

# SG-Net: Spatial Granularity Network for One-Stage Video Instance Segmentation

Dongfang Liu<sup>1\*</sup>, Yiming Cui<sup>2\*</sup>, Wenbo Tan<sup>3</sup>, Yingjie Chen<sup>1</sup>

<sup>1</sup>Purdue University, USA

<sup>2</sup>University of Florida, USA

<sup>3</sup>Hangzhou Dian Zi University, China

{liu2538,victorchen}@purdue.edu, cuiyiming@ufl.edu, 174060003@hdu.edu.cn

## Abstract

Video instance segmentation (VIS) is a new and critical task in computer vision. To date, top-performing VIS methods extend the two-stage Mask R-CNN by adding a tracking branch, leaving plenty of room for improvement. In contrast, we approach the VIS task from a new perspective and propose a one-stage spatial granularity network (SG-Net). Compared to the conventional two-stage methods, SG-Net demonstrates four advantages: 1) Our method has a one-stage compact architecture and each task head (detection, segmentation, and tracking) is crafted interdependently so they can effectively share features and enjoy the joint optimization; 2) Our mask prediction is dynamically performed on the sub-regions of each detected instance, leading to high-quality masks of fine granularity; 3) Each of our task predictions avoids using expensive proposal-based RoI features, resulting in much reduced runtime complexity per instance; 4) Our tracking head models objects' centerness movements for tracking, which effectively enhances the tracking robustness to different object appearances. In evaluation, we present state-of-the-art comparisons on the YouTube-VIS dataset. Extensive experiments demonstrate that our compact one-stage method can achieve improved performance in both accuracy and inference speed. We hope our SG-Net could serve as a strong and flexible baseline for the VIS task. Our code will be available here<sup>1</sup>.

## 1. Introduction

Video instance segmentation (VIS) is a challenging vision task introduced by [43]. Given a video frame, an algorithm aims to perform the task of detection, segmentation, and tracking of instances simultaneously. The VIS task holds valuable possibilities for applications which requires

\*indicates equal contributions.

<sup>1</sup><https://github.com/goodproj13/SG-Net>

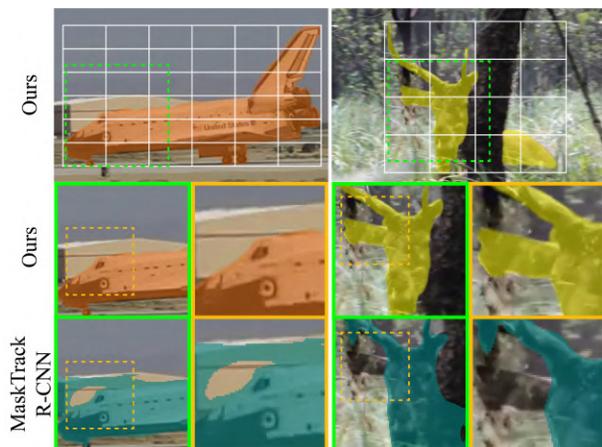


Figure 1. A detailed comparison with MaskTrack R-CNN. The large images in the first row are the results of our method. We further zoom in our results and compare against results from MaskTrack R-CNN. We dynamically divide a target instance into sub-regions based on its bounding box and perform instance segmentation on each sub-region to achieve superior spatial granularity. Both examples are cropped from the original images for better illustration.

video-level object masks and temporal descriptions such as augmented reality, video editing, and autonomous driving.

The seminal work, MaskTrack R-CNN [43] follows a two-stage paradigm as it is extended from Mask R-CNN [13]. MaskTrack R-CNN first uses region proposal network (RPN) from Faster R-CNN [31] to produce a set of candidate proposals. Then, regions-of-interests (RoI) features based on the proposals are cropped out and fed into each task head to predict bounding boxes, instance masks, and object trackings respectively. Despite a few works being proposed recently, the dominant VIS frameworks [11, 16, 19, 25, 26] follow the two-stage paradigm.

However, the two-stage paradigm may encounter a number of issues. First, it is difficult for each sub-task head (detection, segmentation, and tracking) in the two-stage ap-

proach to share features, which causes a trouble for network architecture optimization. Second, the cropped RoI features are resized into patches of a uniformed size (e.g.,  $14 \times 14$  or  $28 \times 28$  in Mask R-CNN [13]), which restricts the output resolution of the instance mask. This practice could particularly compromise the prediction of large instances as they need higher resolutions to retain details of the object boundary. Third, the candidate proposals are redundant representations because their numbers are much larger than the final predictions. The mask head and tracking head have to repeatedly encode the proposal-based RoI features for the final predictions. Specifically, the mask head requires a stack of convolutions (e.g., four  $3 \times 3$  convolutional layers in Mask R-CNN [13]) to obtain a receptive field large enough to understand sufficient image context. Therefore, the inference runtime is largely depended on the number of detected objects that appeared on the video frame and would degrade considerably with the increase of predictions.

Considering the above limitations, we attempt to solve the VIS task from a new perspective. We treat all three sub-tasks, detection, segmentation and tracking in VIS as interconnected problems that should be considered interdependently. Recent advances in object detection illuminate that one-stage methods such as FCOS [39] and CenterNet [47] can outperform their two-stage counterparts in accuracy. Both methods use flexible convolutional operations, which are convenient for multi-task implementations and multi-network optimization (i.e., instance segmentation and tracking). Their successes enable the possibility to implement a one-stage approach for the VIS task.

For the instance segmentation task, we recognize two important predecessors, BlendMask [5] and CondInst [38], which are both built on the FCOS framework. Compared to the Mask R-CNN-based approaches, they only use fully convolutional network (FCN) architectures, which help them to avoid the RoI operations (i.e., RoIpool, RoIAlign, cropping, and resizing). This improvement significantly preserves the feature map resolution and retains details of the mask boundary. Moreover, both methods implement a light-weight mask head, which makes them very robust in real-time video tasks. However, both methods segment objects on the instance level and ignore the potential to achieve more granularity on objects.

