BANet: Bidirectional Aggregation Network
with Occlusion Handling for Panoptic Segmentation

Yifeng Chen[1], Guangchen Lin[1], Songyuan Li[1], Omar Bourahla[1], Yiming Wu[1],
Fangfang Wang[1], Junyi Feng[1], Mingliang Xu[2], Xi Li[1]*

[1]Zhejiang University, [2]Zhengzhou University

{yifengchen, aaronlin, leizungjyun, xilizju}@zju.edu.cn

Abstract

Panoptic segmentation aims to perform instance segmentation for foreground instances and semantic segmentation for background stuff simultaneously. The typical top-down pipeline concentrates on two key issues: 1) how to effectively model the intrinsic interaction between semantic segmentation and instance segmentation, and 2) how to properly handle occlusion for panoptic segmentation. Intuitively, the complementarity between semantic segmentation and instance segmentation can be leveraged to improve the performance. Besides, we notice that using detection/mask scores is insufficient for resolving the occlusion problem. Motivated by these observations, we propose a novel deep panoptic segmentation scheme based on a bidirectional learning pipeline. Moreover, we introduce a plug-and-play occlusion handling algorithm to deal with the occlusion between different object instances. The experimental results on COCO panoptic benchmark validate the effectiveness of our proposed method. Codes will be released soon at https://github.com/Mooonside/BANet.

1. Introduction

Panoptic segmentation [19], an emerging and challenging problem in computer vision, is a composite task unifying both semantic segmentation (for background stuff) and instance segmentation (for foreground instances). A typical solution to the task is in a top-down deep learning manner-whereby instances are first identified and then assigned to semantic labels [22, 23, 28, 38]. In this way, two key issues arise out of a robust solution: 1) how to effectively model the intrinsic interaction between semantic segmentation and instance segmentation, and 2) how to robustly handle the occlusion for panoptic segmentation.
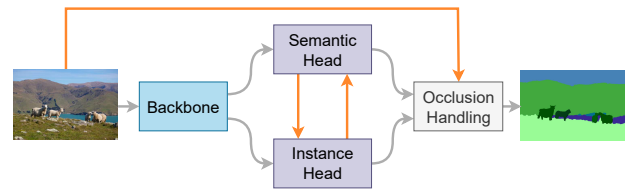

Figure 1. The illustration of BANet. We introduce a bidirectional path to leverage the complementarity between semantic and instance segmentation. To obtain the panoptic segmentation results, low-level appearance information is utilized in the occlusion handling algorithm.

In principle, the complementarity does exist between the tasks of semantic segmentation and instance segmentation. Semantic segmentation concentrates on capturing the rich pixel-wise class information for scene understanding. Such information could work as useful contextual clues to enrich the features for instance segmentation. Conversely, instance segmentation gives rise to the structural information (e.g., shape) on object instances, which enhances the discriminative power of the feature representation for semantic segmentation. Hence, the interaction between these two tasks is bidirectionally reinforced and reciprocal. However, previous works [22, 23, 38] usually take a unidirectional learning pipeline to use score maps from instance segmentation to guide semantic segmentation, resulting in the lack of a path from semantic segmentation to instance segmentation. Besides, the information contained by these instance score maps is often coarse-grained with a very limited channel size, leading to the difficulty in encoding more fine-grained structural information for semantic segmentation.

In light of the above issue, we propose a Bidirectional Aggregation NETwork, dubbed BANet, for panoptic segmentation to model the intrinsic interaction between semantic segmentation and instance segmentation at the feature level. Specifically, BANet possesses bidirectional paths for feature aggregation between these two tasks, which respectively correspond to two modules: Instance-

---

*Corresponding author, xilizju@zju.edu.cn

To-Semantic (I2S) and Semantic-To-Instance (S2I). S2I passes the context-abundant features from semantic segmentation to instance segmentation for localization and recognition. Meanwhile, the instance-relevant features, attached with more structural information, are fed back to semantic segmentation to enhance the discriminative capability of the semantic features. To achieve a precise instance-to-semantic feature transformation, we design the ROIInlay operator based on bilinear interpolation. This operator is capable of restoring the structure of cropped instance features so that they can be aggregated with the semantic features for semantic segmentation.

After the procedures of semantic and instance segmentation, we need to fuse their results into the panoptic format. During this fusion process, a key problem is to reason the occlusion relationships for the occluded parts among object instances. A conventional way [11, 19, 28, 38] relies heavily on detection/mask scores, which are often inconsistent with the actual spatial ranking relationships of object instances. For example, a tie usually overlaps a person, but it tends to get a lower score (due to class imbalance). With this motivation, we propose a learning-free occlusion handling algorithm based on the affinity between the overlapped part and each object instance in the low-level appearance feature space. It compares the similarity between occluded parts and object instances and assigns each part to the object of the closest appearance.

In summary, the contributions of this work are as follows:

- We propose a deep panoptic segmentation scheme based on a bidirectional learning pipeline, namely Instance-To-Semantic (I2S) and Semantic-To-Instance (S2I) to enable feature-level interaction between instance segmentation and semantic segmentation.

- We present the ROIInlay operator to achieve the precise instance-to-semantic feature mapping from the cropped bounding boxes to the holistic scene image.

- We propose a simple yet effective learning-free approach to handle the occlusion, which can be plugged in any top-down based network.

