A Unified Model for Semi-supervised and Interactive Video Object Segmentation using Space-time Memory Networks

Seoung Wug Oh, Yonsei University
Joon-Young Lee, Adobe Research
Ning Xu, Adobe Research
Seon Joo Kim, Yonsei University

Abstract

We propose a novel solution for semi-supervised and interactive video object segmentation. By the nature of the problem, available cues (e.g. video frame(s) with object masks and user annotations) become richer with the intermediate predictions and additional user interactions. To take advantage of this characteristic, we leverage memory networks to learn to read relevant information from all available sources. In the semi-supervised scenario, the past frames with object masks form an external memory, and the current frame as the query is segmented using the mask information in the memory. In the interactive scenario, the frames given user interactions form an external memory, and the frames not given user input become the query to be segmented. Specifically, the query and the memory are densely matched in the feature space, covering all the space-time pixel locations in a feed-forward fashion. The abundant use of the guidance information allows us to better handle the challenges such as appearance changes and occlussions. In DAVIS challenge 2019, our team win 1st place on the interactive track and 4th place on the semi-supervised track without online learning, pre-computed results such as optical flow, and post-processing.

1. Introduction

Video object segmentation is an essential step for many video editing tasks, which is getting more attention as videos have become the most popular form of shared media contents. We tackle the video object segmentation problem in two scenarios: semi-supervised and interactive. In the semi-supervised setting, the ground truth mask of the target object is given in the first frame and the goal is to estimate the object masks in all other frames. In the interactive mode, the user provides annotations on a selected frame and an algorithm computes segmentation maps for all the frames in the video in a batch process. The user can provide additional annotations and an algorithm need to refine the object masks of all the frames. In both scenarios, it is a very challenging task as the appearance of the target object can change drastically over time and also due to occlusions and drifts. In addition, in the interactive mode, an algorithm need to be able to understand user's intention, at the same time.

In this paper, we propose a novel DNN system based on the memory network [16, 8, 6] that computes the spatio-temporal attention on every pixel in multiple frames of the video for each pixel in the query image, to decide whether the pixel belongs to a foreground object or not. With our framework, there is no restriction on the number of frames to use and new information can be easily added by putting them onto the memory. In the semi-supervised scenario, the past frames with object masks form an external memory, and the current frame as the query is segmented using the mask information in the memory [9]. In the interactive scenario, the frames given user interactions form an external memory, and the frames not given user input become the query to be segmented. This memory update greatly helps us to address the challenges and enable us to recover from the failure mode with a aid of user interactions.

In addition to using more temporal information, our network inherently includes non-local spatial pixel matching mechanism that is well suited for pixel-level estimation problems. By exploiting rich reference information, our approach can deal with appearance changes, occlusions, and drifts much better than the previous methods. Our method took first place on the interactive track of DAVIS Challenge on Video Object Segmentation 2019 with with an AUC of 0.783 and a J&F@60s of 0.791. Our method also take fourth position on the semi-supervised track with a global mean of 0.752 without the use of online-learning, post-processing, and optical flow.

2. Algorithm

We consider the video frames with additional information (*e.g.* object masks and user scribbles) as the *memory* frames and the video frame to be segmented as the *query* frame. Both the memory and the query frames are first encoded into pairs of key and value maps through the dedicated deep encoders. Note that the query encoder takes only an image as the input, while the memory encoder takes the

