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 video object segmentation. By the nature of the problem, available cues (*e.g. *video frame(s) with object masks) become richer with the intermediate predictions. However, the existing methods are unable to fully exploit this rich source of information. We resolve the issue by leveraging memory networks and learn to read relevant information from all available sources. In our framework, 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. 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. Contrast to the previous approaches, the abundant use of the guidance information allows us to better handle the challenges such as appearance changes and occlussions. We validate our method on the latest benchmark sets and achieved the state-of-the-art performance (overall score of 79.4 on Youtube-VOS val set, $\mathcal{J}$ of 88.7 and 79.2 on DAVIS 2016/2017 val set respectively) while having a fast runtime (0.16 second/frame on DAVIS 2016 val set).*

1. Introduction

Video object segmentation is a task of separating the foreground and the background pixels in all frames of a given video. It 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 the semi-supervised setting, where 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. 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.

As in most tasks in computer vision, many deep learning based algorithms have been introduced to solve the video object segmentation problem. With deep learning approaches, the essential question is from which frame(s) should the deep networks learn the cues? In some algorithms, the features were extracted and propagated from the previous frame (Fig. 1(a)) [11, 26]. The main strength of this approach is that it can deal with changes in appearance better, while sacrificing robustness against occlusions and error drifts. Another direction for deep learning based video segmentation is to use the first frame as a reference and independently detect the target object at each frame (Fig. 1(b)) [2, 42, 12]. The pros and cons of this approach are exactly the opposite from the previous approach. Methods that use both the first frame and the previous frame to take the advantages of the two approaches were proposed in [24, 40] (Fig. 1(c)). By using two frames as the source for cues, the algorithm [24] achieved the state-of-the-art accuracy with faster running time, as the algorithm does not require online learning as with other methods.

As using two frames has shown to be beneficial for video segmentation, a natural extension is to use more frames, possibly every frame in the video, for the segmentation task.

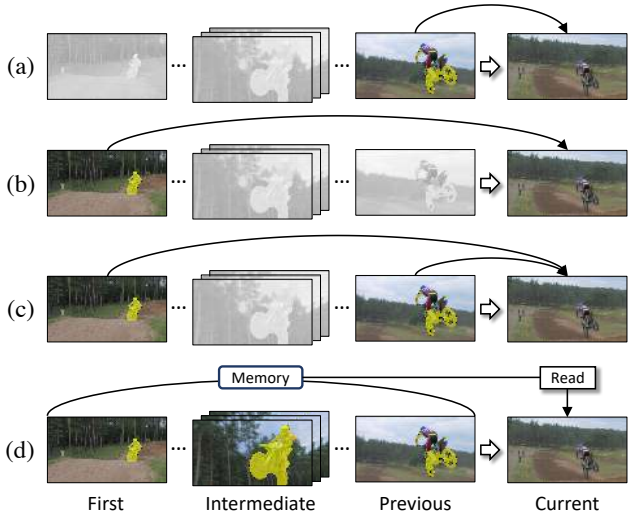

Figure 1: Previous DNN-based algorithms extract features in different frames for video object segmentation (a-c). We propose an efficient algorithm that exploits multiple frames in the given video for more accurate segmentation (d).

*This work was done during an internship at Adobe Research.

The question is how to design an efficient deep neural network (DNN) architecture that exploits all the frames. In this paper, we propose a novel DNN system based on the memory network [30, 22, 16] 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. This memory update greatly helps us to address the challenges like appearance changes and occlussions with no cost. 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. Experimental results show that our method outperforms all the existing methods on public benchmark datasets by a large margin in terms of both speed and accuracy.

2. Related Work

2.1. Semi-supervised Video Object Segmentation

Propagation-based methods [26, 14, 11, 18] learn an object mask propagator, a deep network that refines misaligned mask toward the target object (Fig. 1(a)). To make the network object-specific, online training data is generated from the first frame by deforming the object mask [26] or synthesizing images [14] for fine-tuning. Li *et al*. [18] integrate re-identification module into the system to retrieve missing objects due to drifts.