2. Related Work

Semantic segmentation Semantic segmentation, the task of assigning a semantic category to each pixel in an image, has made great progress recently with the development of the deep CNNs in a fully convolutional fashion (FCN[32]). It has been known that contextual information is beneficial for segmentation [8, 12, 15, 17, 20, 21, 33, 36], and these models usually provide a mechanism to exploit it. For example, PSPNet [41] features global pyramid pooling which provides additional contextual information to FCN. Feature Pyramid Network (FPN) [26] takes features from different layers as multi-scale information and stacks them to a feature pyramid. DeepLab series [5, 6] apply several architectures with atrous convolution to capture multi-scale context. In our work, we focus on utilizing features from semantic segmentation to help instance segmentation instead of designing a sophisticated context mechanism.

Instance segmentation Instance segmentation assigns a category and an instance identity to each object pixel in an image. Methods for instance segmentation fall into two main categories: top-down and bottom-up. The top-down, or proposal-based, methods [4, 9, 10, 16, 24, 30, 35] first generate bounding boxes for object detection, and then perform dense prediction for instance segmentation. The bottom-up, or segmentation-based, methods [1, 7, 13, 25, 29, 31, 34, 37, 39, 40] first perform pixel-wise semantic segmentation, and then extract instances out of grouping. Top-down approaches dominates the leaderboards of instance segmentation. We adopt this manner for the instance segmentation branch in our pipeline. Chen *et al*. [2] made use of semantic features in instance segmentation. Our approach is different from it in that we design a bidirectional path between instance segmentation and semantic segmentation.

Panoptic segmentation Panoptic segmentation unifies semantic and instance segmentation, and therefore its methods can also fall into top-down and bottom-up categories on the basis of their strategy to do instance segmentation. Kirillov *et al*. [19] proposed a baseline that combines the outputs from Mask-RCNN [16] and PSPNet [41] by heuristic fusion. De Geus *et al*. [11] and Kirillov *et al*. [18] proposed end-to-end networks with multiple heads for panoptic segmentation. To model the internal relationship between instance segmentation and semantic segmentation, previous works [22, 23] utilized class-agnostic score maps to guide semantic segmentation.

To solve occlusion between objects, Liu *et al*. [28] proposed a spatial ranking module to predict the ranking of objects and Xiong *et al*. [38] proposed a parameter-free module to bring explicit competition between object scores and semantic logits.

Our approach is different from previous works in three ways. 1) We utilize instance features instead of coarse-grained score maps to improve the discriminative ability of semantic features. 2) We build a path from semantic segmentation to instance segmentation. 3) We make use of low-level appearance to resolve occlusion.

3. Methods

Our BANet contains four major components: a backbone network, the Semantic-To-Instance (S2I) module, the
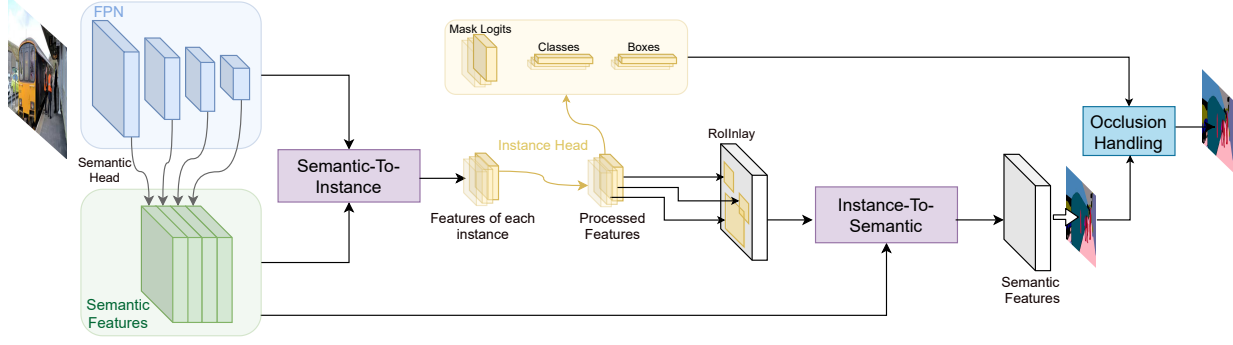
Figure 2. Our framework takes advantage of complementarity between semantic and instance segmentation. This is shown through two key modules, namely, Semantic-To-Instance (S2I) and Instance-To-Semantic (I2S). S2I uses semantic features to enhance instance features. I2S uses instance features restored by the proposed RoIInlay operation for better semantic segmentation. After performing instance and semantic segmentation, the occlusion handling module is applied to determine the belonging of occluded pixels and merge the instance and semantic outputs as the final panoptic segmentation.

Instance-To-Semantic (I2S) module and an occlusion handling module, as shown in Figure 2, We adopt ResNet-FPN as the backbone. The S2I module aims to use semantic features to help instance segmentation as described in Section 3.1. The I2S module assists semantic segmentation with instance features as described in Section 3.2. In Section 3.3, an occlusion handling algorithm is proposed to deal with instance occlusion.

3.1. Instance Segmentation

Instance segmentation is the task of localizing, classifying and predicting a pixel-wise mask for each instance. We propose the S2I module to bring about contextual clues for the benefit of instance segmentation, as illustrated in Figure 3. The semantic features F_S are obtained by applying a regular semantic segmentation head on the FPN features $\{P_i\}_{i=2...5}$.

For each instance proposal, we crop semantic features F_S and the selected FPN features P_i by RoIAlign [16]. These features are denoted by F_S^{crop} and P_i^{crop}. The proposals we use here are obtained by feeding FPN features into a regular RPN head.

After that, F_S^{crop} and P_i^{crop} are aggregated as follows:

$$F_{\text{S2I}} = \phi(F_S^{\text{crop}}) + P_i^{\text{crop}}, \qquad (1)$$

where ϕ is a 1×1 convolution layer to align the feature spaces. The aggregated features F_{S2I} benefit from contextual information from F_S^{crop} and spatial details from P_i^{crop}.

