VIDEO INSTANCE SEGMENTATION

Zhihao Duan, Xueyao Liang

Boston University
Department of Electrical and Computer Engineering
8 Saint Mary's Street
Boston, MA 02215
`www.bu.edu/ece`

DEC. 11, 2019

Technical Report No. ECE-2019-04

Summary

This article is the final report of the project - Video Instance Segmentation in the course EC720 at Boston University. In this project, our task is to spatiotemporally segment all the objects in a video in the form of 3-D binary masks. We start with an image instance segmentation algorithm and build correspondence between resulting detections in different frames. The algorithms we tried include object re-identification and object tracking by deep neural networks. Based on the results, we conclude that our proposed algorithm is not far away from the state-of-the-art method, and there are opportunities that our algorithm can be further improved.

Contents

List of Figures

List of Tables

1 Introduction

Image instance segmentation has attracted a tremendous amount of attention in computer vision due to its wide real-world applications, e.g., image editing, surveillance, autonomous driving, and augmented reality. However, video instance segmentation, which extends it from 2-D image domain into 3-D video domain, is a relatively new topic [9]. In this project, we will focus on video instance segmentation, aiming at developing an algorithm that can solve the problem.

2 Related work

Image Instance Segmentation: Image instance segmentation requires a precise segmentation of each object in an image. Formally, given an $h \times w$ image $I \in [0,1]^{h\times w}$, our algorithm should return n binary masks $\boldsymbol{M}^1, ..., \boldsymbol{M}^n \in \{0,1\}^{h\times w}$ and n category labels $c^1, ..., c^n \in \{\text{person, cat, dog, ...}\}$ corresponding to n ground truth objects in the image. Fig. 1 illustrates the difference between Object Detection and Instance Segmentation.
A variety of image instance segmentation methods based on deep learning have been developed in recent years. Mask R-CNN [5] is one such state-of-the-art method. Given an input image, it first extracts some Region of Interest (RoI), which probably contains an object in the image. Then, in each RoI, it predicts the category label, bounding box, and segmentation mask of the object.

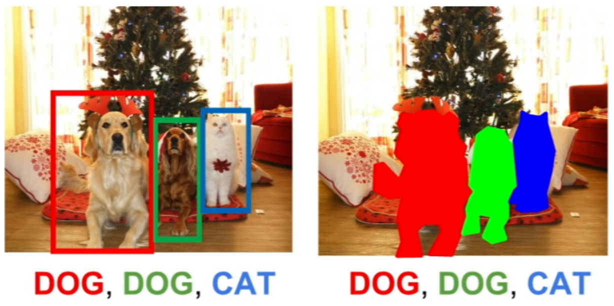


Figure 1: The left image refers to Object Detection, while the right image refers to Instance Segmentation. (image credits: cs231n.stanford.edu/slides/2017/ lecture11)

Video Object Tracking: There are two different problem formulations used in video object tracking problems. One is called 'tracking-by-detection', in which a sequence of images is fed into an algorithm, and the algorithm is supposed to output n tunnels of bounding boxes [7]. Methods of this type usually detect all objects in each frame and then connect them across video frames. The other formulation is 'detection-free' tracking. Given a sequence of images $I_1, ..., I_T \in [0,1]^{h\times w}$ and a bounding box $\boldsymbol{B}_1 \in \mathbb{R}^4$ which contains an object in the first frame, the algorithm is supposed to output the bounding box $\boldsymbol{B}_t \in \mathbb{R}^4, \forall t = 2, ..., T$ in all the following frames [3] [8]. Methods of this type are usually 'class-agnostic'. In other words, they should be able to track an object of any category.

3 Problem statement

3.1 Problem Description

Formally, in video instance segmentation, we have a predefined category label set $C = \{\text{person, cat, dog, ...}\}$. Given a sequence of image frames $I_1, ..., I_T \in [0,1]^{h\times w}$, our algorithm should produce n video instance hypotheses. For each hypothesis j, the following are predicted: a category label $cls^j \in C$, a confidence score $conf^j \in [0,1]$, and a sequence of predicted binary masks $\boldsymbol{M}_1^j, ..., \boldsymbol{M}_T^j \in \{0,1\}^{h\times w}$, where T is the number of frames. In others words, we have to find all the object tunnels given an image sequence [9].

