MaX-DeepLab: End-to-End Panoptic Segmentation with Mask Transformers

Huiyu Wang[1]* Yukun Zhu[2] Hartwig Adam[2] Alan Yuille[1] Liang-Chieh Chen[2]
[1]Johns Hopkins University [2]Google Research

Abstract

We present MaX-DeepLab, the first end-to-end model for panoptic segmentation. Our approach simplifies the current pipeline that depends heavily on surrogate sub-tasks and hand-designed components, such as box detection, non-maximum suppression, thing-stuff merging, etc. *Although these sub-tasks are tackled by area experts, they fail to comprehensively solve the target task. By contrast, our MaX-DeepLab directly predicts class-labeled masks with a mask transformer, and is trained with a panoptic quality inspired loss via bipartite matching. Our mask transformer employs a dual-path architecture that introduces a global memory path in addition to a CNN path, allowing direct communication with any CNN layers. As a result, MaX-DeepLab shows a significant 7.1% PQ gain in the box-free regime on the challenging COCO dataset, closing the gap between box-based and box-free methods for the first time. A small variant of MaX-DeepLab improves 3.0% PQ over DETR with similar parameters and M-Adds. Furthermore, MaX-DeepLab, without test time augmentation, achieves new state-of-the-art 51.3% PQ on COCO test-dev set.*

1. Introduction

The goal of panoptic segmentation [48] is to predict a set of non-overlapping masks along with their corresponding class labels. Modern panoptic segmentation methods address this mask prediction problem by approximating the target task with multiple surrogate sub-tasks. For example, Panoptic-FPN [47] adopts a 'box-based pipeline' with three levels of surrogate sub-tasks, as demonstrated in a tree structure in Fig. 1. Each level of this proxy tree involves manually-designed modules, such as anchors [77], box assignment rules [105], non-maximum suppression (NMS) [7], thing-stuff merging [98], *etc*. Although there are good solutions [77, 12, 33] to individual surrogate sub-tasks and modules, undesired artifacts are introduced when these sub-tasks fit into a pipeline for panoptic segmentation, especially in the challenging conditions (Fig. 2).

*Work done while an intern at Google.

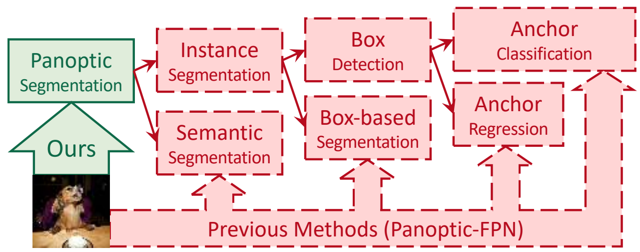
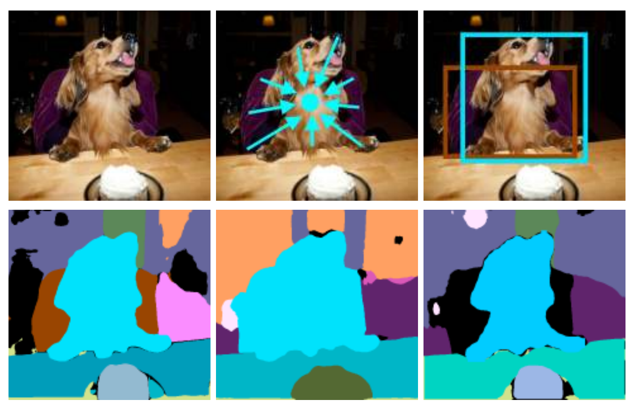


Figure 1. Our method predicts panoptic segmentation masks **directly from images**, while previous methods (Panoptic-FPN as an example) rely on **a tree of surrogate sub-tasks**. Panoptic segmentation masks are obtained by merging semantic and instance segmentation results. Instance segmentation is further decomposed into box detection and box-based segmentation, while box detection is achieved by anchor regression and anchor classification.



(a) Our MaX-DeepLab 51.1 PQ (box-free) (b) Axial-DeepLab [89] 43.4 PQ (box-free) (c) DetectoRS [76] 48.6 PQ (box-based)

Figure 2. A case study for our method and state-of-the-art *box-free* and *box-based* methods. (a) Our end-to-end MaX-DeepLab correctly segments a dog sitting on a chair. (b) Axial-DeepLab [89] relies on a surrogate sub-task of regressing object center offsets [21]. It fails because the centers of the dog and the chair are close to each other. (c) DetectoRS [76] classifies object bounding boxes, instead of masks, as a surrogate sub-task. It filters out the chair mask because the chair bounding box has a low confidence.

Recent work on panoptic segmentation attempted to simplify this box-based pipeline. For example, UPSNet [98] proproses a parameter-free panoptic head, permitting back-propagation to both semantic and instance segmentation modules. Recently, DETR [10] presents an end-to-end approach for box detection, which is used to replace detectors

Method	Anchor -Free	Center -Free	NMS -Free	Merge -Free	Box -Free
Panoptic-FPN [47]	✗	✓	✗	✗	✗
UPSNet [98]	✗	✓	✗	✓	✗
DETR [10]	✓	✓	✓	✓	✗
Axial-DeepLab [89]	✓	✗	✗	✗	✓
MaX-DeepLab	✓	✓	✓	✓	✓

Table 1. Our end-to-end MaX-DeepLab dispenses with these common hand-designed components necessary for existing methods.

in panoptic segmentation, but the whole training process of DETR still relies heavily on the box detection task.

Another line of work made efforts to completely remove boxes from the pipeline, which aligns better with the mask-based definition of panoptic segmentation. The state-of-the-art method in this regime, Axial-DeepLab [89], along with other box-free methods [100, 21, 11], predicts pixel-wise offsets to pre-defined instance centers. But this center-based surrogate sub-task makes it challenging to deal with highly deformable objects, or near-by objects with close centers. As a result, box-free methods do not perform as well as box-based methods on the challenging COCO dataset [60].

In this paper, we streamline the panoptic segmentation pipeline with an end-to-end approach. Inspired by DETR [10], our model *directly* predicts a set of non-overlapping masks and their corresponding semantic labels with a mask transformer. The output masks and classes are optimized with a panoptic quality (PQ) style objective. Specifically, inspired by the definition of PQ [48], we define a similarity metric between two class-labeled masks as the multiplication of their mask similarity and their class similarity. Our model is trained by maximizing this similarity between ground truth masks and predicted masks via one-to-one bipartite matching [51, 82, 10]. This direct modeling of panoptic segmentation enables end-to-end training and inference, removing those hand-coded priors that are necessary in existing box-based and box-free methods (Tab. 1). Our method is dubbed MaX-DeepLab for extending Axial-DeepLab with a **Ma**sk **X**former.

