SOIT: Segmenting Objects with Instance-Aware Transformers

Xiaodong Yu[1*], **Dahu Shi**[1*], **Xing Wei**[2], **Ye Ren**[1], **Tingqun Ye**[1], **Wenming Tan**[1†]

[1] Hikvision Research Institute, Hangzhou, China
[2] School of Software Engineering, Xi'an Jiaotong University
{yuxiaodong7, shidahu}@hikvision.com, weixing@mail.xjtu.edu.cn,
{renye, yetingqun, tanwenming}@hikvision.com

Abstract

This paper presents an end-to-end instance segmentation framework, termed SOIT, that Segments Objects with Instance-aware Transformers. Inspired by DETR (Carion et al. 2020), our method views instance segmentation as a direct set prediction problem and effectively removes the need for many hand-crafted components like RoI cropping, one-to-many label assignment, and non-maximum suppression (NMS). In SOIT, multiple queries are learned to directly reason a set of object embeddings of semantic category, bounding-box location, and pixel-wise mask in parallel under the global image context. The class and bounding-box can be easily embedded by a fixed-length vector. The pixel-wise mask, especially, is embedded by a group of parameters to construct a lightweight instance-aware transformer. Afterward, a full-resolution mask is produced by the instance-aware transformer without involving any RoI-based operation. Overall, SOIT introduces a simple single-stage instance segmentation framework that is both RoI- and NMS-free. Experimental results on the MS COCO dataset demonstrate that SOIT outperforms state-of-the-art instance segmentation approaches significantly. Moreover, the joint learning of multiple tasks in a unified query embedding can also substantially improve the detection performance. Code is available at https://github.com/yuxiaodongHRI/SOIT.

Introduction

Instance segmentation is a fundamental yet challenging task in computer vision, which requires an algorithm to predict a pixel-wise mask with a category label for each instance of interest in an image. As popularized in the Mask R-CNN framework (He et al. 2017), state-of-the-art instance segmentation methods follow a *detect-then-segment* paradigm (Cai and Vasconcelos 2019; Chen et al. 2019a; Vu, Kang, and Yoo 2021). These methods employ an object detector to produce the bounding boxes of instances and crop the feature maps via RoIAlign (He et al. 2017) according to the detected boxes. Then pixel-wise masks are predicted by a fully convolutional network (FCN) (Long, Shelhamer, and Darrell 2015) only in the detected region (as shown in Fig 1a). The detect-then-segment paradigm is sub-optimal since it has

---
*Equal contribution. †Corresponding author.

(a) Detect-then-segment pipeline.

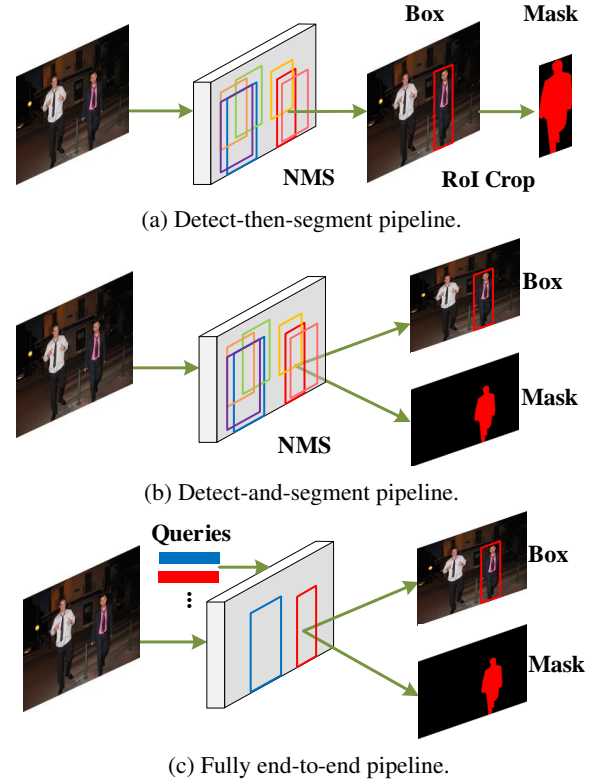(b) Detect-and-segment pipeline.

(c) Fully end-to-end pipeline.

Figure 1: Comparisons of different instance-level perception pipelines. We proposed the fully end-to-end framework as shown in (c), which is RoI-free and NMS-free.

the following drawbacks: 1) Segmentation results heavily rely on the object detector, incurring inferior performance in complex scenarios; 2) RoIs are always resized into patches of the same size (*e.g.*, 14×14 in Mask R-CNN), which restricts the quality of segmentation masks, as large instances would require higher resolution features to retain details at the boundary. To overcome the drawbacks of this paradigm, recent works (Chen et al. 2019b; Xie et al. 2020; Cao et al. 2020a; Peng et al. 2020) start to build instance segmentation frameworks on top of single-stage detectors (Lin et al. 2017b; Tian et al. 2019), getting rid of local RoI operations.

However, these methods still rely on one-to-many label assignment in training and hand-crafted Non-Maximum Suppression (NMS) post-processing to eliminate duplicated instances when testing. As a result, these two categories of instance segmentation methods are not end-to-end fully optimized and suffer from sub-optimal solutions.

