M points for each cluster center. In particular, a MLP is used to predict the reference mask out of cluster center features, followed by a sigmoid activation function. That is, we have:

$$\hat{e} = e + \text{MLP}(\mathbf{C}), \tag{9}$$

$$r^c = \text{sigmoid}(\hat{e}), \tag{10}$$

where $e \in \mathbb{R}^{N\times 2M}$ denotes an embedding projected from the cluster centers, and $r^c = [r^{c,h}, r^{c,w}] \in \mathbb{R}^{N\times 2M}$ are the reference mask represented with M pairs of coordinates $(r_i^{c,h}, r_i^{c,w})$. We utilize a residual update manner [33, 99] to predict the reference mask, with a skip-connection on the projected embedding e across stages. The location space is normalized to $[0, 1] \times [0, 1]$.

We add location information to pixel features and cluster centers through a coordinate convolution [59]. Specifically, we apply coordinate convolutions at the beginning of each transformer layer to ensure location information is considered during the clustering process, as shown below.

$$\hat{\mathbf{C}} = \text{Conv}(\text{Concat}(\mathbf{C}, r^c)), \tag{11}$$

$$\hat{\mathbf{F}} = \text{Conv}(\text{Concat}(\mathbf{F}, r^p)), \tag{12}$$

where $r^p \in \mathbb{R}^{HW\times 2}$ is the coordinates normalized to $[0, 1]$ for pixels in image space, which is fixed and not learnable.

We note that compared to the reference point used in the Deformable DETR [99], the proposed reference mask provides a rough mask shape prior for the whole object mask. Besides, we adopt a much simpler way to incorporate the location information via coordinate convolution.

In order to learn meaningful reference mask predictions, we optimize the reference masks towards ground truth masks by proposing a mask approximation loss.

Mask Approximation Loss. We propose a loss to minimize the distance between the distribution of predicted reference points and that of points of ground-truth object masks. In detail, we utilize the Hungarian matching result to assign the ground-truth mask for each cluster center. Given the predicted M points for each cluster center, we infer their extreme points [68] and mask center. We then apply an L_1 loss to push them to be closer to their ground-truth extreme points and center. Specifically, we have

$$\begin{aligned}\mathcal{L}_{\text{ext}} = \frac{1}{4K}\sum_{i=1}^{K}(&|\min(r_i^{c,h}) - \min(y_i^h)| + |\max(r_i^{c,h}) - \max(y_i^h)| \\ &+ |\min(r_i^{c,w}) - \min(y_i^w)| + |\max(r_i^{c,w}) - \max(y_i^w)|),\end{aligned}$$

$$\mathcal{L}_{\text{cen}} = \frac{1}{2K}\sum_{i=1}^{K}(|\text{avg}(r_i^{c,h}) - \text{avg}(y_i^h)| + |\text{avg}(r_i^{c,w}) - \text{avg}(y_i^w)|),$$

$$\mathcal{L}_{\text{loc}} = \mathcal{L}_{\text{ext}} + \mathcal{L}_{\text{cen}}, \tag{13}$$

where $y = [y^h, y^w]$ are pixels on ground-truth masks and predicted reference masks have been filtered and re-ordered based on Hungarian matching results.

Finally, combining all the proposed designs results in our Clustering Mask Transformer, or CMT-DeepLab, which rethinks the current mask transformer design from the clustering perspective.

3.4. Network Instantiation

We instantiate CMT-DeepLab on top of MaX-DeepLab-S [83] (abbreviated as MaX-S). We first refine its architecture design. Afterwards, we enhance it with the proposed Clustering Mask Transformers.

Base Architecture. We use MaX-S [83] as our base architecture. To better align it with other state-of-the-art architecture designs [62], we use GeLU [35] activation to replace the original ReLU activation functions. Besides, we remove all transformer blocks in the pretrained backbones, which reverts the backbone from MaX-S back to Axial-ResNet-50 [84]. On top of the backbone, we append six dual-path axial-transformer blocks [83] (three at stage-5 w/ channels 2048, and the other three at stage-4 w/ channels 1024), yielding totally six axial self-attention and six cross-attention modules, which aligns with the number of attention operations used in other works [10, 20]. Additionally, we obtain a larger network backbone by scaling up the number of blocks in stage-4 of the backbone [14]. As a result, two different model variants are used: one built upon Axial-ResNet-50 backbone with number of blocks $[3, 4, 6, 3]$ (starting from stage-2), and another built upon Axial-ResNet-104 with number of blocks $[3, 4, 24, 3]$. See the supplementary material for a detailed illustration.

Loss Functions. Following [83], we use the PQ-style loss and three other auxiliary losses for the model training, including the instance discrimination loss, mask-ID cross-entropy, and semantic segmentation loss. However, we note that the instance discrimination loss proposed in [83] aims to push pixel features to be close to the feature center computed based on the ground-truth mask, instead of directly to the cluster centers. Therefore, we adopt the pixel-wise instance discrimination loss, which learns closely aligned representations for all pixels from the same class, allowing better clustering results.

Formally, we sample a set of pixels A from the image, where we add bias to pixels' sampling probability based on

the size of object mask they belong to. Thus, final sampled pixels are more balanced from objects with different scales. Afterwards, we directly perform contrastive loss on top of these pixels with multiple positive targets [44]:

$$\mathcal{L}^{insdis} = \sum_{a \in A} \frac{-1}{|P(a)|} \sum_{p \in P(a)} \log \frac{\exp\left(f_a \cdot f_p / \tau\right)}{\sum_{b \in A} \exp\left(f_a \cdot f_b / \tau\right)}, \tag{14}$$

where $P(a)$ is a subset of pixels of A that belongs to the same cluster (*i.e.*, object mask) with a, and $|P(a)|$ is its cardinally. We use f to denote a pixel feature vector, and τ is the temperature.

Recursive Feature Network. Motivated by DetectoRS [71] and CBNet [60], we adopt a simple strategy, named Recursive Feature Network (RFN), to increase the network capacity by stacking twice the whole model (including the backbone and added transformer blocks). There are two main differences. First, since we do not employ an FPN [57] (as in [71]), we simply connect the features at stride 4 (*i.e.*, same stride as the segmentation output). Second, we do not use the complicated fusion module proposed in [71], but simply average the features between two stacked networks, which we empirically found to be better by around 0.2% PQ.

4. Experimental Results

We report main results on COCO along with state-of-the-art methods, followed by ablation studies on the architecture variants, clustering mask transformers, pretrained weights, post-processing, and scaling strategies. Finally, we analyze the working mechanism behind CMT-DeepLab with visualizations.

Implementation Details. We build CMT-DeepLab on top of MaX-DeepLab [83] with the official code-base [88]. The training strategy mainly follows MaX-DeepLab. If not specified, the model is trained with 64 TPU cores for 100k iterations with the first 5k for warm-up. We use batch size = 64, Adam [46] optimizer, a poly schedule learning rate of 10^{-3}. The ImageNet-pretrained [74] backbone has a learning rate multiplier 0.1. Weight decay is set to 0 and drop-path rate [39] to 0.2. The input images are resized and padded to 1281×1281 for training and inference. We use $|A| = 4096$ for pixel-wise contrastive loss and $M = 8$ for reference masks, we also tried other values but did not observe significant difference. Loss weight is 1.0 for the mask approximation loss. Other losses employ the same setting as [83]. During inference, we adopt a mask-wise merging scheme [20] to obtain the final results.

4.1. Main Results

Our main results on the COCO panoptic segmentation *val* set and *test-dev set* are summarized in Tab. 1.