3.2 Dataset and Metric

We will focus on YouTube Video Instance Segmentation (VIS) dataset since it is very large-scale compared to other related datasets [4]. The metric used in YouTube Video VIS challenge is Average Precision (AP), i.e., the area under the precision-recall curve, which is commonly used in object detection and image instance segmentation. In YouTube VIS, the Intersection over Union (IoU) is computed spatially-temporally:

$$IoU(i,j) = \frac{\sum_{t=1}^{T} \boldsymbol{M}_t^i \cap \boldsymbol{M}_t^j}{\sum_{t=1}^{T} \boldsymbol{M}_t^i \cup \boldsymbol{M}_t^j},$$

where i is a detection, j is ground truth, and T is the number of frames. A detection, which contains T binary masks, is considered as a True Positive if it has a large overlap (IoU ≥ 0.5) with any ground truth. Otherwise, it is considered as a False Positive.

3.3 State-of-the-art Methods

The winner of YouTube VIS 2019 challenge [2] combined detection, segmentation, classification, and tracking in order to achieve state-of-the-art performance. However, their method is not clearly described in their report [2]. To my best understanding, they first apply Mask R-CNN to every frame, then they use a classification network pre-trained on ImageNet to eliminate misclassifications. Also, they adopt a tracking method [6], the winner of DAVIS 2019 Challenge [4], to track all objects.
The second place of the YouTube VIS 2019 challenge uses Mask R-CNN for image instance segmentation as well[1]. After obtaining image-level detections, they compute similarity scores between these image-level detections and existing tunnels, which are maintained by another tracking method called SiamMask [8]. By repeating this step

at each frame, they obtain a set of object tunnels. Finally, they use an image classification network to refine the object tunnels. The diagram of their method is shown in Fig. 2.

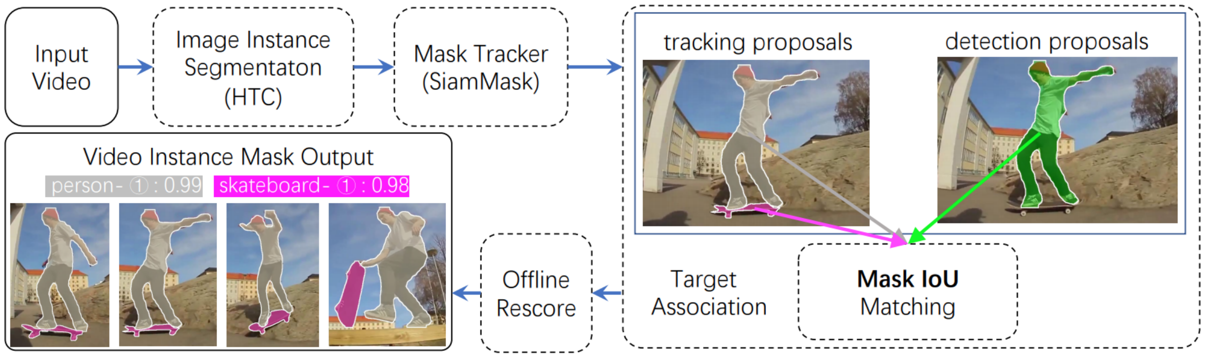


Figure 2: Block diagram of the second-place method in YouTube VIS 2019 challenge.

4 Implementation

In our work, we made the following global assumptions:

- Frame rate ≥ 6 Hz,
- Objects do not disappear and appear again,
- No heavy occlusion.

4.1 Build tracklets (”tunnels”) by correspondence

If we make an additional assumption that Mask R-CNN gives perfect image instance segmentation at each frame, our problem simplifies to a correspondence problem, where at each step we should associate between the current detections $\boldsymbol{M}_1^t, \boldsymbol{M}_2^t, \ldots$ and previous detections $\boldsymbol{M}_1^{t-1}, \boldsymbol{M}_2^{t-1}, \ldots$.

In order to solve a correspondence problem, one way is to treat it as a classification problem. Given two objects $\boldsymbol{M}^i, \boldsymbol{M}^j \in \{0,1\}^{h \times w}$ and their features, we want to classify whether they are the same object or not. Some possible features are:

- Difference between object sizes $= |sum(\boldsymbol{M}^i) - sum(\boldsymbol{M}^j)|$.
- Intersection over Union (IoU) $= \frac{\boldsymbol{M}^i \cap \boldsymbol{M}^j}{\boldsymbol{M}^i \cup \boldsymbol{M}^j}$.
- Difference between locations $\|center(\boldsymbol{M}^i) - center(\boldsymbol{M}^j)\|_2$.
- K-L divergence between HSV histograms: $D_{HSV} = KL(hist_{HSV}(i); hist_{HSV}(j))$.
- K-L divergence between RGB histograms: $D_{RGB} = KL(hist_{RGB}(i); hist_{RGB}(j))$.

