Appendix - Less than Few: Self-Shot Video Instance Segmentation

Pengwan Yang[1,2], Yuki M. Asano[1,2], Pascal Mettes[2], and Cees G. M. Snoek[1,2]

[1] QUVA Lab, University of Amsterdam, Amsterdam, The Netherlands
[2] Institute of Informatics, University of Amsterdam, Amsterdam, The Netherlands
yangpengwan2016@gmail.com

1 Further Details

1.1 Datasets

Dataset statistics. We report the statistics of the self-shot video instance segmentation datasets in Table A. We list the class names for the train, validation and test sets in Table B.

Table A: Overview of the self-shot video instance segmentation datasets. Self-VIS contains a few instances for a single class per video, while Self-OVIS contains videos with more instances of multiple classes

	Self-VIS	Self-OVIS
Video statistics		
mean length (frame)	30.2	69.4
mean class number	1	1.6
mean instance number	1.7	5.9
number of train videos	1,651	449
number of val+test videos	472	158
Class statistics		
number of train classes	30	17
number of val+test classes	10	8

1.2 Self-shot construction

Ranking loss. $R_a(b, c)$ in Equation 2 in the paper is a differentiable function to approximately rank video b among all videos in the set $\{c\}$ with respect to the query video a. It was first introduced for images in [14].

$$R_a(b, c) = 1 + \sum_{\hat{c} \in \{c\}, \hat{c} \neq b} \frac{1}{1 + \exp(-d_{a\hat{c}b}/2)}, \tag{A}$$

Table B: Subset labels on self-shot video instance segmentation datasets. Performance numbers in the paper refer to models evaluated on the test splits. Note that we do not use the labels in our experiments

	Self-VIS	**Self-OVIS**
Training	TennisRacket, Snowboard, Surfboard, EarlessSeal, Horse, Boat, Mouse, Tiger, Frog, Elephant, Truck, Owl, Airplane, Zebra, Train, Deer, Fish, Leopard, Turtle, Fox, Duck, Snake, Skateboard, Rabbit, Cat, Sedan, Parrot, GiantPanda, Ape, Person	Fish, Vehical, Sheep, Zebra, Bird, Poultry, Elephant, Motorcycle, Dog, Monkey, Boat, Turtle, Cow, Parrot, Giraffe, Tiger, GiantPanda
Validation and testing	Eagle, Cow, Motorbike, Bear, Dog, Shark, Giraffe, Monkey, Lizard, Hand	Person, Horse, Bicycle, Rabbit, Cat, Airplane, Bear, Lizard

$$d_{a\hat{c}b} = \frac{a \cdot \hat{c}}{\|a\| \cdot \|\hat{c}\|} - \frac{a \cdot b}{\|a\| \cdot \|b\|}. \tag{B}$$

With this approximation we are able to design the differentiable ranking loss $\mathcal{L}_{Rank}$ of Equation 2.

1.3 Self-shot VIS transformer

Feature extraction. We adopt a modified ResNet-50 [5] with a bigger receptive field for feature extraction. Specifically, we remove the last stage of the ResNet-50 and take the outputs of the fourth stage as final outputs. We further modify the 3×3 convolution in the fourth stage to a dilated convolution with a stride of 2 instead of 1 to increase the receptive field. Finally, a learned 1×1 convolution is applied to reduce the channel dimension. During training, the modified layers are initialized with Xavier [3] init, all other backbone parameters are initialized with a COCO-pretrained ResNet-50.

Query-support fuser. The query-support fuser can effectively leverage the similarity between the query and support videos, the detailed structure of which is shown in Figure A. First, the query and support representations are enhanced by each other simultaneously through multi-head attention. Then, the support branch is fused into the query branch by the multi-head cross-attention mechanism. Finally, a feed-forward neural network (FFN) module in the form of a residual is performed to augment the fitting ability. To better understand the effectiveness of the fuser module, we visualize the attention heatmaps before and after the fuser module in Figure B. The figure shows that the proposed fuser module is able to better highlight the instances of interest in the query video with the aid of supports.