F_{S2I} is fed into a regular instance segmentation head to predict masks, boxes and categories for instances. The specific design of the instance head follows [16]. For mask predictions, three 3×3 convolutions are applied to F_{S2I} to extract instance-wise features F_{ins}. Then a deconvolution layer up-samples the features and predicts object-wise masks of 28×28. Meanwhile, fully connected layers are applied to F_{S2I} to predict boxes and categories. Note that F_{ins} is later used in Section 3.2.
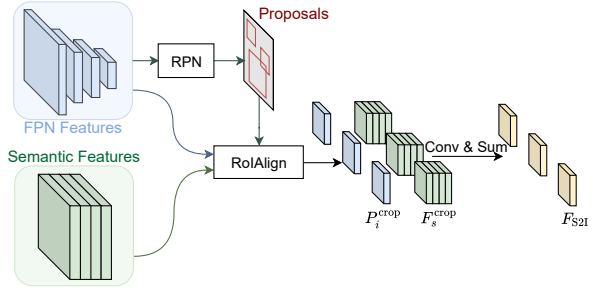

Figure 3. The architecture of our S2I module. For each instance, S2I crops semantic features and the selected FPN features of the instance and then aggregates the cropped features. As a result, it enhances instance segmentation by semantic information.

3.2. Semantic Segmentation

Semantic segmentation assigns each pixel with a class label. Our framework utilizes instance features to introduce structural information to semantic features. It does so through our I2S module which uses F_{ins} from the previous section. However, F_{ins} cannot be fused with semantic feature F_S directly since it is already cropped and resized. To solve this issue, we propose the RoIInlay operation, which maps F_{ins} back into a feature map F_{inlay} with the same spatial size as F_S. This restores the structure of each instance, allowing us to efficiently use it in semantic segmentation.

After obtaining F_{inlay}, we use it along with F_S to perform semantic segmentation. As shown in Figure 4, these two features are aggregated in two modules, namely Structure Injection Module (SIM) and Object Context Module (OCM). In SIM, F_{inlay} and F_S are first projected to the same feature space. Then, they are concatenated and go through a 3×3 convolution layer to alleviate possible distortions caused by RoIInlay. By doing so, we inject the structure information of F_{inlay} into the semantic feature F_S.

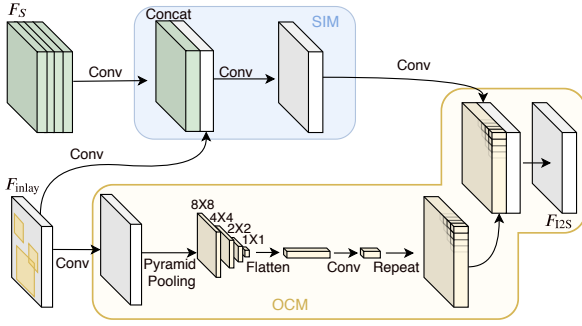OCM takes the output of SIM and further enhances it by information on the objects' layout in the scene.

Figure 4. The architecture of the I2S module. SIM uses instance features restored by RoIInlay and combines them with semantic features. Meanwhile, OCM extracts information on the objects' layout in the scene. After that, OCM combines it with SIM's output for use in semantic segmentation.

As shown in Figure 4, we first project F_{inlay} into a space of E dimension ($E = 10$). Then, a pyramid of max-pooling is applied to get multi-scale descriptions of the objects' layout. These descriptions are flattened, concatenated and projected to obtain an encoding of the layout. This encoding is repeated horizontally and vertically, and concatenated with the output of SIM. Finally, the concatenated features are projected as F_{I2S}.

F_{I2S} is then used to predict semantic segmentation which will be later used to obtain the panoptic result.

Extraction of semantic features   To extract F_S, we use a semantic head with a design that follows [38]. A subnet of three stacked 3×3 convolutions is applied to each FPN feature. After that, they are upsampled and concatenated to form F_S.

RoIInlay   RoIInlay aims to restore features cropped by operations such as RoIAlign back to their original structure. In particular, RoIInaly resizes the cropped feature and inlays it in an empty feature map at the correct location, namely at the position from which it was first cropped.

As a patch-recovering operator, RoIInlay shares a common purpose with RoIUpsample [23], but RoIInlay has two advantages over RoIUpsample thanks to its different interpolation style, as shown in Figure 5. RoIUpsample obtains values through a modified gradient function of bilinear interpolation. RoIInlay applies the bilinear interpolation carried out in the relative coordination of sampling points (used in RoIAlign). Therefore, it can both avoid "holes", *i.e.* pixels whose values cannot be recovered and interpolate more accurately. More comparisons on these two operators can be found in the supplementary material.

Recall that in RoIAlign [16], $m \times m$ sampling points are generated to crop a region. The resulting feature is thus divided into a group of $m \times m$ bins with a sampling point at the center of each bin. Given a region of size (w_r, h_r) the size of each bin will be $b_h = h_r/m$ and $b_w = w_r/m$ and
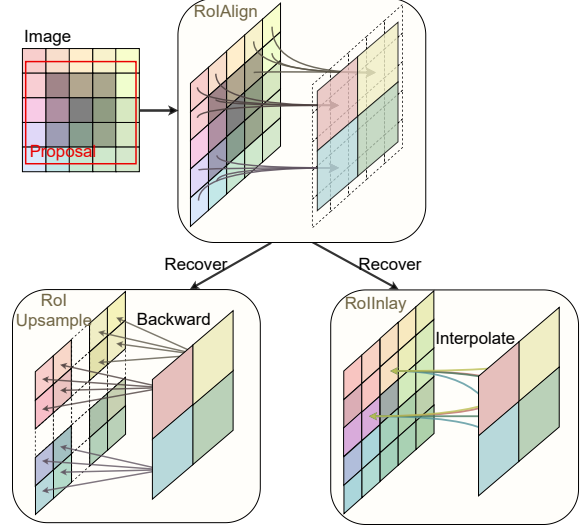

Figure 5. The difference between RoIUpsample and our RoIInlay. Both RoIUpsample and RoIInlay restore features cropped by RoIAlign. However, RoIUpsample only uses a single reference for each pixel whereas RoIInlay uses four references and does not suffer from pixels with unassigned values.

the value at each sampling point is obtained by interpolating from the 4 closest pixels as shown in Figure 5.