In companion with direct training and inference, we equip our mask transformer with a novel architecture. Instead of stacking a traditional transformer [87, 10] on top of a Convolutional Neural Network (CNN) [52], we propose a dual-path framework for combining CNNs with transformers. Specifically, we enable any CNN layer to read and write a global memory, using our dual-path transformer block. This block supports all types of attention between the CNN-path and the memory-path, including memory-path self-attention (*M2M*), pixel-path axial self-attention (*P2P*), memory-to-pixel attention (*M2P*), and finally pixel-to-memory attention (*P2M*). The transformer block can be inserted anywhere in a CNN, enabling communication with the global memory at any layer. Besides this communication module, our MaX-DeepLab employs a stacked-hourglass-style decoder [78, 71, 19]. The decoder aggregates multi-scale features into a high resolution output, which is then multiplied with the global memory feature, to form our mask set prediction. The classes for the masks are predicted with another branch of the mask transformer.

We evaluate MaX-DeepLab on one of the most challenging panoptic segmentation datasets, COCO [60], against the state-of-the-art box-free method, Axial-DeepLab [89], and state-of-the-art box-based method, DetectoRS [93] (Fig. 2). Our MaX-DeepLab, *without* test time augmentation (TTA), achieves the state-of-the-art result of 51.3% PQ on the test-dev set. This result surpasses Axial-DeepLab (with TTA) by 7.1% PQ in the box-free regime, and outperforms DetectoRS (with TTA) by 1.7% PQ, bridging the gap between box-based and box-free methods for the first time. For a fair comparison with DETR [10], we also evaluate a lightweight model, MaX-DeepLab-S, that matches the number of parameters and M-Adds of DETR. We observe that MaX-DeepLab-S outperforms DETR by 3.3% PQ on the val set and 3.0% PQ on the test-dev set. In addition, we perform extensive ablation studies and analyses on our end-to-end formulation, model scaling, dual-path architectures, and our loss functions. We also notice that the extra-long training schedule of DETR [10] is not necessary for MaX-DeepLab.

To summarize, our contributions are four-fold:

- MaX-DeepLab is the first end-to-end model for panoptic segmentation, inferring masks and classes directly without hand-coded priors like object centers or boxes.
- We propose a training objective that optimizes a PQ-style loss function via a PQ-style bipartite matching between predicted masks and ground truth masks.
- Our dual-path transformer enables CNNs to read and write a global memory at any layer, providing a new way of combining transformers with CNNs.
- MaX-DeepLab closes the gap between box-based and box-free methods and sets a new state-of-the-art on COCO, even without using test time augmentation.

2. Related Work

Transformers. Transformers [87], first introduced for neural machine translation, have advanced the state-of-the-art in many natural language processing tasks [27, 79, 26]. Attention [2], as the core component of Transformers, was developed to capture both correspondence of tokens across modalities [2] and long-range interactions in a single context (self-attention) [22, 87]. Later, the complexity of transformer attention has been reduced [49, 90], by introducing local [68] or sparse attention [23], together with a global memory [6, 103, 31, 1]. The global memory, which inspires our dual-path transformer, recovers long-range context by propagating information globally.

Transformer and attention have been applied to computer vision as well, by combining non-local modules [91, 9] with CNNs or by applying self-attention only [72, 37, 89]. Both classes of methods have boosted various vision tasks such as image classification [18, 5, 72, 37, 57, 89, 28], object detection [91, 80, 72, 36, 10, 108], semantic segmentation [15, 106, 39, 29, 109, 107], video recognition [91, 18], image generation [73, 35], and panoptic segmentation [89]. It is worth mentioning that DETR [10] stacked a transformer on top of a CNN for end-to-end object detection.

Box-based panoptic segmentation. Most panoptic segmentation models, such as Panoptic FPN [47], follow a box-based approach that detects object bounding boxes and predicts a mask for each box, usually with a Mask R-CNN [33] and FPN [58]. Then, the instance segments ('thing') and semantic segments ('stuff') [13] are fused by merging modules [54, 56, 74, 67, 101] to generate panoptic segmentation. For example, UPSNet [98] developed a parameter-free panoptic head, which facilitates unified training and inference [55]. Recently, DETR [10] extended box-based methods with its transformer-based end-to-end detector. And DetectoRS [76] advanced the state-of-the-art with recursive feature pyramid and switchable atrous convolution.

Box-free panoptic segmentation. Contrary to box-based approaches, box-free methods typically start with semantic segments [12, 14, 16]. Then, instance segments are obtained by grouping 'thing' pixels with various methods, such as instance center regression [44, 86, 70, 100, 20], Watershed transform [88, 3, 8], Hough-voting [4, 53, 8], or pixel affinity [45, 66, 81, 30, 8]. Recently, Axial-DeepLab [89] advanced the state-of-the-art by equipping Panoptic-DeepLab [21] with a fully axial-attention [35] backbone. In this work, we extend Axial-DeepLab with a mask transformer for end-to-end panoptic segmentation.

3. Method

In this section, we describe how MaX-DeepLab directly predicts class-labeled masks for panoptic segmentation, followed by the PQ-style loss used to train the model. Then, we introduce our dual-path transformer architecture as well as the auxiliary losses that are helpful in training.

3.1. MaX-DeepLab formulation

The goal of panoptic segmentation is to segment the image $I \in \mathbb{R}^{H \times W \times 3}$ into a set of class-labeled 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.

Our MaX-DeepLab directly predicts outputs in the exact same form as the ground truth. MaX-DeepLab segments the image I into a fixed-size set of class-labeled masks:

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

The constant size N of the set is much larger than the typical number of masks in an image [10]. The predicted masks $\hat{m}_i \in [0,1]^{H \times W}$ are softly exclusive to each other, *i.e.*, $\sum_{i=1}^{N} \hat{m}_i = 1^{H \times W}$, and $\hat{p}_i(c)$ denotes the probability of assigning class c to mask $\hat{m}_i$. Possible classes $\mathbb{C} \ni c$ include thing classes, stuff classes, and a $\varnothing$ class (no object). In this way, MaX-DeepLab deals with thing and stuff classes in a unified manner, removing the need for merging operators.