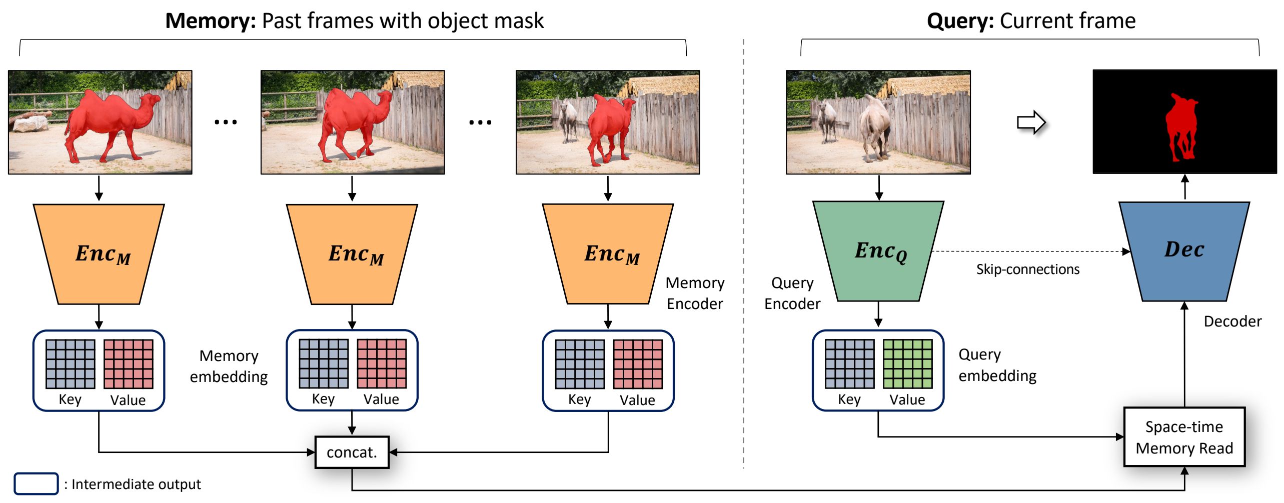

Figure 1: Overview of our framework. Our network consists of two encoders each for the memory and the query frame, a space-time memory read block, and a decoder. The memory encoder (Enc_M) takes an RGB frame and the object mask. The object mask is represented as a probability map (the softmax output is used for estimated object masks). The query encoder (Enc_Q) takes the query image as input.

frames with additional information. Each encoder outputs **Key** and **Value** maps. **Key** is used for addressing. Specifically, similarities between key features of the query and the memory frames are computed to determine when-and-where to retrieve relevant memory **values** from. Therefore, **key** is learned to encode visual semantics robust to appearance variations. On the other hand, **value** stores detailed information for producing the results. **Values** from the query and the memory contain information for somewhat different purposes. Specifically, *value for the query frame* is learned to store detailed appearance information for us to decode accurate object masks. *Value for the memory frames* is learned to encode both the visual semantics and the mask information about whether each feature belongs to the foreground or the background.

The keys and values further go through our space-time memory read block. Every pixel on the key feature maps of the query and the memory frames is densely matched over the spatio-temporal space of the video. Relative matching scores are then used to address the value feature map of the memory frame, and the corresponding values are combined to return outputs. Finally, the decoder takes the output of the read block and reconstructs the mask for the query frame.

2.1. Key and Value Embedding

The query encoder takes the query frame as the input. The encoder outputs two feature maps – key and value – through two parallel convolutional layers attached to the backbone network. The memory encoder has the same structure except for the inputs. The input to the memory encoder consists of an RGB frame and all available information (the object mask and the user scribbles). The object mask is represented as a single channel probability map between 0 and 1 (the softmax output is used for estimated masks). The user scribbles are represented as a two channel binary map where each channel is account for the positive and the negative scribble.

The inputs are concatenated along the channel dimension before being fed into the memory encoder. If there are more than one memory frames, each of them is independently embedded into key and value maps. Then, the key and value maps from different memory frames are stacked along the temporal dimension. We take ResNet50 [5] as the backbone network for both the memory and the query encoder.

2.2. Space-time Memory Read

In the memory read operation (Fig. 2), soft weights are first computed by measuring the similarities between all pixels of the query key map and the memory key map. The similarity matching is performed in a non-local manner by comparing every space-time locations in the memory key map with every spatial location in the query key map. Then, the value of the memory is retrieved by a weighted summation with the soft weights and it is concatenated with the query value. This memory read operates for every location on the query feature map and can be summarized as:

$$\mathbf{y}_i = \left[\mathbf{v}_i^Q,\ \frac{1}{Z}\sum_{\forall j} f(\mathbf{k}_i^Q, \mathbf{k}_j^M)\mathbf{v}_j^M\right], \tag{1}$$

where i and j are the index of the query and the memory location, $Z = \sum_{\forall j} f(\mathbf{k}_i^Q, \mathbf{k}_j^M)$ is the normalizing factor

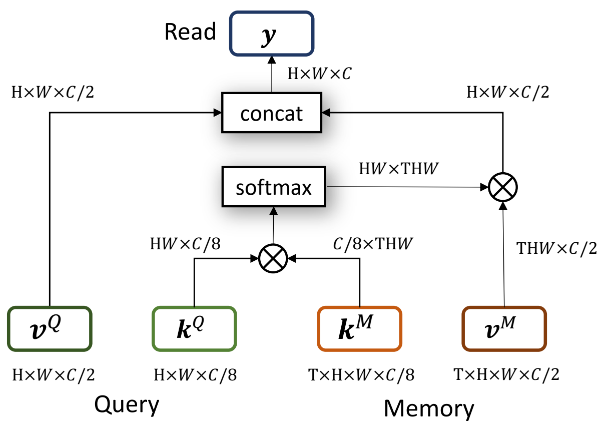

Figure 2: Detailed implementation of the space-time memory read operation using basic tensor operations as described in Sec. 2.2. $\otimes$ denotes matrix inner-product.

and $[\cdot,\cdot]$ denotes the concatenation. The similarity function f is as follows:

$$f(\mathbf{k}^Q_i, \mathbf{k}^M_j) = \exp(\mathbf{k}^Q_i \circ \mathbf{k}^M_j), \tag{2}$$

where $\circ$ denotes the dot-product.

2.3. Decoder

The decoder takes the output of the read operation and reconstructs the current frame's object mask. We employ the refinement module used in [10] as the building block of our decoder.