Val Set. We compare our validation set results with box-based, center-based, and end-to-end panoptic segmentation methods. It is noticeable that CMT-DeepLab, built upon a smaller backbone Axial-ResNet-50, already surpasses all other box-based and center-based methods by a large margin. More importantly, when compared with its end-to-end baseline MaX-DeepLab-S [83], we observe a significant improvement of 4.6% PQ. Our small model even surpasses previous state-of-the-art method MaX-DeepLab-L [83], which has more than $5\times$ parameters, by 1.9% PQ. Compared to recently proposed MaskFormer [20], CMT-DeepLab still shows a significant advantage of 1.2% PQ and 1.4% PQ while being more light-weight over the small and large model variant, respectively. The significant improvement illustrates the importance of introducing the concept of clustering into transformer, which leads to a denser attention preferred by the segmentation task. Our CMT-DeepLab with a deeper backbone Axial-ResNet-104 improves the *single-scale* performance to 54.1% PQ, outperforming *multi-scale* Axial-DeepLab [84] by 10.2% PQ. Moreover, we enhance the model with the proposed RFN, which further improves the PQ to 55.3%.

Test-dev Set. We verify the transfer-ability of CMT-DeepLab on *test-dev* set, which shows consistently better results compared to other methods. Especially, the small version of CMT-DeepLab with Axial-R50 backbone outperforms DETR [10] by 7.4% PQ, MaX-DeepLab-S [83] by 4.4% PQ, and MaX-DeepLab-L [83] by 2.1% PQ. Additionally, employing a deeper backbone Axial-R104 can boost the PQ score by 1.1% PQ. On top of it, using the proposed RFN further improves PQ to 55.7%, surpassing MaskFormer [20] with Swin-L [62] backbone by 2.4% PQ.

4.2. Ablation Studies

Herein, we evaluate the effectiveness of different components of the proposed CMT-DeepLab. For all the following experiments, we use MaX-DeepLab-S [83] with GeLU [35] activation function as our baseline. This improved baseline has a 0.3% higher PQ compared to the original MaX-DeepLab-S. If not specified, we perform all ablation studies with the Axial-R50 backbone [33, 84], ImageNet-1K [74] pretrained, crop size 641×641, and 100k training iterations.

Clustering Mask Transformer. We start with adding the design variants of Clustering Mask Transformer step by step, as summarized in Tab. 2a. Regarding the object queries as cluster centers, and adding a clustering-style update can improve the PQ by 0.9%, illustrating the effectiveness of the cluster center perspective and the importance of including more pixels into the cluster center updates. Next, we utilize pixel-wise contrastive loss instead of the original instance-wise contrastive loss, resulting in another 0.4% PQ improvement, as it provides a better supervision signal from a clustering perspective. In short, re-designing the trans-

method	backbone	TTA	params	val-set PQ	val-set PQ^{Th}	val-set PQ^{St}	test-dev PQ	test-dev PQ^{Th}	test-dev PQ^{St}
box-based panoptic segmentation methods									
Panoptic-FPN [47]	R101			40.3	47.5	29.5	-	-	-
UPSNet [90]	R50			42.5	48.5	33.4	-	-	-
UPSNet [90]	R50	✓		43.2	49.1	34.1	-	-	-
UPSNet [90]	DCN-101 [24]	✓		-	-	-	46.6	53.2	36.7
DETR [10]	R101		61.8M	45.1	50.5	37.0	46.0	-	-
DetectoRS [71]	RX-101 [89]	✓		-	-	-	49.6	57.8	37.1
center-based panoptic segmentation methods									
Panoptic-DeepLab [19]	X-71 [23]		46.7M	39.7	43.9	33.2	-	-	-
Panoptic-DeepLab [19]	X-71 [23]	✓	46.7M	41.2	44.9	35.7	41.4	45.1	35.9
Axial-DeepLab-L [84]	AX-L [84]		44.9M	43.4	48.5	35.6	43.6	48.9	35.6
Axial-DeepLab-L [84]	AX-L [84]	✓	44.9M	43.9	48.6	36.8	44.2	49.2	36.8
end-to-end panoptic segmentation methods									
MaX-DeepLab-S [83]	MaX-S [83]		61.9M	48.4	53.0	41.5	49.0	54.0	41.6
MaX-DeepLab-L [83]	MaX-L [83]		451M	51.1	57.0	42.2	51.3	57.2	42.4
MaskFormer [20]	Swin-B‡ [62]		102M	51.8	56.9	44.1	-	-	-
MaskFormer [20]	Swin-L‡ [62]		212M	52.7	58.5	44.0	53.3	59.1	44.5
CMT-DeepLab	Axial-R50‡ [84]		94.9M	53.0	57.7	45.9	53.4	58.3	46.0
CMT-DeepLab	Axial-R104‡		135.2M	54.1	58.8	**47.1**	54.5	59.6	46.9
CMT-DeepLab	Axial-R104‡-RFN		270.3M	55.1	60.6	46.8	55.4	61.0	**47.0**
CMT-DeepLab (iter 200k)	Axial-R104‡-RFN		270.3M	**55.3**	**61.0**	46.6	**55.7**	**61.6**	46.8

Table 1. Results comparison on COCO val and test-dev set. **TTA:** Test-time augmentation. ‡: ImageNet-22K pretraining. We provide more comparisons with concurrent works in the supplementary materials.

	PQ	PQ^{Th}	PQ^{St}
baseline	46.2	50.0	40.5
+ clustering transformer	47.1	51.0	41.1
+ pixel-wise contrastive loss	**47.5**	**51.1**	**42.1**

(a) **CMT-DeepLab: clustering update**.

	PQ	PQ^{Th}	PQ^{St}
baseline	46.2	50.0	40.5
+ ref. mask pred.	46.6	50.3	40.9
+ coord-conv	**46.9**	**50.6**	**41.3**

(b) **CMT-DeepLab: location-senseitive clustering**.

clustering update	location	decoder	params	PQ	PQ^{Th}	PQ^{St}
			61.9M	46.2	50.0	40.5
✓			61.9M	47.5	51.1	42.1
	✓		65.5M	46.9	50.6	41.3
		✓	91.0M	47.1	51.3	40.9
✓		✓	91.0M	48.1	51.9	42.2
✓	✓	✓	94.9M	**48.4**	**52.1**	**42.8**

(c) **CMT-DeepLab: architecture**.

ImageNet-22K	RFN	mask-wise merge	PQ	PQ^{Th}	PQ^{St}
			48.4	52.1	42.8
✓			49.3	53.3	43.4
✓	✓		50.1	**54.8**	43.0
✓	✓	✓	**50.6**	**54.8**	**44.3**

(d) **CMT-DeepLab: pretraining, post-processing, scaling**.

Table 2. CMT-DeepLab ablation experiments. Baseline is labeled with grey color. Results are reported in accumulative manner.

res.	backbone	iters	PQ	PQ^{Th}	PQ^{St}
641	Axial-R50	100k	50.1	53.5	44.9
641	Axial-R50	200k	50.6	54.5	44.8
1281	Axial-R50	100k	53.0	57.7	45.9
1281	Axial-R50	200k	53.5	58.5	45.9
641	Axial-R104	100k	51.7	55.4	46.4
641	Axial-R104	200k	52.2	56.4	46.0
1281	Axial-R104	100k	54.1	58.8	47.1
1281	Axial-R104-RFN	100k	55.1	60.6	46.8

Table 3. Ablation on **input resolution/backbone/training iterations**. ImageNet-22K, mask-wise merge are used for all results.

former layer from a clustering perspective leads to a 1.3% PQ improvement overall.