Simple inference. End-to-end inference of MaX-DeepLab is enabled by adopting the same formulation for both ground truth definition and model prediction. As a result, the final panoptic segmentation prediction is obtained by simply performing argmax twice. Specifically, the first argmax predicts a class label for each mask:

$$\hat{c}_i = \arg\max_{c} \hat{p}_i(c) \, . \tag{3}$$

And the other argmax assigns a mask-ID $\hat{z}_{h,w}$ to each pixel:

$$\begin{gathered} \hat{z}_{h,w} = \arg\max_{i} \hat{m}_{i,h,w} \, , \\ \forall h \in \{1, 2, \ldots, H\}, \quad \forall w \in \{1, 2, \ldots, W\} \, . \end{gathered} \tag{4}$$

In practice, we filter each argmax with a confidence threshold – Masks or pixels with a low confidence are removed as described in Sec. 4. In this way, MaX-DeepLab infers panoptic segmentation directly, dispensing with common manually-designed post-processing, *e.g.*, NMS and thing-stuff merging in almost all previous methods [47, 98]. Besides, MaX-DeepLab does not rely on hand-crafted priors such as anchors, object boxes, or instance mass centers, *etc*.

3.2. PQ-style loss

In addition to simple inference, MaX-DeepLab enables end-to-end training as well. In this section, we introduce how we train MaX-DeepLab with our PQ-style loss, which draws inspiration from the definition of *panoptic quality* (PQ) [48]. This evaluation metric of panoptic segmentation, PQ, is defined as the multiplication of a *recognition quality* (RQ) term and a *segmentation quality* (SQ) term:

$$\text{PQ} = \text{RQ} \times \text{SQ} \, . \tag{5}$$

Based on this decomposition of PQ, we design our objective in the same manner: First, we define a PQ-style similarity metric between a class-labeled ground truth mask and a predicted mask. Next, we show how we match a predicted mask to each ground truth mask with this metric, and finally how to optimize our model with the same metric.

Mask similarity metric. Our mask similarity metric $\text{sim}(\cdot, \cdot)$ between a class-labeled ground truth mask $y_i = (m_i, c_i)$ and a prediction $\hat{y}_j = (\hat{m}_j, \hat{p}_j(c))$ is defined as

$$\text{sim}(y_i, \hat{y}_j) = \underbrace{\hat{p}_j(c_i)}_{\approx \text{RQ}} \times \underbrace{\text{Dice}(m_i, \hat{m}_j)}_{\approx \text{SQ}} \, , \tag{6}$$

where $\hat{p}_j(c_i) \in [0,1]$ is the probability of predicting the correct class (recognition quality) and $\mathrm{Dice}(m_i, \hat{m}_j) \in [0,1]$ is the Dice coefficient between a predicted mask $\hat{m}_j$ and a ground truth m_i (segmentation quality). The two terms are multiplied together, analogous to the decomposition of PQ.

This mask similarity metric has a lower bound of 0, which means either the class prediction is incorrect, OR the two masks do not overlap with each other. The upper bound, 1, however, is only achieved when the class prediction is correct AND the mask is perfect. The AND gating enables this metric to serve as a good optimization objective for both model training and mask matching.

Mask matching. In order to assign a predicted mask to each ground truth, we solve a one-to-one bipartite matching problem between the prediction set $\{\hat{y}_i\}_{i=1}^N$ and the ground truth set $\{y_i\}_{i=1}^K$. Formally, we search for a permutation of N elements $\sigma \in \mathfrak{S}_N$ that best assigns the predictions to achieve the maximum total similarity to the ground truth:

$$\hat{\sigma} = \underset{\sigma \in \mathfrak{S}_N}{\arg\max} \sum_{i=1}^{K} \mathrm{sim}(y_i, \hat{y}_{\sigma(i)}) \,. \tag{7}$$

The optimal assignment is computed efficiently with the Hungarian algorithm [51], following prior work [10, 82]. We refer to the K matched predictions as positive masks which will be optimized to predict the corresponding ground truth masks and classes. The $(N-K)$ masks left are negatives, which should predict the $\varnothing$ class (no object).

Our one-to-one matching is similar to DETR [10], but with a different purpose: DETR allows only one positive match in order to remove duplicated boxes in the absence of NMS, while in our case, duplicated or overlapping masks are precluded by design. But in our case, assigning multiple predicted masks to one ground truth mask is problematic too, because multiple masks cannot possibly be optimized to fit a single ground truth mask at the same time. In addition, our one-to-one matching is consistent with the PQ metric, where only one predicted mask can theoretically match (*i.e.*, have an IoU over 0.5) with each ground truth mask.

PQ-style loss. Given our mask similarity metric and the mask matching process based on this metric, it is straight forward to optimize model parameters θ by maximizing this same similarity metric over matched (*i.e.*, positive) masks:

$$\max_{\theta} \sum_{i=1}^{K} \mathrm{sim}(y_i, \hat{y}_{\hat{\sigma}(i)}) \Leftrightarrow \max_{\theta, \sigma \in \mathfrak{S}_N} \sum_{i=1}^{K} \mathrm{sim}(y_i, \hat{y}_{\sigma(i)}) \,. \tag{8}$$

Substituting the similarity metric (Equ. (6)) gives our PQ-style objective $\mathcal{O}_{\mathrm{PQ}}^{\mathrm{pos}}$ to be maximized for positive masks:

$$\max_{\theta} \mathcal{O}_{\mathrm{PQ}}^{\mathrm{pos}} = \sum_{i=1}^{K} \underbrace{\hat{p}_{\hat{\sigma}(i)}(c_i)}_{\approx \text{ RQ}} \times \underbrace{\mathrm{Dice}(m_i, \hat{m}_{\hat{\sigma}(i)})}_{\approx \text{ SQ}} \,. \tag{9}$$

In practice, we rewrite $\mathcal{O}_{\mathrm{PQ}}^{\mathrm{pos}}$ into two common loss terms by applying the product rule of gradient and then changing a probability $\hat{p}$ to a log probability $\log \hat{p}$. The change from $\hat{p}$ to $\log \hat{p}$ aligns with the common cross-entropy loss and scales gradients better in practice for optimization:

$$\begin{aligned} \mathcal{L}_{\mathrm{PQ}}^{\mathrm{pos}} = & \sum_{i=1}^{K} \underbrace{\hat{p}_{\hat{\sigma}(i)}(c_i)}_{\text{weight}} \cdot \underbrace{\big[-\mathrm{Dice}(m_i, \hat{m}_{\hat{\sigma}(i)})\big]}_{\text{Dice loss}} \\ & + \sum_{i=1}^{K} \underbrace{\mathrm{Dice}(m_i, \hat{m}_{\hat{\sigma}(i)})}_{\text{weight}} \cdot \underbrace{\big[-\log \hat{p}_{\hat{\sigma}(i)}(c_i)\big]}_{\text{Cross-entropy loss}} \,, \end{aligned} \tag{10}$$

where the loss weights are constants (*i.e.*, no gradient is passed to them). This reformulation provides insights by bridging our objective with common loss functions: Our PQ-style loss is equivalent to optimizing a dice loss weighted by the class correctness and optimizing a cross-entropy loss weighted by the mask correctness. The logic behind this loss is intuitive: we want *both* of the mask and class to be correct at the same time. For example, if a mask is far off the target, it is a false negative anyway, so we disregard its class. This intuition aligns with the down-weighting of class losses for wrong masks, and vice versa.