Inspired by the recent application of transformer architecture in object detection (Carion et al. 2020; Zhu et al. 2021), we present a transformer-based instance segmentation framework, namely SOIT (Segment Objects with Instance-aware Transformer) in this paper. We reformulate instance segmentation as a direct set prediction problem and builds a fully end-to-end approach. Concretely, given multiple randomly initialized object queries, SOIT learns to reason a set of object embeddings of semantic category, bounding-box, and pixel-wise mask simultaneously, under the global image context. SOIT adopts the bipartite matching strategy to assign a learning target for each object query. As shown in Fig. 1c, this training approach is advantageous to conventional one-to-many instance segmentation training strategies (He et al. 2017; Wang et al. 2020b; Tian, Shen, and Chen 2020) as it avoids the heuristic label assignment and eliminates the need for NMS post-processing.

A compact fixed-length vector can easily embed the semantic category and bounding-box in the end-to-end learning framework. However, it is not trivial to represent a spatial binary mask of each object for learning as the mask is high-dimensional and varies from each instance. To solve this problem, we embed the pixel-wise mask to a group of instance-aware parameters, whereby a unique instance-aware transformer is constructed. Moreover, we propose a novel relative positional encoding for the transformer, which provides strong location cues to distinguish different objects. Then the instance-aware transformer is employed to segment the object in a high-resolution feature map directly. It is expected that the instance-aware parameters and relative positional encoding can encode the characteristics of each instance. Thus it can only fire on the pixels of the particular object. As described above, our method is naturally RoI-free and NMS-free, which eliminates many extra hand-crafted operations involved in previous instance segmentation methods.

Our main contributions are summarized as follows:

- We attempt to solve instance segmentation from a new perspective that uses parallel instance-aware transformers in an end-to-end framework. This novel solution enables the framework to directly generate pixel-wise mask results of each instance without RoI cropping or NMS post-processing.

- In our method, queries learn to encode multiple object representations simultaneously, including categories, locations, and pixel-wise masks. This multi-task joint learning paradigm establishes a collaboration between objection detection and instance segmentation, encouraging these two tasks to benefit from each other. We demonstrate that our architecture can also significantly improve object detection performance.

- To show the effectiveness of the proposed framework,

we conduct extensive experiments on the COCO dataset. SOIT with ResNet-50 achieves 42.5% mask AP and 49.1% box AP on the `test-dev` split without any bells and whistles, outperforming the complex well-tuned HTC (Chen et al. 2019a) by 2.8% in mask AP and 4.2% in box AP.

Related Work

Instance Segmentation

Instance segmentation is a challenging task, as it requires instance-level and pixel-wise predictions simultaneously. The existing approaches can be summarized into three categories: top-down, bottom-up, and single-stage methods. In top-down methods, the Mask R-CNN family (He et al. 2017; Cai and Vasconcelos 2019; Chen et al. 2019a; Cao et al. 2020b) follow the detect-then-segment paradigm, which first performs object detection and then segments objects in the boxes. Moreover, some recent works (Lee and Park 2020; Wang et al. 2020a; Chen et al. 2020b) are proposed to improve the segmentation performance further. Bottom-up methods (Liu et al. 2017; Gao et al. 2019) view the task as a label-then-cluster problem. They first learn per-pixel embeddings and then cluster them into instance groups. Besides, YOLACT (Bolya et al. 2019), CondInst (Tian, Shen, and Chen 2020) and SOLO (Wang et al. 2020b) build single-stage instance segmentation framework on the top of one-stage detectors (Tian et al. 2019), achieving competitive performance. Concurrently, QueryInst (Fang et al. 2021) and SOLQ (Dong et al. 2021) aim at building end-to-end instance segmentation frameworks, eliminating NMS post-processing. However, they still need RoI cropping to separate different instances first, which may have the same limitations of the detect-then-segment pipeline. In this paper, we go for an end-to-end instance segmentation framework that neither relies on RoI cropping nor NMS post-processing.

Transformer in Vision

Transformer (Vaswani et al. 2017) introduces the self-attention mechanism to model long-range dependencies and has been widely applied in natural language processing (NLP). Recently, several works attempted to involve Transformer architecture in computer vision tasks and showed promising performances. ViT series (Dosovitskiy et al. 2020; Touvron et al. 2021) take an image as a sequence of patches and achieve the cross-patch interactions by transformer architecture in image classification. DETR (Carion et al. 2020), and Deformable DETR (Zhu et al. 2021) adopted learnable queries and transformer architecture together with bipartite matching to perform object detection in an end-to-end fashion, without any hand-crafted process such as NMS. SETR (Zheng et al. 2021) reformulates the image semantic segmentation problem from a sequence-to-sequence learning perspective, offering an alternative to the dominating encoder-decoder FCN model design. Despite transformer architecture is being widely used in many computer vision tasks, few efforts are conducted to build a transformer-based instance segmentation framework. We aim to achieve this goal in this paper.
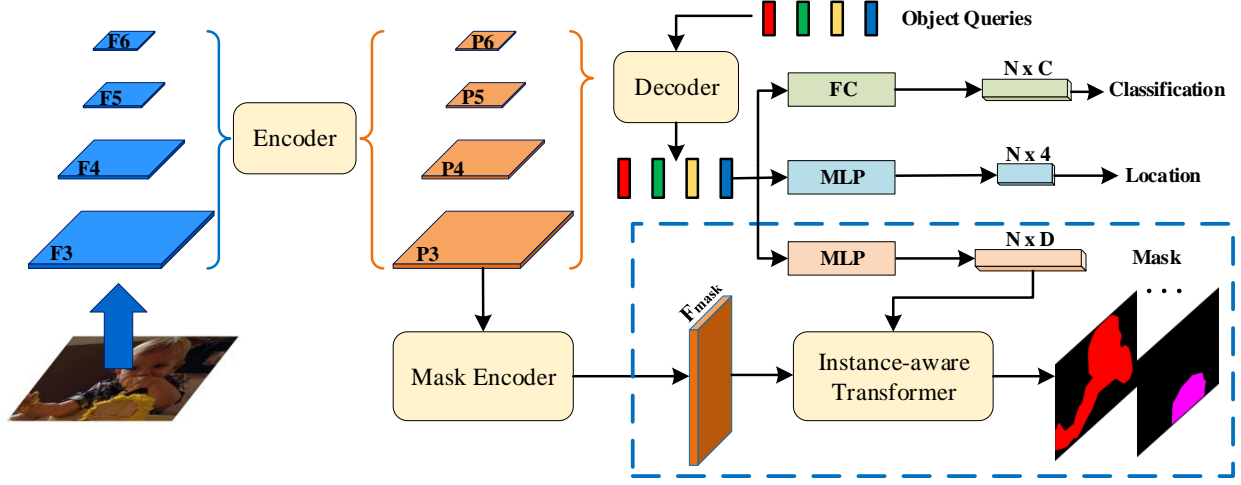
Figure 2: Illustration of the overall architecture of SOIT. F_3 to F_6 are the multi-scale image feature maps extracted from the backbone (*e.g.*, ResNet-50). P_3 to P_6 are the multi-scale feature memory refined by the transformer encoder. F_{mask} represents mask features produced by the mask encoder. D-dimensional (*e.g.* 441 by default) dynamic parameters generated in the mask branch are used to construct the instance-aware transformer. As shown in the blue dashed box, the pixel-wise mask is produced via the instance-aware transformer, of which the details are described in Figure 3.