Location-Sensitive Clustering. Location information plays an important role in the clustering process, as shown in Tab. 2b. Each cluster center needs to predict a reference mask without using pixel features (*i.e.*, appearance information), which requires cluster centers to include more location information in the feature embedding and thus benefits clustering. Adding reference masks prediction alone brings a gain of 0.4% PQ. Using the coordinate convolution (coord-conv) [59] to include the reference mask information yields another 0.3% PQ improvement. In sum, the location-sensitive clustering brings up the PQ score by 0.7%.

Stronger Decoder. We study the effect of using a stronger decoder design [10, 20]. We remove all transformer layers from the pretrained backbone, which reverts the MaX-S backbone [83] to Axial-ResNet-50 [84]. Then we stack more axial-blocks with transformer module in the decoder

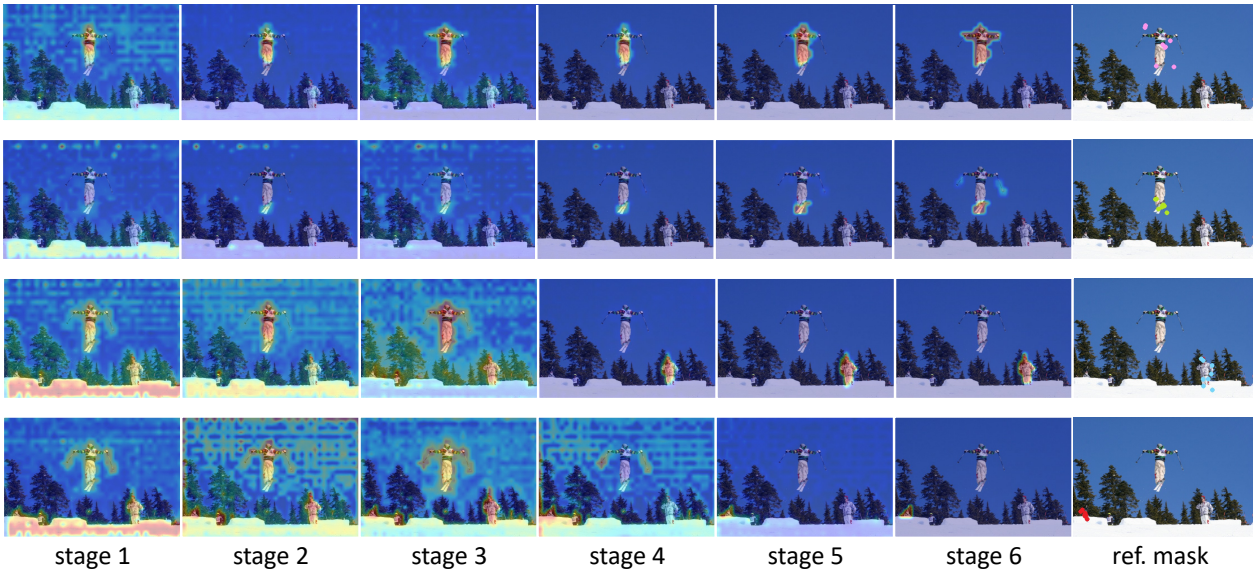

Figure 5. Visualization of clustering results at different stages (*i.e.*, transformer layers), with last column for reference masks. The clustering results, providing denser attention maps, are close-to-random at the beginning and are gradually refined to focus on corresponding object.

part. More specifically, we use six self-attention modules and six cross-attention modules in total for the decoder, which aligns to the design of DETR [10]. As shown in Tab. 2c, this stronger decoder brings 0.9% PQ improvement (47.1% *vs*. 46.2%).

As shown in Tab. 2c, these improvements are complementary to each other, while combining them together can further boost the performance. Adding all of them leads to CMT-DeepLab, which improves 2.2% PQ over the MaX-DeepLab-S-GeLU baseline. We note that the major cost comes from the stronger decoder, which accounts for the increase of 29.1M parameters, while clustering update and location-sensitive clustering improve the PQ by 1.3% and 0.7%, respectively, with neglectable extra parameters.

Pretraining, Post-processing, and Scaling. We further verify the effect of better pretraining, post-processing, and scaling-up, with results summarized in Tab. 2d and Tab. 3. Specifically, we find that using ImageNet-22K for pretraining can improve the performance by 0.9% PQ. Furthermore, we empirically find that using the mask-wise merge strategy [20] to obtain panoptic results, compared to the simple per-pixel strategy [83], improves PQ by 0.5%. Next, we scale up CMT-DeepLab from different dimensions. With a longer training strategies (from 100k to 200k iterations), we observe a consistent 0.5% PQ improvement over various settings, where the improvement mainly comes from PQ^{Th} (*i.e.*, thing classes), indicating that the model needs a longer training schedule to better segment thing objects. We also find that using a larger input resolution (from 641 to 1281) significantly boosts the performance by more than 2% PQ. Besides, increasing the model size by using a deeper backbone or stacking the model with RFN can improve the performance by 1.6% and 1.0%, respectively.

Visualization. In Fig. 5, we visualize the clustering results in each stage as well as the learned reference masks. As shown in the figure, the clustering results, starting with a close-to-random assignment, gradually learn to focus on the target instances. For example, in the last two rows of Fig. 5, the clustering results firstly focus on all the 'person' instances and the background 'snow', and then they start to concentrate on the specific person instance, showing a refinement from "semantic segmentation" to "instance segmentation". Moreover, as shown in the last column of Fig. 5, the learned reference mask provides a reasonable prior for the object mask.

5. Conclusion

In this work, we have introduced CMT-DeepLab, which rethinks object queries, used in the current mask transformers for panoptic segmentation, from a clustering perspective. Considering object queries as cluster centers, our framework additionally incorporates the proposed cluster center update in the cross-attention module, which significantly enriches the learned cross-attention maps and further facilitates the segmentation prediction. As a result, CMT-DeepLab achieves new state-of-the-art performance on the COCO dataset, and sheds light on the working mechanism behind mask transformers for segmentation tasks.

Acknowledgments. We thank Jun Xie for the valuable feedback on the draft. This work was supported in part by ONR N00014-21-1-2812.

References

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE TPAMI*, 2012. 2, 4

[2] Joshua Ainslie, Santiago Ontanon, Chris Alberti, Philip Pham, Anirudh Ravula, and Sumit Sanghai. Etc: Encoding long and structured data in transformers. In *EMNLP*, 2020. 2

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015. 1, 2

[4] Min Bai and Raquel Urtasun. Deep watershed transform for instance segmentation. In *CVPR*, 2017. 2

[5] Dana H Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 1981. 2

[6] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *ICCV*, 2019. 2

[7] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020. 2

[8] Ujwal Bonde, Pablo F Alcantarilla, and Stefan Leutenegger. Towards bounding-box free panoptic segmentation. *arXiv:2002.07705*, 2020. 2

[9] Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *CVPR*, 2005. 2

[10] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 1, 2, 3, 4, 5, 6, 7, 8, 12

[11] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *ICLR*, 2015. 2

[12] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE TPAMI*, 2017. 2

[13] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv:1706.05587*, 2017. 2

[14] Liang-Chieh Chen, Huiyu Wang, and Siyuan Qiao. Scaling wide residual networks for panoptic segmentation. *arXiv:2011.11675*, 2020. 5

[15] Liang-Chieh Chen, Yi Yang, Jiang Wang, Wei Xu, and Alan L Yuille. Attention to scale: Scale-aware semantic image segmentation. In *CVPR*, 2016. 2

[16] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018. 2

[17] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. Aˆ 2-nets: Double attention networks. In *NeurIPS*, 2018. 2