Given the positions and values of each sampling point, RoIInlay aims to recover values of pixels within the region. To achieve this, it is designed as a bilinear interpolation carried out in the relative coordinates of sampling points. Specifically, for a pixel located at (a, b), we find its four nearest sampling points $\{(x_i, y_i), i \in [1, 4]\}$. The value at (a, b) is calculated as:

$$v(a,b) = \sum_{i=1}^{4} G(a, x_i, b_w) G(b, y_i, b_h) v(x_i, y_i), \quad (2)$$

where $v(x_i, y_i)$ is the value of sampling point (x_i, y_i), (b_h, b_w) is the size of each sampling bin and G is the bilinear interpolation kernel in the relative coordinates of sampling points:

$$G(a, x_i, b_w) = 1.0 - \frac{|a - x_i|}{b_w}. \quad (3)$$

Pixels within the region but out of the boundary of sampling points are calculated as if they were positioned at the boundary. To handle cases where different objects may generate values at the same position, we take the average of these values to maintain the scale.

3.3. Occlusion Handling

Occlusion occurs during instance segmentation when a pixel x is claimed by multiple objects $\{O_1, \ldots, O_k\}$. To get the final panoptic result, we must resolve the overlap relationships among objects so that x is assigned to just one

object. We argue that low-level appearance is a strong visual cue for the spatial ranking of objects compared to semantic features or instance features. The former contains mostly category information, which cannot resolve the occlusion of the objects belonging to the same class, while the latter loses details after RoIAlign, which are fatal when small objects (*e.g.* tie) overlaps big ones (*e.g.* person).

By utilizing appearance as the reference, we propose a novel occlusion handling algorithm that assigns pixels to the most similar object instance. To compare the similarity between a pixel x and an object instance O_i, we need to define a measure $f(x, O_i)$. In this algorithm, we adopt the cosine similarity between the RGB of pixel x and each object instance O_i (represented by its average RGB values).

After calculating the similarity between x and each object, we assign x to O^*, where

$$O^* = \text{argmax}_{O_i} f(x, O_i) \qquad (4)$$

In practice, instead of considering individual pixels, we consider them in sets, which will lead to more stable results. To compare between an object and a pixel set, we average over the similarity of that object with each pixel in the set.

Through this learning-free algorithm, the instance assignment of each pixel is resolved. After that, we combine it with the semantic segmentation for the final panoptic results according to the procedures in [19].

3.4. Training and Inference

Training During training, we sample ground truth detection boxes and only apply RoIInlay on features of sampled objects. The sampling rate is chosen randomly from 0.6 to 1, where at least one ground truth box is kept. There are seven loss items in total. The RPN proposal head contains two losses: L_{rpn_cls} and L_{rpn_box}. The instance head contains three losses: L_{cls} (bbox classification loss), L_{box} (bbox regress loss) and L_{mask} (mask prediction loss). The semantic head contains two losses: L_{seg} (semantic segmentation from F_S) and L_{I2S} (semantic segmentation from F_{I2S}). The total loss function L is :

$$L = \underbrace{L_{rpn\_cls} + L_{rpn\_box}}_{\text{rpn proposal loss}} + \underbrace{L_{cls} + L_{box} + L_{mask}}_{\text{instance segmentation loss}}$$
$$+ \underbrace{\lambda_s L_{seg} + \lambda_i L_{I2S}}_{\text{semantic segmentation loss}} , \qquad (5)$$

where λ_s and λ_i are loss weights to control the balance between semantic segmentation and other tasks.

Inference During inference, predictions from instance head are sent to the occlusion handling module. It first performs non-maximum-suppression (NMS) to remove duplicate predictions. Then the occluded objects are identified and their conflicts are solved based on appearance similarity. Afterwards, the occlusion-resolved instance prediction

is combined with semantic segmentation prediction following [19], where instances always overwrite stuff regions. Finally, stuff regions are removed and labeled as "void" if their areas are below a certain threshold.

4. Experiments

4.1. Datasets

We evaluate our approach on MS COCO [27], a large-scale dataset with annotations of both instance segmentation and semantic segmentation. It contains 118k training images, 5k validation images, and 20k test images. The panoptic segmentation task in COCO includes 80 thing categories and 53 stuff categories. We train our model on the *train* set without extra data and report results on both *val* and *test-dev* sets.

4.2. Evaluation Metrics

Single-task metrics For semantic segmentation, the mIoU$^{\text{Sf}}$ (mean Intersection-over-Union averaged over stuff categories) is reported. We do not report the mIoU over thing categories since the semantic segmentation prediction of thing classes will not be used in the fusion algorithm. For instance segmentation, we report AP$_{\text{mask}}$, which is averaged between categories and IoU thresholds [27].