Apart from the $\mathcal{L}_{\mathrm{PQ}}^{\mathrm{pos}}$ for positive masks, we define a cross-entropy term $\mathcal{L}_{\mathrm{PQ}}^{\mathrm{neg}}$ for negative (unmatched) masks:

$$\mathcal{L}_{\mathrm{PQ}}^{\mathrm{neg}} = \sum_{i=K+1}^{N} \big[-\log \hat{p}_{\hat{\sigma}(i)}(\varnothing)\big] \,. \tag{11}$$

This term trains the model to predict $\varnothing$ for negative masks. We balance the two terms by α, as a common practice to weight positive and negative samples [59]:

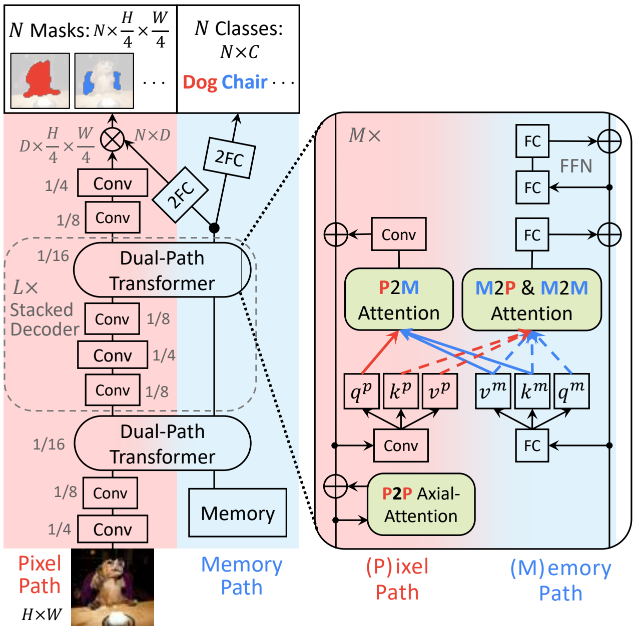
$$\mathcal{L}_{\mathrm{PQ}} = \alpha \mathcal{L}_{\mathrm{PQ}}^{\mathrm{pos}} + (1-\alpha) \mathcal{L}_{\mathrm{PQ}}^{\mathrm{neg}} \,, \tag{12}$$

where $\mathcal{L}_{\mathrm{PQ}}$ denotes our final PQ-style loss.

3.3. MaX-DeepLab Architecture

As shown in Fig. 3, MaX-DeepLab architecture includes a dual-path transformer, a stacked decoder, and output heads that predict the masks and classes.

Dual-path transformer. Instead of stacking a transformer on top of a CNN [10], we integrate the transformer and the CNN in a dual-path fashion, with bidirectional communication between the two paths. Specifically, we augment a 2D *pixel*-based CNN with a 1D global *memory* of size N (*i.e.*, the total number of predictions) and propose a transformer block as a drop-in replacement for any CNN block or an add-on for a pretrained CNN block. Our transformer block enables all four possible types of communication between the 2D pixel-path CNN and the 1D memory-path: (1) the traditional memory-to-pixel (*M2P*) attention, (2) memory-to-memory (*M2M*) self-attention, (3) pixel-to-memory (*P2M*) feedback attention that allows pixels to



(a) Overview of MaX-DeepLab (b) Dual-path transformer block

Figure 3. (a) An image and a global memory are fed into a dual-path transformer, which directly predicts a set of masks and classes (residual connections omitted). (b) A dual-path transformer block is equipped with all 4 types of attention between the two paths.

read from the memory, and (4) pixel-to-pixel (*P2P*) self-attention, implemented as axial-attention blocks [39, 35, 89]. We select axial-attention [89] rather than global 2D attention [10, 91, 5] for efficiency on high resolution feature maps. One could optionally approximate the pixel-to-pixel self-attention with a convolutional block that only allows local communication. This transformer design with a memory path besides the main CNN path is termed dual-path transformer. Unlike previous work [10], it allows transformer blocks to be inserted anywhere in the backbone at any resolution. In addition, the *P2M* feedback attention enables the pixel-path CNN to refine its feature given the memory-path features that encode mask information.

Formally, given a 2D input feature $x^p \in \mathbb{R}^{\hat{H}\times\hat{W}\times d_{in}}$ with height $\hat{H}$, width $\hat{W}$, channels d_{in}, and a 1D global memory feature $x^m \in \mathbb{R}^{N\times d_{in}}$ with length N (*i.e.*, the size of the prediction set). We compute pixel-path queries q^p, keys k^p, and values v^p, by learnable linear projections of the pixel-path feature map x^p at each pixel. Similarly, q^m, k^m, v^m are computed from x^m with another set of projection matrices. The query (key) and value channels are d_q and d_v, for both paths. Then, the output of feedback attention (*P2M*), $y^p_a \in \mathbb{R}^{d_{out}}$, at pixel position a, is computed as

$$y^p_a = \sum_{n=1}^{N} \text{softmax}_n \left(q^p_a \cdot k^m_n\right) v^m_n \,, \tag{13}$$

where the softmax$_n$ denotes a softmax function applied to the whole memory of length N. Similarly, the output of memory-to-pixel (*M2P*) and memory-to-memory (*M2M*) attention $y^m_b \in \mathbb{R}^{d_{out}}$, at memory position b, is

$$y^m_b = \sum_{n=1}^{\hat{H}\hat{W}+N} \text{softmax}_n \left(q^m_b \cdot k^{pm}_n\right) v^{pm}_n \,, \qquad k^{pm} = \begin{bmatrix} k^p \\ k^m \end{bmatrix} , \quad v^{pm} = \begin{bmatrix} v^p \\ v^m \end{bmatrix} , \tag{14}$$

where a single softmax is performed over the concatenated dimension of size $(\hat{H}\hat{W}, +N)$, inspired by ETC [1].