In this work, our primary focus is to investigate ways to circumvent the heavy proposal-based two-stage approach and find a flexible solution for the VIS task. Inspired by [5, 38], we generalize the mask prediction by modeling spatial scales of objects with semantic information for lower-level granularity. We also propose an easy track strategy, which directly uses the object centerness from detection to delineate the video temporal coherence. Concretely, our work delivers the following contributions:

- We attempt to solve the VIS task from a new per-

spective. To this end, we devise a compact one-stage method called SG-Net. Our method dynamically divides instance into sub-regions and performs segmentation on each region for Spatial Granularity, thus the name of our network, SG-Net. Compared to MaskTrack R-CNN [43], our method achieves more appealing segmentation behavior as it can enrich object details and produce masks with more accurate edges (As shown in Figure 1).

- Our method is proposal-free and efficient. Removing proposals allows us to assign heavier duties to the mask prediction module with an affordable computation overhead. Particularly, our inference time does not increase with the growing number of predictions as the two-stage methods do.
- Our entire architecture consists of only convolutional operations, which is tied to the state of the art one-stage object detector, FCOS [39]. We organically craft each task head interdependently so they can effectively share features and enjoy joint optimization.
- Our tracking head models the movement of object centerness for tracking, which is simple and effective. Compared to MaskTrack R-CNN [43], our tracking head is more robust to different object shapes and sizes as well as appearance changes.

## 2. Related Work

Although VIS is a new task in computer vision, its sub-tasks (a.k.a., detection, instance segmentation, and tracking) have been well studied in the literature. This section summarizes the related work in these tasks.

### 2.1. Object Detection

One-stage detectors can achieve a faster inference speed by avoiding proposal generation [24, 30]. However, one-stage approaches are generally inferior in accuracy compared to their two-stage counterparts. The latest one-stage object detection methods (i.e, FCOS [39] and CenterNet [47]) prove the effectiveness of removing box anchors to simplify the detection pipeline. Without any restriction of the pre-defined anchors, target objects are matched freely to the prediction features of their suitable receptive fields. Compared to its anchor-based counterparts [21, 24, 30], FCOS [39] and CenterNet [47] not only improve the detection accuracy, but also remain a fast inference. The flexibility of one-stage detection framework benefits the instance segmentation methods [5, 38] as it removes the computation overhead caused by proposal or anchor operations. Building on the lessons learned from the concurrent approaches [5, 38], our approach is based on the one-stage FCOS framework.

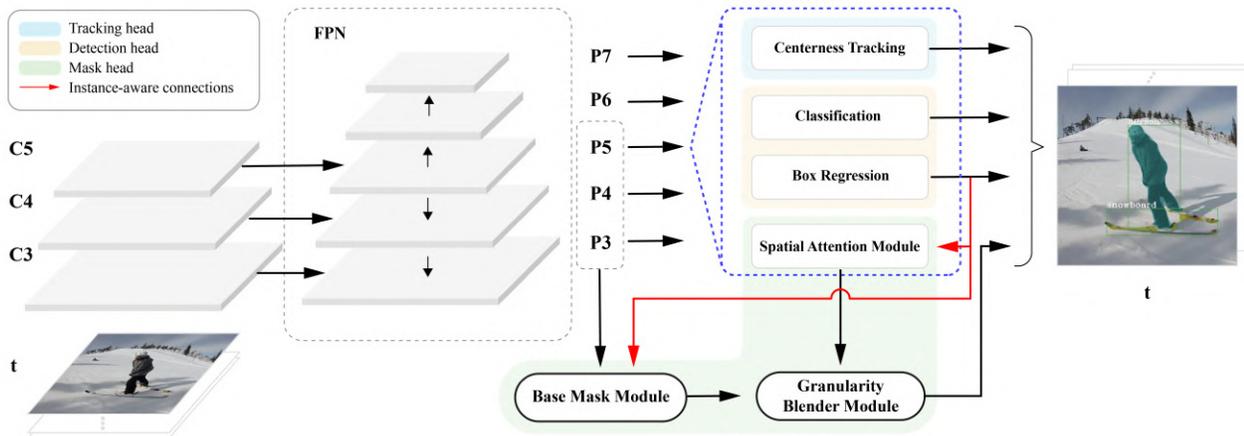


Figure 2. Illustration of SG-Net framework. Our detection head includes the classification branch, the box regression branch, the centerness branch (not shown for simplicity) as in FCOS [39]. Our tracking head directly uses the centerness outputs from FCOS to model object movements for tracking. We use the instance-aware connections to propagate the bounding box information and guide the instance segmentation task. Therefore, our mask head can efficiently encode on the pixel of the instance, instead of repeatedly performing predictions on proposal-based features as in two-stage methods. Specifically, the spatial attention module divides the target instance into equally-sized sub-regions based on its size and shape and predict an attention score for each sub-region. The base mask module produces a set of base masks which is equal to the number of the sub-regions of the same instance. The granularity blender combines the attention scores and the base masks to predict the final mask. Note that the functional modules inside of the blue dashed box are repeatedly applied to  $\{P3, \dots, P7\}$ .

## 2.2. Instance Segmentation

To date, the dominant instance segmentation methods still follow the two-stage paradigm based on Mask R-CNN[13]. These methods [6, 15, 23] rely heavily on RoI features and operations, which first produce a set of candidate proposals and then predict the foreground masks on each of the RoIs. Although there are several attempts been made for using alternative approaches [7, 10, 27, 28, 29], Mask R-CNN still outperforms these methods in accuracy.

More recently, YOLACT [3], SipMask [4], and BlendMask [5] reformulate the instance segmentation pipeline based on Mask R-CNN. YOLACT [3] and SipMask [4] generate a number of prototype masks for each instance and combine them with per-pixel predictions using a single coefficient for the entire instance (YOLACT) and  $2 \times 2$  coefficients for each sub-region of the instance (SipMask), respectively. However, these overly simplified assembling designs may not achieve the optimal representation of the object features. Similarly, BlendMask [5] first predicts base masks and corresponding 2D attention maps for each instance, and then blends them together for the final prediction. Since BlendMask performs segmentation directly at the instance level, its representation of object details is not sufficient. Inspired by the aforementioned methods [4, 5, 38], our approach models objects' spatial scales with semantic information. Based on objects' sizes and shapes, we dynamically divide them into different sub-regions for segmentation to achieve improved mask granularity.