Dynamic Networks

Unlike traditional network layers with fixed filters once trained, the filters of dynamic networks are conditioned on the input and dynamically generated by another network. This idea has been explored previously in convolution modules like dynamic filter networks (Jia et al. 2016), and CondConv (Yang et al. 2019), to increase the capacity of a classification network. Recently, some works (Tian, Shen, and Chen 2020; Shi et al. 2021) employ the dynamic filters, conditioned on each instance in the image, to implement instance-level vision tasks. In this work, we extend this idea to transformer architecture and build instance-aware transformers to solve the challenging instance segmentation task.

Method

In this section, we first introduce the overall architecture of our framework. Next, we elaborate on the proposed instance-aware transformer employed to produce the full-resolution mask for each instance. Then, we describe relative positional encoding to improve instance segmentation performance further. At last, the training losses of our model are summarized.

Overall Architecture

As depicted in Fig. 2, the proposed framework is composed of three main components: a backbone network to extract multi-scale image feature maps, a transformer encoder-decoder to produce object-related query features in parallel, and a multi-task prediction network to perform object detection and instance segmentation simultaneously.

Multi-Level Features. Given an image $I \in \mathbb{R}^{H \times W \times 3}$, we extract multi-scale feature maps $\mathcal{F} = \{F_3, F_4, F_5, F_6\}$ (blue feature maps in Fig 2) from the backbone (*e.g.*, ResNet (He et al. 2016)). Specifically, $\{F^l\}_{l=3}^{5}$ are produced by adding a 1×1 convolution on the output feature maps of stage C_3 through C_5 in the backbone, where C_l is of resolution 2^l lower than the input images. The lowest resolution feature map F_6 is obtained via a 3×3 stride 2 convolution on the final C_5 stage. Multi-scale image feature maps in $\mathcal{F}$ are all of 256 channels.

Transformer Encoder-Decoder. In this work, we employ the deformable transformer encoder (Zhu et al. 2021) to produce multi-scale feature memory. Each encoder layer comprises a multi-scale deformable attention module (Zhu et al. 2021) and a feed-forward network (FFN). There are six encoder layers stacked in sequence in our framework. The encoder takes the image feature maps $\mathcal{F}$ as input and output the refined multi-scale feature memory $\mathcal{P} = \{P^l\}_{l=3}^{6}$ (orange feature maps in Fig. 2) with the same resolutions.

Given the refined multi-scale feature memory $\mathcal{P}$ and N learnable object queries, we then generate the instance-aware query embeddings for target objects by the deformable transformer decoder (Zhu et al. 2021). Similar to the encoder, six decoder layers are applied sequentially. Each one is composed of a self-attention module, and a deformable cross-attention module (Zhu et al. 2021), where object queries interact with each other and the global context, respectively. In the end, the instance-aware query features are collected and then fed into the multi-task prediction network.

Multi-Task Predictions. After query feature extraction, each query embedding represents the features of the corresponding instance. Subsequently, we simultaneously apply three branches to generate the category, bounding-box location, and pixel-wise mask of the targeting instance. The
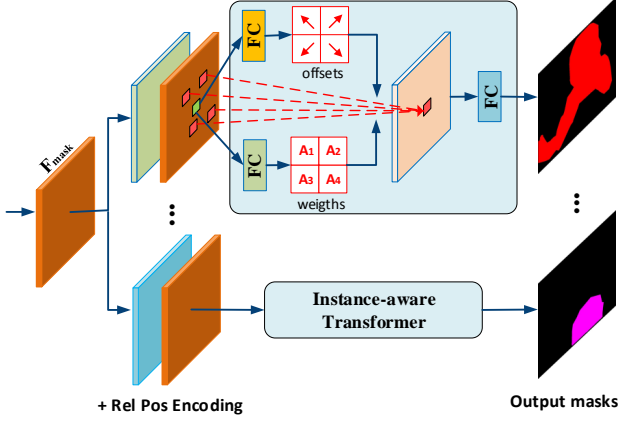
Figure 3: Detailed structure of the instance-aware transformer. Two linear projections (*i.e.*, FC) predict sampling locations and attention weights for different feature points, respectively. Another linear projection is employed for output projection. In our instance-aware transformer, all weights of these three layers are dynamically generated in the mask branch and conditioned on the target object.