Stacked decoder. Unlike previous work [21, 89] that uses a light-weight decoder, we explore stronger hourglass-style stacked decoders [78, 71, 19]. As shown in Fig. 3, our decoder is stacked L times, traversing output strides (4, 8, and 16 [16, 61]) multiple times. At each decoding resolution, features are fused by simple summation after bilinear resizing. Then, convolutional blocks or transformer blocks are applied, before the decoder feature is sent to the next resolution. This stacked decoder is similar to feature pyramid networks [58, 63, 84, 76] designed for pyramidal anchor predictions [64], but our purpose here is only to aggregate multi-scale features, *i.e.*, intermediate pyramidal features are not directly used for prediction.

Output heads. From the memory feature of length N, we predict mask classes $\hat{p}(c) \in \mathbb{R}^{N\times|\mathbb{C}|}$ with two fully-connected layers (2FC) and a softmax. Another 2FC head predicts mask feature $f \in \mathbb{R}^{N\times D}$. Similarly, we employ two convolutions (2Conv) to produce a normalized feature $g \in \mathbb{R}^{D\times\frac{H}{4}\times\frac{W}{4}}$ from the decoder output at stride 4. Then, our mask prediction $\hat{m}$ is simply the multiplication of transformer feature f and decoder feature g:

$$\hat{m} = \text{softmax}_N \left(f \cdot g\right) \in \mathbb{R}^{N\times\frac{H}{4}\times\frac{W}{4}} \,. \tag{15}$$

In practice, we use batch norm [41] on f and $(f \cdot g)$ to avoid deliberate initialization, and we bilinear upsample the mask prediction $\hat{m}$ to the original image resolution. Finally, the combination $\{(\hat{m}_i, \hat{p}_i(c))\}_{i=1}^N$ is our mask transformer output to generate panoptic results as introduced in Sec. 3.1.

Our mask prediction head is inspired by CondInst [85] and SOLOv2 [92], which extend dynamic convolution [43, 99] to instance segmentation. However, unlike our end-to-end method, these methods require hand-designed object centers and assignment rules for instance segmentation, and a thing-stuff merging module for panoptic segmentation.

3.4. Auxiliary losses

In addition to the PQ-style loss (Sec. 3.2), we find it beneficial to incorporate auxiliary losses in training. Specifically, we propose a pixel-wise instance discrimination loss that helps cluster decoder features into instances. We also use a per-pixel mask-ID cross-entropy loss that classifies each pixel into N masks, and a semantic segmentation loss.

Our total loss function thus consists of the PQ-style loss $\mathcal{L}_{\rm PQ}$ and these three auxiliary losses.

Instance discrimination. We use a per-pixel instance discrimination loss to help the learning of the feature map $g \in \mathbb{R}^{D\times\frac{H}{4}\times\frac{W}{4}}$. Given a downsampled ground truth mask $m_i \in \{0,1\}^{\frac{H}{4}\times\frac{W}{4}}$, we first compute a normalized feature embedding $t_{i,:} \in \mathbb{R}^D$ for each annotated mask by averaging the feature vectors $g_{:,h,w}$ inside the mask m_i:

$$t_{i,:} = \frac{\sum_{h,w} m_{i,h,w} \cdot g_{:,h,w}}{||\sum_{h,w} m_{i,h,w} \cdot g_{:,h,w}||}\,,\; i = 1, 2, \dots, K\,. \quad (16)$$

This gives us K instance embeddings $\{t_{i,:}\}_{i=1}^K$ representing K ground truth masks. Then, we let each pixel feature $g_{:,h,w}$ perform an instance discrimination task, *i.e.*, each pixel should correctly identify which mask embedding (out of K) it belongs to, as annotated by the ground truth masks. The contrastive loss at a pixel (h, w) is written as:

$$\mathcal{L}^{\rm InstDis}_{h,w} = -\log\frac{\sum_{i=1}^K m_{i,h,w}\exp\left(t_{i,:}\cdot g_{:,h,w}/\tau\right)}{\sum_{i=1}^K \exp\left(t_{i,:}\cdot g_{:,h,w}/\tau\right)}\,, \quad (17)$$

where τ denotes the temperature, and note that $m_{i,h,w}$ is non-zero only when pixel (h, w) belongs to the ground truth mask m_i. In practice, this per-pixel loss is applied to all instance pixels in an image, encouraging features from the same instance to be similar and features from different instances to be distinct, in a contrastive fashion, which is exactly the property required for instance segmentation.

Our instance discrimination loss is inspired by previous works [96, 94, 40, 17, 32, 46]. However, they discriminate instances either unsupervisedly or with image classes [46], whereas we perform a pixel-wise instance discrimination task, as annotated by panoptic segmentation ground truth.

Mask-ID cross-entropy. In Equ. (4), we describe how we infer the mask-ID map given our mask prediction. In fact, we can train this per-pixel classification task by applying a cross-entropy loss on it. This is consistent with the literature [42, 83, 10] that uses a cross-entropy loss together with a dice loss [69] to learn better segmentation masks.

Semantic segmentation. We also use an auxiliary semantic segmentation loss to help capture per pixel semantic feature. Specifically, we apply a semantic head [21] on top of the backbone if no stacked decoder is used (*i.e.*, $L = 0$). Otherwise, we connect the semantic head to the first decoder output at stride 4, because we find it helpful to separate the final mask feature g with semantic segmentation.

4. Experiments

We report our main results on COCO, comparing with state-of-the-art methods. Then, we provide a detailed ablation study on the architecture variants and losses. Finally, we analyze how MaX-DeepLab works with visualizations.

Technical details. Most of our default settings follow Axial-DeepLab [89]. Specifically, we train our models with 32 TPU cores for 100k (400k for main results) iterations (54 epochs), a batch size of 64, Radam [62] Lookahead [104], a 'poly' schedule learning rate of 10^{-3} (3×10^{-4} for MaX-DeepLab-L), a backbone learning rate multiplier of 0.1, a weight decay of 10^{-4}, and a drop path rate [38] of 0.2. We resize and pad images to 641 $\times$ 641 [21, 89] (1025 $\times$ 1025 for main results) for inference and M-Adds calculation. During inference, we set masks with class confidence below 0.7 to void and filter pixels with mask-ID confidence below 0.4. Finally, following previous work [98, 21, 89], we filter stuff masks with an area limit of 4096 pixels, and instance masks with a limit of 256 pixels. In training, we set our PQ-style loss weight (Equ. (12), normalized by N) to 3.0, with $\alpha = 0.75$. Our instance discrimination uses $\tau = 0.3$, and a weight of 1.0. We set the mask-ID cross-entropy weight to 0.3, and semantic segmentation weight to 1.0. We use an output size $N = 128$ and $D = 128$ channels. We fill the initial memory with learnable weights [10].

4.1. Main results

We present our main results on COCO *val* set and *test-dev* set [60], with a small model, MaX-DeepLab-S, and a large model, MaX-DeepLab-L.