## 2.3. Multiple Object Tracking

Recent multi-object tracking mainly follows the tracking-by-detection paradigm [9, 18, 34, 42, 48], which first detects objects and then predicts the temporal association of all detections. Some methods use motion priors, such as Kalman Filter [12, 45] or optical flow [41] to associate the detected bounding boxes. Others learn tracking representations by exploiting the appearance similarity [33, 42, 35] or re-identify vanished objects [17, 36, 37]. Our tracking also follows the tracking-by-detection paradigm. Instead of using the object bounding box or mask as the tracking clue, we track object movement by its center, which is the output from the centerness branch of FCOS [39]. Therefore, we can optimize detection and tracking jointly and achieve an improved tracking behavior.

## 2.4. Video Instance Segmentation

Majority of VIS methods [2, 8, 11, 16, 19, 25, 26] follow the two-stage paradigm based on Mask R-CNN. Although the two-stage paradigm is intuitively simple, it has several performance bottlenecks: 1) feature sharing cross task heads is insufficient so joint optimization of network architecture is difficult; 2) the quality of the instance mask is restricted by the RoI operation (e.g., resizing or pooling); 3) the RoI feature representation is redundant so the inference speed is largely effected by the number of candidate proposals. Compared to the two-stage methods, our SG-Net is fast and one-stage. Our mask head is fully-convolutional

(no RoI operations) and proposal-free (no redundant RoI features), so we can produce high-quality instance masks while achieving a fast inference. Each of our task heads is crafted interdependently and can be jointly optimized.

### 3. Our SG-Net

#### 3.1. Overall Architecture

Built on the FCOS [39], our SG-Net includes a feature extraction backbone, a detection head, a mask head, and a tracking head. Our overall architecture is shown in Figure 2. Our backbone network employs ResNet [14] with FPNs [20]. We exploit the feature maps from  $\{P3, P4, P5, P6, P7\}$  of FPNs and apply different functional modules to fulfill different sub-tasks. Our detection adopts the original FCOS [39], which consists of the classification, box regression, centerness branch. At time  $t$ , we denote the detection of the  $i^{th}$  object as  $D_t^i = (c_t^i, o_t^i, b_t^i)$ , where  $c_t^i \in \{0, \dots, C-1\}$  is the class of the detected object,  $o_t^i \in \mathbb{R}^2$  is the centerness location of the detected object, and  $b_t^i \in \mathbb{R}^4$  is the object bounding box. Refer to [39] for more details. Since our mask and tracking head are the key components of SG-Net, we elaborate the two techniques in the following sections.

#### 3.2. Mask Head

Our mask head consists of: 1) a spatial attention module; 2) a base mask module; 3) a granularity blender. Each module works collaboratively to predict the final mask.

**Spatial attention module:** Our spatial attention module divides the object bounding box into multiple sub-regions and predicts attention scores on each sub-region (the green area in Figure 3). We use the instance-aware connection to propagate the bounding box information  $\{b^1, \dots, b^n\}$  from detection results to the spatial attention module. Based on objects' shapes and sizes, we dynamically divide the object bounding box into  $r_1 \times r_2$  sub-regions:

$$r_1, r_2 = \min(6, \lceil \frac{w}{50} \rceil), \min(6, \lceil \frac{h}{50} \rceil) \quad (1)$$

where  $w$  and  $h$  are the width and height of the bounding box respectively and the number 50 is in a pixel unit. Instead of pooling features into 3D parameter-heavy attention maps in BlendMask [38], we predict a set of 1D attention scores (similar to coefficients in YOLACT [3]) for every sub-region by appending two  $3 \times 3$  convolution layers to each of the  $P3 - P7$  levels. For a detected object, we define its attention scores as  $\mathbb{A} = \{a^j \in \mathbb{R} | j = 1, \dots, r_1 \times r_2\}$  where  $a^j$  is the attention score for the  $j^{th}$  sub-region.

**Base mask module:** The base mask generation pipeline has two major steps: 1) extract and prepare FPNs features; 2) generate the base masks for each target instance. In the first step, we first extract the  $P3 - P5$  features from FPNs.

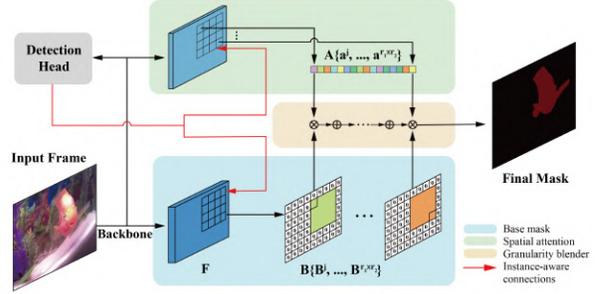


Figure 3. The working pipeline of our mask head.

$P4 - P5$  features are upsampled to the same resolution of  $P3$  using bilinear interpolation. Afterward, the three-level features ( $P3 - P5$ ) are concatenated together and fed into two  $3 \times 3$  convolutions with 128 channels. In order to reduce the computational parameters, we append another convolution layer to downsize the feature channels from 128 to 36. The outputs from the previous step are upsampled  $\times 4$  using bilinear interpolation to produce high-resolution feature maps  $F$ . Our later evaluations show that the feature upsampling is beneficial to the final mask prediction.

In the second step, we use the instance-aware connection to guide the base mask production (the blue area in Figure 3). Informed by the bounding box locations  $\{b^1, \dots, b^n\}$ , we can perform prediction accurately within the region of the detected object. To produce the base masks for a specific instance, we first assign the local features outside of its bounding box to be zero and obtain new target-oriented feature maps, where we apply a  $1 \times 1$  convolution to dynamically produce our base masks. The number of the base masks for an instance is restricted by the volume of its sub-regions ( $r_1 \times r_2$ ) from the spatial attention module. For a detected object, we define its base masks as  $\mathbb{B} = \{B^j \in \mathbb{R}^{\frac{H}{2} \times \frac{W}{2}} | j = 1, \dots, r_1 \times r_2\}$  where  $B^j$  is the  $j^{th}$  base mask and  $\frac{H}{2} \times \frac{W}{2}$  is the base mask resolutions ( $H \times W$  is the input size). The spatial attention module and base mask module progress in parallel. Therefore, each base mask  $B^j$  has its corresponding attention score  $a^j$ .