classification branch is a linear projection layer (FC) that predicts the class confidence for each object. The location branch is a multi-layer perceptron (MLP) with a hidden size of 256 and predicts the normalized center coordinates, height, and width of the box *w.r.t.* the input image. The mask branch architecture is the same as the location branch except that the channel of the output layer is set to D. It is worth noting that the output of the mask branch is a group of dynamic parameters conditioned on the particular instance. These parameters are later employed to construct instance-aware transformers to directly generate masks from full-image feature maps, elaborated in the following subsection.

Instance-Aware Transformers

Unlike semantic category and bounding-box, it is challenging to represent the per-pixel mask by a compact fixed-length vector without RoI cropping. Our core idea is that for an image with N instances, N different transformer encoder networks will be dynamically generated. It is expected that the instance-aware transformer can encode the characteristics of each instance and only fires on the pixels of the corresponding object. To avoid the quadratic growth of the computational complexity in the original transformer encoder (Vaswani et al. 2017), we build our instance-aware transformer on the deformable transformer encoder (Zhu et al. 2021) for efficiency.

Concretely, given an input feature map $x \in \mathbb{R}^{C \times H \times W}$, let q indexes a query (*e.g.*, the green grid point in Fig. 3) with content feature z_q and a 2-d reference point p_q, the deformable multi-head attention feature is calculated by

$$H_m^n = \sum_{k=1}^{K} A_{mqk}^n \cdot x(p_q + \Delta p_{mqk}^n), \qquad (1)$$

where $m \in [1, 2, \ldots, M]$ indexes the attention head, k in-

dexes the sampled keys, and K is the total sampled key number ($K \ll HW$). n denotes the n-th object query (*i.e.*, instance). As shown in Fig. 3, Δp_{mqk} and A_{mqk} are the sampling offset and attention weight of the kth sampling point in the mth attention head, respectively. Both Δp_{mqk} and A_{mqk} are obtained via a linear projection (*i.e.*, FC) layer over the query feature z_q. Afterwards, another linear projection layer (*i.e.*, W^n) is applied for output projection, which can be formulated as

$$\text{Mask}^n = W^n \left[\text{Concat} \left(H_1^n, H_2^n, \ldots, H_M^n \right) \right], \qquad (2)$$

where "Concat" represents the concatenating operation. To establish our instance-aware transformer encoder, the weights of these three linear projection layers are dynamically generated, conditioned on the target instance. Specifically, for n-th object query, the D parameters predicted in the mask branch are split into three parts and converted as the weights of the three linear projections. Moreover, the channel of the output projection layer is set to 1 for the mask prediction, followed by a sigmoid activate function. Note that the attention locations and weights for each instance are different even at the same feature point, so each instance has a particular preference for where to focus in the feature map.

Shared Mask Features. To get high-quality masks, our method generates pixel-wise masks on a full-image feature map, not a cropped region with fixed size (*e.g.*, 14×14 in Mask R-CNN (He et al. 2017)). As shown in Fig. 2, the mask encoder branch is employed to provide the high-resolution feature map $F_{mask} \in \mathbb{R}^{H_{mask} \times W_{mask} \times C_{mask}}$ that the instance-aware transformers take as input to predict the per-instance mask. The mask encoder branch is connected to aggregated feature P_3, and thus, its output resolution is $1/8$ of the input image. It consists of a deformable transformer encoder layer, whose feature dimension is 256 (same as the feature channels of P_3). Afterward, a linear projection layer with layer normalization (LN) is employed to reduce the feature dimension from 256 to 8 (*i.e.*, $C_{mask} = 8$). As described above, the instance-aware transformer mask head is very compact due to the few channels of the shared mask feature.

Relative Positional Encodings

As described in (Vaswani et al. 2017), the original positional encoding in transformer is calculated by sine and cosine functions of different frequencies:

$$
\begin{aligned}
PE_{(pos, 2i)} &= sin(pos/10000^{2i/d_{model}}) \\
PE_{(pos, 2i+1)} &= cos(pos/10000^{2i/d_{model}})
\end{aligned}
\qquad (3)
$$

where pos is the absolute position, i is the dimension and d_{model} is the embedding dimension. DETR (Carion et al. 2020) extends the above positional encoding to the 2D case. Specifically, for both spatial coordinates (x, y) of each embedding in the 2D feature map, DETR independently uses $d_{model}/2$ sine and cosine functions with different frequencies. Then they are concatenated to get the final d_{model} channel positional encoding.

For our instance-aware transformer encoder, the input is the sum of the shared mask feature and the absolute posi-

tional encoding as described above. To further utilize the location information of each object query, we propose a new relative positional encoding, which can be written as:

$$PE_{(pos,2i)} = sin((pos - pos_q)/10000^{2i/d_{model}})$$
$$PE_{(pos,2i+1)} = cos((pos - pos_q)/10000^{2i/d_{model}})$$
(4)

where pos_q is the center location of the box predicted by current object query. Please note that the proposed relative positional encoding provides a strong cue for predicting the instance mask. The performance improvement in the ablation study demonstrates its superiority compared to the original absolute positional encoding.