[18] Bowen Cheng, Maxwell D Collins, Yukun Zhu, Ting Liu, Thomas S Huang, Hartwig Adam, and Liang-Chieh Chen. Panoptic-DeepLab. In *ICCV COCO + Mapillary Joint Recognition Challenge Workshop*, 2019. 1

[19] Bowen Cheng, Maxwell D Collins, Yukun Zhu, Ting Liu, Thomas S Huang, Hartwig Adam, and Liang-Chieh Chen. Panoptic-DeepLab: A Simple, Strong, and Fast Baseline for Bottom-Up Panoptic Segmentation. In *CVPR*, 2020. 2, 7

[20] Bowen Cheng, Alexander G Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. In *NeurIPS*, 2021. 3, 5, 6, 7, 8, 12, 13

[21] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. In *EMNLP*, 2016. 2

[22] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv:1904.10509*, 2019. 2

[23] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *CVPR*, 2017. 7

[24] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017. 7, 13

[25] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. In *ACL*, 2019. 2

[26] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019. 2

[27] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv:2010.11929*, 2020. 2

[28] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *CVPR*, 2019. 2

[29] Naiyu Gao, Yanhu Shan, Yupei Wang, Xin Zhao, Yinan Yu, Ming Yang, and Kaiqi Huang. Ssap: Single-shot instance segmentation with affinity pyramid. In *ICCV*, 2019. 2

[30] Ankit Gupta and Jonathan Berant. Gmat: Global memory augmentation for transformers. *arXiv:2006.03274*, 2020. 2

[31] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation. In *ECCV*, 2014. 1

[32] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 1, 2

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4, 5, 6

[34] Xuming He, Richard S Zemel, and Miguel Á Carreira-Perpiñán. Multiscale conditional random fields for image labeling. In *CVPR*, 2004. 1

[35] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 5, 6

[36] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. *arXiv:1912.12180*, 2019. 2

[37] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *CVPR*, 2018. 2

[38] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *ICCV*, 2019. 2

[39] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *ECCV*, 2016. 6

[40] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *ICCV*, 2019. 2

[41] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. Dynamic filter networks. In *NeurIPS*, 2016. 3

[42] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, 2018. 2

[43] Margret Keuper, Evgeny Levinkov, Nicolas Bonneel, Guillaume Lavoué, Thomas Brox, and Bjorn Andres. Efficient decomposition of image and mesh graphs by lifted multicuts. In *ICCV*, 2015. 2

[44] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *NeurIPS*, 2020. 6

[45] Dahun Kim, Jun Xie, Huiyu Wang, Siyuan Qiao, Qihang Yu, Hong-Seok Kim, Hartwig Adam, In So Kweon, and Liang-Chieh Chen. TubeFormer-DeepLab: Video Mask Transformer. In *CVPR*, 2022. 2

[46] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 6

[47] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, 2019. 1, 2, 7

[48] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *ICLR*, 2020. 2

[49] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006. 2, 5

[50] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[51] Bastian Leibe, Ales Leonardis, and Bernt Schiele. Combined object categorization and segmentation with an implicit shape model. In *Workshop on statistical learning in computer vision, ECCV*, 2004. 2

[52] Jie Li, Allan Raventos, Arjun Bhargava, Takaaki Tagawa, and Adrien Gaidon. Learning to fuse things and stuff. *arXiv:1812.01192*, 2018. 2

[53] Qizhu Li, Xiaojuan Qi, and Philip HS Torr. Unifying training and inference for panoptic segmentation. In *CVPR*, 2020. 2

[54] Yanwei Li, Xinze Chen, Zheng Zhu, Lingxi Xie, Guan Huang, Dalong Du, and Xingang Wang. Attention-guided unified network for panoptic segmentation. In *CVPR*, 2019. 2

[55] Yingwei Li, Xiaojie Jin, Jieru Mei, Xiaochen Lian, Linjie Yang, Cihang Xie, Qihang Yu, Yuyin Zhou, Song Bai, and Alan Yuille. Neural architecture search for lightweight non-local networks. In *CVPR*, 2020. 2

[56] Yanwei Li, Hengshuang Zhao, Xiaojuan Qi, Liwei Wang, Zeming Li, Jian Sun, and Jiaya Jia. Fully convolutional networks for panoptic segmentation. In *CVPR*, 2021. 2

[57] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 2, 6, 13

[58] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 2, 14

[59] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. In *NeurIPS*, 2018. 2, 5, 7

[60] Yudong Liu, Yongtao Wang, Siwei Wang, TingTing Liang, Qijie Zhao, Zhi Tang, and Haibin Ling. Cbnet: A novel composite backbone network architecture for object detection. In *AAAI*, 2020. 6

[61] Yiding Liu, Siyu Yang, Bin Li, Wengang Zhou, Jizheng Xu, Houqiang Li, and Yan Lu. Affinity derivation and graph merge for instance segmentation. In *ECCV*, 2018. 2

[62] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 2, 5, 6, 7, 12, 13

[63] Huanyu Liu1, Chao Peng, Changqian Yu, Jingbo Wang, Xu Liu, Gang Yu, and Wei Jiang. An end-to-end network for panoptic segmentation. In *CVPR*, 2019. 2

[64] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982. 2, 4

[65] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1

[66] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *EMNLP*, 2015. 2

[67] Davy Neven, Bert De Brabandere, Marc Proesmans, and Luc Van Gool. Instance segmentation by jointly optimizing spatial embeddings and clustering bandwidth. In *CVPR*, 2019. 2

[68] Dim P Papadopoulos, Jasper RR Uijlings, Frank Keller, and Vittorio Ferrari. Extreme clicking for efficient object annotation. In *ICCV*, 2017. 5

[69] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, 2018. 2

[70] Lorenzo Porzi, Samuel Rota Bulò, Aleksander Colovic, and Peter Kontschieder. Seamless scene segmentation. In *CVPR*, 2019. 2

[71] Siyuan Qiao, Liang-Chieh Chen, and Alan Yuille. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. *arXiv:2006.02334*, 2020. 2, 6, 7, 12

[72] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. In *NeurIPS*, 2019. 2

[73] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1

[74] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 115:211–252, 2015. 6, 14

[75] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *NAACL*, 2018. 2

[76] Zhuoran Shen, Mingyuan Zhang, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Efficient attention: Attention with linear complexities. In *WACV*, 2021. 2

[77] Konstantin Sofiiuk, Olga Barinova, and Anton Konushin. Adaptis: Adaptive instance selection network. In *ICCV*, 2019. 2

[78] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *ICCV*, 2021. 3

[79] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. In *ECCV*, 2020. 3

[80] Jonas Uhrig, Eike Rehder, Björn Fröhlich, Uwe Franke, and Thomas Brox. Box2pix: Single-shot instance segmentation by assigning pixels to object boxes. In *IEEE Intelligent Vehicles Symposium (IV)*, 2018. 2

[81] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1, 2

[82] Luc Vincent and Pierre Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE TPAMI*, 1991. 2

[83] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Max-deeplab: End-to-end panoptic segmentation with mask transformers. In *CVPR*, 2021. 1, 2, 3, 5, 6, 7, 8, 12, 13, 17

[84] Huiyu Wang, Yukun Zhu, Bradley Green, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Axial-DeepLab: Stand-Alone Axial-Attention for Panoptic Segmentation. In *ECCV*, 2020. 2, 5, 6, 7, 12, 13

[85] Sinong Wang, Belinda Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv:2006.04768*, 2020. 2

[86] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 2

[87] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. SOLOv2: Dynamic and fast instance segmentation. In *NeurIPS*, 2020. 3