MaX-DeepLab-S augments ResNet-50 [34] with axial-attention blocks [89] in the last two stages. After pretaining, we replace the last stage with dual-path transformer blocks and use an $L = 0$ (not stacked) decoder. We match parameters and M-Adds to DETR-R101 [10], for fair comparison.

MaX-DeepLab-L stacks an $L = 2$ decoder on top of Wide-ResNet-41 [102, 95, 11]. And we replace all stride 16 residual blocks by our dual-path transformer blocks with wide axial-attention blocks [89]. This large variant is meant to be compared with state-of-the-art results.

Val set. In Tab. 2, we report our validation set results and compare with both box-based and box-free panoptic segmentation methods. As shown in the table, our *single-scale* MaX-DeepLab-S already outperforms all other *box-free* methods by a large margin of more than 4.5 % PQ, no matter whether other methods use test time augmentation (TTA, usually flipping and multi-scale) or not. Specifically, it surpasses *single-scale* Panoptic-DeepLab by 8.7% PQ, and *single-scale* Axial-DeepLab by 5.0% PQ with similar M-Adds. We also compare MaX-DeepLab-S with DETR [10], which is based on an end-to-end detector, in a controlled environment of similar number of parameters and M-Adds. Our MaX-DeepLab-S outperforms DETR [10] by 3.3% PQ in this fair comparison. Next, we scale up MaX-DeepLab to a wider variant with stacked decoder, MaX-DeepLab-L. This scaling further improves the *single-scale* performance to 51.1% PQ, outperforming *multi-scale* Axial-DeepLab [89] by 7.2% PQ with similar inference M-Adds.

Method	Backbone	TTA	Params	M-Adds	PQ	PQ^{Th}	PQ^{St}
Box-based panoptic segmentation methods							
Panoptic-FPN [47]	RN-101				40.3	47.5	29.5
UPSNet [98]	RN-50				42.5	48.5	33.4
Detectron2 [93]	RN-101				43.0	-	-
UPSNet [98]	RN-50	✓			43.2	49.1	34.1
DETR [10]	RN-101		61.8M	314B[1]	45.1	50.5	37.0
Box-free panoptic segmentation methods							
Panoptic-DeepLab [21]	X-71 [24]		46.7M	274B	39.7	43.9	33.2
Panoptic-DeepLab [21]	X-71 [24]	✓	46.7M	3081B	41.2	44.9	35.7
Axial-DeepLab-L [89]	AX-L [89]		44.9M	344B	43.4	48.5	35.6
Axial-DeepLab-L [89]	AX-L [89]	✓	44.9M	3868B	43.9	48.6	36.8
MaX-DeepLab-S	MaX-S		61.9M	324B	48.4	53.0	41.5
MaX-DeepLab-L	MaX-L		451M	3692B	51.1	57.0	42.2

Table 2. COCO val set. **TTA:** Test-time augmentation

Method	Backbone	TTA	PQ	PQ^{Th}	PQ^{St}
Box-based panoptic segmentation methods					
Panoptic-FPN [47]	RN-101		40.9	48.3	29.7
DETR [10]	RN-101		46.0	-	-
UPSNet [98]	DCN-101 [25]	✓	46.6	53.2	36.7
DetectoRS [76]	RX-101 [97]	✓	49.6	57.8	37.1
Box-free panoptic segmentation methods					
Panoptic-DeepLab [21]	X-71 [24, 75]	✓	41.4	45.1	35.9
Axial-DeepLab-L [89]	AX-L [89]		43.6	48.9	35.6
Axial-DeepLab-L [89]	AX-L [89]	✓	44.2	49.2	36.8
MaX-DeepLab-S	MaX-S		49.0	54.0	41.6
MaX-DeepLab-L	MaX-L		51.3	57.2	42.4

Table 3. COCO test-dev set. **TTA:** Test-time augmentation

Test-dev set. Our improvements on the *val* set transfers well to the test-dev set, as shown in Tab. 3. On the test-dev set, we are able to compare with more competitive methods and stronger backbones equipped with group convolution [50, 97], deformable convolution [25], or recursive backbone [65, 76], while we do not use these improvements in our model. In the regime of no TTA, our MaX-DeepLab-S outperforms Axial-DeepLab [89] by 5.4% PQ, and DETR [10] by 3.0% PQ. Our MaX-DeepLab-L without TTA further attains 51.3% PQ, surpassing Axial-DeepLab with TTA by 7.1% PQ. This result also outperforms the best box-based method DetectoRS [76] with TTA by 1.7% PQ, closing the large gap between box-based and box-free methods on COCO for the first time. Our MaX-DeepLab sets a new state-of-the-art on COCO, even without using TTA.

4.2. Ablation study

In this subsection, we provide more insights by teasing apart the effects of MaX-DeepLab components on the *val* set. We first define a default baseline setting and then vary each component of it: We augment Wide-ResNet-41 [102, 95, 11] by applying dual-path transformer to all blocks at stride 16, enabling all four types of attention. For faster wall-clock training, we use an $L = 0$ (not stacked) decoder and approximate *P2P* attention with convolutional blocks.

[1]https://github.com/facebookresearch/detr

Res	Axial	L	Iter	Params	M-Adds	PQ	PQ^{Th}	PQ^{St}
641	✗	0	100k	196M	746B	45.7	49.8	39.4
641	✓	0	100k	277M	881B	47.8	51.9	41.5
1025	✗	0	100k	196M	1885B	46.1	50.7	39.1
1025	✓	0	100k	277M	2235B	**49.4**	**54.5**	**41.8**
641	✗	*1*	100k	271M	1085B	47.1	51.6	40.3
641	✗	*2*	100k	347M	1425B	47.5	52.3	40.2
641	✗	0	*200k*	196M	746B	46.9	51.5	40.0
641	✗	0	*400k*	196M	746B	47.7	52.5	40.4

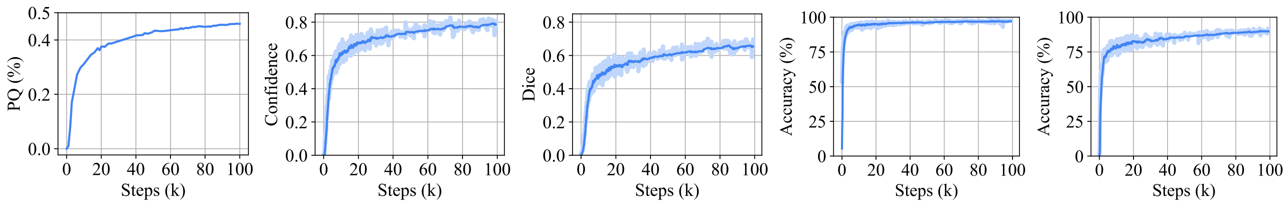
Table 4. Scaling MaX-DeepLab by using a larger input **Res**olution, replacing convolutional blocks with **Axial**-attention blocks, stacking decoder $\boldsymbol{L}$ times, and training with more **Iter**ations.