Detection-based methods [2, 21, 42, 1, 3, 12] work by learning an object detector using the object appearance on the first frame (Fig. 1(b)). In [2, 21], an object-specific detector learned by fine-tuning the deep networks at the test time is used to segment out the target object. In [3, 12], to avoid the online learning, pixels are embedded into feature space and classified by matching to templates.

Hybrid methods [40, 24] are designed to take advantages of both detection and propagation approaches (Fig. 1(c)). In [24, 40], networks that exploit both the visual guidance from the first frame and the spatial priors from the previous frame were proposed. Furthermore, some methods tried to exploit all previous information [38, 34]. In [38], a sequence-to-sequence network that learns the long-term information in videos was proposed. Voigtlaender and Leibe [34] employ the idea of online adaptation and continuously update the detector using the intermediate outputs.

Online/Offline learning. Many of aforementioned methods fine-tune deep network models on the initial object mask in the first frame to remember the appearance of the target object [2, 34, 26, 14, 26, 11, 18] during the test time. While the online learning improves accuracy, it is computationally expensive, limiting its practical use. Offline learning methods attempted to bypass the online learning while retaining the accuracy [24, 40, 3, 12, 13, 32, 33]. A common idea is to design deep networks capable of object-agnostic segmentation at the test time, given guidance information.

Our framework belongs to the offline learning method. Our framework maintains intermediate outputs in the external memory rather than fixing which frame(s) to use as the guidance, and adaptively selects necessary information in runtime. This flexible use of the guidance information makes our method to outperform the aforementioned methods by a large margin. Our memory network is also fast, as the memory reading is done as a part of the network forward pass, thus no online learning is required.

2.2. Memory Networks

Memory networks refer to the neural networks that have external memory where information can be written and read by purposes. Memory networks that can be trained end-to-end were first proposed in the NLP research for the purpose of document Q&A [30, 22, 16]. Commonly in those approaches, memorable information is separately embedded into key (input) and value (output) feature vectors. Keys are used to address relevant memories whose corresponding values are returned. Recently, the memory networks have been applied to some vision problems such as personalized image captioning [25], visual tracking [41], movie understanding [23], and summarization[17].

While our work is based on the memory networks, we extend the idea of the memory networks to make it suitable for our task, semi-supervised video object segmentation. Obviously, frames with object masks are put to the memory, and a frame to be segmented acts as the query. The memory is dynamically updated with newly predicted masks and it greatly helps us to address the challenges like appearance changes, occlusions, and error accumulations without the online learning.

Our goal is to have pixel-wise predictions given a set of annotated frame(s) as memory. Thus each pixel in the query frame needs to access information in the memory frames at different space-time locations. To this end, we coin our memory into 4D tensors to contain pixel-level information and propose the space-time memory read operation to localize and read relevant information from the 4D memory. Conceptually, our memory reading can be considered as a spatio-temporal attention algorithm because we are computing *when-and-where* to attend for each query pixel to decide whether the pixel belongs to a foreground object or not.