Panoptic segmentation metrics We use PQ [19] (averaged over categories) as the metric for panoptic segmentation. It captures both recognition quality (RQ) and segmentation quality (SQ):

$$\text{PQ} = \underbrace{\frac{\sum_{(p,g)\in TP} \text{IoU}(p,g)}{|TP|}}_{\text{segmentation quality(SQ)}} \times \underbrace{\frac{|TP|}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}}_{\text{recognition quality(RQ)}}, \qquad (6)$$

where $\text{IoU}(p,g)$ is the intersection-over-union between a predicted segment p and the ground truth g, TP refers to matched pairs of segments, FP denotes the unmatched predictions and FN represents the unmatched ground truth segments. Additionally, PQ$^{\text{Th}}$ (average over thing categories) and PQ$^{\text{Sf}}$ (average over stuff categories) are reported to reflect the improvement on instance and semantic segmentation segmentation.

4.3. Implementation Details

Our model is based on the implementation in [3]. We extend the Mask-RCNN with a stuff head, and treat it as our baseline model. ResNet-50-FPN and DCN-101-FPN [10] are chosen as our backbone for *val* and *test-dev* respectively.

We use the SGD optimization algorithm with momentum of 0.9 and weight decay of 1e-4. For the model based on ResNet-50-FPN, we follow the 1x training schedule in [14]. In the first 500 iterations, we adopt the linear warmup policy to increase the learning rate from 0.002 to 0.02. Then it

Models	Subset	Backbone	PQ	SQ	RQ	PQTh	SQTh	RQTh	PQ^{Sf}	SQ^{Sf}	RQ^{Sf}
JSIS-Net [11]	*val*	ResNet-50-FPN	26.9	72.4	35.7	29.3	72.1	39.2	23.3	72.0	30.4
Panoptic FPN [18]	*val*	ResNet-50-FPN	39.0	-	-	45.9	-	-	28.7	-	-
OANet [28]	*val*	ResNet-50-FPN	39.0	77.1	47.8	48.3	**81.4**	58.0	24.9	70.6	32.5
AUNet [23]	*val*	ResNet-50-FPN	39.6	-	-	**49.1**	-	-	25.2	-	-
Ours	*val*	ResNet-50-FPN	**41.1**	**77.2**	**51**	**49.1**	80.4	**60.3**	**29.1**	**72.4**	**37.1**
UpsNet† [38]	*val*	ResNet-50-FPN	42.5	78.0	52.4	48.5	79.5	59.6	**33.4**	**76.3**	**41.6**
Ours†	*val*	ResNet-50-FPN	**43.0**	**79.0**	**52.8**	**50.5**	**81.1**	**61.5**	31.8	75.9	39.4
AUNet [23]	*test-dev*	ResNeXt-152-FPN	46.5	**81.0**	56.1	**55.9**	**83.7**	**66.3**	32.5	77.0	40.7
UpsNet† [38]	*test-dev*	DCN-101-FPN	46.6	80.5	56.9	53.2	81.5	64.6	**36.7**	78.9	**45.3**
Ours†	*test-dev*	DCN-101-FPN	**47.3**	80.8	**57.5**	54.9	82.1	**66.3**	35.9	**78.9**	44.3

Table 1. Comparison with state-of-the-art methods on COCO *val* and *test-dev* set. † refers to deformable convolution.

is divided by 10 at 60k iterations and 80k iterations respectively. For the model based on DCN-101-FPN, we follow the 3x training schedule in [14] and apply multi-scale training. The learning rate setting of the 3x schedule is adjusted in proportion to the 1x schedule. As for data augmentation, the shorter edge is resized to 800, while the longer side is kept below 1333. Random crop and horizontal flip are used. When training models containing I2S, we set λ_s to 0.2 and λ_i to 0.3. For models without I2S, λ_s is set to 0.5 since there is no L_{I2S} left. For models that contain deformable convolutions, we set λ_s to 0.1 and λ_i to 0.2.

NMS is applied to all candidates whose scores are higher than 0.6 in a class-agnostic way, and its threshold is set to 0.5. In the occlusion handling algorithm, we first define the occluded pair as follows. For two objects A and B, the pair (A, B) is treated an occluded pair when the overlap area is larger than 20% of either A or B. When overlap ratio is less than 20%, objects with higher scores simply overwrite the others. For all occluded pairs, we assign the overlapping part to the object with closer appearance as described in Section 3.3. To handle the occlusion involving more than two objects, we deal with overlapping object pairs in descending order of pair scores, the higher object's score in each pair. As for interweaving cases, where objects overlap each other, we would set aside the contradictory pairs with lower scores. For example, let $A \rightarrow B$ denotes that object A overlaps object B. Given $A \rightarrow B$, $C \rightarrow A$, $B \rightarrow C$ in an image with their pair scores in descending order, we would set $B \rightarrow C$ aside. If more than 50% of an object is assigned to other objects, we remove it from the scene.

After that, we resolve the conflicts between instances and stuff by prioritizing instances. Finally, we remove stuff regions whose areas are under 4096, as described in [19].

4.4. Comparison with State-of-the-Art Methods

In Table 1, we compare our method with other state-of-the-art methods [11] on COCO *val* and *test-dev* set.

When comparing to methods without deformable convolution, our model outperforms them with respect to nearly all metrics on COCO *val*. It achieves especially higher re-

sults at both SQ and RQ, showing that it is well-balanced between segmentation and recognition. By applying deformable convolutions in the network, our approach gains a clear improvement at PQ (from 41.1% to **43.0%**) and outperforms UpsNet on most of the metrics. When it comes to the performance on things, we achieved 50.5% at PQTh which exceeds UpsNet by 2%. The improvement of PQTh comes from having better SQTh(+1.6%) and RQTh(+1.9%). As for the performance on stuff, our method is inferior to UpsNet since we simply resolve the conflict between instances and segmentation in favor of instances.

