Panoptic-PartFormer: Learning a Unified model for Panoptic Part Segmentation

Xiangtai Li[1] Shilin Xu[1] Yibo Yang[1,3]
Guangliang Cheng[2✉] Yunhai Tong[1✉] Dacheng Tao[3]

[1] Key Laboratory of Machine Perception, MOE, School of Artificial Intelligence, Peking University
[2] SenseTime Research
[3] JD Explore Academy
lxtpku@pku.edu.cu, xushilin@stu.pku.edu.cn,chengguangliang@sensetime.com

Abstract. Panoptic Part Segmentation (PPS) aims to unify panoptic segmentation and part segmentation into one task. Previous work mainly utilizes separated approaches to handle thing, stuff, and part predictions individually without performing any shared computation and task association. In this work, we aim to unify these tasks at the architectural level, designing the first end-to-end unified method named Panoptic-PartFormer. In particular, motivated by the recent progress in Vision Transformer, we model things, stuff, and part as object queries and directly learn to optimize the all three predictions as unified mask prediction and classification problem. We design a decoupled decoder to generate part feature and thing/stuff feature respectively. Then we propose to utilize all the queries and corresponding features to perform reasoning jointly and iteratively. The final mask can be obtained via inner product between queries and the corresponding features. The extensive ablation studies and analysis prove the effectiveness of our framework. Our Panoptic-PartFormer achieves the new state-of-the-art results on both Cityscapes PPS and Pascal Context PPS datasets with around 70% GFlops and 50% parameters decrease. Given its effectiveness and conceptual simplicity, we hope the Panoptic-PartFormer can serve as a strong baseline and aid future research in PPS. Our code and models will be available at https://github.com/lxtGH/Panoptic-PartFormer.

Keywords: Panoptic Part Segmentation, Scene Understanding, Vision Transformer

1 Introduction

One essential problem in computer vision is to understand a scene at multiple levels of concept. In particular, when people perceive a scene, they can catch each visual entities such as car, bus, or person, and they can also understand the parts of entities, such as person-head and car-wheel, etc. The former is named as scene parsing, while the latter is termed as part parsing. One representative direction of unified scene parsing is Panoptic Segmentation (PS) [23,22]. It predicts a class

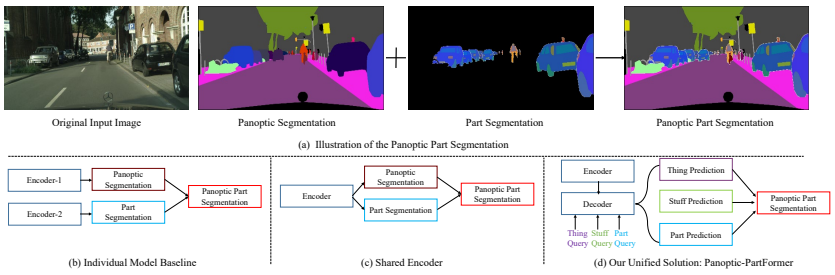

Fig. 1: Illustration of Panoptic Part Segmentation (PPS) and different Approaches for solving such PPS task. (a) An illustration of the Panoptic Part Segmentation task. It combines the Panoptic Segmentation and Part Segmentation in a unified manner that provides the multi-level concept understanding of the scene (b) The baseline method proposed in [15] combines results of panoptic segmentation and part segmentation. (c) Panoptic-FPN-like baseline [22,29,64] adds part segmentation into the current panoptic segmentation frameworks. (d) Our proposed approach represents things, stuff, and part via object queries and performs joint learning. Best view in color.

label and an instance ID for each pixel. The part parsing has a wide range of definitions, such as human part or car part [33,14]. Both directions are independent, while both are equally important for many vision systems, including auto-driving and robot navigation [8].

Recently, the Panoptic Part Segmentation (PPS, or called Part-aware Panoptic Segmentation) [15] is proposed to unify these multiple levels of abstraction into one single task. As shown in Fig. 1(a), it requires a model to output per-pixel scene-level classification for background stuff, segment things into individual instances and segment each instance into specific parts. Several baselines using combined approaches [18,73,75] were proposed to tackle this task. As shown in Fig. 1(a), they fuse different individual model predictions to obtain PPS results. In particular, they fuse the panoptic segmentation and part segmentation results from the non-shared backbone networks. This makes the entire process exceptionally complex with huge engineering efforts. Also, the shared computation and task association are ignored, which leads to inferior results. Another solution for this task is to make the part segmentation as an extra head with shared backbone as shown in Fig. 1(b). Such design is well explored in PS studies [64,22,31,4,25,47,68,63,5,58]. However, most of them treat PS as separated tasks [64] or sequential tasks with several post processing components [5].

Since Detection Transformer (DETR) [1], there are several works [57,6,72] unifying both thing and stuff learning via *object queries* in PS, which makes the entire pipeline elegant and achieves strong results where the mask classification and prediction can be performed directly. These results show that many complex components including NMS and box detection can be removed. In particular, such design considers the full scene understanding via performing interactions among things, stuff, and part simultaneously. Joint training with things, stuff, and part leads to better part segmentation results since the full scene infor-

mation renders part representation a more discriminative information such as global context. Motivated by these analysis, we take a further step on the more challenging PPS task and propose the **first unified model** for this task.

In this paper, we present a simple yet effective baseline named Panopic-PartFormer, a unified model for PPS task. As shown in Fig. 1(c), we introduce three different types of queries for modeling thing, stuff, and part prediction, respectively. Then a decoupled decoder is proposed to generate fine-grained features. These features are used to decode thing, stuff, and part mask prediction. The decoupled decoder contains a part decoder and a scene decoder. For part decoder, we design a feature aligned decoder to keep more fine details in part. Rather than directly using the pixel-level self-attention in Transformer, we consider the recent works [52,72] that perform self-attention on query level. To be more specific, we focus on refining queries via their corresponding query features. We define the *query feature* as *grouped features* that are generated from the *corresponding mask* of each query and *decoder features*. The masks are generated via dot product between queries and decoder features. In particular, the initial query features are grouped via the initial mask prediction from the decoupled decoder. Then we perform updating object queries with the query features. This operation is implemented with one dynamic convolution [72,52] and multi-head self-attention layers [56] between query and query features iteratively. The former poses instance-wise information from feature to enhance the query learning where the parameters are generated by the features itself while the latter performs inner reasoning among different types of queries to model the relationship among thing, stuff, and part. Moreover, the entire procedure avoids pixel-level computation in other vision Transformer decoder [1,6]. In this way, since all thing, stuff, and part information is encoded into query, the relation between these queries can be well explored and optimized jointly via pure mask based supervison. Extensive experiments (Sec.4) show that our approach achieves much better results than the previous design in Fig. 1(b).