[88] Mark Weber, Huiyu Wang, Siyuan Qiao, Jun Xie, Maxwell D. Collins, Yukun Zhu, Liangzhe Yuan, Dahun Kim, Qihang Yu, Daniel Cremers, Laura Leal-Taixe, Alan L. Yuille, Florian Schroff, Hartwig Adam, and Liang-Chieh Chen. DeepLab2: A TensorFlow Library for Deep Labeling. *arXiv: 2106.09748*, 2021. 6

[89] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 7

[90] Yuwen Xiong, Renjie Liao, Hengshuang Zhao, Rui Hu, Min Bai, Ersin Yumer, and Raquel Urtasun. Upsnet: A unified panoptic segmentation network. In *CVPR*, 2019. 1, 2, 7

[91] Tien-Ju Yang, Maxwell D Collins, Yukun Zhu, Jyh-Jing Hwang, Ting Liu, Xiao Zhang, Vivienne Sze, George Papandreou, and Liang-Chieh Chen. Deeperlab: Single-shot image parser. *arXiv:1902.05093*, 2019. 2

[92] Yibo Yang, Hongyang Li, Xia Li, Qijie Zhao, Jianlong Wu, and Zhouchen Lin. Sognet: Scene overlap graph network for panoptic segmentation. In *AAAI*, 2020. 2

[93] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. In *NeurIPS*, 2020. 2

[94] Wenwei Zhang, Jiangmiao Pang, Kai Chen, and Chen Change Loy. K-net: Towards unified image segmentation. In *NeurIPS*, 2021. 12, 13

[95] Hengshuang Zhao, Yi Zhang, Shu Liu, Jianping Shi, Chen Change Loy, Dahua Lin, and Jiaya Jia. Psanet: Point-wise spatial attention network for scene parsing. In *ECCV*, 2018. 2

[96] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, 2016. 12

[97] Song Chun Zhu and Alan Yuille. Region competition: Unifying snakes, region growing, and bayes/mdl for multiband image segmentation. *IEEE TPAMI*, 1996. 4

[98] Xizhou Zhu, Dazhi Cheng, Zheng Zhang, Stephen Lin, and Jifeng Dai. An empirical study of spatial attention mechanisms in deep networks. In *ICCV*, pages 6688–6697, 2019. 2

[99] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv:2010.04159*, 2020. 2, 5

[100] Zhen Zhu, Mengde Xu, Song Bai, Tengteng Huang, and Xiang Bai. Asymmetric non-local neural networks for semantic segmentation. In *CVPR*, 2019. 2

In the supplementary materials, we provide more technical details, along with more ablation and comparison results with other concurrent works. We also include more visualizations and comparisons over the baselines. Additionally, we provide a comprehensive comparison, in terms of training epochs, memory cost, parameters, FLOPs, and FPS, across different methods. We also report results with a ResNet-50 backbone for a fair comparison across different methods, along with additional results on Cityscapes. Finally, we summarize the limitations of our work and potential negative impacts.

6. More Technical Details

Backbones. In Fig. 6, we provide an architectural comparison of MaX-DeepLab-S [83] and CMT-DeepLab built upon Axial-R50/104 [84]. Specifically, we simplify the backbone from MaX-DeepLab-S [83] by removing transformer modules in the backbone (light blue), and stacking more blocks in the decoder module (light orange). The Axial-R104 backbone is obtained by scaling up Axial-R50 (*i.e.*, four times more layers in the stage-4).

Recursive Feature Network. We construct Recursive Feature Network (RFN) in a manner similar to [71]. More specifically, we stack two models together, with a skip-connection from the decoder features at stride 4 in the first network to the encoder features at stride 4 in the second network. Instead of using the complicated fusion module proposed in [71], we simply average the features for fusion. Moreover, the two networks share the same set of cluster centers (*i.e.*, object queries), which are sequentially updated from the first network to the second one. We also add supervision for the first network but use the Hungarian matching results based on the final output.

7. More Results and In-depth Analysis

Effect of frequent pixel update (our second solution). As discussed in the main paper, the clustering results will be also used to update pixel features besides cluster centers to ensure a frequent pixel update. We tried removing the pixel feature updates from clustering transformer, which leads to a degradation of 0.4% PQ.

Comparison with more concurrent works. Also shown in Tab. 4, we compare our CMT-DeepLab with the baseline MaX-DeepLab [83], and *concurrent* works MaskFormer [20] and K-Net [94] on the *test-dev* set. As shown in the table, our best model (using 200K iterations and RFN) attains the performance of 55.7% PQ on the test-dev set, which is 4.4% and 2.4% better than MaX-DeepLab-L [83] and MaskFormer [20]. Our best model is 0.5% PQ better than K-Net [94], which adopts a different framework (*i.e.*, dynamic kernels) than mask-transformer-based approaches. In addition to PQ, we further look into RQ and SQ for performance analysis. We observe that with a similar performance to K-Net [94] in RQ, our best model performs better in SQ. Specifically, our best model yields 83.6% SQ, which is 1.2%, 1.6%, and 1.1% better than K-Net, MaskFormer, and MaX-DeepLab-L, respectively. Interestingly, our lightweight variant, CMT-DeepLab with Axial-R50, achieves 83.0% SQ, which is still better than all the other methods. We attribute our better performance in SQ to the proposed clustering mask transformer layer, which yields denser attention maps to facilitate segmentation tasks.

Accuracy-cost Trade-off Comparison. We provide a comprehensive comparison of training cost (epochs, memory), model size (params, FLOPs, FPS), and performance (PQ) in Tab. 5. The training memory is measured on a TPU-v4, while other statistics are measured with a Tesla V100-SXM2 GPU. We use TensorFlow 2.7, cuda 11.0, input size 1200×800, and batch size 1. For MaskFormer (PyTorch-based), we cite the numbers from their paper. As shown in the table, our CMT-DeepLab-S (Axial-R50) outperforms MaskFormer-SwinB by 1.2% PQ with comparable model size and inference cost. Our CMT-DeepLab-S also outperforms MaskFormer-SwinL while using much fewer model parameters and running faster. All our models outperform MaX-DeepLab. Notably, our best model CMT-DeepLab-L-RFN (Axial-R104-RFN) outperforms MaX-DeepLab-L by 4.2% PQ while using only 60% model parameters and 33.6% FLOPs.

Backbone Differences. As different backbones are adopted for different methods (*e.g.*, MaX-S/L [83], Swin [62]), it hinders a direct and fair comparison across different methods. To this end, we provide results based on a ResNet-50 backbone across different models on COCO *val* set. As shown in Tab. 6, our CMT-DeepLab significantly outperforms MaX-DeepLab and concurrent works (MaskFormer and K-Net).

Results on Cityscapes. We provide additional results on Cityscapes in Tab. 7. For a fair comparison, we adopt the **same** setting, including pretrain weights (IN-1k), training hyper-parameters (e.g., iterations 60k, learning rate 3e-4, crop size 1025×2049), and post-processing scheme (pixel-wise argmax as in MaX-DeepLab). As shown in the table, our CMT-DeepLab-S significantly outperforms MaX-DeepLab-S by 2.9% PQ and 1.6% mIoU.