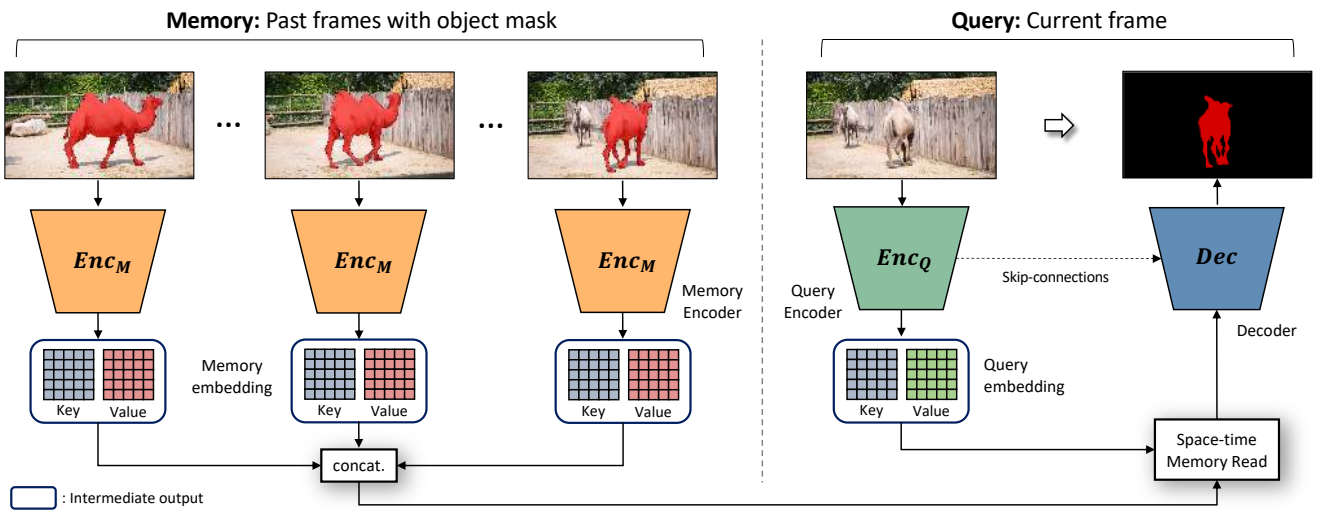

Figure 2: Overview of our framework. Our network consists of two encoders each for the memory and the query frame, a space-time memory read block, and a decoder. The memory encoder (Enc_M) takes an RGB frame and the object mask. The object mask is represented as a probability map (the softmax output is used for estimated object masks). The query encoder (Enc_Q) takes the query image as input.

3. Space-Time Memory Networks (STM)

In our framework, 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. The overview of our framework is shown in Fig. 2.

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 both an image and an object mask. 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 for matching robust to appearance variations. On the other hand, **value** stores detailed information for producing the mask estimation (*e.g.* the target object and object boundaries). **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.

3.1. Key and Value Embedding

Query encoder. 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. These convolutional layers serve as bottleneck layers to reduce the feature channel size of the backbone network output (to 1/8 for the key and 1/2 for the value) and no non-linearity is applied. The output of the query embedding is a pair of 2D key and value maps ($\mathbf{k}^Q \in \mathbb{R}^{H\times W\times C/8}, \mathbf{v}^Q \in \mathbb{R}^{H\times W\times C/2}$), where H is the height, W is the width, and C is the feature dimension of the backbone network output feature map.

Memory encoder. The memory encoder has the same structure except for the inputs. The input to the memory encoder consists of an RGB frame and the object mask. The object mask is represented as a single channel probability map between 0 and 1 (the softmax output is used for estimated masks). 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,

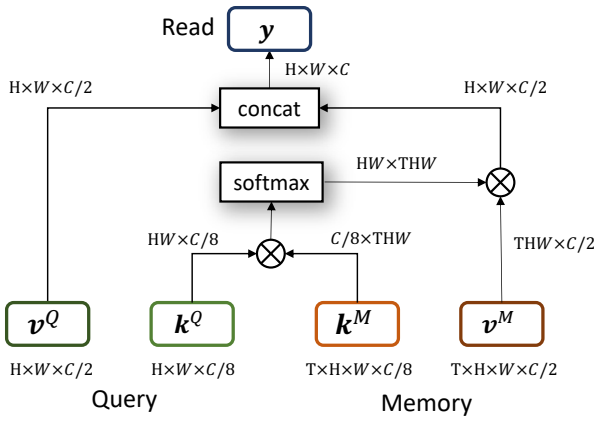

Figure 3: Detailed implementation of the space-time memory read operation using basic tensor operations as described in Sec. 3.2. $\otimes$ denotes matrix inner-product.

the key and value maps from different memory frames are stacked along the temporal dimension. The output of the memory embedding is a pair of 3D key and value maps ($\mathbf{k}^M \in \mathbb{R}^{T\times H\times W\times C/8}, \mathbf{v}^M \in \mathbb{R}^{T\times H\times W\times C/2}$), where T is the number of the memory frames.