**Granularity blender module:** Our granularity blender takes the attention scores and the base masks as inputs to produce the final mask (the yellow area in Figure 3). For the  $i^{th}$  detected object on the frame, we perform a summation of an element-wise multiplication between its attention scores  $\mathbb{A}$  and base masks  $\mathbb{B}$  to predict its instance mask  $M^i$ :

$$M^i = \sum_{j=1}^{r_1 \times r_2} \text{sigmoid}(a^j \times B^j), \quad (2)$$

where  $\text{sigmoid}()$  is to binarize the summation. Accordingly, we can obtain the instance masks  $M$  for the whole

frame as the follows:

$$M = \sum_{i=1}^n M^i, \quad (3)$$

where  $n$  is the total number of predictions on the frame.

At the first glance, our mask head may incur a large number of network parameters, when processing video frames with a high volume of objects. However, we circumvent inefficient proposal generations and ROI operations in two-stage methods, resulting in much-reduced runtime complexity per instance and improved segmentation behavior.

### 3.3. Tracking Head

Our tracking head uses centerness to represent each instance. Since FCOS already has a centerness branch, we can conveniently craft our tracking head into the FCOS framework. After FCOS’s centerness branch, we append one  $1 \times 1$  convolutional layer to process the centerness features. The work similar to us is CenterTrack [46] using box centers. Our centerness tracking is more robust to object appearance changes, while box centers can reside at the same position in multiple bounding boxes, causing confusion in tracking.

At time  $t$ , tracking for one object  $t_t = (o_t, id_t)$  is described by its centerness location  $o_t \in \mathbb{R}^2$  from the detection result and a unique identity  $id_t \in [0, \tilde{N}]$ , where  $\tilde{N}$  is the total number of distinct instances so far. A newly detected object can only be assigned to one of the  $\tilde{N}$  identities if it is one of the previous instances or a new  $id$  if it is a new unique instance. To achieve this goal, our tracking head predicts a 2D movement  $D \in \mathbb{R}^{\tilde{H} \times \tilde{W} \times 2}$  to associate centerness through time. For the  $i^{th}$  detection at time  $t$ ,  $D_t^i$  describes the centerness movements  $o_t^i - o_{t-1}^i$  between the current frame  $t$  and the previous frame  $t - 1$ . Therefore, we can model the movement regression objective by:

$$\mathcal{L}_{track} = \frac{1}{N} \sum_{t \in T} \sum_{i \in n} \left| D_t^i - (o_t^i - o_{t-1}^i) \right|, \quad (4)$$

where  $n$  is the number of detections on a single frame at time  $t$ , and  $N$  is all detections within the video time span  $T$ . In training,  $T$  is the number of video sequences. Figure 4 shows an example of this movement prediction.

With the movement prediction, we can use a simple greedy matching to link objects across frame. For a detection  $i$  at position  $o^i$  of time  $t$ , we greedily associate it with the closest prior detection at position  $o_{t-1}^i - D_t^i$ . If no matched candidate within a radius  $r$  is found, we spawn a new tracklet. We define  $r$  as the mean of the bounding box width and height of the tracking object. Our simple tracking technique is empirically effective.

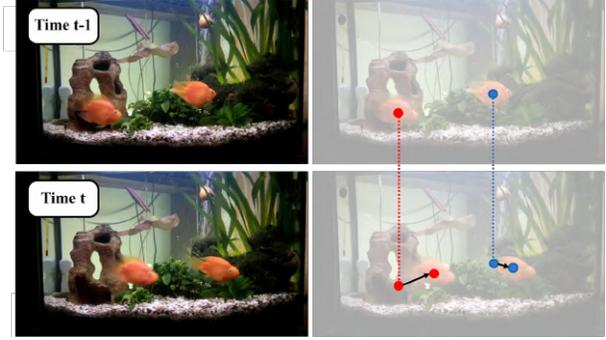


Figure 4. The centerness tracking. We use object centers to model object movements. By calculating object centerness movement across frames, we can depict the video temporal coherence.

### 3.4. Training Objective

Our method follows the top-down paradigm. We use the bounding box locations from detection to guide the spatial attention module and base mask module for instance segmentation. Therefore, we argue that the improved bounding box regression can benefit the instance segmentation task. Bear this in mind, we modify the GIoU loss ( $\mathcal{L}_{GIoU} = 1 - GIoU$ ) in [32] to accommodate our own system design. We endeavor to facilitate the hard sample learning when dealing with small GIoU. To achieve this goal, we use a logarithmic function to increase the bounding box losses:

$$\mathcal{L}_{box} = -\ln \frac{1 + GIoU}{2} \quad (5)$$

Our overall loss is the sum of individual loss from the detection head, the mask head, and the tracking head:

$$\mathcal{L}_{all} = \mathcal{L}_{det} + \mathcal{L}_{mask} + \mathcal{L}_{track} \quad (6)$$

where  $\mathcal{L}_{mask}$  is the mask loss as in [13] and  $\mathcal{L}_{track}$  is the movement loss from Eq. 4.  $\mathcal{L}_{det}$  include three terms  $\mathcal{L}_{cls}$ ,  $\mathcal{L}_{cent}$ , and  $\mathcal{L}_{box}$ .  $\mathcal{L}_{cls}$  and  $\mathcal{L}_{cent}$  are the focal loss and the centerness loss in [21] respectively.  $\mathcal{L}_{box}$  is the modified GIoU loss from Eq. 5.

## 4. Experiments

We assess SG-Net on YouTube-VIS dataset [43]. Following [43], we perform training on the 3,471 training videos and testing on the 507 validation videos with the same input size  $640 \times 360$ . Our main results and ablation experiments are all reported on the validation set.