Training Loss

In this work, the final outputs of our framework are supervised by three sub-tasks: classification, localization, and segmentation. We use the same loss functions for classification and localization as in (Zhu et al. 2021), and adopt the Dice Loss (Milletari, Navab, and Ahmadi 2016) and the binary cross-entropy (BCE) loss for instance segmentation. The overall loss function is written as:

$$L = \lambda_{cls}L_{cls} + \lambda_{L1}L_{L1} + \lambda_{iou}L_{iou} + \lambda_{dice}L_{dice} + \lambda_{bce}L_{bce}.$$

Following (Zhu et al. 2021), we set $\lambda_{cls} = 2$, $\lambda_{L1} = 5$ and $\lambda_{iou} = 2$. We empirically find $\lambda_{dice} = 8$ and $\lambda_{bce} = 2$ work best for the proposed framework.

Experiments

Dataset and Metrics

We validate our method on COCO benchmark (Lin et al. 2014). COCO 2017 dataset contains 115k images for training (split `train2017`), 5k for validation (split `val2017`) and 20k for testing (split `test-dev`), involving 80 object categories with instance-level segmentation annotations. Following the common practice, our models are trained with split `train2017`, and all the ablation experiments are evaluated on split `val2017`. Our main results are reported on the `test-dev` split for comparisons with state-of-the-art methods. Consistent with previous methods, the standard mask AP is used to evaluate the performance of instance segmentation. Moreover, we also report the box AP to show the object detection performance.

Implementation Details

ImageNet (Deng et al. 2009) pre-trained ResNet (He et al. 2016) is employed as the backbone and multi-scale feature maps $\{F^l\}_{l=1}^L$ are extracted without FPN (Lin et al. 2017a). Unless otherwise noted, the deformable attention has 8 attention heads, and the number of sampling points is set as 4. The feature channels in the encoder and decoder are 256, and the hidden dim of FFNs is 1024. We train our model with Adam optimizer (Kingma and Ba 2015) with base learning rate of 2.0×10^{-4}, momentum of 0.9 and weight decay of 1.0×10^{-4}. Models are trained for 50 epochs, and the initial learning rate is decayed at 40th epoch by a factor of 0.1. Multi-scale training is adopted, where the shorter side is randomly chosen within [480, 800] and the longer side is less
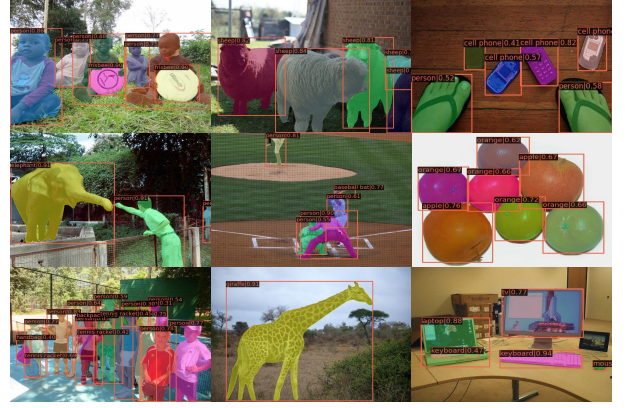

Figure 4: Qualitative results of object detection and instance segmentation on COCO `val2017` split. The model is trained on COCO `train2017` split with ResNet-50 backbone.

or equal to 1333. When testing, the input image is resized to have the shorter side being 800 and the longer side less or equal 1333. All experiments are conducted on 16 NVIDIA Tesla V100 GPUs with a total batch size of 32.

Main Results

As shown in Table 1, we compare SOIT with state-of-the-art instance segmentation methods on COCO `test-dev` split. Without bells and whistles, our method achieves the best performance on object detection and instance segmentation. Compared to the typical two-stage method Mask R-CNN (He et al. 2017), SOIT with ResNet-50 significantly improves box AP and mask AP by 7.8% and 5.0%, respectively. The performance of SOIT is also better than the well-tuned HTC (Chen et al. 2019a) by 4.2% box AP and 2.8% mask AP, which is an improved version of Mask R-CNN presenting interleaved execution and complicated mask information flow. CondInst (Tian, Shen, and Chen 2020) is the latest state-of-the-art one-stage instance segmentation approach based on dynamic convolutions. SOIT with the same ResNet-50 backbone outperforms CondInst with 4.7% mask AP. With a stronger backbone, ResNet-101, SOIT still outperforms the state-of-the-art methods over 2.0% mask AP. Benefiting from the RoI-free scheme, our method with ResNet-50 surpasses the recent SOLQ (Dong et al. 2021) and QueryInst (Fang et al. 2021) by 2.8% and 1.9%, respectively. We also apply SOIT to the recent Swin Transformer backbone (Liu et al. 2021) without further modification, building a pure transformer-based instance segmentation framework. Our model with Swin-L can achieve 56.9% and 49.2% in box AP and mask AP, respectively. As shown in Fig 4, we provide some qualitative results of SOIT with ResNet-50 backbone on COCO `val2017` split. Our masks are generally of high quality (*e.g.*, preserving more details at object boundaries), and the detected boxes are precise.