Moreover, our Panopic-PartFormer can support both CNN backbones [19] and Transformer backbones [39] for feature extraction. Panopic-PartFormer is also memory and computation efficient, which is mainly benefited by avoiding pixel-level computation of self-attention layers. Panopic-PartFormer can directly output thing, stuff, and part segmentation predictions in box-free and NMS-free manner. It can also be evaluated by sub-task of PPS such as Panoptic Segmentation. In the experiment part, we verify our panoptic segmentation predictions on Cityscaeps datasets [8] and it also achieves better results than the recent works [31]. To sum up, our main contributions are as follows:

- We present a novel, simple and effective baseline named Panopic-PartFormer for the PPS task. To the best of our knowledge, it is the first unified end-to-end model for this task.
- We propose a decoupled decoder and a joint query updating and reasoning framework for the joint feature learning of thing, stuff, and part. Besides, a joint loss function is proposed to supervise the whole model.

– Extensive experiments and analyses indicate the effectiveness and generalization of our model. In particular, with our framework, we find the part segmentation can be improved significantly via joint training. We achieve the new state-of-the-art results on two challenging PPS benchmarks including Pascal Context PPS dataset (about 6-7% PartPQ gain on ResNet101, 10% PartPQ gain using swin Transformer [39]) and Cityscapes PPS dataset (about 1-2% PartPQ gain).

2 Related Work

Part Segmentation. Most previous approaches for both instance and semantic part segmentation mainly focus on human analysis [48,51]. Several works [11,38,60] design specific methods for semantic part segmentation which are in category-level settings. There are two paradigms for human part segmentation: *top-down* pipelines [26,66,50,21,65] and *bottom-up* pipelines [17,24,74,76]. Meanwhile, there are also several works focusing on task-specific part segmentation [75,34,42]. Compared with these methods, the focus of this paper is to solve the PPS task which naturally contains the part segmentation as a sub-task.

Panoptic Segmentation. Earlier work [23] mainly performs segmentation for things and stuff via separated networks where the original benchmark directly combines predictions of things and stuff from different models. To alleviate the computation cost, recent works [22,29,4,25,47,68,63] are proposed to model both stuff segmentation and thing segmentation in one model with different task heads. Detection based methods [64,22,49,20] usually represent things with the box prediction while several bottom-up models [5,67,13,58] perform grouping instance via pixel-level affinity or center heat maps from semantic segmentation results. The former introduces complex process while the latter suffers from the performance drops in complex scenarios. Recently, several works [57,6,72] propose to directly obtain segmentation masks without box supervision. However, these works do *not* cover the *knowledge of part-level semantics* of images which can provide more comprehensive information for scene understanding.

Panoptic Part Segmentation. To better understand the full scene, the PPS task [15] is proposed. This work annotates two datasets (Cityscape PPS [8] and Pascal Context PPS [10]) and proposes a new metric named PartPQ [15] for evaluation. This work also presents several baseline methods to obtain the final results. However, these methods are all separated networks for instance and semantic segmentation to obtain the panoptic segmentation or use existing panoptic segmentation algorithms with part semantic segmentation as an isolated sub-network (shown in Fig .1(a) and (b)). For the comparison, our goal is to design a unified and effective network for all the tasks.

Vision Transformer. There are mainly two different usages for Transformer in vision: feature extractor and query modeling. Compared with CNN, vision Transformers [9,39,55] have more advantages in modeling global-range relation among the image patch features. The second design is to use the object query representation. DETR [1] models the object detection task as a set prediction

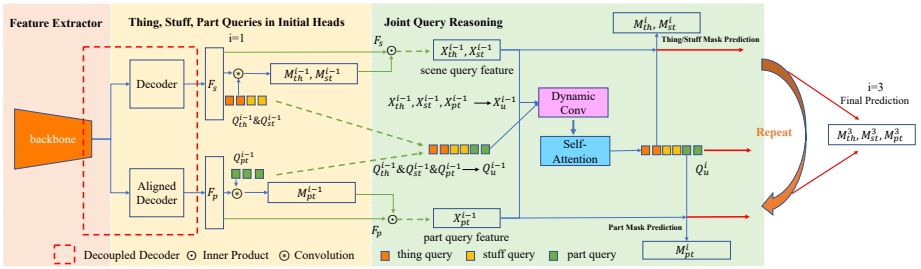

Fig. 2: Our proposed Panoptic-PartFormer. It contains three parts: (1) a backbone to extract features (Red area) (b) a decoupled decoder to generate scene features and part features along with the initial prediction heads to generate initial mask predictions. (Yellow area) (c) a cascaded Transformer decoder to jointly do reasoning between the query and query features. (Green area) Green arrows mean input (come from previous stage) while Red arrows represent current stage output (used for next stage). $i = 3$ is the final mask outputs.

problem with learnable queries. The following works [52,78] explore the locality of the learning process to improve the performance of DETR. Query based learning can also be applied to other fields [12,41]. Our methods are inspired by these works with the goal of unifying and simplifying the PPS task based on query learning.

3 Method

3.1 Basic Architecture

Fig. 2 presents an overall illustration of the proposed method. Our method contains three parts: (1) an encoder backbone to extract features; (2) a decoupled decoder to obtain the scene features and part features individually. We denote the scene features are used to generate things, stuff masks while the part features are used to generate part masks; (3) a Transformer decoder which takes three different types of queries and backbone features as inputs and provides thing, stuff, and part mask predictions.

Encoder network: We first extract image features for each input image. It contains a backbone network (Convolution Network [19] or Vision Transformer [39]) with Feature Pyramid Network [35] as neck. This results in a set of multiscale features which are the inputs of the decoupled decoder.