Instance segmenter. The instance segmenter predicts the mask sequence for each instance. To predict mask sequences that are informed across time, we predict the masks based on the accumulation of mask features per instance. For each frame in the query video, we feed the instance features $\mathcal{D}$ and the fused feature $\mathcal{F}$ into an attention module to obtain the attention maps. Then the

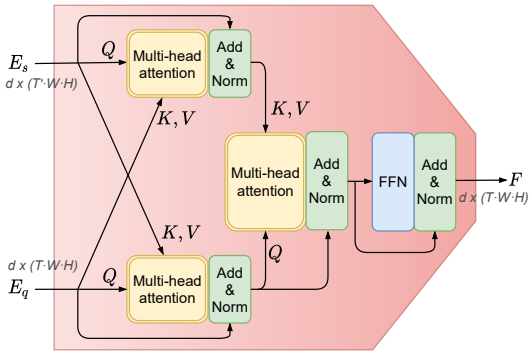

Fig. A: Structure of query-support fuser. First, the query and support representations are enhanced by each other simultaneously through message passing. Then, the support branch is fused into the query branch by the multi-head cross-attention mechanism. Finally, a feed-forward neural network (FFN) module in the form of a residual is performed to augment the fitting ability. **Q, K, V** denote different roles in the multi-head attention mechanism

attention maps are concatenated with the encoded query feature E_q and the fused feature $\mathcal{F}$, followed by a deformable convolution layer [2]. In this way, we obtain the mask features for each instance of different frames in the query video. Assume that the mask feature for instance i of frame t is $g_{i,t} \in \mathbb{R}^{1\times a\times W_0\times H_0}$, where a is the channel number, W_0 and H_0 are the feature width and height. Then we concatenate the mask features of T frames in the query video to form the $G_i \in \mathbb{R}^{1\times a\times T\times W_0\times H_0}$. The instance segmenter takes the instance mask features G_i as input, and outputs the mask sequence $m_i \in \mathbb{R}^{1\times 1\times T\times W_0\times H_0}$ for the instance i directly. The instance segmenter consists of three 3D convolutional layers and group normalization layers [15] with ReLU activation function, yielding T mask predictions for each instance i.

Optimal bipartite matching. Let $\hat{y}=\{\hat{y}\}_{i=1}^{n}$ denote the predicted instance sequences, and y the ground truth set of instance sequences. Assuming n is larger than the number of instances in the query video, we consider y also a set of size n by padding with $\emptyset$ (background). To find a bipartite matching between the two sets we search for a permutation of n elements σ with the lowest cost. As computing the instance mask sequence similarity directly is computationally intensive, we replace the mask sequence with the box sequence to perform the matching. Following the same practice of [1,16], we get the normalized center coordinates, height and width of the boxes and the binary *foreground/background* labels. The matching function is defined as [1]:

$$\mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}) = -\hat{p}_{\sigma(i)}(c_i) + \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)}), \tag{C}$$

where c_i=*foreground*, $\hat{p}_{\sigma(i)}(c_i)$ denotes the probability of c_i with index $\sigma(i)$, b_i denotes the ground truth box sequences, and $\hat{b}_{\sigma(i)}$ denotes the predicted box