We take ResNet50 [9] as the backbone network for both the memory and the query encoder. We use the stage-4 (`res4`) feature map of the ResNet50 as the base feature map for computing the key and value feature maps. For the memory encoder, the first convolution layer is modified to be able to take a 4-channel tensor by implanting additional single channel filters. The network weights are initialized from the ImageNet pre-trained model, except for the newly added filters which are initialized randomly.

3.2. Space-time Memory Read

In the memory read operation, 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:

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

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

$$f(\mathbf{k}_i^Q, \mathbf{k}_j^M) = \exp(\mathbf{k}_i^Q \circ \mathbf{k}_j^M), \tag{2}$$

where $\circ$ denotes the dot-product.

Our formulation can be seen as an extension of the early formulation of the differential memory networks [30, 22, 16] to 3D spatio-temporal space for video pixel matching. Accordingly, the proposed read operation localizes the space-time location of the memory for retrieval. It is also related to non-local self-attention mechanisms [31, 36] in that it performs non-local matching. However, our formulation is motivated for a different purpose as it is designed to attend to others (memory frames) for the information retrieval, not to itself for the self-attention. As depicted in Fig. 3, our memory read operation can be easily implemented by a combination of basic tensor operations in modern deep learning platforms.

3.3. 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 [24] as the building block of our decoder. The read output is first compressed to have 256 channels by a convolutional layer and a residual block [10], then a number of refinement modules gradually upscale the compressed feature map by a factor of two at a time. The refinement module at every stage takes both the output of the previous stage and a feature map from the query encoder at the corresponding scale through skip-connections. The output of the last refinement block is used to reconstruct the object mask through the final convolutional layer followed by a softmax operation. Every convolutional layer in the decoder uses 3×3 filters, producing 256-channel output except for the last one that produces 2-channel output. The decoder estimates the mask in 1/4 scale of the input image.

3.4. Multi-object Segmentation

The description of our framework is based on having one target object in the video. However, recent benchmarks require a method that can deal with multi-objects [28, 38]. To meet this requirement, we extend our framework with a mask merging operation. We run our model for each object independently and compute mask probability maps for all objects. Then, we merge the predicted maps using a soft aggregation operation similar to [24]. In [24], the mask merging is performed only during the testing as a post-processing step. In this work, we coin the operation as a differential network layer and apply it during both the training and the testing. More details are included in the supplementary materials.

3.5. 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.

Pre-training on images. One advantage of our framework is that it does not require long training videos. This is because the method learns the semantic spatio-temporal matching between distant pixels without any assumption on temporal smoothness. This means that we can train our network with only a few frames[1] with object masks. This enables us to simulate training videos using image datasets. Some previous works [26, 24] trained their networks using static images and we take a similar strategy. A synthetic video clip that consists of 3 frames is generated by applying random affine transforms[2] to a static image with different parameters. We leverage the image datasets annotated with object masks (salient object detection – [29, 5], semantic segmentation – [7, 8, 19]) to pre-train our network. By doing so, we can expect our model to be robust against a wide variety of object appearance and categories.

Main training on videos. After the pre-training, we use real video data for the main training. In this step, either Youtube-VOS [38] or DAVIS-2017 [28] is used, depending on the target evaluation benchmark. To make a training sample, we sample 3 temporally ordered frames from a training video. To learn the appearance change over a long time, we randomly skip frames during the sampling. The maximum number of frames to be skipped is gradually increased from 0 to 25 during the training as in curriculum learning [39].

Training with dynamic memory. During the training, the memory is dynamically updated with the network's previous outputs. As the system moves forward frame-by-frame, the computed segmentation output at the previous step is added to the memory for the next frame. The raw network output without thresholding, which is a probability map of being a foreground object, is directly used for the memory embedding to model the uncertainty of the estimation.

Training details. We used randomly cropped 384×384 patches for training. For all experiments, we set the mini-batch size to 4 and disabled all the batch normalization layers. We minimize the cross-entropy loss using Adam optimizer [15] with a fixed learning rate of 1e-5. Pre-training takes about 4 days and main training takes about 3 days using four NVIDIA GeForce 1080 Ti GPUs.