Method	Backbone	RoI-free	NMS-free	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AP^{box}
Mask R-CNN (He et al. 2017)				37.5	59.3	40.2	21.1	39.6	48.3	41.3
CMR (Cai and Vasconcelos 2019)				38.8	60.4	42.0	19.4	40.9	53.9	44.5
HTC (Chen et al. 2019a)				39.7	61.4	43.1	22.6	42.2	50.6	44.9
BlendMask (Chen et al. 2020a)				37.0	58.9	39.7	17.3	39.4	52.5	42.7
CondInst (Tian, Shen, and Chen 2020)		✓		37.8	59.2	40.4	18.2	40.3	52.7	41.9
SOLOv2 (Wang et al. 2020c)	ResNet-50	✓		38.2	59.3	40.9	16.0	41.2	55.4	40.4
DSC (Ding et al. 2021)				40.5	61.8	44.1	-	-	-	46.0
RefineMask (Zhang et al. 2021)				40.2	-	-	-	-	-	-
SCNet (Vu, Kang, and Yoo 2021)				40.2	62.3	43.4	22.4	42.8	53.4	45.0
SOLQ (Dong et al. 2021)			✓	39.7	-	-	21.5	42.5	53.1	47.8
QueryInst (Fang et al. 2021)			✓	40.6	63.0	44.0	23.4	42.5	52.8	45.6
SOIT (Ours)		✓	✓	**42.5**	**65.3**	**46.0**	**23.8**	**45.4**	**55.7**	**49.1**
Mask R-CNN (He et al. 2017)				38.8	60.9	41.9	21.8	41.4	50.5	43.1
CMR (Cai and Vasconcelos 2019)				39.9	61.6	43.3	19.8	42.1	55.7	45.7
HTC (Chen et al. 2019a)				40.7	62.7	44.2	23.1	43.4	52.7	46.2
MEInst (Zhang et al. 2020)				33.9	56.2	35.4	19.8	36.1	42.3	-
BlendMask (Chen et al. 2020a)				39.6	61.6	42.6	22.4	42.2	51.4	44.8
CondInst (Tian, Shen, and Chen 2020)		✓		39.1	60.9	42.0	21.5	41.7	50.9	43.3
SOLOv2 (Wang et al. 2020c)	ResNet-101	✓		39.7	60.7	42.9	17.3	42.9	57.4	42.6
DCT-Mask (Shen et al. 2021)				40.1	61.2	43.6	22.7	42.7	51.8	-
DSC (Ding et al. 2021)				40.9	62.5	44.5	-	-	-	46.7
RefineMask (Zhang et al. 2021)				41.2	-	-	-	-	-	-
SCNet (Vu, Kang, and Yoo 2021)				41.3	63.9	44.8	22.7	44.1	55.2	46.4
SOLQ (Dong et al. 2021)			✓	40.9	-	-	22.5	43.8	54.6	48.7
QueryInst (Fang et al. 2021)			✓	42.8	65.6	46.7	24.6	45.0	55.5	48.1
SOIT (Ours)		✓	✓	**43.4**	**66.3**	**46.9**	**23.9**	**46.4**	**57.4**	**50.0**
SOLQ (Dong et al. 2021)			✓	46.7	-	-	29.2	50.1	60.9	56.5
QueryInst (Fang et al. 2021)	Swin-L		✓	49.1	74.2	**53.8**	**31.5**	51.8	63.2	56.1
SOIT (Ours)		✓	✓	**49.2**	**74.3**	53.5	30.2	**52.7**	**65.2**	**56.9**

Table 1: Comparisons with state-of-the-art instance segmentation methods on the COCO `test-dev`. "CMR" is short for "Cascade Mask RCNN". AP^{box} denotes box AP, and AP without superscript denotes mask AP. All models are trained with multi-scale and tested with single scale.

Ablation Study

Number of Heads in Instance-Aware Transformers. The multi-head attention mechanism is of great importance for the transformer. In this section, we discuss the effect of this design on our instance-aware transformer encoder. We vary the number of heads of multi-head attention, and the performance of instance segmentation is shown in Table 2. We find that using only one head of attention already has a moderate capacity and leads to a qualified performance with 37.8% mask AP. The performance of instance segmentation improves gradually with the increased number of attention heads in the instance-aware transformer. Besides, when the number of attention heads increases up to 8, segmentation performance does not improve further. We assume there are two reasons for this saturation in performance. One is that 4 different spaces of representation are sufficient for distinguishing various instances. The other reason is that predicting too many parameters (873 parameters) makes optimizing the mask branch difficult. Therefore, we set the number of attention heads in the instance-aware transformer to 4 by default in the following experiments.

Architectures of Mask Encoder. We then investigate the impact of the proposed mask encoder with different architectures. We first change C_{mask}, i.e., the number of channels

Heads	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
1	37.8	61.6	39.5	18.1	41.1	57.6
2	38.1	61.9	39.9	18.5	41.3	58.1
4	**38.4**	**62.0**	**40.1**	**18.6**	41.7	**58.4**
8	38.3	62.0	40.1	18.4	**41.9**	58.4

Table 2: Instance segmentation results on COCO `val2017` split with different number of heads of multi-head attention in instance-aware transformer. The input feature channel (i.e., C_{mask}) is fixed to 8 by default.

of the mask encoder's output feature maps (i.e., F_{mask}). As shown in Table 3a, the performance drops 0.8% in mask AP (from 38.4% to 37.6%) when the channel of F_{mask} shrinks from 8 to 4. In this case, the multi-heads attention only has a single-channel map in each attention head. It is hard for the attention module to obtain sufficient information on each instance. Besides, the performance keeps almost the same when C_{mask} increases from 8 to 16. Thus, we fix the mask feature channels to 8 in all other experiments by default. As the $C_{mask} = 8$ and the number of attention heads is 4, there are a total of 441 parameters predicted by the mask branch for constructing the instance-aware transformer.

Channels	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$
4	37.6	61.8	39.2	18.2	40.8	57.5
8	**38.4**	**62.0**	**40.1**	**18.6**	**41.7**	**58.4**
16	38.3	62.0	40.0	18.5	**41.7**	58.3

(a) Vary the output channels of the mask encoder.