P2M	M2M	Stride	Params	M-Adds	PQ	PQ^{Th}	PQ^{St}
✓	✓	16	196M	746B	45.7	49.8	39.4
	✓	16	188M	732B	45.0	48.9	39.2
✓		16	196M	746B	45.1	49.3	38.9
		16	186M	731B	44.7	48.5	39.0
✓	✓	16 & 8	220M	768B	**46.7**	**51.3**	**39.7**
✓	✓	16 & 8 & 4	234M	787B	46.3	51.1	39.0

Table 5. Varying transformer **P2M** feedback attention, **M2M** self-attention, and the **Stride** where we apply the transformer.

Scaling. We first study the scaling of MaX-DeepLab in Tab. 4. We notice that replacing convolutional blocks with axial-attention blocks gives the most improvement. Further changing the input resolution to 1025×1025 improves the performance to 49.4% PQ, with a short 100k schedule (54 epochs). Stacking the decoder $L = 1$ time improves 1.4% PQ, but further scaling to $L = 2$ starts to saturate. Training with more iterations helps convergence, but we find it not as necessary as DETR which is trained for 500 epochs.

Dual-path transformer. Next, we vary attention types of our dual-path transformer and the stages (strides) where we apply transformer blocks. Note that we always apply *M2P* attention that attaches the transformer to the CNN. And *P2P* attention is already ablated above. As shown in Tab. 5, removing our *P2M* feedback attention causes a drop of 0.7% PQ. On the other hand, we find MaX-DeepLab robust (-0.6% PQ) to the removal of *M2M* self-attention. We attribute this robustness to our non-overlapping mask formulation. Note that DETR [10] relies on *M2M* self-attention to remove duplicated boxes. In addition, it is helpful (+1.0% PQ) to apply transformer blocks to stride 8 also, which is impossible for DETR without our dual-path design. Pushing it further to stride 4 does not show more improvements.



(a) Validation PQ (b) Matched class confidence (c) Matched mask dice (d) Instance discrimination (e) Mask-ID prediction

Figure 4. Training curves for (a) validation PQ, (b) average class confidence, $\hat{p}_{\hat{\sigma}(i)}(c_i)$, of matched masks, (c) average mask dice, $\mathrm{Dice}(m_i, \hat{m}_{\hat{\sigma}(i)})$, of matched masks, (d) per-pixel instance discrimination accuracy, and (e) per-pixel mask-ID prediction accuray.

sim	InstDis	Mask	Sem	PQ	PQ$^{\mathrm{Th}}$	PQ$^{\mathrm{St}}$	SQ	RQ
RQ $\times$ SQ	✓	✓	✓	**45.7**	49.8	**39.4**	**80.9**	**55.3**
RQ + SQ	✓	✓	✓	44.9	48.6	39.3	80.2	54.5
RQ $\times$ SQ	✓	✓		45.1	**50.1**	37.6	80.6	54.5
RQ $\times$ SQ		✓		43.3	46.4	38.6	80.1	52.6
RQ $\times$ SQ	✓			42.6	48.1	34.1	80.0	52.0
RQ $\times$ SQ				39.5	41.8	36.1	78.9	49.0

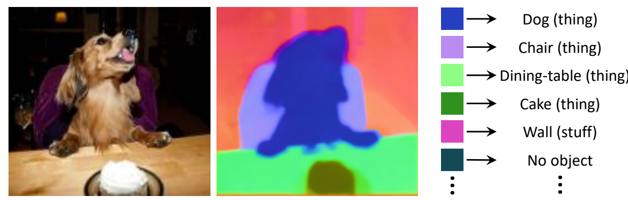
Table 6. Varying the similarity metric **sim** and whether to apply the auxiliary **Inst**ance **Dis**crimination loss, **Mask**-ID cross-entropy loss or the **Sem**antic segmentation loss.

Loss ablation. Finally, we ablate our PQ-style loss and auxiliary losses in Tab. 6. We first switch our PQ-style similarity in Equ. (6) from RQ $\times$ SQ to RQ + SQ, which differs in the hungarian matching (Equ. (7)) and removes dynamic loss weights in Equ. (10). We observe that RQ + SQ works reasonably well, but RQ $\times$ SQ improves 0.8% PQ on top of it, confirming the effect of our PQ-style loss in practice, besides its conceptual soundness. Next, we vary auxiliary losses applied to MaX-DeepLab, without tuning loss weights for remaining losses. Our PQ-style loss alone achieves a reasonable performance of 39.5% PQ. Adding instance discrimination significantly improves PQ$^{\mathrm{Th}}$, showing the importance of a clustered feature embedding. Mask-ID prediction shares the same target with the Dice term in Equ. (10), but helps focus on large masks when the Dice term is overwhelmed by small objects. Combining both of the auxiliary losses leads to a large 5.6% PQ gain. Further multi-tasking with semantic segmentation improves 0.6% PQ, because its class-level supervision helps stuff classes but not instance-level discrimination for thing classes.

4.3. Analysis

We provide more insights of MaX-DeepLab by plotting our training curves and visualizing the mask output head.

Training curves. We first report the validation PQ curve in Fig. 4(a), with our default ablation model. MaX-DeepLab converges quickly to around 46% PQ within 100k iterations (54 epochs), 1/10 of DETR [10]. In Fig. 4(b) and Fig. 4(c), we plot the characteristics of all matched masks in an image. The matched masks tend to have a better class correctness than mask correctness. Besides, we report per-pixel accuracies for instance discrimination (Fig. 4(d)) and mask-ID prediction (Fig. 4(e)). We see that most pixels learn quickly to find their own instances (out of K) and predict their own mask-IDs (out of N). Only 10% of all pixels predict wrong mask-IDs, but they contribute to most of the PQ error.



(a) Original image (b) Decoder feature g (c) Transformer output

Figure 5. (b) Pixels of the same instance have similar colors (features), while pixels of different instances have distinct colors. (c) The transformer predicts mask colors (features) and classes.