### 4.1. Implementation Details

ImageNet pretrained ResNet [14] (the 50 and 101 version) with FPN is used as our backbone network. For the newly added layers, we initialize them as in [5]. Our models are trained with SGD with momentum 0.9 and an initial

|           | Method               | Backbone   | Time (ms)   | AP          | AP@0.5      | AP@0.75     | AR@1        | AR@10       |
|-----------|----------------------|------------|-------------|-------------|-------------|-------------|-------------|-------------|
| Two Stage | OSMN MaskProp [44]   | ResNet-50  | -           | 23.4        | 36.5        | 25.7        | 28.9        | 31.1        |
|           | IoUTracker+ [43]     | ResNet-50  | -           | 23.6        | 39.2        | 25.5        | 26.2        | 30.9        |
|           | FEELVOS [40]         | ResNet-50  | -           | 26.9        | 42.0        | 29.7        | 29.9        | 33.4        |
|           | OSMN [44]            | ResNet-50  | -           | 27.5        | 45.1        | 29.1        | 28.6        | 31.1        |
|           | DeepSORT             | ResNet-50  | -           | 26.1        | 42.9        | 26.1        | 27.8        | 31.3        |
|           | SeqTracker [43]      | ResNet-50  | -           | 27.5        | 45.7        | 28.7        | 29.7        | 32.5        |
|           | MaskTrack R-CNN [43] | ResNet-50  | 98.8        | 30.3        | 51.1        | 32.6        | 31.0        | 35.5        |
|           | MaskTrack R-CNN [43] | ResNet-101 | 142.8       | 31.8        | 53.0        | 33.6        | 33.2        | 37.6        |
| One Stage | STEm-Seg [1]         | ResNet-50  | 83.3        | 30.6        | 50.7        | 33.5        | 31.6        | 37.1        |
|           | SipMask [4]          | ResNet-50  | 35.7        | 33.7        | 54.1        | 35.8        | 35.4        | 40.1        |
|           | <b>Ours</b>          | ResNet-50  | <b>43.5</b> | <b>34.8</b> | <b>56.1</b> | <b>36.8</b> | <b>35.8</b> | <b>40.8</b> |
|           | STEm-Seg [1]         | ResNet-101 | 125.0       | 34.6        | 55.8        | 37.9        | 34.4        | 41.6        |
|           | SipMask [4]          | ResNet-101 | 41.7        | 35.8        | 56.0        | 39.0        | 35.4        | 42.4        |
|           | <b>Ours</b>          | ResNet-101 | <b>50.6</b> | <b>36.3</b> | <b>57.1</b> | <b>39.6</b> | <b>35.9</b> | <b>43.0</b> |

Table 1. Quantitative results on YouTube-VIS validation set. Metrics for MaskTrack R-CNN, STEm-Seg, and SipMask are obtained using their official code and trained model. Other results are retrieved from [43]. AP and AR are the Average Precision and Recall respectively.

the learning rate of 0.005 which decays exponentially. our models are trained with one TITAN RTX GPU, while all tests are conducted on a single 1080Ti GPU.

## 4.2. Main Results

**Quantitative results:** We compare our SG-Net against previous state-of-the-art methods on the validation set of YouTube-VIS. Table 4 presents the comparison results. Our methods achieve competitive results under all evaluation metrics. In head-to-head comparisons, our SG-Net outperforms all two-stage methods. Specifically, our margins over the state-of-the-art MaskTrack R-CNN [43] are 8.2% and 7.9% in AP using ResNet-50 and -101 respectively. We also compare SG-Net against the emerging one-stage methods, STEm-Seg [1] and SipMask [4]. Our SG-Net consistently surpasses STEm-Seg by a significant margin (34.8% vs. 30.6% using ResNet-50 and 36.3% vs. 34.6% using ResNet-101). Moreover, we observe steady improvements from SipMask [4] (34.8% vs. 33.7% using ResNet-50 and 36.3% vs. 35.8% using ResNet-101).

SG-Net is also efficient. Compared to the two-stage methods, our simpler method, without any bells and whistles, achieves both higher accuracy and faster inference speed. Our models are about  $\times 2$  and  $\times 3$  faster than MaskTrack R-CNN [43] with ResNet-50 and ResNet-101 respectively. Measured on a 1080Ti GPU, the best SG-Net (ResNet-50) typically runs at 0.043s/image, with nearly negligible impact of scene complexity. On the contrary, the two-stage MaskTrack R-CNN has a more expensive mask and tracking computation and its inference time closely relies on the number of predictions. Furthermore, our method achieves a faster speed than STEm-Seg [1] (43.5 ms vs. 83.3 ms with ResNet-50 and 50.6 ms vs. 125 ms with

ResNet-101), as it has three parameter-heavy decoders for mask generation, object tracking, and class prediction respectively. Compared to SipMask [4], our methods achieve better accuracy with on par execution time. For fair comparisons, all the inference times here are measured by the same local hardware with the official codes.

**Qualitative results:** We compare our qualitative results with those from MaskTrack R-CNN [43]. To demonstrate our advantages, we select some challenging samples where MaskTrack R-CNN struggles to deal with (as shown in Figure 6). For instance, in the skydiver examples, MaskTrack R-CNN [43] intends to assign parts to the wrong instance and leave large false-negative regions; in the tiger examples, MaskTrack R-CNN [43] has difficulties to discriminate instances of the same class close to each other; finally, in the zebra examples, occlusions could also compromise the tracking prediction of MaskTrack R-CNN [43]. Our method demonstrates improved segmenting and tracking behaviors on these difficult cases because: 1) Our base masks have higher resolution (stride-2 features). Thus details are better retained. 2) Our attention module captures instance details, which guide the base masks to output results of fine granularity. 3) Our tracking head uses centerness to track objects, which is flexible and robust to different object appearances.

The last row in Figure 6 demonstrates examples of our weak predictions. Our method fails to recognize the right earless seal for the first three frames and constantly classify the left seal as an ape. We argue that one critical reason, which causes the erroneous prediction, is the data bias on YouTube-VIS, as the “earless seal” class is underrepresented in the training set. More qualitative examples can be found here <https://youtu.be/zft0T3YUgPM>.