Decoupled decoder: The decoupled decoder has two separate decoder networks to obtain features for scene feature and part feature, respectively. The former is used to decode both thing and stuff predictions, while the latter is applied to decode the part prediction. Our motivation is that part segmentation has different properties from panoptic segmentation. First, part features need a more precise location and fine details. Second, scene features focus on mask proposal level prediction while part features pay more attention to the inner parts

of mask proposal, which conflicts with each other. We show that the decoupled design leads to better results in the experimental section. (see Sec. 4.2).

For implementation, we adopt semantic FPN design [22] to fuse features in a top-down manner. Thus, we obtain the two features named F_s and F_p. In particular, for part segmentation, we design a light-weight aligned feature decoder for part segmentation. Rather than the naive bilinear upsampling, we propose to learn feature flow [28,76] to warp the low resolution feature to high resolution feature. Then we sum all the predictions into the highest resolution as semantic FPN. Moreover, to preserve more locational information, we add the positional embedding to each stage of the semantic FPN following the previous works [61,62]. In summary, decoupled decoder outputs two separate features: scene features F_s and part features F_p. The former is used to generate thing and stuff masks while the latter is used to generate part masks and both have the same resolution.

3.2 Thing, Stuff, and Part as Queries with Initial Head Prediction

Previous works [6,72,57] show that single mask classification and prediction can achieve the state-of-the-art results on COCO [37]. Motivated by this, our model treats thing, stuff, and part as the input queries to directly obtain the final panoptic part segmentation. Following previous works [72,52], the initial weights of these queries are directly obtained from the first stage weights of the initial decoupled decoder prediction. For mask predictions of thing, stuff, and part, we use three 1×1 convolution layers to obtain the initial outputs of thing, stuff, and part masks. These layers are appended at the end of the decoupled decoder.

All these predictions are directly supervised with corresponding ground truth masks. As shown in [52,72], using such initial heads can avoid heavier Transformer encoder layers for pixel-level computation, since the corresponding query features can be obtained via mask grouping from the initial mask prediction.

In this way, we obtain the three different queries for thing, stuff, and part along with their initial mask prediction. We term them as Q_{th}, Q_{st}, Q_{pt} and M_{th}, M_{st}, M_{pt} with shapes $N_{th} \times d$, $N_{st} \times d$, $N_{pt} \times d$ and shapes $N_{th} \times H \times W$, $N_{st} \times H \times W$, $N_{pt} \times H \times W$. d, W, H are the channel number, width, height of feature F_p and F_s. N_{th}, N_{st}, N_{pt} are numbers of categories for thing, stuff, and part classification.

3.3 Joint Thing, Stuff, and Part Query Reasoning

The cascaded Transformer decoder takes previous mask predictions, previous object queries and decoupled features as inputs and outputs the current refined mask predictions and object queries. The refined mask predictions and object queries along with decoupled features will be the inputs of the next stage. The relationship between queries and query features is jointly learned and reasoned.

Our key insights are: Firstly, joint learning can learn the full correlation between scene features and part features. For example, car parts must be on the

road rather than in the sky. Secondly, joint reasoning can avoid several scene noisy cases, such as car parts on the human body or human parts on the car. We find joint learning leads to better results (see Sec. 4.2).

We combine the three queries and the three mask predictions into a unified query Q_u^{i-1} and M_u^{i-1} where $Q_u^{i-1} = concat(Q_{th}, Q_{st}, Q_{pt})$, $Q_u^{i-1} \in \boldsymbol{R}^{(N_{th}+N_{st}+N_{pt})\times d}$ and $M_u^{i-1} = concat(M_{th}^{i-1}, M_{st}^{i-1}, M_{pt}^{i-1})$, $M_u^{i-1} \in \boldsymbol{R}^{(N_{th}+N_{st}+N_{pt})\times H\times W}$. i is the stage index of our Transformer decoder. $i = 1$ means the predictions come from the initial heads. Otherwise, it means predictions from the outputs of previous stage. *concat* is preformed along the first dimension.

In particular, following the previous work [72], we first obtained query features X^i via grouping from previous mask predictions M_u^{i-1} and input features (F_p, F_s) shown in Equ. 1 (dot product in Fig. 2). We present this process in one formulation for simplicity.

$$X^i = \sum_u^W \sum_v^H M^{i-1}(u, v) \cdot F(u, v), \tag{1}$$

where $X^i \in \boldsymbol{R}^{(N_{th}+N_{st}+N_{pt})\times d}$ is the per-instance extracted feature with the same shape as Q_u, M^{i-1} is the per-instance mask extracted from the previous stage $i-1$, and F is the input feature extracted for the decoupled decoder head. u, v are the indices of spatial location. i is layer number and starts from 1. As shown in center part of Fig 2, the part mask prediction and scene mask prediction are applied on corresponding features (F_p, F_s) individually where we obtain part query features X_{pt}^i and scene query features X_{th}^i and X_{st}^i. Then we combine these query features through $X_u^i = concat(X_{th}^i, X_{st}^i, X_{pt}^i)$. These inputs are shown in the green arrows in Fig. 2.

Then we perform a dynamic convolution [54,72,52] to refine input queries Q_u^{i-1} with the query features X_u^i which are grouped from their masks.

$$\hat{Q}_u^{i-1} = DynamicConv(X_u^i, Q_u^{i-1}), \tag{2}$$

where the dynamic convolution uses the query features X_u^i to generate parameters to weight input queries Q_u^{i-1}. To be more specific, *DynamicConv* uses input query features X_u^i to generate gating parameters via MLP and multiply back to the original query input Q_u^{i-1}. Our motivation has two folds: Compared with pixel-wise MHSA [6,1], dynamic convolution introduces less computation and faster convergence for limited computation. Secondly, it poses the instance-wised information to each query dynamically during training and inference, which shows better generalization and has complementary effects with MHSA. More details can be found in Sec. 4.2.

This operation absorbs more fine-grained information to help query look for more precise location.

In particular, we adopt the same design [72] by learning gating functions to update the refined queries. The *DynamicConv* operation is shown as follows:

$$\hat{Q}_u^{i-1} = Gate_x(X_u^i)X_u^i + Gate_q(X_u^i)Q_u^{i-1}, \tag{3}$$

where $Gate$ is implemented with a fully connected (FC) layers followed by LayerNorm (LN), and a sigmoid layer. We adopt two different gate functions including $Gate_x$ and $Gate_q$. The former is to weight the query features, while the latter is to weight corresponding queries.