8. Visual Comparison

Visualization Details. To visually compare the clustering results/attention maps, we firstly follow DETR [10] to average values across multi-heads to obtain a single attention map, which is then transformed into a heatmap in a manner similar to CAM [96] by normalizing the values to the range [0, 255]. Note that we do not apply any smoothing techniques (*e.g.*, square root), which in fact adjust the learned attention values. These differences make the visualization

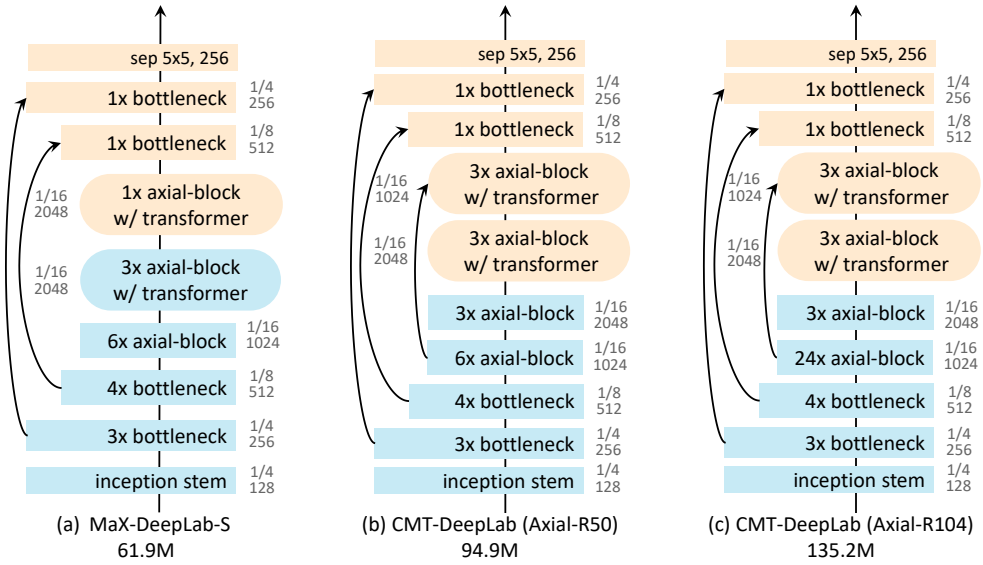

Figure 6. A visual comparison of architecture between MaX-DeepLab-S and CMT-DeepLab. Pretrained backbone part is labeled in blue color.

method	backbone	params	val-set PQ	val-set PQ^{Th}	val-set PQ^{St}	test-dev PQ	test-dev PQ^{Th}	test-dev PQ^{St}	test-dev SQ	test-dev RQ
MaX-DeepLab-S [83]	MaX-S [83]	61.9M	48.4	53.0	41.5	49.0	54.0	41.6	-	-
MaX-DeepLab-L [83]	MaX-L [83]	451M	51.1	57.0	42.2	51.3	57.2	42.4	82.5	61.3
MaskFormer† [20]	Swin-B‡ [62]	102M	51.8	56.9	44.1	-	-	-	-	-
MaskFormer† [20]	Swin-L‡ [62]	212M	52.7	58.5	44.0	53.3	59.1	44.5	82.0	64.1
K-Net† [94]	R101-FPN [57]	-	49.6	55.1	41.4	-	-	-	-	-
K-Net† [94]	R101-FPN-DCN [24]	-	48.3	54.0	39.7	-	-	-	-	-
K-Net† [94]	Swin-L‡ [62]	-	54.6	60.2	46.0	55.2	61.2	46.2	82.4	**66.1**
CMT-DeepLab	Axial-R50‡ [84]	94.9M	53.0	57.7	45.9	53.4	58.3	46.0	83.0	63.6
CMT-DeepLab	Axial-R104‡	135.2M	54.1	58.8	**47.1**	54.5	59.6	46.9	83.2	64.7
CMT-DeepLab	Axial-R104‡-RFN	270.3M	55.1	60.6	46.8	55.4	61.0	**47.0**	83.5	65.6
CMT-DeepLab (iter 200k)	Axial-R104‡-RFN	270.3M	**55.3**	**61.0**	46.6	**55.7**	**61.6**	46.8	**83.6**	65.9

Table 4. Results comparison on COCO val and test-dev set. ‡: ImageNet-22K pretraining. †: Concurrent works. We update comparison with concurrent works, and also our improved results with longer training iterations.

method	epochs	memory	params	FLOPs	FPS	PQ
MaskFormer-SwinB [20]	300	-	102M	411G	8.4	51.8
MaskFormer-SwinL [20]	300	-	212M	792G	5.2	52.7
MaX-DeepLab-S [83]	216	6.3G	62M	291G	11.9	48.4
MaX-DeepLab-L [83]	216	28.7G	451M	3317G	2.2	51.1
CMT-DeepLab-S	54	10.2G	95M	396G	8.1	53.0
CMT-DeepLab-L	54	11.8G	135M	553G	6.0	54.1
CMT-DeepLab-L-RFN	54	25.8G	270M	1114G	3.2	55.1
CMT-DeepLab-L-RFN	108	25.8G	270M	1114G	3.2	55.3

Table 5. A comprehensive accuracy-cost trade-off comparison.

	MaskFormer [20]	K-Net [94]	MaX-DeepLab [83]	CMT-DeepLab
PQ	46.5	47.1	46.0	**48.5**

Table 6. Results comparison with ResNet-50 as the backbone.

method	PQ	RQ	SQ	mIoU
MaX-DeepLab-S	61.7	74.5	81.5	79.8
CMT-DeepLab-S	64.6	77.4	82.6	81.4

Table 7. Cityscapes *val* set results.

differ from those in the paper of MaX-DeepLab [83]. All visualizations are done with CMT-DeepLab based on Axial-R50, and MaX-DeepLab-S, with input size 641×641.

Clustering results. In Fig. 7, Fig. 8, Fig. 9, and Fig. 10, we provide more clustering visualization results. We observe the same trend as we presented in the main paper that the clustering results, providing denser attention maps, are close-to-random at the beginning and are gradually refined to focus on different objects. Interestingly, we also observe some exceptions (see Fig. 7, Fig. 8, Fig. 9), where the clus-

tering results start with a good semantic-level clustering, indicating that some cluster centers can embed semantic information and thus specialize in some classes.

Attention map comparison with MaX-DeepLab. In Fig. 11, Fig. 12, and Fig. 13, we show more attention map comparison with MaX-DeepLab. As shown in those figures, CMT-DeepLab provides a much denser attention map than MaX-DeepLab.

9. Limitations

Motivated from a clustering perspective, CMT-DeepLab generates denser attention maps and thus leads to a superior performance in the segmentation task. However, the proposed clustering mask transformer, though significantly improves the segmentation quality (SQ), does not bring the same-level performance boost on the recognition ability (RQ). Specifically, we have adopted some simple scaling-up strategies, including increasing model size, input size, or training iterations. Those strategies result in a large performance gain in RQ as a compensation, but with a cost at parameters, computation, or training time. It thus remains an interesting problem to explore in the future that how to improve its recognition ability efficiently and effectively.

10. Potential Negative Impacts

In this paper, we present a new panoptic segmentation framework, inspired by the traditional clustering-based algorithm, generates denser attention maps and further achieves new state-of-the-art performance. The findings described in this paper can potentially help advance the research in developing stronger, faster, and more elegant end-to-end segmentation methods. However, we also note that there is a long-lasting debate on the impacts of AI on human world. As a method improving the fundamental task in computer vision, our work also advances the development of AI, which means there could be both beneficial and harmful influences depending on the users.

License of used assets. COCO dataset [58]: CC-by 4.0. ImageNet [74]: https://image-net.org/download.php.