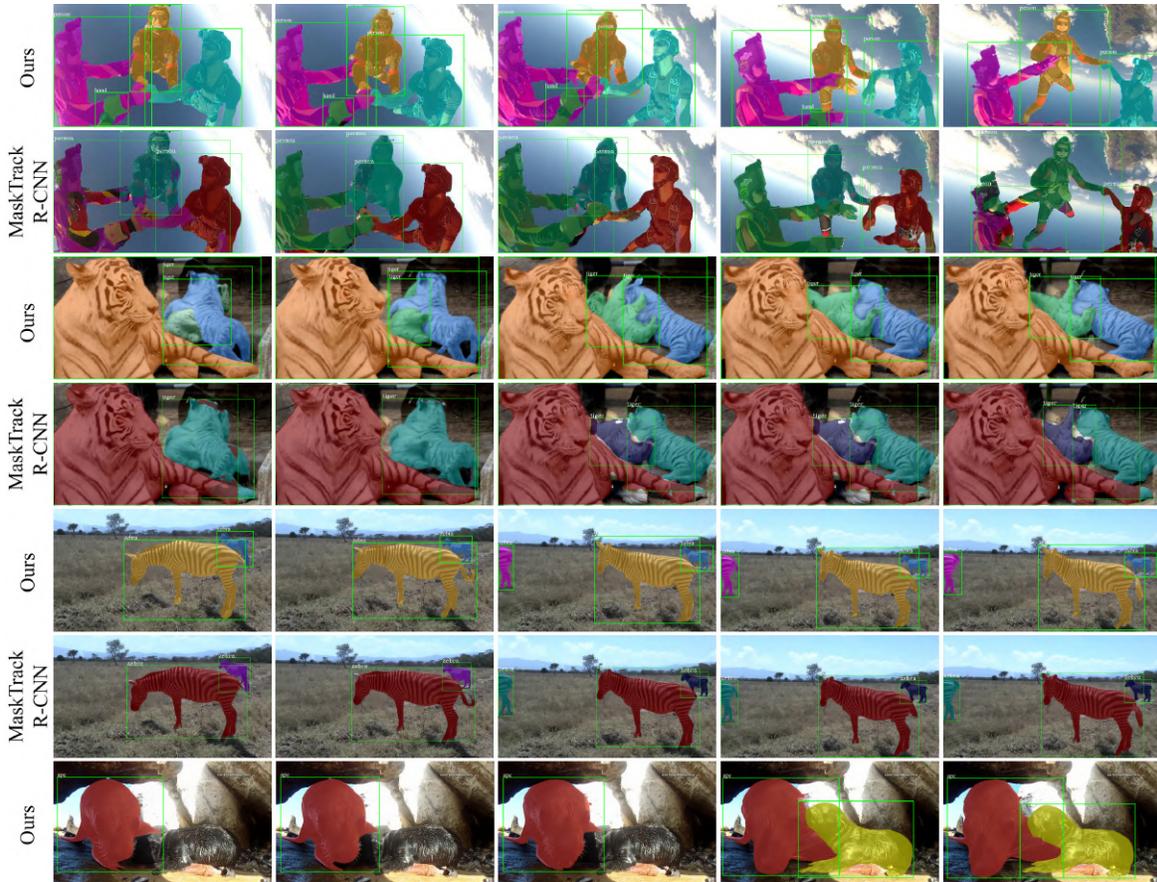


Figure 5. Selected results of MaskTrack R-CNN and SG-Net. Both methods use ResNet-101 with FPN. Each row has five frames from a video sequence. Objects with the same predicated identity have the same mask color. Best viewed in the digital format with zoom.

### 4.3. Ablation Study

We perform an ablation study to investigate the effectiveness of the different contributing components and configurable parameters in our method. The following results are based on ResNet101-FPN [20].

| Method    | a    | b    | c    | d    | e           |
|-----------|------|------|------|------|-------------|
| Baseline  | ✓    | ✓    | ✓    | ✓    | ✓           |
| BMM       |      | ✓    |      | ✓    | ✓           |
| SAM       |      |      | ✓    | ✓    | ✓           |
| GL        |      |      |      | ✓    | ✓           |
| AP        | 31.1 | 34.3 | 35.2 | 36.1 | <b>36.3</b> |
| Time (ms) | 33.3 | 45.7 | 51.9 | 50.6 | <b>50.6</b> |

Table 2. Impact of integrating different functional modules into the baseline to the accuracy. BMM, SAM, GL stand for base mask module, spatial attention module, and GIoU loss respectively.

**Contributing components:** We first show the contributions of progressively integrating different components (Table 2): the base mask module (BMM), the spatial attention module (SAM), and the GIoU loss (GL), to the baseline.

Our baseline, which is similar to YOLACT [3] with our tracking head, only uses a single set of attention scores and obtain the base masks from  $P_3$  level of FPN. The baseline achieves an AP of 31.1% with a speed of 33.3 ms. Each component (BMM, SAM, and GL) contributes towards improving the overall performance in accuracy. Method b adds BMM with multi-level FPN features ( $P_3 - P_5$ ) to the baseline and uses a single set of attention scores for mask prediction. We observe that FPN features are beneficial for improving the segmentation performance, especially objects with varying sizes being presented. Method c adds SAM to the baseline but uses the original 36 channels of FPN features (as discussed in Section 3.2). Since SAM introduces more non-linearity in mask predictions, it obtains further improvements from Method b in accuracy, but its speed is slower as it uses more feature channels for predictions. The most improvement over the baseline comes from Model e as it includes our important components BMM and SAM as well as GIoU loss. Compared to the baseline, our full model (Model e) obtains an absolute gain of 5.2% in accuracy while still maintains a fast inference.