After that, we adopt one self-attention layer with feed forward layers [56,57] to learn the correspondence among each query and update them accordingly. This operation leads to the full correlation among queries, shown as follows:

$$Q_u^i = FFN(MHSA(\hat{Q}_u^{i-1}) + \hat{Q}_u^{i-1}), \tag{4}$$

where $MHSA$ means Multi Head Self Attention, FFN is the Feed Forward Network that is commonly used in current vision Transformers [1,9]. The output refined query has the same shape as the input, i.e. $Q_u^i \in \boldsymbol{R}^{(N_{th}+N_{st}+N_{pt})\times d}$.

Finally, the refined masks are obtained via dot product between the refined queries Q_u^i and the input features F_p, F_s. For mask classification, we adopt several feed forward layers on Q_u^i and directly output the class scores (thing, stuff, and part). For mask prediction, we also adopt several feed forward layers on Q_u^i and then we perform the inner product between learned queries and features (F_s F_p) to generate scene masks (thing and stuff) and part masks of stage i. These masks will be used for the next stage input as shown in the red arrows in Fig 2. The process of Equ. 1, Equ. 2 and Equ. 4 will be repeated for several times. We set the iteration number to 3 by default. The inter mask predictions are also optimized by mask supervision.

Discussion. We admit that we use the dynamic convolution and self-attention among queries that are proposed by [52,72]. However, we **do not claim** this is our contribution for Panoptic-PartFormer. Our main contribution is a system level unified model for this challenging task(PPS) and we are the first work to prove that joint learning of the thing, stuff and part learning benefits PPS tasks than many other designs. More details can be found in supplementary.

3.4 Training and inference

Training: To train the Panoptic-PartFormer, we need to assign ground truth according to the pre-defined cost since all the outputs are encoded via queries. In particular, we mainly follow the design of [6] to use bipartite matching as a cost by considering both mask and classification results. After the bipartite matching, we apply a loss jointly considering mask prediction and classification for each thing, stuff, and part. In particular, we apply focal loss [36] on both classification and mask prediction. We also adopt dice loss [43] on mask predictions (L_{part}, L_{thing}, L_{stuff}). Such settings are *the same as previous works* [1,6]. The loss for each stage i can be formulated as follows:

$$\mathcal{L}_i = \lambda_{part} \cdot \mathcal{L}_{part} + \lambda_{thing} \cdot \mathcal{L}_{thing} + \lambda_{stuff} \cdot \mathcal{L}_{stuff} + \lambda_{cls} \cdot \mathcal{L}_{cls} \tag{5}$$

Note that the losses are applied to each stage $\mathcal{L}_{final} = \sum_i^N \mathcal{L}_i$, where N is the total stages applied to the framework. We adopt $N = 3$ and all λs are set to 1 by default.

Panoptic seg. method	Part seg. method	PQ			PartPQ		
		All	P	NP	All	P	NP
Cityscapes Panoptic Parts *validation set*							
UPSNet [64](ResNet50)	DeepLabv3+ [3](ResNet50)	59.1	57.3	59.7	55.1	42.3	59.7
DeepLabv3+(ResNet50) & Mask R-CNN(ResNet50) [18]	DeepLabv3+ [3] (Xception- 65)	61.0	58.7	61.9	56.9	43.0	61.9
Panoptic-PartFormer (ResNet50)		**61.6**	**60.0**	**62.2**	**57.4**	**43.9**	**62.2**
EfficientPS [45](EfficientNet) [53]	BSANet [75](ResNet101)	65.0	64.2	65.2	60.2	**46.1**	65.2
HRNet-OCR (HRNetv2-W48) [70,59] & PolyTransform [32]	BSANet [75](ResNet101)	66.2	64.2	67.0	61.4	45.8	67.0
Panoptic-PartFormer (Swin-base)		**66.6**	**65.1**	**67.2**	**61.9**	**45.6**	**68.0**

Table 1: **Experiment Results on CPP.** Previous works combine results from commonly used (top), and state-of-the-art methods (bottom) for semantic segmentation, instance segmentation, panoptic segmentation and part segmentation. Metrics split into *P* and *NP* are evaluated on scene-level classes with and without parts, respectively.

Inference: We directly get the output masks from the corresponding queries according to their sorted scores. To obtain the final panoptic part segmentation, we first obtain the panoptic segmetnation results and then merge part masks into panoptic segmentation results. For panoptic segmentation results, we adopt the method used in Panoptic-FPN [22] to merge panoptic mask. For part merging process, we follow the original PPS task to obtain the final panoptic part segmentation results. For scene-level semantic classes that do not have part classes, we simply copy the predictions from panoptic segmentation. For predicted instances with the part, we extract the part predictions for the pixels corresponding to this segment. Otherwise, if a part prediction contains a part class that does not correspond to the scene-level class, we set it to *void* label. This setting mainly follows the previous work [15].

4 Experiment

Datasets. We mainly carry out experiments on two datasets including Cityscapes Panoptic Parts (CPP) and PASCAL Panoptic Parts (PPP), which are based on the established scene understanding datasets Cityscapes [8] and PASCAL VOC [10], respectively. The CPP extends with part-level semantics the Cityscapes dataset [8] and is annotated with 23 part-level semantic classes. In particular, 5 scene-level semantic classes from the human and vehicle high-level categories are annotated with parts. The CPP contains 2975 training and 500 validation images. PPP extends the PASCAL VOC 2010 benchmark [10] with part-level and scene-level semantics. PPP has 4998 training and 5105 validation images. To perform fair comparison, following previous settings [15,10], we perform experiments 59 scene-level classes (20 things, 39 stuff), and 57 part classes. We further report Cityscapes Panoptic Segmentation validation set [8] results as sub-task comparison.

Experiment Settings. ResNet [19] and Swin Transformer [39] are adopted as the backbone networks and other layers use Xavier initialization [16]. The optimizer is AdamW [40] with weight decay 0.0001. The training batch size is set to 16 and all models are trained with 8 GPUs. For PPP datasets, we first pretrain our model on COCO dataset [37] since most previous baselines [15] are

pretrained on COCO. For PPP dataset, we adopt the multiscale training [1] by resizing the input images from the scale 0.5 to scale 2.0. We also apply random crop augmentations during training, where the train images are cropped with probability 0.5. For CPP dataset, we follow the similar setting in Panoptic-Deeplab [5] where we resize the images with scale rang from 0.5 to 2.0 and randomly crop the whole image during training with batch size 16. All the results are obtained via single scale inference.