One thing we can do is to use YouTube VIS dataset [9] to generate a same/different binary classification dataset of object pairs: same-object pairs and different-object pairs. More specifically, we can find the same object in adjacent frames and calculate the above features so that we have positive samples. Similarly, for negative samples, we can find different objects in adjacent frames and calculate the features.

After obtaining the same/different classification dataset, we can run any classification algorithm to associate new object pairs.

However, in this project, we are doing something simpler. Instead of dealing with these features jointly, we look at them separately. For example, we only calculate the distance between RGB histograms $D_{RGB}(i, j)$. If the distance is smaller than a predefined threshold θ, then both masks belong to the same object; otherwise, they belong to different objects. Our approach is summarized in Algorithm 1. We choose the threshold by analyzing the statistics in the YouTube VIS dataset. We build the histogram of K-L divergence between RGB histograms of same-object pairs, as shown in Fig. 3. We assume that such feature of different-object pairs is uniformly distributed. Then, we use maximum likelihood estimation to choose the threshold. In this case, the threshold $\theta = 1.4$.

Algorithm 1: Correspondence by thresholding

```
Select $f(\cdot)$ from the above similarity(-distance) functions, e.g., $f(\cdot) = IoU(\cdot)$;
for $t = 1$ to $T$ do
    Receive frame $\boldsymbol{I}_t \in [0,1]^{h\times w}$;
    Feed $\boldsymbol{I}_t$ into Mask R-CNN, get $N_t$ detections $\boldsymbol{M}_t^1, ..., \boldsymbol{M}_t^{N_t} \in \{0,1\}^{h\times w}$;
    for $i = 1$ to $N_t$ do
        $f^*, j^* = \max_{\text{unmatched } j} f(\boldsymbol{M}_t^i, \boldsymbol{M}_{t-1}^j)$;
        if $f^* \geq \theta$ then
            $\boldsymbol{M}_t^i$ matches $\boldsymbol{M}_{t-1}^{j^*}$;
        else
            Create a new tunnel containing $\boldsymbol{M}_t^i$;
        end
    end
end
Return all tunnels;
```

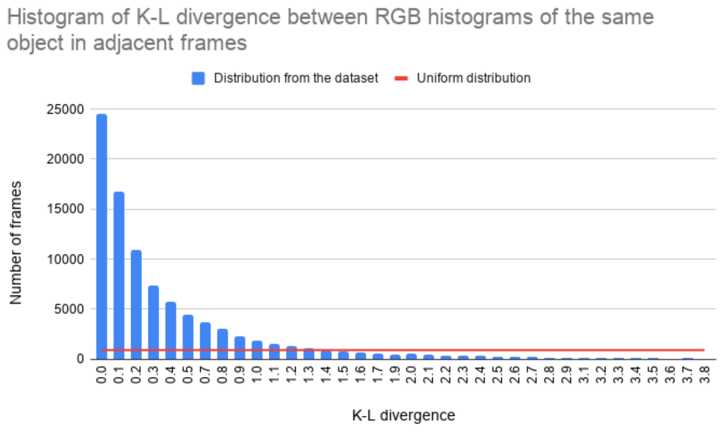


Figure 3

4.2 Build tracklets ("tunnels") by tracking

In this section, we make a weaker assumption on Mask R-CNN. It does not have to be perfect, but we assume that it has a high Precision (the reason will be discussed later).

Since Mask R-CNN is not always perfect, we may face difficulties using the previously discussed correspondence algorithm. In particular, if Mask R-CNN misses an object i in several frames but then detects it again, the previous algorithm will fail to append the current detection to i's tracklet. It is natural to think of video object tracking, which aims at tracking an object in a sequence of images. So, we investigated several video object tracking algorithms and chose SiamMask as our new tool since it outputs masks. We will first briefly introduce SiamMask before we describe our algorithm.

SiamMask: The name of SiamMask comes from the fact that it is based on a Siamese network and it produces masks. At each iteration t, SiamMask receives a small image $\boldsymbol{I}'_{t-1}$, which is an image window containing an object in the previous frame, and a large image $\boldsymbol{I}_t$, which is the whole current frame. It first converts these two images into feature maps using CNNs, and then calculates the cross-correlation between these two feature maps. Finally, it uses CNNs to regress the mask in the current frame.