Layers	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$
0	37.9	61.4	39.4	18.0	40.9	57.6
1	**38.4**	**62.0**	**40.1**	**18.6**	**41.7**	58.4
2	38.4	61.9	40.1	18.5	41.6	**58.6**

(b) Vary the layers of stacked mask encoder.

Table 3: Instance segmentation results on COCO `val2017` split with different architectures of the mask encoder. "Channels": the number of channels of the mask encoder's output. "Layers": the number of stacked mask encoder.

PE	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$
None	37.9	61.4	39.6	18.3	41.2	58.0
Abs	38.4	62.0	40.1	18.6	41.7	58.4
Rel	**39.2**	**62.9**	**41.3**	**19.7**	**43.0**	**59.2**

Table 4: Impact of the positional encoding in instance-aware transformer on COCO `val2017` split. "None" means removing positional encoding, "Abs" represents the traditional absolute positional encoding and "Rel" represents the proposed relative positional encoding.

To demonstrate the effectiveness of the mask encoder, we directly connect a linear projection (output channel is 8) with layer normalization to the feature map P_3 instead of the proposed mask encoder. As shown in Table 3b, the segmentation performance drops 0.5% (from 38.4% to 37.9%). This result proves the importance of the mask encoder, which generates the specialized mask feature and decouples it from the generic image context feature. Moreover, when more mask encoders are stacked, no noticeable improvement of performance is obtained, as shown in Table 3b (3rd row). This indicates that one mask encoder is sufficient, resulting in a compact instance segmentation model.

Relative Positional Encodings. We further investigate the effect of our proposed relative positional encodings for the instance-aware transformers. *Abs* is the absolute positional encodings used in many transformer-based architectures (Carion et al. 2020; Zhu et al. 2021). *Rel* is the proposed relative positional encodings in Equation (4), which employ the box center coordinates of object queries to obtain the instance-aware location information. As shown in Table 4 (1st row), the performance of our model drops 0.5% in mask AP after removing absolute positional encodings to the mask features. The instance-aware transformer cannot distinguish the instances with similar appearances at different locations without the positional information. As shown in Table 4 (3rd row), the relative positional encodings improve the segmentation performance of our SOIT by 0.8% compared to the absolute positional encodings. We argue that the relative positional encoding is highly correlated with the corresponding object query and provides a strong location cue, for instance mask prediction. Therefore, in the sequel, we use the proposed relative positional encoding for all the following experiments.

Stages Enabling Mask Loss. Ultimately, we ablate the impact of the number of decoder stages enabling mask loss

Stages	AP	AP$_{50}$	AP$_{75}$	AP^{box}	AP$^{box}_{50}$	AP$^{box}_{75}$
0	-	-	-	46.8	66.3	50.7
1	39.2	62.9	41.3	47.3	66.2	52.0
2	40.7	63.6	43.4	47.6	66.4	52.5
3	41.2	63.9	44.1	48.1	66.5	52.8
4	41.7	64.2	44.5	48.2	66.4	53.0
5	42.0	64.5	44.9	48.5	66.7	53.2
6	**42.2**	**64.6**	**45.3**	**48.9**	**67.0**	**53.4**

Table 5: Ablation of the number of decode stages enabling mask loss on COCO `val2017` split. Stages is K means that enable the last K decoder layers with mask loss. 0 stages represents a object detection model without any mask supervision. AP^{box} denotes box AP.

in training. The classification and localization loss are enabled in all decoder stages in these ablations by default. Note that we throw away all the predicted mask parameters in the intermediate stages when the training is completed and only use the final stage predictions for inference. As shown in Table 5, enabling more decoder layers with mask loss can improve both instance segmentation and object detection performance consistently. The experimental results show that adding mask loss on all decoders can improve 3.0% mask AP and 1.6% box AP compared to enabling mask loss on only one decoder, respectively. The gain of detection performance is mainly derived from the joint training with instance segmentation. As shown in Table 5 (last row), the detection performance of the SOIT surpasses the pure object detector by 2.1% (from 46.8% to 48.9%) with all decoder stages enabling mask loss. This indicates the advantages of our framework, which learns a unified query embedding to perform instance segmentation and object detection simultaneously.

Conclusion

In this paper, we present a transformer-based instance segmentation approach, termed SOIT. It reformulates instance segmentation as a direct set prediction problem and builds a fully end-to-end framework. SOIT is naturally RoI-free and NMS-free, avoiding many hand-crafted operations involved in previous instance segmentation methods. Extensive experiments on the MS COCO dataset show that SOIT achieves state-of-the-art performance in instance segmentation as well as object detection. We hope that our simple end-to-end framework could serve as a strong baseline for instance-level perception.

Acknowledgments

References

Bolya, D.; Zhou, C.; Xiao, F.; and Lee, Y. J. 2019. Yolact: Real-time instance segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 9157–9166.

Cai, Z.; and Vasconcelos, N. 2019. Cascade r-cnn: High quality object detection and instance segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Cao, J.; Anwer, R. M.; Cholakkal, H.; Khan, F. S.; Pang, Y.; and Shao, L. 2020a. Sipmask: Spatial information preservation for fast image and video instance segmentation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, 1–18. Springer.

Cao, J.; Cholakkal, H.; Anwer, R. M.; Khan, F. S.; Pang, Y.; and Shao, L. 2020b. D2det: Towards high quality object detection and instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11485–11494.