3.6. Inference

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 previous frame with object masks are the most important reference information [26, 24, 40]. The first frame always provides reliable information as it comes with the ground truth mask. The previous frame is similar in appearance to the current frame, thus we can expect accurate pixel matching and mask propagation. 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. N is a hyperparameter that controls the trade-off between speed and accuracy, and we use $N = 5$ unless mentioned otherwise. It is noteworthy that our framework achieves the effect of online learning and online adaptation without additional training. The effect of online model updating is easily accomplished by putting the previous frames into the memory without updating model parameters. Thus, our method runs considerably faster than most of the previous methods while achieving state-of-the-art accuracy.

[1] Minimum 2; one as the memory frame and the other as the query.

[2] We used rotation, sheering, zooming, translation, and cropping.

4. Evaluation

We evaluate our model on Youtube-VOS [37] and DAVIS [27, 28] benchmarks. We prepared two models trained on each benchmarks' training set. For the evaluation on Youtube-VOS, we used 3471 training videos following the official split [37]. For DAVIS, we used 60 videos from the DAVIS-2017 train set. Both DAVIS 2016 and 2017 are evaluated using a single model trained on DAVIS-2017 for a fair comparison with the previous works [24, 40]. In addition, we provide the results for the DAVIS with our model trained with additional training data from Youtube-VOS. Note that we use the network output directly without post-processing to evaluate our method.

We measured region similarity $\mathcal{J}$ and contour accuracy $\mathcal{F}$. For Youtube-VOS, we uploaded our results to the online evaluation server [37]. For DAVIS, we used the official benchmark code [27]. Our code and model will be available online.

4.1. Youtube-VOS

Youtube-VOS [38] is the latest large-scale dataset for the video object segmentation that consists of 4453 videos annotated with multiple objects. The dataset is about 30 times larger than the popular DAVIS benchmark that consists of 120 videos. It also has validation data for the unseen object categories. Thus, it is good for measuring the generalization performance of different algorithms. The validation set consists of 474 videos including 91 object categories. It has separate measures for 65 of seen and 26 of unseen object categories. We compare our method to existing methods that are trained on Youtube-VOS training set by [13, 37]. As shown in Table 1, our method significantly outperforms all other methods in every evaluation metric.

	Overall	Seen		Unseen	
		$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$
OSMN [40]	51.2	60.0	60.1	40.6	44.0
MSK [26]	53.1	59.9	59.5	45.0	47.9
RGMP [24]	53.8	59.5	-	45.2	-
OnAVOS [34]	55.2	60.1	62.7	46.6	51.4
RVOS [32]	56.8	63.6	67.2	45.5	51.0
OSVOS [2]	58.8	59.8	60.5	54.2	60.7
S2S [38]	64.4	71.0	70.0	55.5	61.2
A-GAME [13]	66.1	67.8	-	60.8	-
PreMVOS [20]	66.9	71.4	75.9	56.5	63.7
BoLTVOS [35]	71.1	71.6	-	64.3	-
Ours	**79.4**	**79.7**	**84.2**	**72.8**	**80.9**

Table 1: The quantitative evaluation of multi-object video object segmentation on Youtube-VOS [38] validation set. Results for other methods are directly copied from [37, 13, 32, 35].

	OL	$\mathcal{J}$ Mean	$\mathcal{F}$ Mean	Time
S2S **(+YV)** [38]	✓	79.1	-	$9s$
MSK [26]	✓	79.7	75.4	$12s$
OSVOS [2]	✓	79.8	80.6	$9s$
MaskRNN [11]	✓	80.7	80.9	-
VideoMatch [12]		81.0	-	$0.32s$
FEELVOS **(+YV)** [33]		81.1	82.2	$0.45s$
RGMP [24]		81.5	82.0	$0.13s$
A-GAME **(+YV)** [13]		82.0	82.2	$0.07s$
FAVOS [4]		82.4	79.5	$1.8s$
LSE [6]	✓	82.9	80.3	-
CINN [1]	✓	83.4	85.0	$>30s$
PReMVOS [20]	✓	84.9	88.6	$>30s$
OSVOS^S [21]	✓	85.6	86.4	$4.5s$
OnAVOS [34]	✓	86.1	84.9	$13s$
DyeNet [18]	✓	86.2	-	$2.32s$
Ours		84.8	88.1	$0.16s$
Ours **(+YV)**		**88.7**	**89.9**	$0.16s$