Our algorithm: We propose a simple combination of Mask R-CNN and SiamMask. Our main idea is to keep tracking the object by SiamMask after we detect it by Mask R-CNN. To achieve this goal, we run both Mask R-CNN and SiamMask at each iteration. Fig. 4 illustrates how our algorithm works on a single object.

Let $\boldsymbol{I}_1, ..., \boldsymbol{I}_T \in [0,1]^{h \times w}$ be a sequence of images. At each iteration, we run Mask R-CNN on the current frame. Let's say Mask R-CNN first detects an object $\boldsymbol{M}_{dt,t}$ in frame $\boldsymbol{I}_t$ at iteration t. Then, we initialize SiamMask with the bounding box of $\boldsymbol{M}_{dt,t}$. From now on, SiamMask will keep tracking the object independent of Mask R-CNN. At iteration $t+1$ and $t+2$, let us assume Mask R-CNN misses an object. Clearly, we have no mask in the images. However, in the hidden back end, SiamMask will still output tracking results $\boldsymbol{M}_{tr,t+1}$ and $\boldsymbol{M}_{tr,t+2}$ of this object. At iteration $t+3$, we run Mask R-CNN and SiamMask as usual. As a result, we get outputs $\boldsymbol{M}_{dt,t+3}$ and $\boldsymbol{M}_{tr,t+3}$ respectively. At this time, there is an Mask R-CNN output as well as a tracklet, so we calculate $f(\boldsymbol{M}_{dt,t+3}, \boldsymbol{M}_{tr,t+3})$ where $f(\cdot)$ is a similarity function, e.g., the Intersection over Union (IoU). If $\boldsymbol{M}_{dt,t+3}$ and $\boldsymbol{M}_{tr,t+3}$ are similar enough, we merge the detection $\boldsymbol{M}_{dt,t+3}$ with the tracklet. Indeed, our algorithm is very similar to one of the state-of-the-art algorithms [1].

For the multi-object case, at each iteration t, we first sort the N_t detections $\boldsymbol{M}^1_{dt,t}, ..., \boldsymbol{M}^{N_t}_{dt,t}$ by their confidence scores and repeat the above procedure for each object. The algorithm is shown in Algorithm 2. Since we will start a new tracklet if there exists a detection that does not match any existing tracklet, we require Mask R-CNN to have a high Precision. If the Precision is low, i.e., many detections are not an object, the algorithm will create too many tracklets, which will result in a high time complexity.

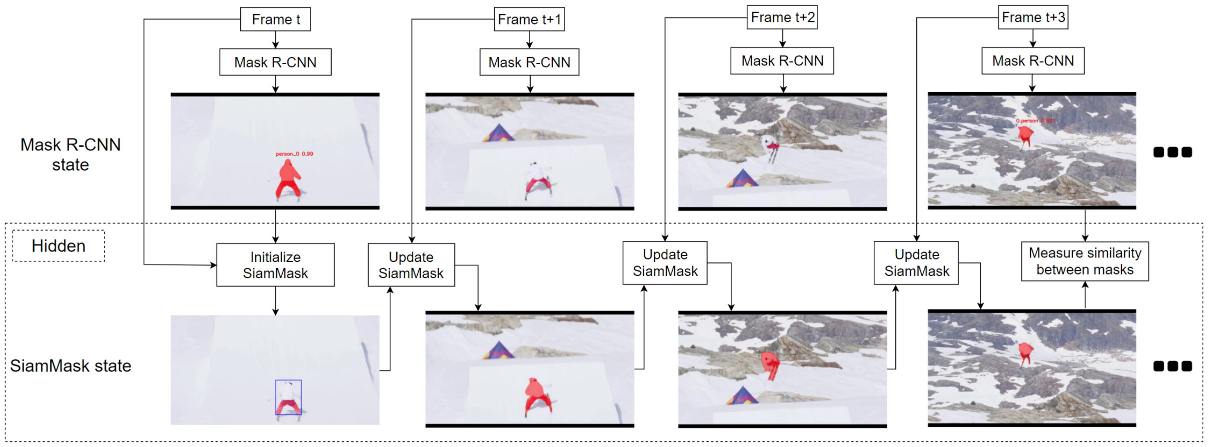
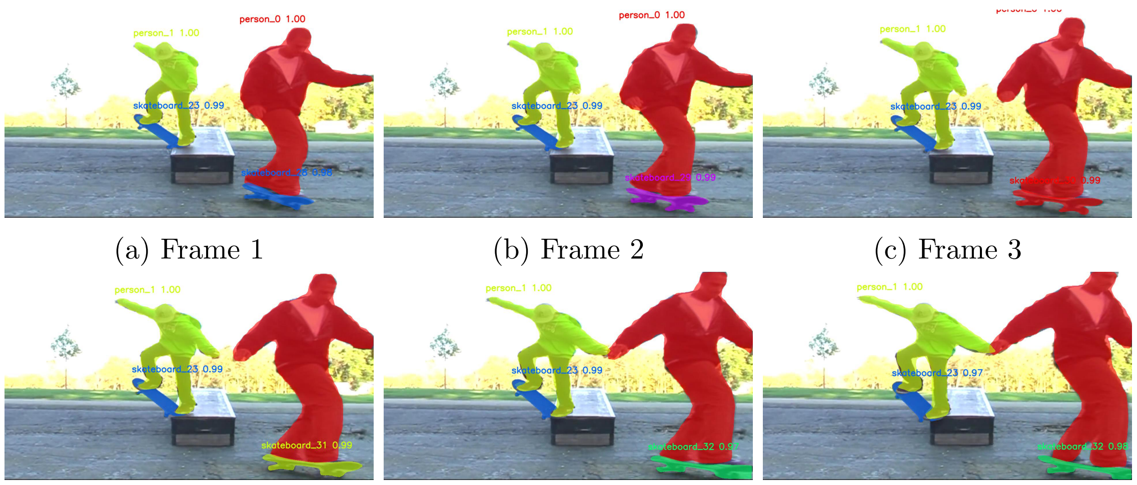