Visualization. In order to intuitively understand the normalized decoder output g, the transformer mask feature f, and how they are multiplied to generate our mask output $\hat{m}$, we train a MaX-DeepLab with $D = 3$ and directly visualize the normalized features as RGB colors. As shown in Fig. 5, the decoder feature g assigns similar colors (or feature vectors) to pixels of the same mask, no matter the mask is a thing or stuff, while different masks are colored differently. Such effective instance discrimination (as colorization) facilitates our simple mask extraction with an inner product.

5. Conclusion

In this work, we have shown for the first time that panoptic segmentation can be trained end-to-end. Our MaX-DeepLab directly predicts masks and classes with a mask transformer, removing the needs for many hand-designed priors such as object bounding boxes, thing-stuff merging, *etc.* Equipped with a PQ-style loss and a dual-path transformer, MaX-DeepLab achieves the state-of-the-art result on the challenging COCO dataset, closing the gap between box-based and box-free methods for the first time.

Acknowledgments. We thank Maxwell Collins, Sergey Ioffe, Jiquan Ngiam, Siyuan Qiao, Chen Wei, Jieneng Chen, and Google Mobile Vision for the support and discussions. This work is supported by Google Faculty Research Award.

References

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

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

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

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

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

[7] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms–improving object detection with one line of code. In *ICCV*, 2017.

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

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

[10] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.

[11] Liang-Chieh Chen, Raphael Gontijo Lopes, Bowen Cheng, Maxwell D Collins, Ekin D Cubuk, Barret Zoph, Hartwig Adam, and Jonathon Shlens. Naive-Student: Leveraging Semi-Supervised Learning in Video Sequences for Urban Scene Segmentation. In *ECCV*, 2020.

[12] 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.

[13] 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.

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

[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.

[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.

[17] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

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

[19] Bowen Cheng, Liang-Chieh Chen, Yunchao Wei, Yukun Zhu, Zilong Huang, Jinjun Xiong, Thomas S Huang, Wen-Mei Hwu, and Honghui Shi. Spgnet: Semantic prediction guidance for scene parsing. In *ICCV*, 2019.

[20] 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.

[21] 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.

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

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

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

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

[26] 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.

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

[28] 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.

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

[30] 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.

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

[32] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

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

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

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

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

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

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

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

[40] Jyh-Jing Hwang, Stella X Yu, Jianbo Shi, Maxwell D Collins, Tien-Ju Yang, Xiao Zhang, and Liang-Chieh Chen. SegSort: Segmentation by discriminative sorting of segments. In *ICCV*, 2019.

[41] Sergey Ioffe and Christian Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

[42] Fabian Isensee, Jens Petersen, Andre Klein, David Zimmerer, Paul F Jaeger, Simon Kohl, Jakob Wasserthal, Gregor Koehler, Tobias Norajitra, Sebastian Wirkert, et al. nnu-net: Self-adapting framework for u-net-based medical image segmentation. *arXiv:1809.10486*, 2018.

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

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

[45] 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.

[46] 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.

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

[48] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *CVPR*, 2019.

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

[50] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[51] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[52] 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.

[53] 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.

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

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

[56] 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.

[57] 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.

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

[59] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017.

[60] 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.

[61] Chenxi Liu, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, Wei Hua, Alan Yuille, and Li Fei-Fei. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In *CVPR*, 2019.

[62] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *ICLR*, 2020.

[63] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *CVPR*, 2018.

[64] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016.

[65] 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.

[66] 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.

[67] 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.

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

[69] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *3DV*, 2016.

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

[71] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, 2016.
[72] Niki Parmar, Prajit Ramachandran, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. In *NeurIPS*, 2019.
[73] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, 2018.
[74] Lorenzo Porzi, Samuel Rota Bulò, Aleksander Colovic, and Peter Kontschieder. Seamless scene segmentation. In *CVPR*, 2019.
[75] Haozhi Qi, Zheng Zhang, Bin Xiao, Han Hu, Bowen Cheng, Yichen Wei, and Jifeng Dai. Deformable convolutional networks – coco detection and segmentation challenge 2017 entry. *ICCV COCO Challenge Workshop*, 2017.
[76] Siyuan Qiao, Liang-Chieh Chen, and Alan Yuille. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. *arXiv:2006.02334*, 2020.
[77] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.
[78] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 2015.
[79] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *NAACL*, 2018.
[80] Zhuoran Shen, Mingyuan Zhang, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Efficient attention: Attention with linear complexities. In *WACV*, 2021.
[81] Konstantin Sofiiuk, Olga Barinova, and Anton Konushin. Adaptis: Adaptive instance selection network. In *ICCV*, 2019.
[82] Russell Stewart, Mykhaylo Andriluka, and Andrew Y Ng. End-to-end people detection in crowded scenes. In *CVPR*, 2016.
[83] Saeid Asgari Taghanaki, Yefeng Zheng, S Kevin Zhou, Bogdan Georgescu, Puneet Sharma, Daguang Xu, Dorin Comaniciu, and Ghassan Hamarneh. Combo loss: Handling input and output imbalance in multi-organ segmentation. *Computerized Medical Imaging and Graphics*, 75:24–33, 2019.
[84] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *CVPR*, 2020.
[85] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. In *ECCV*, 2020.
[86] 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.
[87] 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.
[88] Luc Vincent and Pierre Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE TPAMI*, 1991.
[89] 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.
[90] Sinong Wang, Belinda Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv:2006.04768*, 2020.
[91] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018.
[92] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. SOLOv2: Dynamic and fast instance segmentation. In *NeurIPS*, 2020.
[93] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019.
[94] Zhirong Wu, Alexei A Efros, and Stella X Yu. Improving generalization via scalable neighborhood component analysis. In *ECCV*, 2018.
[95] Zifeng Wu, Chunhua Shen, and Anton Van Den Hengel. Wider or deeper: Revisiting the ResNet model for visual recognition. *Pattern Recognition*, 2019.
[96] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, 2018.
[97] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.
[98] Yuwen Xiong, Renjie Liao, Hengshuang Zhao, Rui Hu, Min Bai, Ersin Yumer, and Raquel Urtasun. Upsnet: A unified panoptic segmentation network. In *CVPR*, 2019.
[99] Brandon Yang, Gabriel Bender, Quoc V Le, and Jiquan Ngiam. Condconv: Conditionally parameterized convolutions for efficient inference. In *NeurIPS*, 2019.
[100] 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.
[101] Yibo Yang, Hongyang Li, Xia Li, Qijie Zhao, Jianlong Wu, and Zhouchen Lin. Sognet: Scene overlap graph network for panoptic segmentation. In *AAAI*, 2020.
[102] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016.
[103] 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.
[104] Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps forward, 1 step back. In *NeurIPS*, 2019.
[105] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *CVPR*, 2020.

[106] 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.

[107] 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.

[108] 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.

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