Metric. We report PartPQ [15] and PQ [23] as the main metrics, since PartPQ is a unified metric that contain both scene-level output and part-level output. Part segmentation results such as mIoU can be found in the appendix file. The PartPQ per scene-level class l is formalized as $\text{PartPQ} = \frac{\sum_{(p,g)\in TP} \text{IOU}_\text{p}(p,g)}{|TP|+\frac{1}{2}|FP|+\frac{1}{2}|FN|}$. TP is true positive, FP is false positive, and FN is false negative segments, receptively. The definition of these is based on the Intersection Over Union (IOU) between a predicted segment p and a ground-truth segment g for a class l (where $l \in \mathcal{L}$, $\mathcal{L}$ is the label set). A prediction is a TP if it has an overlap with a ground-truth segment with an IOU > 0.5. An FP is a predicted segment that is not matched with the ground-truth, and an FN is a ground-truth segment not matched with a prediction. IOU contains two cases (part and non-part):

$$\text{IOU}_\text{p}(p,g) = \begin{cases} \text{mean IOU}_\text{part}(p,g), & l \in \mathcal{L}^\text{parts} \\ \text{IOU}_\text{inst}(p,g), & l \in \mathcal{L}^\text{no-parts} \end{cases}$$

4.1 Main Results

Results on Cityscape Panoptic Part Dataset. In Tab. 1, we compare our Panoptic-PartFormer with previous baselines. All the models use the single scale inference without test time augmentation. Our method with ResNe50 backbone achieves 57.4% PartPQ which outperforms the previous work using complex pipelines [3,18] with even stronger backbone [7]. For the same backbone, our method results in 2.3% PartPQ gain over the previous baseline. For large model comparison, our method with Swin-Transformer achieves 61.9 %PartPQ. It outperforms previous works that use state-of-the-art individual models [70,59,75,32] by 0.5%. Note that the best model from HRNet [70] is pretrained using Mapillary dataset [46]. We follow the same pipeline for fair comparison. Both settings prove the effectiveness of our approaches.

GFlops and Parameter Comparison. Since our method is one single unified model, our Panoptic-PartFormer has advantages on both GFlops and Parameters. Since the work [32] is not public available, we estimate the lower bound by its baseline model [18]. As shown in Tab. 2, our model obtain around 60% GFlops drop and 70% parameter drop.

Results on Pascal Panoptic Part Dataset. We further compare our method with previous work on the Pascal Panoptic Part dataset in Tab. 3. For different settings, our methods achieve state-of-the-art results on both PQ and PartPQ with a very significant gain. For ResNet backbone, our methods achieve **6-7% gains** on PartPQ. Moreover, our resnet101 model can achieve better results than

Method	PQ	PartPQ	Param(M)	GFlops
UPSNet + DeepLabv3+ (ResNet50)	59.1	55.1	>87	>890
Panoptic-PartFormer (ResNet50)	**61.6**	**57.4**	**37.35**	**185.84**
HRNet(OCR) +PolyTransform + BSANet	66.2	61.4	>181	>1154
Panoptic-PartFormer (Swin-base)	**66.6**	**61.9**	**100.32**	**408.52**

Table 2: More detailed comparison on Cityscapes PPS dataset. GFlops are measured with 1200 × 800 input.

Panoptic seg. method	**Part seg. method**	PQ	PartPQ
Pascal Panoptic Parts *validation set*			
DeepLabv3+ & Mask R-CNN [18](ResNet50)	DeepLabv3+ [3](ResNet50)	35.0	31.4
DLv3-ResNeSt269 [71] & DetectoRS [49]	BSANet [75]	42.0	38.3
Our Unified Approach			
Panoptic PartFormer (ResNet50)		**47.6**	**37.8**
Panoptic PartFormer (ResNet101)		**49.2**	**39.3**

Table 3: **Experiment Results on PPP dataset.** All the methods use single scale inference.

DD	DC	SA	I=1	I=3	PQ	PartPQ
✓	✓	✓	-	✓	61.6	57.4
-	✓	✓	-	✓	61.2	55.9
✓	-	✓	-	✓	57.0	52.2
✓	✓	-	-	✓	57.3	53.4
✓	✓	✓	✓	-	58.3	54.2

(a) Effect of each component. DD: Decoupled Decoder. DC: Dynamic Convolution. SA: Self Attention. I: Interaction number.

Setting	PQ	PartPQ
Joint Reasoning	61.6	57.4
Separate Reasoning	61.1	56.8
Sequential Reasoning	60.8	56.3

(b) Ablation on Query Reasoning Design

Method	PQ	PartPQ
Joint Query	61.6	57.4
DP-Based	59.8	55.9
DP-Based w ASPP [2]	59.9	56.1

(c) Dense Prediction or Query Prediction on Part. DP: Dense Prediction. w: with. ASPP: Atrous Spatial Pyramid Pooling [2].

Settings	PQ	PartPQ	P	NP
w Aligned Part Decoder	61.6	57.4	43.9	62.2
w/o Aligned Part Decoder	61.4	56.3	41.2	62.1
Aligned Part Decoder on Both Features	61.4	57.2	43.7	62.0

(d) Effect of Aligned Decoder Design.

Setting	Panoptic-GT	Part-GT	PartPQ	P
baseline	-	-	57.4	43.9
	-	✓	61.6	56.1
	✓	-	88.4	56.4

(e) Upper bound Analysis. GT: Ground Truth

Table 4: Ablation studies and analysis on Cityscapes Panoptic Part validation set with ResNet50 as backbone. Best view it in color.

previous work using *larger* backbone [49]. Using Swin Transformer base as backbone [39], our method achieves **47.4% PartPQ** which shows the generalization ability on large model.

Results on Cityscapes Panoptic Segmentation. We also compare our method with several previous works on cityscapes panoptic validation set. As shown in Tab. 5, our Panoptic-PartFormer also achieves state-of-the-art results compared with previous works [31,5]. This proves the generalization ability of our framework.

4.2 Ablation Study and Model Design

In this section, we present ablation study and several model designs of our Panoptic-PartFormer.