Figure 4: Block diagram of the proposed algorithm

Algorithm 2: Multi-object video instance segmentation using Mask R-CNN and SiamMask

```
Input: similarity function $f(\cdot,\cdot)$ and threshold $\theta$;
Initialization: number of tracklets $P = 0$;
for $t = 1$ to $T$ do
    Receive frame $\boldsymbol{I}^t \in [0,1]^{h\times w}$ ;
    Get $P$ tracking outputs $\boldsymbol{M}^1_{tr,t}, ..., \boldsymbol{M}^P_{tr,t} = \text{SiamMask}(\boldsymbol{I}_t)$;
    Get $N_t$ detection outputs $\boldsymbol{M}^1_{dt,t}, ..., \boldsymbol{M}^{N_t}_{dt,t}, = \text{MaskRCNN}(\boldsymbol{I}_t)$;
    for $i = 1$ to $N_t$ do
        Find the best matched tracklet $f^*, j^* = \max_j \; f(\boldsymbol{M}^i_{dt,t}, \boldsymbol{M}^j_{tr,t})$;
        if $f^* \geq \theta$ then
            $\boldsymbol{M}^i_{dt,t}$ is appended to the tracklet $j^*$;
        else
            Do nothing
        end
    end
    for All $i$ : $\boldsymbol{M}^i_{dt,t}$ is unmatched do
        Initialize a SiamMask tracklet using $\boldsymbol{M}^i_{dt,t}$;
        $P \leftarrow P + 1$
    end
end
Return all tracklets;
```



(a) Frame 1 (b) Frame 2 (c) Frame 3

(d) Frame 4 (e) Frame 5 (f) Frame 6

Figure 5: Results for our first method. The color of the mask indicates the tracklet, i.e., "tunnel".

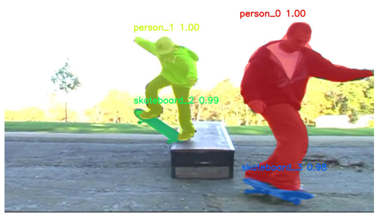
	Test set split	# of videos	# of categories	AP
State-of-the-art 2019	Official	343	40	0.467
Baseline 2018	Official	343	40	0.303
Mask R-CNN + SiamMask (ours)	Our split	200	19	0.403

Table 1: Performance of our method compared to other published methods

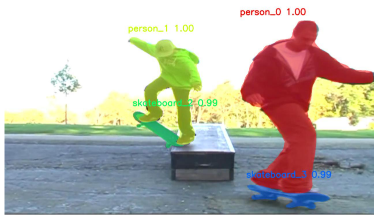
5 Experiments

The results for our first method, which builds tracklets by correspondence, are shown in Fig. 5. We choose IoU to be our similarity function in this case. We can observe that two people and one skateboard are correctly tracked. However, the bottom-right skateboard is continuously identified as a new tunnel. Its speed is too fast compared to its size, resulting in a low IoU between its masks in adjacent frames. Since we concluded this algorithm is not robust in many hard cases, we did not compute its quantitative performance.