Table 2: The quantitative evaluation on DAVIS-2016 validation set. OL indicates online learning. **(+YV)** indicates the use of Youtube-VOS for training. Methods with $\mathcal{J}$ Mean below 79 are omitted due to the space limit and the complete table is available in the supplementary material.

4.2. DAVIS

Single object (DAVIS-2016). DAVIS-2016 [27] is one of the most popular benchmark datasets for video object segmentation tasks. We use the validation set that contains 20 videos annotated with high-quality masks each for a single target object. We compare our method with state-of-the-art methods in Table 2. In the table, we indicate the use of online learning and provide approximate runtimes of each method. Most of the previous top-performing methods rely on online learning that severely harms the running speed. Our method achieves the best accuracy among all competing methods without online learning, and shows competitive results with the top-performing online learning based methods while running in a fraction of time. Our method trained with additional data from Youtube-VOS outperforms all the methods by a large margin.

	OL	$\mathcal{J}$ Mean	$\mathcal{F}$ Mean
OSMN [40]		52.5	57.1
FAVOS [4]		54.6	61.8
VidMatch [12]		56.5	68.2
OSVOS [2]	✓	56.6	63.9
MaskRNN [11]	✓	60.5	-
OnAVOS [34]	✓	64.5	71.2
OSVOS^S [2]	✓	64.7	71.3
RGMP [24]		64.8	68.6
CINN [1]	✓	67.2	74.2
A-GAME **(+YV)** [13]		67.2	72.7
FEELVOS **(+YV)** [33]		69.1	74.0
DyeNet [18]	✓	*74.1	
PReMVOS [20]	✓	73.9	81.7
Ours		69.2	74.0
Ours **(+YV)**		**79.2**	**84.3**

Table 3: The quantitative evaluation on DAVIS-2017 validation set. OL indicates online learning. **(+YV)** indicates the use of Youtube-VOS for training. *: average of $\mathcal{J}$ Mean and $\mathcal{F}$ Mean.

Multiple objects (DAVIS-2017). DAVIS-2017 [28] is a multi-object extension of DAVIS-2016. The validation set consists of 59 objects in 30 videos. In Table Table 3, we report the results of multi-object video segmentation on the validation set. Again, our method shows the best performance among fast methods without online learning. With additional Youtube-VOS data, our method largely outperforms all the previous state-of-the-art methods including the 2018 DAVIS challenge winner [20]. Our results on the test-dev set is included in the supplementary materials.

The large performance leap by using additional training data indicates that DAVIS is too small to train a generalizable deep network due to over-fitting. It also explains why top performing online learning methods on the DAVIS benchmark do not show good performance on the large-scale Youtube-VOS benchmark. Online learning methods are hardly aided by large training data. Those methods usually require an extensive parameter search (*e.g.* data syn-

Figure 4: The qualitative results on Youtube-VOS and DAVIS. Frames are sampled at important moments (*e.g.* before and after occlusions).

Variants	Youtube-VOS	DAVIS-2017	
	Overall	$\mathcal{J}$	$\mathcal{F}$
Pre-training only	69.1	57.9	62.1
Main-training only	68.2	38.1	47.9
Full training	**79.4**	69.2	74.0
Cross validation	56.3	**78.6**	**83.5**

Table 4: Training data analysis on Youtube-VOS and DAVIS-2017 validation sets. We compare models trained through different training stages (Sec. 3.5). In addition, we report the cross-validation results (*i.e.* evaluating DAVIS using the model trained on Youtube-VOS, and vice versa.).