2.4. Two-stage Training

Our network is first pre-trained on a simulation dataset generated from static image data. Then, it is further fine-tuned for real-world videos through the main training. Note that we trained two individual models each for semi-supervised and interactive scenario.

Pre-training on images. We take similar strategy [10] to synthesize training data from static images datasets (salient object detection – [15, 1], semantic segmentation – [2, 4, 7]).

Main-training on videos. After the pre-training, we use real video data for the main training. In this step, either Youtube-VOS [17] or DAVIS-2017 [14] is used, depending on the target evaluation benchmark.

Scribble synthesis. We use morphological skeletonization and random walk algorithm to automatically generate realistic scribbles. We use a fast implementation of the thinning algorithm [3] for the skeletonization.

3. Inference

3.1. Semi-supervised mode

In semi-supervised mode, video frames are sequentially processed starting from the second frame using the ground truth annotation given in the first frame. During the video processing, we consider the past frames with object masks (either given at the first frame or estimated at other frames) as the memory frames and the current frame without the object mask as the query frame.

However, writing all previous frames on to the memory may raise practical issues such as GPU memory overflow and slow running speed. Instead, we select frames to be put onto the memory by a simple rule. The first and the latest previous frame with object masks are the most important reference information [13, 10, 18]. Therefore, we put these two frames into the memory by default. For the intermediate frames, we simply save a new memory frame every N frames.

3.2. Interactive mode

In interactive mode, every video frames are segmented in a batch process using information in the memory. We consider frames given with user interaction as the memory frames. Therefore, the number of memory frame is grows with user interactions. Note that, in interactive mode, memory encoder takes frame image with both the current object mask and corrective user scribbles.

4. DAVIS Challenge 2019

4.1. Interactive track

In the challenge, each method interacts with a robot agent up to 8 times and is expected to compute masks within 30 seconds per object for each interaction. The performance of each method is evaluated using two metrics: area under the curve (AUC) and J&F score at 60 seconds (J&F@60s). AUC is designed to measure the overall accuracy during the evaluation. J&F@60 measures the accuracy with a limited time budget (60 seconds). Our method took first place on this challenge with an AUC of 0.783 and a J&F@60s of 0.791, which greatly outperforms the winning entry on the interactive track of DAVIS Challenge 2018 [11, 12].

4.2. Semi-supervised track

Our team takes fourth position on the semi-supervised track with a global mean of 0.752. To maximize the speed of our method, we avoid time-consuming tricks for squeezing out performance. In other words, we did not employ online learning, post-processing, model/scale ensemble. Our method does not require pre-computed optical flow.

References

[1] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. Torr, and S.-M. Hu. Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):569–582, 2015. 3

[2] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. 3

[3] Z. Guo and R. W. Hall. Parallel thinning with two-subiteration algorithms. *Commun. ACM*, 32(3):359–373, Mar. 1989. 3

[4] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *IEEE International Conference on Computer Vision (ICCV)*, pages 991–998. IEEE, 2011. 3

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 2

[6] A. Kumar, O. Irsoy, P. Ondruska, M. Iyyer, J. Bradbury, I. Gulrajani, V. Zhong, R. Paulus, and R. Socher. Ask me anything: Dynamic memory networks for natural language processing. In *International Conference on Machine Learning*, pages 1378–1387, 2016. 1

[7] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pages 740–755. Springer, 2014. 3

[8] A. Miller, A. Fisch, J. Dodge, A.-H. Karimi, A. Bordes, and J. Weston. Key-value memory networks for directly reading documents. *arXiv preprint arXiv:1606.03126*, 2016. 1

[9] S. W. Oh, J. Lee, N. Xu, and S. J. Kim. Video object segmentation using space-time memory networks. *CoRR*, abs/1904.00607, 2019. 1

[10] S. W. Oh, J.-Y. Lee, K. Sunkavalli, and S. J. Kim. Fast video object segmentation by reference-guided mask propagation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[11] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim. Fast user-guided video object segmentation by deep networks. *The 2018 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2018. 3

[12] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim. Fast user-guided video object segmentation by interaction-and-propagation networks. In *CVPR*, 2019. 3

[13] F. Perazzi, A. Khoreva, R. Benenson, B. Schiele, and A. Sorkine-Hornung. Learning video object segmentation from static images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[14] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017. 3

[15] J. Shi, Q. Yan, L. Xu, and J. Jia. Hierarchical image saliency detection on extended cssd. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(4):717–729, 2016. 3

[16] S. Sukhbaatar, J. Weston, R. Fergus, et al. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448, 2015. 1

[17] N. Xu, L. Yang, D. Yue, J. Yang, B. Price, J. Yang, S. Cohen, Y. Fan, Y. Liang, and T. Huang. Youtube-vos: Sequence-to-sequence video object segmentation. In *European Conference on Computer Vision (ECCV)*, 2018. 3

[18] L. Yang, Y. Wang, X. Xiong, J. Yang, and A. K. Katsaggelos. Efficient video object segmentation via network modulation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3