Method	Backbone	PQ	PQ_{th}	PQ_{st}
Panoptic FPN [22]	ResNet101	58.1	52.0	62.5
UPSNet [64]	ResNet50	59.3	54.6	62.7
SOGNet [68]	ResNet50	60.0	56.7	62.5
Seamless [47]	ResNet50	60.2	55.6	63.6
Unifying [27]	ResNet50	61.4	54.7	66.3
Panoptic-DeepLab [5]	ResNet50	59.7	-	-
Panoptic FCN* [31]	ResNet50	61.4	54.8	66.6
Panoptic FCN++ [30]	Swin-large	64.1	55.6	70.2
Panoptic-PartFormer	ResNet50	61.6	54.9	66.8
Panoptic-PartFormer	Swin-base	66.6	61.7	70.3

Table 5: **Experiment Results on Cityscapes Panoptic validation set.** * indicates using DCN [77].

Method	PQ	PartPQ
baseline	61.6	57.4
w/o positional encoding	59.0	55.1

(a) Effect on Position Encoding.

Method	PQ	PartPQ
baseline	61.6	57.4
+ boundary loss	61.5	57.2

(b) Effect of adding boundary supervision

Method	PQ	PartPQ
things and stuff only	61.2	-
+ part annotation (ours)	61.6	57.4

(c) Effect on adding part annotations.

Table 6: More analysis using our Panoptic-PartFormer.

Effectiveness of each component. As shown in Tab. 4a, we start with the effectiveness of each component of our framework by removing it from the original design. Removing Decoupled Decoder (DD) results in a 1.4 % drop on PartPQ. Removing Dynamic Convolution (DC) or Self Attention (SA) results in a large drop on PQ which means both are important for the interaction between queries and corresponding query features. Decreasing stage number to 1 also leads to a significant drop. Performing more interaction results in more accurate feature location for each query, which is the same as previous works [72,52].

Whether the part query depends more on thing query? With our framework, we can easily analyze the relationship among stuff query, thing query, and part query. We present several ways of reasoning and fusing different queries. From intuitive thought, part information is more related to thing query. We design two different query interaction methods, shown in Tab 4b. For separate reasoning, we adopt DD and SA on two query pairs, including stuff-thing query and thing-part query. For sequential reasoning, we perform DD and SA with thing-part query first and stuff-thing query second. However, we find the best model is the joint reasoning, which is the default setting described in method part. We argue that better part segmentation needs the whole scene context rather than thing features only.

Choose joint query modeling or separate modeling on part? Following PanopticFPN settings [22], we also adopt semantic-FPN like model for part segmentation. Dense Prediction (DP) is the baseline method shown in Fig. 1(b). We adopt the same merging process for panoptic segmentation and part segmentation. As shown in Tab. 4c, our joint query based method achieves the better results and outperforms previous dense prediction based approach and its im-

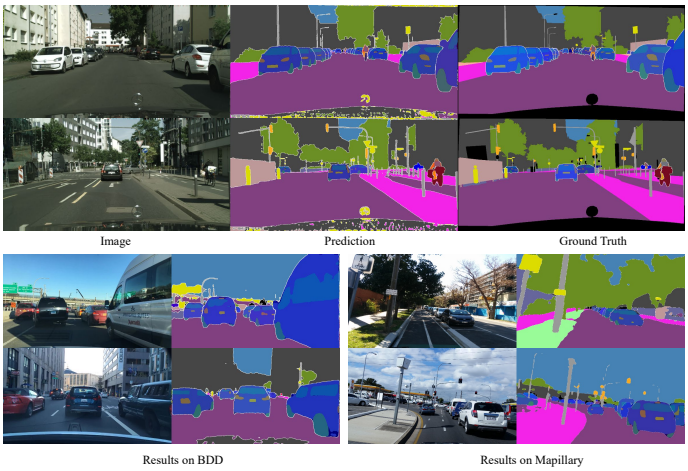

Fig. 3: Visualization of our Panoptic-PartFormer. Top: results on Cityscapes PPS validation set. Bottom left: prediction on BDD dataset [69]. Bottom right: prediction on Mapillary dataset [46]. Best view it on screen.

proved version [2]. This indicates that joint learning benefits part segmentation a lot, which proves the effectiveness of our framework.

Aligned decoder is more important for part segmentation. As shown in Tab. 4d, using aligned part decoder results in better PartPQ especially for the things with Part (P). Adding both paths with aligned decoder does not bring extra gain. This verifies our motivation: part segmentation needs more detailed information, while thing and stuff predictions do not need it.

Necessity of Positional Encoding on X_p and X_u. In Tab. 6a, removing positional encoding leads to inferior results on both PQ and PartPQ which indicates the importance of position information [62,72,1].

Will boundary supervision help for part segmentation? In Tab. 6b, we also add boundary supervision for part segmentation where we use the dice loss [44] and binary cross entropy loss. However, we find no gains on this since our mask is generated from aligned decoder since it already contains detailed information.

Will joint training help for panoptic segmentation? As shown in Tab. 6c, joint learning benefits the panoptic segmentation baseline. However, the benefit is *limited* since both thing and stuff prediction does not need much detailed information.

4.3 Analysis and Visualization

Visualization and Generalization. We give several visualization examples using our model on Cityscapes PPS validation set. Moreover, we also visualize several examples on the Mapillary dataset [46] and BDD dataset [69] to show

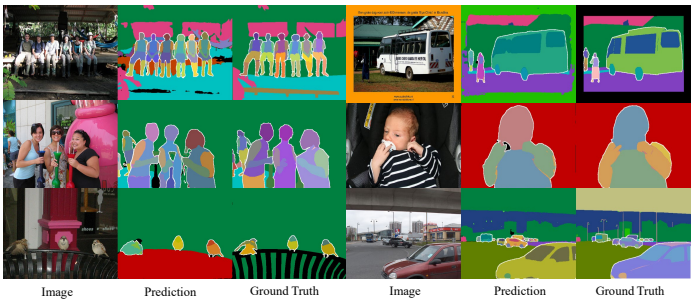

Fig. 4: More visualization results on Pascal Context Panoptic Part validation set. Best viewed in color and by zooming in. Note that stuff classes have the same color, while thing classes are not.

the generalization ability of our method. As shown in the first row of Fig. 3, our method achieves considerable results. Moreover, on the Mapillary [46] and BDD datasets [69] which do not have part annotations, our method can still work well as shown in the last row of the Fig. 3. Moreover, we also visualize the results on PPP datasets. The first two rows show the crowded human scene and outdoor scene. Both cases show that our model can obtain the convincing results. The last row shows the small objects cases. The failure cases are due to tiny objects including their parts.