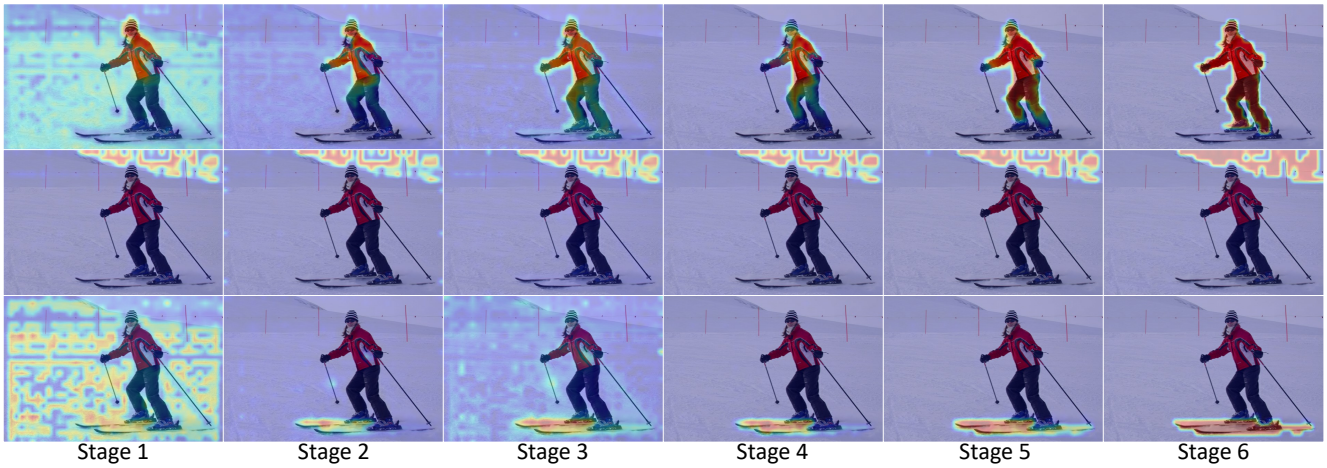

Figure 7. Visualization of clustering results at different stages (*i.e.*, transformer layers). We note that clustering results for person (row 1) and skis (row 3) start from a close-to-random distribution at the beginning and are gradually refined to focus on corresponding target. But we also find some cluster centers, *e.g.*, sky in row 2, are specialized in some semantic classes and start at a good semantic clustering.

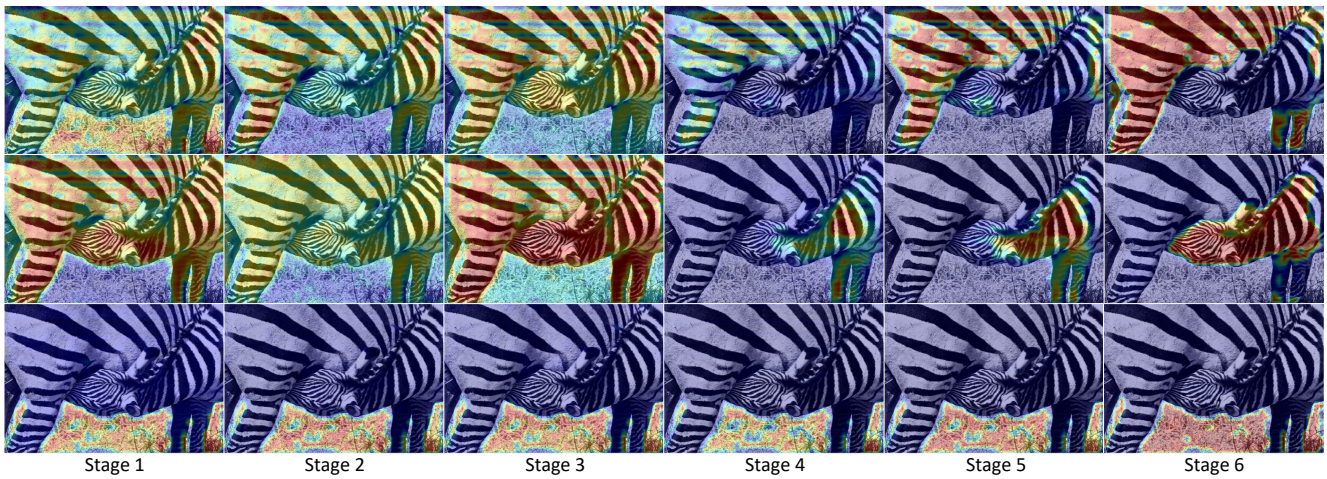

Figure 8. Visualization of clustering results at different stages (*i.e.*, transformer layers). Both row 1 and 2 experience a semantic-to-instance refinement during the clustering process (*e.g.*, in col 3, both clustering results capture all zebras.), which finally falls onto corresponding zebra. The cluster center on row 3 initializes with a good clustering result for grass, which coincides with the observation that some cluster centers intrinsically embed semantic information.

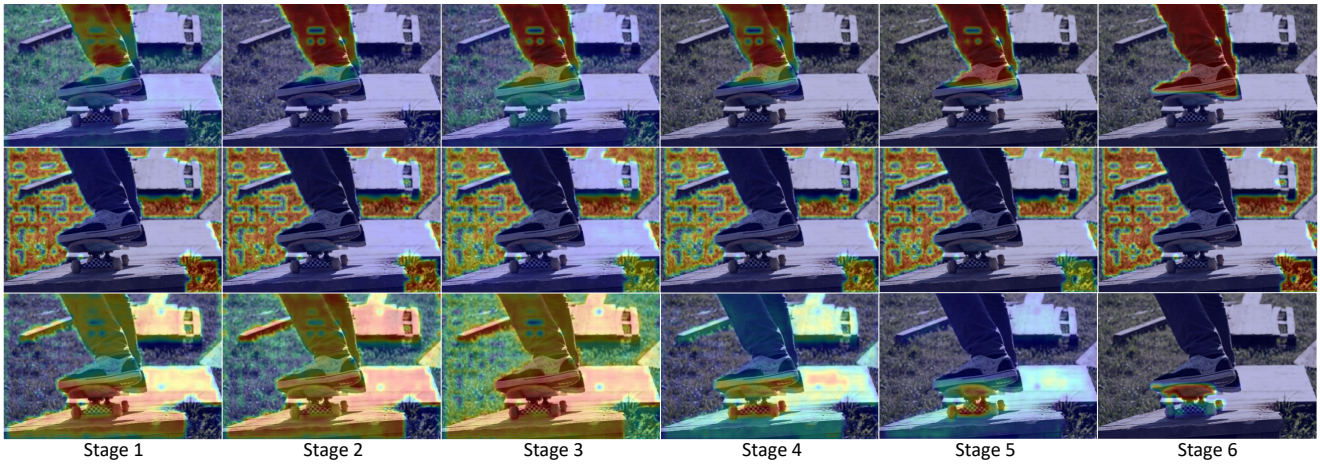

Figure 9. Visualization of clustering results at different stages (*i.e.*, transformer layers). Row 1, 3 gradually falls into the target person and skateboard, while row 2 starts with a good clustering for grass.