On COCO *test-dev* set, our model based on DCN-101-FPN achieves a consistently higher performance of **47.3%** PQ (0.7% higher than UPSNet).

4.5. Ablation Study

We perform ablation studies on COCO *val* with our model based on ResNet50-FPN. We study the effectiveness of our modules by adding them one-by-one to the baseline.

Instance-to-semantic To study the effect of Instance-To-Semantic (I2S), we run experiments with SIM alone and with both SIM and OCM. As shown in the second row of Table 2, applying SIM alone leads to a 0.4% gain in terms of PQ. We notice that both SQTh and SQ^{Sf} get improved by more than 1%. This demonstrates that SIM utilizes the recovered structural information to help semantic segmentation. Applying OCM together with SIM leads to another 0.5% improvement in terms of PQ. Thanks to the object layout context provided by OCM, our model recognizes stuff regions better, resulting in 1.3% improvement w.r.t. RQ^{Sf}.

Semantic-to-instance We apply S2I together with I2S, *i.e.*, SIM and OCM. It turns out that S2I module can effectively improve RQTh(+0.4%) by introducing complementary contextual information from semantic segmentation. The instance segmentation metric AP$_{mask}$ gets improved by 0.3% as well. Although the semantic segmentation on stuff region (mIoU^{Sf}) maintains the same, PQ^{Sf} is slightly improved by 0.2% due to better thing predictions.
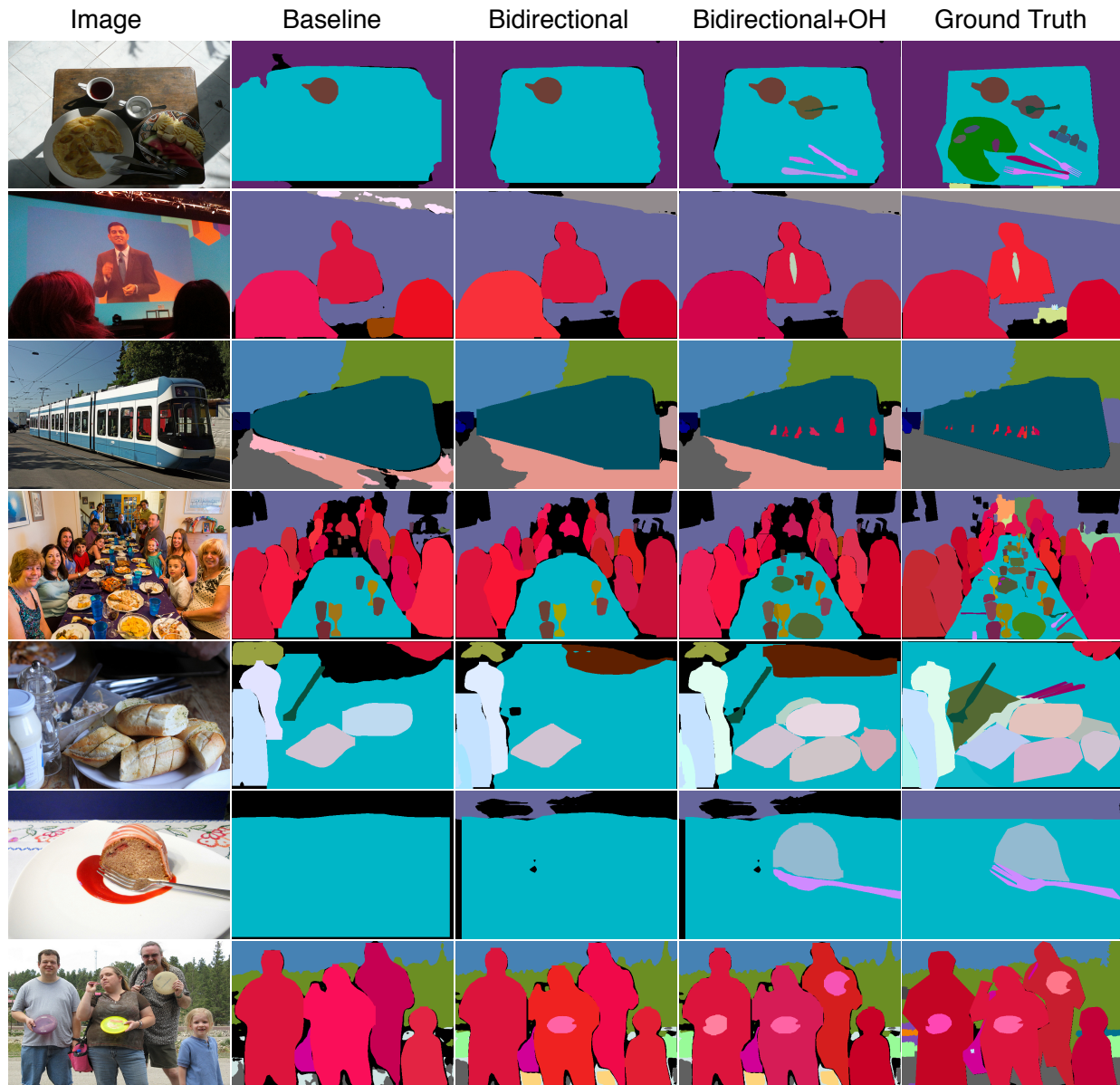
Figure 6. Visualization of panoptic segmentation results on COCO *val*. "Bidirectional" refers to the combination of S2I and I2S. "OH" represents the occlusion handling module. The figure shows the improvements gained from our modules.

Deformable convolution To validate our modules' compatibility with deformable convolution, we replace the vanilla convolution layers in the semantic head with deformable convolution layers. As shown in Table 2, deformable convolution improves our model's performance by 1.5% and is extremely helpful for "stuff" regions, as evidenced by the 1.3% increment of PQ^{Sf}.