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-end object detection with transformers. In *European Conference on Computer Vision*, 213–229. Springer.

Chen, H.; Sun, K.; Tian, Z.; Shen, C.; Huang, Y.; and Yan, Y. 2020a. BlendMask: Top-down meets bottom-up for instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 8573–8581.

Chen, K.; Pang, J.; Wang, J.; Xiong, Y.; Li, X.; Sun, S.; Feng, W.; Liu, Z.; Shi, J.; Ouyang, W.; et al. 2019a. Hybrid task cascade for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4974–4983.

Chen, X.; Girshick, R.; He, K.; and Dollár, P. 2019b. Tensormask: A foundation for dense object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2061–2069.

Chen, X.; Lian, Y.; Jiao, L.; Wang, H.; Gao, Y.; and Lingling, S. 2020b. Supervised edge attention network for accurate image instance segmentation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16*, 617–631. Springer.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Ding, H.; Qiao, S.; Yuille, A.; and Shen, W. 2021. Deeply Shape-guided Cascade for Instance Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8278–8288.

Dong, B.; Zeng, F.; Wang, T.; Zhang, X.; and Wei, Y. 2021. SOLQ: Segmenting Objects by Learning Queries. *NeurIPS*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Fang, Y.; Yang, S.; Wang, X.; Li, Y.; Fang, C.; Shan, Y.; Feng, B.; and Liu, W. 2021. Instances As Queries. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 6910–6919.

Gao, N.; Shan, Y.; Wang, Y.; Zhao, X.; Yu, Y.; Yang, M.; and Huang, K. 2019. Ssap: Single-shot instance segmentation with affinity pyramid. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 642–651.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2961–2969.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Jia, X.; De Brabandere, B.; Tuytelaars, T.; and Gool, L. V. 2016. Dynamic filter networks. *Advances in neural information processing systems*, 29: 667–675.

Kingma, D. P.; and Ba, J. 2015. Adam: A method for stochastic optimization. *ICLR*, 9.

Lee, Y.; and Park, J. 2020. Centermask: Real-time anchor-free instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 13906–13915.

Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; and Belongie, S. 2017a. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2117–2125.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017b. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2980–2988.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, 740–755. Springer.

Liu, S.; Jia, J.; Fidler, S.; and Urtasun, R. 2017. Sgn: Sequential grouping networks for instance segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, 3496–3504.

Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; and Guo, B. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3431–3440.

Milletari, F.; Navab, N.; and Ahmadi, S.-A. 2016. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, 565–571. IEEE.

Peng, S.; Jiang, W.; Pi, H.; Li, X.; Bao, H.; and Zhou, X. 2020. Deep snake for real-time instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8533–8542.

Shen, X.; Yang, J.; Wei, C.; Deng, B.; Huang, J.; Hua, X.-S.; Cheng, X.; and Liang, K. 2021. Dct-mask: Discrete cosine transform mask representation for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8720–8729.

Shi, D.; Wei, X.; Yu, X.; Tan, W.; Ren, Y.; and Pu, S. 2021. InsPose: Instance-Aware Networks for Single-Stage Multi-Person Pose Estimation. In *Proceedings of the 29th ACM International Conference on Multimedia*, 3079–3087.

Tian, Z.; Shen, C.; and Chen, H. 2020. Conditional convolutions for instance segmentation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, 282–298. Springer.

Tian, Z.; Shen, C.; Chen, H.; and He, T. 2019. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, 9627–9636.

Touvron, H.; Cord, M.; Douze, M.; Massa, F.; Sablayrolles, A.; and Jégou, H. 2021. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, 10347–10357. PMLR.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Vu, T.; Kang, H.; and Yoo, C. D. 2021. SCNet: Training Inference Sample Consistency for Instance Segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 2701–2709.

Wang, S.; Gong, Y.; Xing, J.; Huang, L.; Huang, C.; and Hu, W. 2020a. Rdsnet: A new deep architecture forreciprocal object detection and instance segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 12208–12215.

Wang, X.; Kong, T.; Shen, C.; Jiang, Y.; and Li, L. 2020b. Solo: Segmenting objects by locations. In *European Conference on Computer Vision*, 649–665. Springer.

Wang, X.; Zhang, R.; Kong, T.; Li, L.; and Shen, C. 2020c. SOLOv2: Dynamic and Fast Instance Segmentation. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M. F.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 17721–17732. Curran Associates, Inc.

Xie, E.; Sun, P.; Song, X.; Wang, W.; Liu, X.; Liang, D.; Shen, C.; and Luo, P. 2020. Polarmask: Single shot instance segmentation with polar representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 12193–12202.

Yang, B.; Bender, G.; Le, Q. V.; and Ngiam, J. 2019. Condconv: Conditionally parameterized convolutions for efficient inference. *arXiv preprint arXiv:1904.04971*.

Zhang, G.; Lu, X.; Tan, J.; Li, J.; Zhang, Z.; Li, Q.; and Hu, X. 2021. RefineMask: Towards High-Quality Instance Segmentation with Fine-Grained Features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6861–6869.

Zhang, R.; Tian, Z.; Shen, C.; You, M.; and Yan, Y. 2020. Mask encoding for single shot instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10226–10235.

Zheng, S.; Lu, J.; Zhao, H.; Zhu, X.; Luo, Z.; Wang, Y.; Fu, Y.; Feng, J.; Xiang, T.; Torr, P. H.; et al. 2021. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6881–6890.

Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; and Dai, J. 2021. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *ICLR 2021: The Ninth International Conference on Learning Representations*.