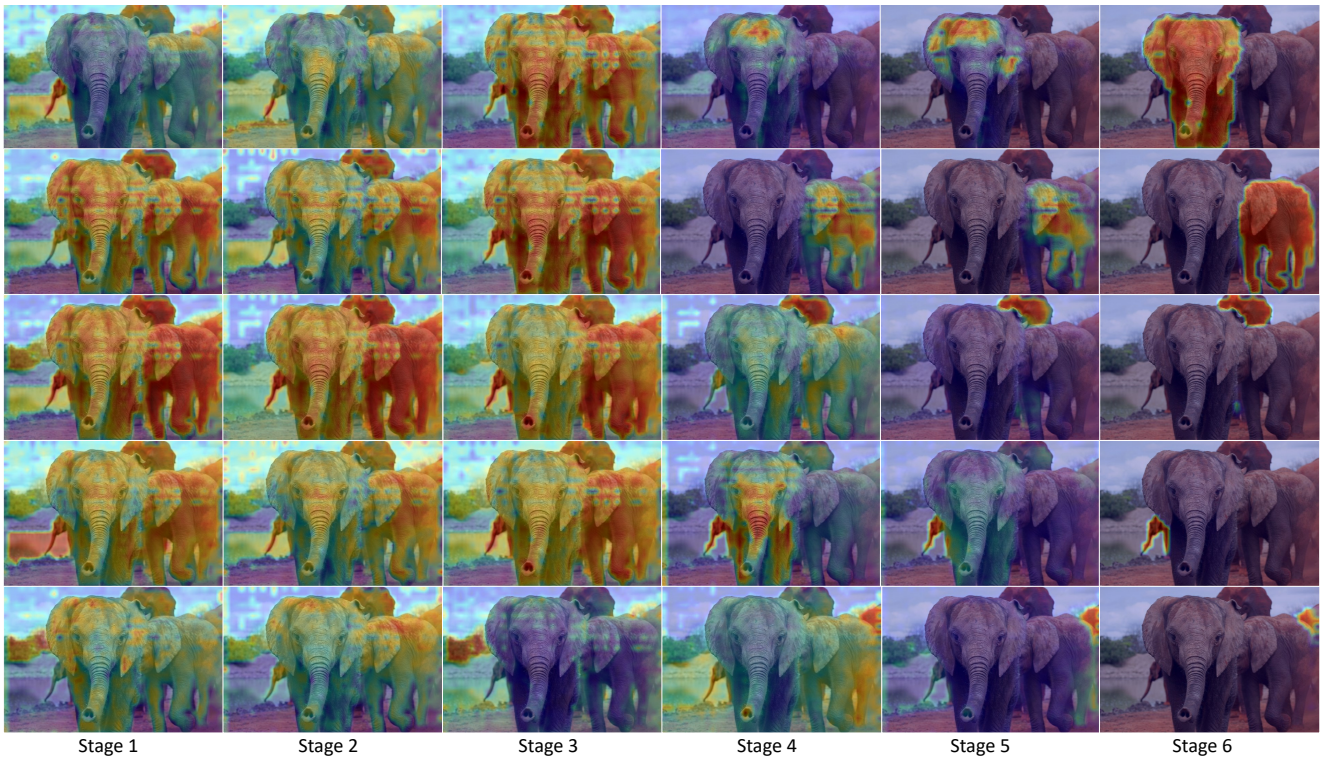

Figure 10. Visualization of clustering results at different stages (*i.e.*, transformer layers). Each row corresponds to an elephant instance prediction. Similarly, most results start from a close-to-random clustering and gradually converge to the target in a semantic-to-instance manner.

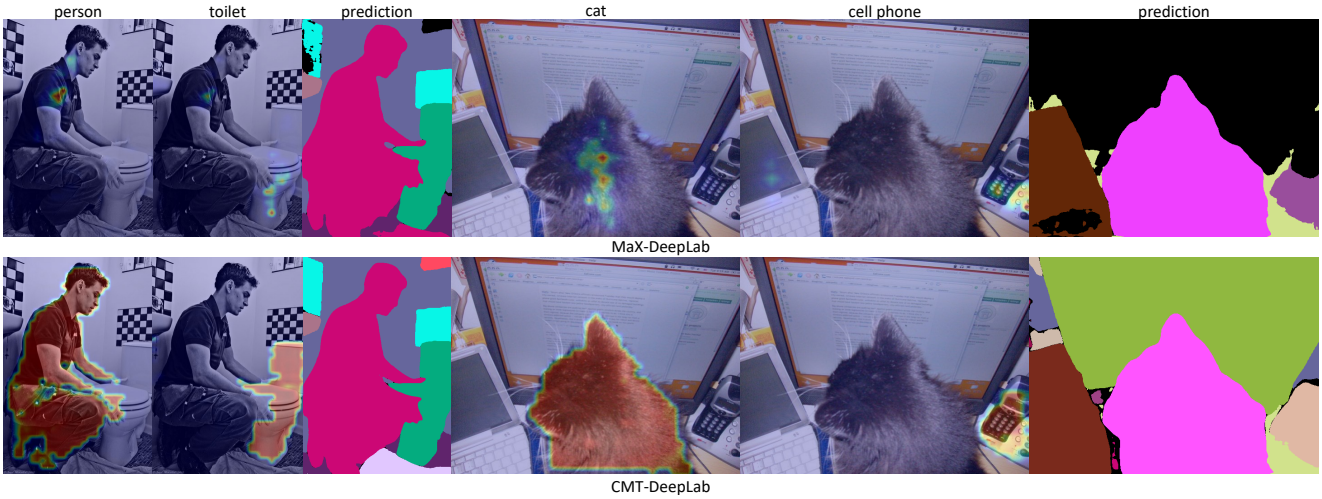

Figure 11. Visual comparison between CMT-DeepLab and MaX-DeepLab [83]. CMT-DeepLab provides a denser attention map to update cluster centers, which leads to superior performance in dense prediction tasks.

Figure 12. Visual comparison between CMT-DeepLab and MaX-DeepLab [83]. CMT-DeepLab provides a denser attention map to update cluster centers, which leads to superior performance in dense prediction tasks.

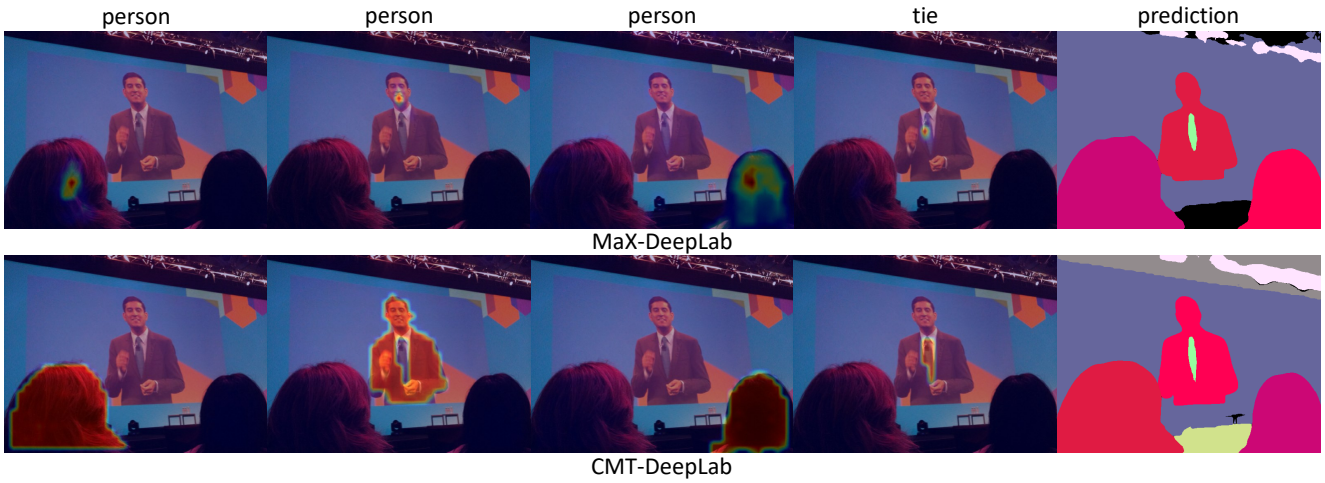

Figure 13. Visual comparison between CMT-DeepLab and MaX-DeepLab [83]. CMT-DeepLab provides a denser attention map to update cluster centers, which leads to superior performance in dense prediction tasks.