**Box divisions:** As discussed in Section 3.2, our SAM

dynamically divides the bounding box into different sub-regions and yields a set of attention scores for each sub-region. We perform a study to evaluate the impact of changing the maximum number of sub-regions on accuracy and efficiency. Table 3 shows that we keep improving accuracy by increasing the box divisions from  $2 \times *$  to  $6 \times *$ . However, the accuracy improvement tends to be marginal when it comes to  $8 \times *$ , and the inference encounters a significant speed penalty. This is because of that more sub-regions require more base masks, which increase the network parameters. Note that the  $2 \times *$  version is similar to SipMask [4], but our result is superior in speed (with similar accuracy) as we use dynamic divisions instead of using a fixed  $2 \times 2$  grid for mask prediction. In default, we use  $6 \times *$ .

| Box division | $2 \times *$ | $4 \times *$ | $6 \times *$ | $8 \times *$ |
|--------------|--------------|--------------|--------------|--------------|
| AP           | 34.0         | 35.4         | <b>36.3</b>  | 36.4         |
| Time (ms)    | 33.2         | 42.1         | <b>50.6</b>  | 88.6         |

Table 3. Impact of the maximum number of sub-regions (i.e.,  $4 \times *$  means the maximum divisions of the sub-regions is 4).

**Base mask resolutions:** As mentioned in Section 3.2, feature upsampling is of great importance to the final mask prediction. Results in Table 4 confirm this argument. Without the upsampling (last row), SG-Net produces the base mask with stride-8 features and only achieves 34.9% in AP. If our base mask features are upsampled to  $1/2$  of the input resolution (factor = 4), the accuracy performance is improved by 1.4% from 34.9% to 36.3%. Particularly, the improvement on the larger IoU threshold (AP@0.75) is large (from 38.1% to 39.6%), which indicates that the upsampling can greatly preserve the object details. However, we also observe that increasing base resolution introduces the speed penalty. In default, we use  $1/2$  of the input resolution (factor = 4) as we obtain the best accuracy.

| Factor   | AP          | AP@0.5      | AP@0.75     | Time (ms)   |
|----------|-------------|-------------|-------------|-------------|
| <b>4</b> | <b>36.3</b> | <b>57.1</b> | <b>39.6</b> | <b>50.6</b> |
| 2        | 35.6        | 55.9        | 38.6        | 47.3        |
| 1        | 34.9        | 55.9        | 38.1        | 40.9        |

Table 4. Impact of base mask resolution to the accuracy. The first column is the upsampling factor which indicates the feature map resolutions (i.e., 1 is no upsampling and 2 is  $\times 2$  upsampling)

**Base feature locations:** We evaluate the feature sampling locations for the base mask module. By using deeper FPN features, we can improve the performance in accuracy with an affordable penalty on speed (Table 5).

#### 4.4. Discussions

Compared to the two-stage methods [11, 19, 25, 26], our SG-Net has a number of improvements as discussed below.

| Feat. locat. | AP          | AP@0.5      | AP@0.75     | Time (ms)   |
|--------------|-------------|-------------|-------------|-------------|
| P3           | 35.0        | 56.1        | 37.0        | 35.7        |
| P3-P4        | 35.7        | 55.9        | 38.8        | 43.2        |
| <b>P3-P5</b> | <b>36.3</b> | <b>57.1</b> | <b>39.6</b> | <b>50.6</b> |

Table 5. Performance using different feature locations for base mask generation. P3, P4, and P5 are features from FPN.

**Feature sharing:** Since FCOS [39] is flexible and only use FCNs, we can craft each task network more closely. Particularly, we implement instance-aware connections to facilitate feature sharing across tasks (detection to segmentation and tracking). Therefore, each task head in our method can be jointly optimized in an end-to-end fashion.

**Mask quality:** Our mask prediction is performed on each sub-region of an instance, leading to a mask with fine granularity. In the meta-experiments on the COCO image benchmark [22], our method outperforms recent strong methods. For instance, our margins over SipMask [4] and CondInst [38] are 1.3-1.8% and 0.8-1.3% respectively with the same backbone settings.

**Tracking robustness:** Using centerness for tracking offers us two advantages. First, centerness tracking is robust to difficult cases (Fig. 6). When dealing with the highly overlapped bounding boxes, our centerness tracking can robustly handle the intractable ambiguity. Two, we can craft our tracking head seamlessly with FCOS, as centerness is collaterally produced in detection.

**Inference efficiency:** Our one-stage design is efficient. Our mask head avoids expensive proposal generation and RoI operations. Therefore, compared to MaskTrack R-CNN [43] whose inference time is proportional to the number of predictions, our method holds the potential for real-time applications and the additional inference time caused by the increased predictions is marginal.

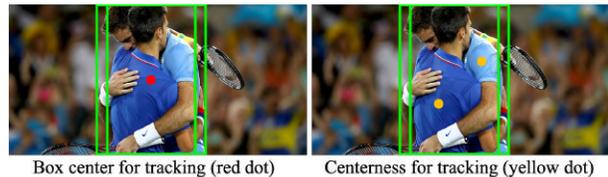


Figure 6. Box centers can reside at the same position (left) in multiple bounding boxes, causing confusion in tracking. Our centerness tracking is robust to object appearance variation (right).

## 5. Conclusion

In this study, we propose a simple yet effective one-stage framework for VIS task, named SG-Net. Compared the two-stage solutions, we effectively improve the mask quality and inference speed. We believe that our SG-Net can be a valuable addition to the existing VIS solutions.