Occlusion handling Occlusion handling is aimed at resolving occlusion between object instances and assigning occluded pixels to the correct object. Our occlusion handler makes use of local appearance (RGB) information and is completely learning-free. By applying the proposed occlusion handling algorithm, we greatly improve the recognition of things, as reflected by a 2% increase w.r.t. PQ^{Th}. Due to the better object arrangement provided by our algorithm, PQ^{Sf} is also slightly improved (+0.1%).

Different backbones We analyze the effect of the backbone by comparing different backbone networks. The performance of our model can be further improved to 44.0% by adopting a deeper ResNet-101-FPN backbone. As shown in Table 3, without the occlusion handling algorithm, the model based on ResNet-101-FPN is 0.8% higher

SIM	OCM	S2I	DFM	OH	PQ	SQ	RQ	PQ^{Th}	SQ^{Th}	RQ^{Th}	PQ^{Sf}	SQ^{Sf}	RQ^{Sf}	AP_{mask}	$mIoU^{Sf}$
					39.1	77.3	48.1	46.7	80.4	56.6	27.7	72.5	35.4	34.2	38.6
✓					39.5	78.0	48.6	47.1	81.2	57.1	28.0	73.1	35.8	34.6	39.5
✓	✓				40.0	78.4	49.1	47.2	81.6	57.0	29.2	73.5	37.1	34.8	39.7
✓	✓	✓			40.3	78.1	49.5	47.5	81.5	57.4	29.4	73.2	37.3	35.1	39.7
✓	✓	✓	✓		41.8	79.6	50.8	48.5	82.1	58.3	31.7	75.9	39.4	36.4	41.1
✓	✓	✓	✓	✓	43.0	79.0	52.8	50.5	81.1	61.5	31.8	75.9	39.5	36.4	41.1

Table 2. Ablation study on COCO *val*. 'SIM', 'OCM' are modules used in Instance-To-Semantic. S2I stands for Semantic-To-Instance. DFM stands for deformable convolution. OH refers to the occlusion handling algorithm. All results without OH are obtained by the heuristic fusion [19].

Backbone	OH	PQ	PQ^{Th}	PQ^{Sf}
ResNet-50-FPN		41.8	48.5	31.7
ResNet-101-FPN		42.5	48.6	33.4
ResNet-50-FPN	✓	43.0	50.5	31.8
ResNet-101-FPN	✓	44.0	51.0	33.4

Table 3. Experimental results for our method with different backbones.

GT Box	GT ICA	GT Occ	GT Seg	PQ	PQ^{Th}	PQ^{Sf}
				43.0	50.5	31.8
		✓		44.6	53.2	31.8
✓				47.1	56.6	32.8
✓	✓			58.4	74.8	33.5
✓	✓	✓		59.3	76.3	33.5
			✓	60.8	50.5	76.4

Table 4. Bottleneck analysis on COCO *val*. We feed different types of ground truth into our model. GT Box stands for ground truth boxes. GT ICA refers to assigning the ground truth classes to instances. GT Occ means the ground truth overlap relationship. GT Seg denotes ground truth semantic segmentation.

than ResNet-50-FPN. When both applying the occlusion handling algorithm, our model based on ResNet-101-FPN achieves 1.0% better performance than ResNet-50-FPN. This also reveals that our occlusion handling algorithm can improve PQ^{Th} consistently based on different backbones.

Bottleneck analysis To analyze the performance bottleneck of our approach, we replace parts of the intermediate results with the ground truth to see how much improvement it will lead to. Specifically, we study ground truth overlap relationships, ground truth boxes, ground truth instance class assignment and ground truth segmentation as input.

To estimate the potential of the occlusion algorithm, we feed ground truth overlaps into the model. Specifically, the predicted boxes are first matched with ground truth boxes. Then the occlusion among matched predictions is resolved using ground truth overlap relationship. The rest of the unmatched occluded predictions are handled by our occlusion handling algorithm. As shown in Table 4, when feeding ground truth overlaps, the performance PQ^{Th} increases to 53.2%. This demonstrates that there still exists a large gap between our occlusion algorithm and an ideal one.

By feeding ground truth boxes, PQ for both things and stuff sees an increase of 6.1% and 1% respectively, which indicates the maximum performance gain of a better RPN. We further assign the predictions of boxes to ground truth labels, which increases PQ^{Th} by more than 20%. This demonstrates that the lack of recognition ability on things is a main bottleneck of our model. Meanwhile, We also test feeding ground truth overlap along with ground truth box and class assignment, PQ^{Th} gets a further improvement of 2%. This shows that the occlusion problem has to be carefully dealt with even if ground truth boxes and labels are fed. Finally, we test the case when ground truth segmentation is given, the performance of PQ^{Sf} is only 76.4%. This indicates that the common fusion process that prioritizes things over stuff is far from optimal.

Visualization We show visual examples of the results obtained by our method in Figure 6. By comparing the second and third columns, we can see large improvements brought by using the bidirectional architecture, specifically, many large misclassified regions are corrected. After adding the occlusion handling module (fourth column) we notice that several conflicts of instances are resolved. This causes the accuracy of overlapping objects to increase significantly.

5. Conclusion

In this paper, we show that our proposed bidirectional learning architecture for panoptic segmentation is able to effectively utilize both instance and semantic features in a complementary fashion. Additionally, we use our occlusion handling module to demonstrate the importance of low-level appearance features for resolving the pixel to instance assignment problem. The proposed approach achieves the state-of-the-art result and the effectiveness of each of our modules is validated in the experiments.