Upper bound analysis of Our Model. In Tab. 4e, we give the upper bound analysis to our model by replacing the panoptic segmentation Ground Truth or part segmentation Ground Truth into our prediction. Replacing panoptic segmentation GT leads to a huge gain on PartPQ while replacing part segmentation only results in a limited gain. That indicates PartPQ is more *sensitive to panoptic segmentationt than part segmentation on CPP dataset*. We conclude that a stronger panoptic segmentation model maybe the key for better PPS results.

5 Conclusion

In this work, we present Panoptic-PartFormer, the first unified end-to-end model for Panoptic Part Segmentation. We present a decoupled decoder with three different queries to generate thing, stuff and part masks at the same time. We propose to jointly learn the three queries with corresponding query features. With this framework, we present detailed analysis of the relationship among things, stuff, and part. As a result, our method achieves the new state-of-the-art results on Cityscapes Panoptic PPS dataset and Pascal Contexct PPS dataset. Panoptic-PartFormer would serve as a unified baseline and benefit multiple level concepts scene understanding by easing the idea development.

Acknowledgement. This research is also supported by the National Key Research and Development Program of China under Grant No. 2020YFB2103402. We thank the computation resource provided by SenseTime Research.

References

1. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: ECCV (2020)
2. Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv:1706.05587 (2017)
3. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: ECCV (2018)
4. Chen, Y., Lin, G., Li, S., Bourahla, O., Wu, Y., Wang, F., Feng, J., Xu, M., Li, X.: Banet: Bidirectional aggregation network with occlusion handling for panoptic segmentation. In: CVPR (2020)
5. Cheng, B., Collins, M.D., Zhu, Y., Liu, T., Huang, T.S., Adam, H., Chen, L.C.: Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation. In: CVPR (2020)
6. Cheng, B., Schwing, A.G., Kirillov, A.: Per-pixel classification is not all you need for semantic segmentation. NeurIPS (2021)
7. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: CVPR (2017)
8. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: CVPR (2016)
9. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
10. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The Pascal Visual Object Classes (VOC) Challenge. IJCV **88**(2), 303–338 (2010)
11. Fang, H.S., Lu, G., Fang, X., Xie, J., Tai, Y.W., Lu, C.: Weakly and semi supervised human body part parsing via pose-guided knowledge transfer. In: CVPR (2018)
12. Fang, Y., Yang, S., Wang, X., Li, Y., Fang, C., Shan, Y., Feng, B., Liu, W.: Instances as queries. arXiv preprint arXiv:2105.01928 (2021)
13. Gao, N., Shan, Y., Wang, Y., Zhao, X., Yu, Y., Yang, M., Huang, K.: Ssap: Single-shot instance segmentation with affinity pyramid. In: ICCV (2019)
14. Geng, Q., Zhang, H., Lu, F., Huang, X., Wang, S., Zhou, Z., Yang, R.: Part-level car parsing and reconstruction in single street view images. IEEE Transactions on Pattern Analysis & Machine Intelligence (01) (2021)
15. de Geus, D., Meletis, P., Lu, C., Wen, X., Dubbelman, G.: Part-aware panoptic segmentation. In: CVPR (2021)
16. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. pp. 249–256. JMLR Workshop and Conference Proceedings (2010)
17. Gong, K., Liang, X., Li, Y., Chen, Y., Yang, M., Lin, L.: Instance-level human parsing via part grouping network. In: ECCV (2018)
18. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV (2017)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
20. Hou, R., Li, J., Bhargava, A., Raventos, A., Guizilini, V., Fang, C., Lynch, J., Gaidon, A.: Real-time panoptic segmentation from dense detections. In: CVPR (2020)

21. Ji, R., Du, D., Zhang, L., Wen, L., Wu, Y., Zhao, C., Huang, F., Lyu, S.: Learning semantic neural tree for human parsing. In: ECCV (2020)
22. Kirillov, A., Girshick, R., He, K., Dollár, P.: Panoptic feature pyramid networks. In: CVPR (2019)
23. Kirillov, A., He, K., Girshick, R., Rother, C., Dollár, P.: Panoptic segmentation. In: CVPR (2019)
24. Li, J., Zhao, J., Wei, Y., Lang, C., Li, Y., Sim, T., Yan, S., Feng, J.: Multiple-human parsing in the wild. arXiv preprint arXiv:1705.07206 (2017)
25. Li, J., Raventos, A., Bhargava, A., Tagawa, T., Gaidon, A.: Learning to fuse things and stuff. arXiv:1812.01192 (2018)
26. Li, Q., Arnab, A., Torr, P.H.: Holistic, instance-level human parsing. arXiv preprint arXiv:1709.03612 (2017)
27. Li, Q., Qi, X., Torr, P.H.: Unifying training and inference for panoptic segmentation. In: CVPR (2020)
28. Li, X., You, A., Zhu, Z., Zhao, H., Yang, M., Yang, K., Tong, Y.: Semantic flow for fast and accurate scene parsing. In: ECCV (2020)
29. Li, Y., Chen, X., Zhu, Z., Xie, L., Huang, G., Du, D., Wang, X.: Attention-guided unified network for panoptic segmentation. In: CVPR (2019)
30. Li, Y., Zhao, H., Qi, X., Chen, Y., Qi, L., Wang, L., Li, Z., Sun, J., Jia, J.: Fully convolutional networks for panoptic segmentation with point-based supervision. arXiv preprint arXiv:2108.07682 (2021)
31. Li, Y., Zhao, H., Qi, X., Wang, L., Li, Z., Sun, J., Jia, J.: Fully convolutional networks for panoptic segmentation. CVPR (2021)
32. Liang, J., Homayounfar, N., Ma, W.C., Xiong, Y., Hu, R., Urtasun, R.: Polytransform: Deep polygon transformer for instance segmentation. In: CVPR (2020)
33. Liang, X., Xu, C., Shen, X., Yang, J., Liu, S., Tang, J., Lin, L., Yan, S.: Human parsing with contextualized convolutional neural network. In: ICCV (2015)
34. Lin, J., Yang, H., Chen, D., Zeng, M., Wen, F., Yuan, L.: Face Parsing with RoI Tanh-Warping. In: CVPR (2019)
35. Lin, T.Y., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection. In: CVPR (2017)
36. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: ICCV (2017)
37. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014)
38. Liu, S., Sun, Y., Zhu, D., Ren, G., Chen, Y., Feng, J., Han, J.: Cross-domain human parsing via adversarial feature and label adaptation. In: AAAI (2018)
39. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. ICCV (2021)
40. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization (2017)
41. Meinhardt, T., Kirillov, A., Leal-Taixe, L., Feichtenhofer, C.: Trackformer: Multi-object tracking with transformers. arXiv preprint arXiv:2101.02702 (2021)
42. Michieli, U., Borsato, E., Rossi, L., Zanuttigh, P.: GMNet: Graph Matching Network for Large Scale Part Semantic Segmentation in the Wild. In: ECCV (2020)
43. Milletari, F., Navab, N., Ahmadi, S.: V-Net: Fully convolutional neural networks for volumetric medical image segmentation. In: 3DV (2016)
44. Milletari, F., Navab, N., Ahmadi, S.A.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: 3DV (2016)
45. Mohan, R., Valada, A.: Efficientps: Efficient panoptic segmentation. International Journal of Computer Vision **129**(5), 1551–1579 (2021)