## References

- [1] Ali Athar, Sabarinath Mahadevan, Aljoša Ošep, Laura Leal-Taixé, and Bastian Leibe. Stem-seg: Spatio-temporal embeddings for instance segmentation in videos. *arXiv preprint arXiv:2003.08429*, 2020. 6
- [2] Gedas Bertasius and Lorenzo Torresani. Classifying, segmenting, and tracking object instances in video with mask propagation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9739–9748, 2020. 3
- [3] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact: Real-time instance segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 9157–9166, 2019. 3, 4, 7
- [4] Jiale Cao, Rao Muhammad Anwer, Hisham Cholakkal, Fahad Shahbaz Khan, Yanwei Pang, and Ling Shao. Sipmask: Spatial information preservation for fast image and video instance segmentation. *arXiv preprint arXiv:2007.14772*, 2020. 3, 6, 8
- [5] Hao Chen, Kunyang Sun, Zhi Tian, Chunhua Shen, Yongming Huang, and Youliang Yan. Blendmask: Top-down meets bottom-up for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8573–8581, 2020. 2, 3, 5
- [6] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4974–4983, 2019. 3
- [7] Jifeng Dai, Kaiming He, Yi Li, Shaoqing Ren, and Jian Sun. Instance-sensitive fully convolutional networks. In *European Conference on Computer Vision*, pages 534–549. Springer, 2016. 3
- [8] Minghui Dong, Jian Wang, Yuanyuan Huang, Dongdong Yu, Kai Su, Kaihui Zhou, Jie Shao, Shiping Wen, and Changhu Wang. Temporal feature augmented network for video instance segmentation. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2019. 3
- [9] Kuan Fang, Yu Xiang, Xiaocheng Li, and Silvio Savarese. Recurrent autoregressive networks for online multi-object tracking. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 466–475. IEEE, 2018. 3
- [10] Alireza Fathi, Zbigniew Wojna, Vivek Rathod, Peng Wang, Hyun Oh Song, Sergio Guadarrama, and Kevin P Murphy. Semantic instance segmentation via deep metric learning. *arXiv preprint arXiv:1703.10277*, 2017. 3
- [11] Qianyu Feng, Zongxin Yang, Peike Li, Yunchao Wei, and Yi Yang. Dual embedding learning for video instance segmentation. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2019. 1, 3, 8
- [12] Weitao Feng, Zhihao Hu, Wei Wu, Junjie Yan, and Wanli Ouyang. Multi-object tracking with multiple cues and switcher-aware classification. *arXiv preprint arXiv:1901.06129*, 2019. 3
- [13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 1, 2, 3, 5
- [14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4, 5
- [15] Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask scoring r-cnn. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6409–6418, 2019. 3
- [16] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Video panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9859–9868, 2020. 1, 3
- [17] Laura Leal-Taixé, Cristian Canton-Ferrer, and Konrad Schindler. Learning by tracking: Siamese cnn for robust target association. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 33–40, 2016. 3
- [18] Laura Leal-Taixé, Anton Milan, Konrad Schindler, Daniel Cremers, Ian Reid, and Stefan Roth. Tracking the trackers: an analysis of the state of the art in multiple object tracking. *arXiv preprint arXiv:1704.02781*, 2017. 3
- [19] Chung-Ching Lin, Ying Hung, Rogerio Feris, and Linglin He. Video instance segmentation tracking with a modified vae architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13147–13157, 2020. 1, 3, 8
- [20] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 4, 7
- [21] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 2, 5
- [22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 8
- [23] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8759–8768, 2018. 3
- [24] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 2
- [25] Xiaoyu Liu, Haibing Ren, and Tingmeng Ye. Spatio-temporal attention network for video instance segmentation. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2019. 1, 3, 8
- [26] Eslam Mohamed, Mahmoud Ewaisha, Mennatullah Siam, Hazem Rashed, Senthil Yogamani, and Ahmad El-Sallab. Instancemotseg: Real-time instance motion segmentation

- for autonomous driving. *arXiv preprint arXiv:2008.07008*, 2020. 1, 3, 8
- [27] Davy Neven, Bert De Brabandere, Marc Proesmans, and Luc Van Gool. Instance segmentation by jointly optimizing spatial embeddings and clustering bandwidth. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8837–8845, 2019. 3
- [28] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *Advances in neural information processing systems*, pages 2277–2287, 2017. 3
- [29] David Novotny, Samuel Albanie, Diane Larlus, and Andrea Vedaldi. Semi-convolutional operators for instance segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 86–102, 2018. 3
- [30] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 2
- [31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 1
- [32] Hamid Rezaatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 658–666, 2019. 5
- [33] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 300–311, 2017. 3
- [34] Samuel Schuster, Paul Vernaza, Wongun Choi, and Manmohan Chandraker. Deep network flow for multi-object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6951–6960, 2017. 3
- [35] Jeany Son, Mooyeol Baek, Minsu Cho, and Bohyung Han. Multi-object tracking with quadruplet convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5620–5629, 2017. 3
- [36] Siyu Tang, Mykhaylo Andriluka, Bjoern Andres, and Bernt Schiele. Multiple people tracking by lifted multicut and person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3539–3548, 2017. 3
- [37] Zheng Tang, Milind Naphade, Ming-Yu Liu, Xiaodong Yang, Stan Birchfield, Shuo Wang, Ratnesh Kumar, David Anastasiu, and Jenq-Neng Hwang. Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8797–8806, 2019. 3
- [38] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. *arXiv preprint arXiv:2003.05664*, 2020. 2, 3, 4, 8
- [39] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 9627–9636, 2019. 2, 3, 4, 8
- [40] Paul Voigtlaender, Yuning Chai, Florian Schroff, Hartwig Adam, Bastian Leibe, and Liang-Chieh Chen. Feelvos: Fast end-to-end embedding learning for video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9481–9490, 2019. 6
- [41] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pages 466–481, 2018. 3
- [42] Jiarui Xu, Yue Cao, Zheng Zhang, and Han Hu. Spatial-temporal relation networks for multi-object tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3988–3998, 2019. 3
- [43] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5188–5197, 2019. 1, 2, 5, 6, 8
- [44] Linjie Yang, Yanran Wang, Xuehan Xiong, Jianchao Yang, and Aggelos K Katsaggelos. Efficient video object segmentation via network modulation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6499–6507, 2018. 6
- [45] Fengwei Yu, Wenbo Li, Quanquan Li, Yu Liu, Xiaohua Shi, and Junjie Yan. Poi: Multiple object tracking with high performance detection and appearance feature. In *European Conference on Computer Vision*, pages 36–42. Springer, 2016. 3
- [46] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *European Conference on Computer Vision*, pages 474–490. Springer, 2020. 5
- [47] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 2
- [48] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. Online multi-object tracking with dual matching attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 366–382, 2018. 3