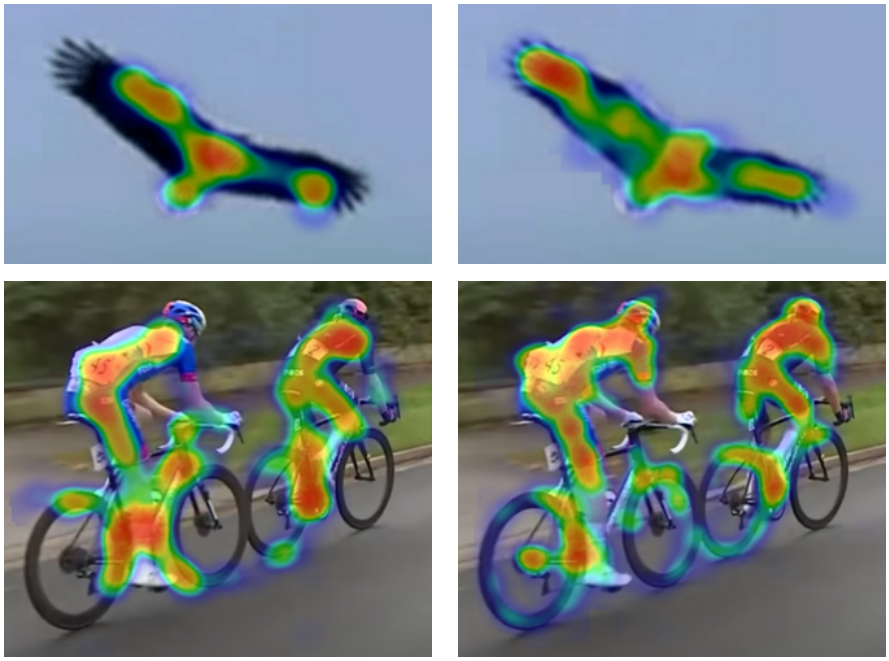

Fig. B: Attention heatmap visualization before and after the fuser module. For both examples, the left and right show the attention heatmaps before and after the fuser module. The heatmap results show the fuser module can better highlight the individual instances in the query video with the aid of supports.

sequences with index $\sigma(i)$. Based on the above criterion, a one-to-one matching of the sequences is found by the Hungarian algorithm [6], following prior work [13,1].

Components of training loss. For the bounding box loss $\mathcal{L}_{\text{box}}$, we use a linear combination of the sequence level $L1$ loss and the generalized IoU loss [11]:

$$\begin{aligned}\mathcal{L}_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}(i)}) = \frac{1}{T}\sum_{t=1}^{T}[\lambda_{iou} \cdot \mathcal{L}_{\text{iou}}(b_{i,t}, \hat{b}_{\hat{\sigma}(i),t}) \\ + \lambda_{L1}\|b_{i,t} - \hat{b}_{\hat{\sigma}(i),t}\|_1],\end{aligned} \tag{D}$$

The mask loss is defined as a combination of the dice loss [10] and focal loss [7]:

$$\begin{aligned}\mathcal{L}_{\text{mask}}(m_i, \hat{m}_{\hat{\sigma}(i)}) = \frac{1}{T}\sum_{t=1}^{T}[\mathcal{L}_{\text{dice}}(m_{i,t}, \hat{m}_{\hat{\sigma}(i),t}) \\ + \mathcal{L}_{\text{focal}}(m_{i,t}, \hat{m}_{\hat{\sigma}(i),t})],\end{aligned} \tag{E}$$

2 Further results

2.1 Few-shot experiments with oracle supports

Ablation of transformer depth. We ablate the effect of the layer depth of the encoder, fuser and decoder modules in Table C under the one- and five-shot setting on Self-VIS. We find the increase in performance is steady across all three modules when adding further capacity and use the setting of 6 encoder layers, 3 fuser layers and 6 decoder layers for all other experiments.

Table C: Ablation of transformer depth of the encoder, fuser and decoder on Self-OVIS under the one- and five- shot setting. The default layer number is 6,3,6 for encoder, fuser and decoder (highlighted in gray). When we ablate the layers of one module, the other two modules have default layers. Performance improves gradually with more layers in the modules

(a) Encoder size

	Encoder layers			
	2	4	6	8
One-shot	51.7	52.4	53.2	53.2
Five-shot	55.1	56.1	56.6	56.4

(b) Fuser size

	Fuser layers				
	1	2	3	4	5
One-shot	51.3	52.5	53.2	53.4	53.5
Five-shot	54.0	55.8	56.6	56.9	57.1

(c) Decoder size

	Decoder layers			
	2	4	6	8
One-shot	49.4	51.8	53.2	53.6
Five-shot	53.1	55.7	56.6	56.8

Benefit of positional encoding. We report results in Table D. Including the spatio-temporal positional encoding is always beneficial, ideally in both the support and query encoder. Not passing any spatio-temporal positional encoding to the features leads to an mAP drop of 3.8 under the one-shot and an mAP drop of 4.3 under the five-shot setting.