References

[1] M. Bai and R. Urtasun. Deep watershed transform for instance segmentation. In *CVPR*, pages 2858–2866, 2017.

[2] K. Chen, J. Pang, J. Wang, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Shi, W. Ouyang, C. C. Loy, and D. Lin. Hybrid task cascade for instance segmentation. In *CVPR*, pages 4969–4978, 2019.

[3] K. Chen, J. Wang, J. Pang, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *CoRR*, abs/1906.07155, 2019.

[4] L. Chen, A. Hermans, G. Papandreou, F. Schroff, P. Wang, and H. Adam. Masklab: Instance segmentation by refining object detection with semantic and direction features. In *CVPR*, pages 4013–4022, 2018.

[5] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE TPAMI*, 40(4):834–848, 2018.

[6] L. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, pages 801–818, 2018.

[7] R. Cipolla, Y. Gal, and A. Kendall. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, pages 7482–7491, 2018.

[8] J. Dai, K. He, and J. Sun. Convolutional feature masking for joint object and stuff segmentation. In *CVPR*, pages 3992–4000, 2015.

[9] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *CVPR*, pages 3150–3158, 2016.

[10] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *ICCV*, pages 764–773, 2017.

[11] D. de Geus, P. Meletis, and G. Dubbelman. Panoptic segmentation with a joint semantic and instance segmentation network. *CoRR*, abs/1809.02110, 2018.

[12] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE TPAMI*, 35(8):1915–1929, 2013.

[13] A. Fathi, Z. Wojna, V. Rathod, P. Wang, H. Song, S. Guadarrama, and K. P. Murphy. Semantic instance segmentation via deep metric learning. *CoRR*, abs/1703.10277, 2017.

[14] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He. Detectron. https://github.com/facebookresearch/detectron, 2018.

[15] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, pages 1–8, 2009.

[16] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, pages 2980–2988, 2017.

[17] X. He, R. S. Zemel, and M. A. Carreira-Perpinan. Multiscale conditional random fields for image labeling. In *CVPR*, volume 2, pages II–II, 2004.

[18] A. Kirillov, R. Girshick, K. He, and P. Dollár. Panoptic feature pyramid networks. In *CVPR*, pages 6392–6401, 2019.

[19] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár. Panoptic segmentation. In *CVPR*, pages 9396–9405, 2019.

[20] P. Kohli, L. Ladický, and P.H.S. Torr. Robust higher order potentials for enforcing label consistency. *IJCV*, 82(3):302–324, 2009.

[21] L. Ladický, C. Russell, P. Kohli, and P. H. S. Torr. Associative hierarchical crfs for object class image segmentation. In *ICCV*, pages 739–746, 2009.

[22] J. Li, A. Raventos, A. Bhargava, T. Tagawa, and A. Gaidon. Learning to fuse things and stuff. *CoRR*, abs/1812.01192, 2018.

[23] Y. Li, X. Chen, Z. Zhu, L. Xie, G. Huang, D. Du, and X. Wang. Attention-guided unified network for panoptic segmentation. In *CVPR*, pages 7019–7028, 2019.

[24] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei. Fully convolutional instance-aware semantic segmentation. In *CVPR*, pages 4438–4446, 2017.

[25] X. Liang, L. Lin, Y. Wei, X. Shen, J. Yang, and S. Yan. Proposal-free network for instance-level object segmentation. *IEEE TPAMI*, 40(12):2978–2991, 2018.

[26] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 936–944, 2017.

[27] T. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. Zitnick, and P. Dollár. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014.

[28] H. Liu, C. Peng, C. Yu, J. Wang, X. Liu, G. Yu, and W. Jiang. An end-to-end network for panoptic segmentation. In *CVPR*, pages 6165–6174, 2019.

[29] S. Liu, J. Jia, S. Fidler, and R. Urtasun. SGN: Sequential grouping networks for instance segmentation. In *ICCV*, pages 3516–3524, 2017.

[30] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia. Path aggregation network for instance segmentation. In *CVPR*, pages 8759–8768, 2018.

[31] Y. Liu, S. Yang, B. Li, W. Zhou, J. Xu, H. Li, and Y. Lu. Affinity derivation and graph merge for instance segmentation. In *ECCV*, pages 686–703, 2018.

[32] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015.

[33] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich. Feedforward semantic segmentation with zoom-out features. In *CVPR*, pages 3376–3385, 2015.

[34] A. Newell, Z. Huang, and J. Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *NIPS*, pages 2277–2287, 2017.

[35] C. Peng, T. Xiao, Z. Li, Y. Jiang, X. Zhang, K. Jia, G. Yu, and J. Sun. Megdet: A large mini-batch object detector. In *CVPR*, pages 6181–6189, 2018.

[36] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *IJCV*, 81(1):2–23, 2009.

[37] J. Uhrig, M. Cordts, U. Franke, and T. Brox. Pixel-level encoding and depth layering for instance-level semantic labeling. In *German Conference on Pattern Recognition*, pages 14–25. Springer, 2016.

[38] Y. Xiong, R. Liao, H. Zhao, R. Hu, M. Bai, E. Yumer, and R. Urtasun. Upsnet: A unified panoptic segmentation network. In *CVPR*, pages 8810–8818, 2019.

[39] Z. Zhang, S. Fidler, and R. Urtasun. Instance-level segmentation for autonomous driving with deep densely connected mrfs. In *CVPR*, pages 669–677, 2016.

[40] Z. Zhang, A. G. Schwing, S. Fidler, and R. Urtasun. Monocular object instance segmentation and depth ordering with cnns. In *ICCV*, pages 2614–2622, 2015.

[41] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, pages 6230–6239, 2017.