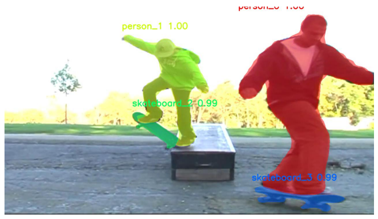
Fig. 6 shows the results for our second method. We observe that there is a huge improvement over our first method. Table 1 shows the quantitative performance of our second algorithm compared to some published methods. The metric we use is Average Precision (AP), as described in previous sections. A higher AP indicates better performance. Note that the test sets are different, so we cannot directly compare them. We observe, however, that by simply combining an image instance segmentation method with an object tracking method, we can approach state-of-the-art methods.



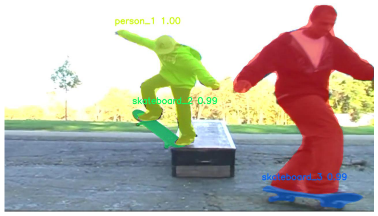
(a) Frame 1



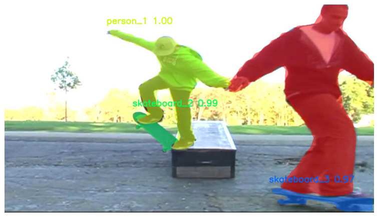
(b) Frame 2



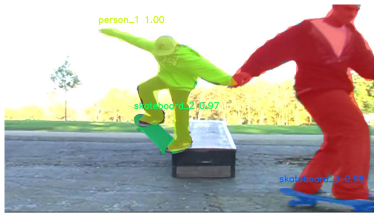
(c) Frame 3



(d) Frame 4



(e) Frame 5



(f) Frame 6

Figure 6: Results for our second method. The color of the mask indicates the tracklet, i.e., "tunnel".

6 Conclusion

In this project, we explored some simple ways to solve the video instance segmentation problem. We conclude that by simply combining an image instance segmentation method with an object tracking method, we can approach state-of-the-art methods. Our future work may focus on how to relax assumptions, i.e., high frame rate, objects do not disappear and appear again, and no heavy occlusion. Possible directions might be to use re-identification methods to identify the object if it appears again.

References

[1] "An empirical study of detection-based video instance segmentation." https://youtube-vos.org/assets/challenge/2019/reports/YouTube-VOS-08-An$_{E}mpirical_{S}tudy_{o}f_{D}etection - Based_{V}ideo_{I}nstance_{S}egmentation.pdf$.

[2] "Video instance segmentation 2019: A winning approach for combined detection, segmentation, classification and tracking." https://youtube-vos.org/assets/challenge/2019/reports/YouTube-VOS-07-A$_{W}inning_{A}pproach_{f}or_{C}ombined_{D}etection_{S}egmentation_{C}lassification_{T}racking.pdf$.

[3] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-Convolutional Siamese Networks for Object Tracking," *arXiv e-prints*, p. arXiv:1606.09549, Jun 2016.

[4] S. Caelles, J. Pont-Tuset, F. Perazzi, A. Montes, K.-K. Maninis, and L. Van Gool, "The 2019 davis challenge on vos: Unsupervised multi-object segmentation," *arXiv:1905.00737*, 2019.

[5] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *arXiv e-prints*, p. arXiv:1703.06870, Mar 2017.

[6] B. L. I. E. Zulfikar, J. Luiten, "Unovost: Unsupervised offline video object segmentation and tracking for the 2019 unsupervised davis challenge," *The 2019 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2019.

[7] A. Sadeghian, A. Alahi, and S. Savarese, "Tracking The Untrackable: Learning To Track Multiple Cues with Long-Term Dependencies," *arXiv e-prints*, p. arXiv:1701.01909, Jan 2017.

[8] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. S. Torr, "Fast Online Object Tracking and Segmentation: A Unifying Approach," *arXiv e-prints*, p. arXiv:1812.05050, Dec 2018.

[9] L. Yang, Y. Fan, and N. Xu, "Video Instance Segmentation," *arXiv e-prints*, p. arXiv:1905.04804, May 2019.