Table D: Positional encoding analysis. Adding a positional encoding for both the query and the support yields best performances

Positional encoding		Few-shot setting	
support	query	one-shot	five-shot
		49.4	52.3
	✓	50.3	53.7
✓		52.4	55.9
✓	✓	**53.2**	**56.6**

Comparison against related methods. As the task of few-shot VIS is novel, we cannot compare with existing methods that are not designed for this setting. We nevertheless try and thus adapt existing methods intended for other tasks. Michaelis *et al.* [9] provides a strong baseline for few-shot image instance segmentation based on Mask R-CNN [4] (FMRCNN). They generate many

segmentation proposals for the query image and match the proposals with the support image to find the fitting ones. To build a baseline for few-shot VIS, we first extract the frame features for the support and query videos by our backbone. Then we compute the average of the support frame features as in prototypical networks [12]. Finally, with the averaged support feature, the instance segmenter can predict the instance masks for each query frame. We also compare against Yang *et al*. [16], which is a transformer-based approach for few-shot common action detection (FST). They propose a dedicated encoder-decoder structure to learn the commonality and predict a spatio-temporal localization. We transplant their transformer structure in our pipeline by replacing our common transformer. We compare to both baselines under one- and five-shot settings on Self-VIS and Self-OVIS in Table E. Our approach achieves the best results with large margins on both datasets and settings.

Table E: Few-shot video instance segmentation on Self-VIS and Self-OVIS. We constructed two baselines [9,16] from author-provided code. Our common transformer obtains favourable results under one- and five- shot settings on both datasets

	Self-VIS		**Self-OVIS**	
	1	5	1	5
SMRCNN [9]	43.2	44.7	16.8	17.7
FST [16]	48.5	51.7	20.5	22.3
This paper	**53.2**	**56.6**	**22.0**	**24.9**

Video instance segmentation from images. Besides using videos, we can also perform few-shot video instance segmentation using only images as support. This is a novel and more challenging setting, as the model cannot exploit temporal information from the support videos. In Table F, (img→vid), we perform few-shot video instance segmentation task with support images that are only a single middle frame for each support video using the Self-VIS dataset. We again compare against other methods and find that our approach performs better under both the one- and five- shot setting on this new task. Extending this task even further, we also consider the problem of few-shot image instance segmentation: Given a few support images containing the same object instances, find and segment the common object instances in a query image. As shown in Table F, (img→img), our method can even generalize to this task on the challenging COCO dataset [9,8] where the backbone network is pretrained on ImageNet dataset, outperforming the other methods.

2.2 Self-shot experiments

Transformer ablation. In the Table 3 of the paper, we show the effect of the three components in our proposed transformer under few-shot setting with oracle supports. Here we show in Table G the results of the transformer ablation under

Table F: Generalisation to images. We report few-shot instance segmentation performance on the video and images when only image support input is provided. For video evaluation we use Self-VIS and for image evaluation MS-COCO from [9]. For MS-COCO we evaluate mAP at IoU level of 0.5 as in [9]

	img→vid		**img→img**	
	1	5	1	5
SMRCNN [9]	41.8	44.1	14.2	16.3
FST [16]	43.3	46.4	15.1	17.7
This paper	**48.7**	**52.0**	**15.6**	**18.4**

Table G: VIS transformer ablation under self-shot setting. Besides the transformer ablation under the few-shot setting reported in the paper (Table 3), we also report the effectiveness of the three modules in the proposed transformer under the self-shot setting.

Encoder	**Fuser**	**Decoder**	**Self-VIS**		**Self-OVIS**	
			1 self-shot	5 self-shots	1 self-shot	5 self-shots
		✓	40.7	40.8	13.7	14.4
✓		✓	42.4	42.2	14.3	15.7
	✓	✓	49.1	51.5	19.6	21.0
✓	✓	✓	**51.4**	**54.3**	**20.6**	**23.7**

self-shot setting and observe that the proposed transformer is also effective for video instance segmentation with self-shot supports.

Self-shot versus zero-shot learning. As a supplement to the experiments in Table 2 in the paper, we construct another two zero-shot baselines by randomly cropping segments in time or in space from the query video as supports. Results in Table H show that self-shot supports from a unlabelled pool of videos are more effective than the supports from the query video itself. This demonstrates that sample diversity in the support helps improve performance.

Further examples. In Figure C, we provide further qualitative examples of self-shot learning.