46. Neuhold, G., Ollmann, T., Rota Bulo, S., Kontschieder, P.: The mapillary vistas dataset for semantic understanding of street scenes. In: ICCV (2017)
47. Porzi, L., Bulo, S.R., Colovic, A., Kontschieder, P.: Seamless scene segmentation. In: CVPR (2019)
48. Qi, S., Wang, W., Jia, B., Shen, J., Zhu, S.C.: Learning human-object interactions by graph parsing neural networks. In: ECCV (2018)
49. Qiao, S., Chen, L.C., Yuille, A.: Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. In: CVPR (2021)
50. Ruan, T., Liu, T., Huang, Z., Wei, Y., Wei, S., Zhao, Y.: Devil in the details: Towards accurate single and multiple human parsing. In: AAAI (2019)
51. Shen, Z., Wang, W., Lu, X., Shen, J., Ling, H., Xu, T., Shao, L.: Human-aware motion deblurring. In: ICCV (2019)
52. Sun, P., Zhang, R., Jiang, Y., Kong, T., Xu, C., Zhan, W., Tomizuka, M., Li, L., Yuan, Z., Wang, C., Luo, P.: SparseR-CNN: End-to-end object detection with learnable proposals. CVPR (2021)
53. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: ICML. pp. 6105–6114. PMLR (2019)
54. Tian, Z., Shen, C., Chen, H.: Conditional convolutions for instance segmentation. arXiv preprint arXiv:2003.05664 (2020)
55. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. In: ICML. PMLR (2021)
56. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. arXiv preprint arXiv:1706.03762 (2017)
57. Wang, H., Zhu, Y., Adam, H., Yuille, A., Chen, L.C.: Max-deeplab: End-to-end panoptic segmentation with mask transformers. CVPR (2021)
58. Wang, H., Zhu, Y., Green, B., Adam, H., Yuille, A., Chen, L.C.: Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In: ECCV (2020)
59. Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., et al.: Deep high-resolution representation learning for visual recognition. PAMI (2020)
60. Wang, W., Zhang, Z., Qi, S., Shen, J., Pang, Y., Shao, L.: Learning compositional neural information fusion for human parsing. In: ICCV (2019)
61. Wang, X., Kong, T., Shen, C., Jiang, Y., Li, L.: SOLO: Segmenting objects by locations. In: ECCV (2020)
62. Wang, X., Zhang, R., Kong, T., Li, L., Shen, C.: SOLOv2: Dynamic and fast instance segmentation. In: NeurIPS (2020)
63. Wu, Y., Zhang, G., Xu, H., Liang, X., Lin, L.: Auto-panoptic: Cooperative multi-component architecture search for panoptic segmentation. NIPS (2020)
64. Xiong, Y., Liao, R., Zhao, H., Hu, R., Bai, M., Yumer, E., Urtasun, R.: Upsnet: A unified panoptic segmentation network. In: CVPR (2019)
65. Yang, L., Song, Q., Wang, Z., Hu, M., Liu, C., Xin, X., Jia, W., Xu, S.: Renovating parsing R-CNN for accurate multiple human parsing. In: ECCV (2020)
66. Yang, L., Song, Q., Wang, Z., Jiang, M.: Parsing R-CNN for instance-level human analysis. In: CVPR (2019)
67. Yang, T.J., Collins, M.D., Zhu, Y., Hwang, J.J., Liu, T., Zhang, X., Sze, V., Papandreou, G., Chen, L.C.: Deeperlab: Single-shot image parser. arXiv:1902.05093 (2019)
68. Yang, Y., Li, H., Li, X., Zhao, Q., Wu, J., Lin, Z.: Sognet: Scene overlap graph network for panoptic segmentation. In: AAAI (2020)

69. Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., Darrell, T.: Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In: CVPR. pp. 2636–2645 (2020)
70. Yuan, Y., Chen, X., Wang, J.: Object-contextual representations for semantic segmentation. ECCV (2020)
71. Zhang, H., Wu, C., Zhang, Z., Zhu, Y., Lin, H., Zhang, Z., Sun, Y., He, T., Mueller, J., Manmatha, R., et al.: Resnest: Split-attention networks. arXiv preprint arXiv:2004.08955 (2020)
72. Zhang, W., Pang, J., Chen, K., Loy, C.C.: K-net: Towards unified image segmentation. NeurIPS (2021)
73. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: CVPR (2017)
74. Zhao, J., Li, J., Cheng, Y., Sim, T., Yan, S., Feng, J.: Understanding humans in crowded scenes: Deep nested adversarial learning and a new benchmark for multi-human parsing. In: MM (2018)
75. Zhao, Y., Li, J., Zhang, Y., Tian, Y.: Multi-Class Part Parsing With Joint Boundary-Semantic Awareness. In: ICCV (2019)
76. Zhou, T., Wang, W., Liu, S., Yang, Y., Van Gool, L.: Differentiable multi-granularity human representation learning for instance-aware human semantic parsing. In: CVPR (2021)
77. Zhu, X., Hu, H., Lin, S., Dai, J.: Deformable convnets v2: More deformable, better results. In: CVPR (2019)
78. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. ICLR (2020)