Table H: Self-shot versus zero-shot learning for video instance segmentation and temporal action localization. The experiment setups follow Table 2 in the paper. We build two simple zero-shot baselines by adapting the temporal or spatial segments of the query video as supports. We find that self-shot supports are more effective than segments from the query video itself

	Self-VIS		**Thumos14**	
	1	5	1	5
Zero-shot learning (temporal segments)	44.0	46.6	40.7	41.9
Zero-shot learning (spatial segments)	44.9	47.8	41.1	42.4
Self-shot learning	51.4	54.3	45.8	47.3
Self-shot learning k+(5)	54.6	55.4	47.7	48.0

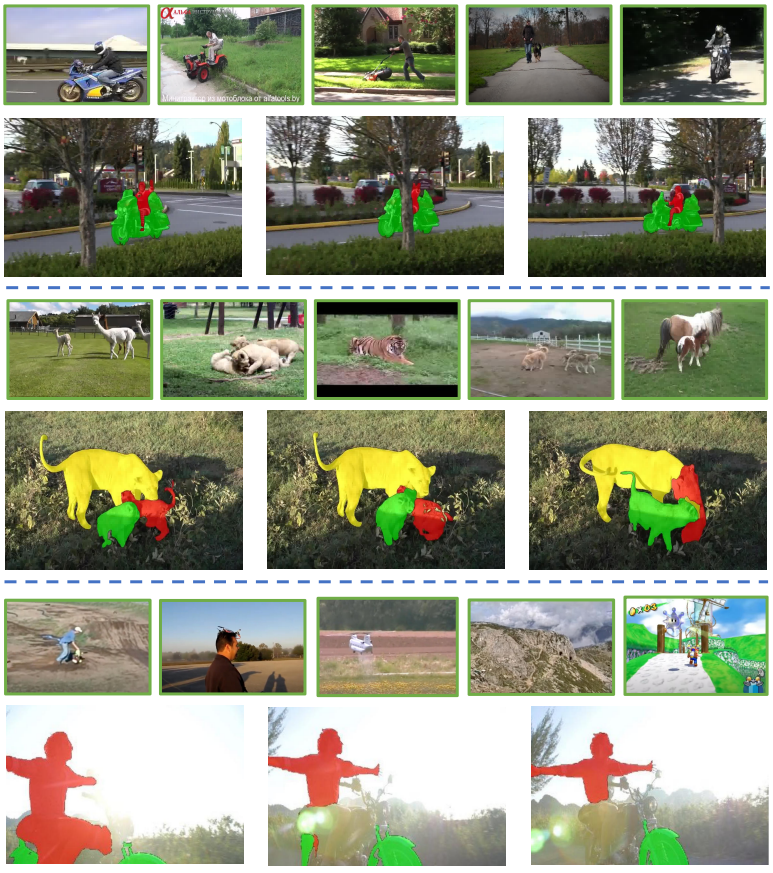

Fig. C: Qualitative examples. Three examples of self-shot videos (top) and the resulting instance segmented query video (bottom). The first and second rows are successful cases while the third example is failure case

References

1. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: ECCV (2020) 3, 4
2. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: ICCV (2017) 3
3. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: AISTATS (2010) 2
4. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV (2017) 5
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016) 2
6. Kuhn, H.W.: The hungarian method for the assignment problem. Naval research logistics quarterly (1955) 4
7. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: ICCV (2017) 4
8. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014) 6
9. Michaelis, C., Ustyuzhaninov, I., Bethge, M., Ecker, A.S.: One-shot instance segmentation. arXiv (2018) 5, 6, 7
10. Milletari, F., Navab, N., Ahmadi, S.A.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: 3DV (2016) 4
11. Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., Savarese, S.: Generalized intersection over union: A metric and a loss for bounding box regression. In: CVPR (2019) 4
12. Snell, J., Swersky, K., Zemel, R.S.: Prototypical networks for few-shot learning. In: NeurIPS (2017) 6
13. Stewart, R., Andriluka, M., Ng, A.Y.: End-to-end people detection in crowded scenes. In: CVPR (2016) 4
14. Varamesh, A., Diba, A., Tuytelaars, T., Van Gool, L.: Self-supervised ranking for representation learning. NeurIPS (2020) 1
15. Wu, Y., He, K.: Group normalization. In: ECCV (2018) 3
16. Yang, P., Mettes, P., Snoek, C.G.M.: Few-shot transformation of common actions into time and space. In: CVPR (2021) 3, 6, 7