Memory frame(s)	Youtube -VOS	DAVIS		Time
		2016	2017	
First	68.9	81.4	67.0	0.06
Previous	69.7	83.2	69.6	0.06
First & Previous	78.4	87.8	77.7	0.07
Every 5 frames	**79.4**	**88.7**	**79.2**	0.16

Table 5: Memory management analysis on the validation sets of Youtube-VOS and DAVIS. We compare results obtained by different memory management rules. We report *Overall* and $\mathcal{J}$ Mean scores for Youtube-VOS and DAVIS, respectively. Time is measured on DAVIS-2016.

thesis methods, optimization iterations, learning rate, and post-processing), which is not easy to do for a large-scale benchmark.

4.3. Qualitative Results.

Fig. 4 shows qualitative examples of our results. We choose challenging videos from Youtube-VOS and DAVIS validation sets and sample important frames (*e.g.* before and after occlusions). As can be seen in the figure, our method is robust to occlusions and complex motions. More results will be provided in the supplementary material.

5. Analysis

5.1. Training Data

We trained our model through two training stages: the pre-training on static images [29, 5, 7, 8, 19] and the main training using DAVIS [28] or Youtube-VOS [38]. In Table 4, we compare the performance of our method with different training data. In addition, we provide a cross-dataset validation to measure the generalization performance.

Pre-training only. It is interesting that our pre-train only model outperforms the main-train only model as well as all other methods on YouTube-VOS, without using any real video. However, we get maximum performance by using both training strategies.

Main-training only. Without the pre-training stage, the performance of our model drops by 11.2 in *Overall* score on Youtube-VOS [38]. This indicates that the amount of training video data is still not enough to bring out the potential of our network even though Youtube-VOS [38] provides more than 3000 training videos. In addition, very low performance on DAVIS implies a severe over-fitting issue as the training loss was similar to the complete model

Figure 5: Visualization of our space-time read operation. We first compute the similarity scores in Eq. (2) for the pixels inside the object area of the query image (marked in red), then visualize the normalized soft weights to the memory frames.

Figure 6: Visual comparisons of the results with and without using the intermediate frame memories.

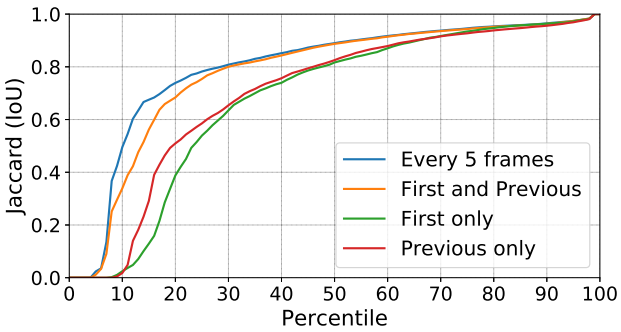

Figure 7: Jaccard score distribution on DAVIS-2017.

(We did not apply early stopping). We conjecture that diverse objects encountered during the pre-training helped our model's generalization and also to prevent over-fitting.

Cross validation. We evaluate our model trained on DAVIS to Youtube-VOS, and vice versa. Our model trained on DAVIS shows limited performance on Youtube-VOS. This is an expected result because DAVIS is too small to learn a generalization ability to other datasets. On the other hand, our model trained on Youtube-VOS performs well on DAVIS and outperforms all other methods.

5.2. Memory Management

For the minimal memory consumption and fastest runtime, we can save either the first and/or the previous frames in the memory. For the maximum accuracy, our final model saves a new intermediate memory frame at every 5 frames in addition to the first and the previous frames as explained in Sec. 3.6.

We compare different memory management rules in Table 5. Saving both the first and the previous frame into the memory is the most important, and our model achieves state-of-the-art accuracy with the two memory frames. This is because our model is strong enough to handle large appearance changes while being robust to drifting and error accumulation by effectively exploiting the memory. On top of that, having the intermediate frame memories further boosts performance by tackling extremely challenging cases as shown in Fig. 6.

For a deeper analysis, we show the frame-level accuracy distribution in Fig. 7. We sort Jaccard scores of all objects in all video frames and plot the scores to analyze the performance on challenging scenes. We compare our final model (*Every 5 frames*) with *First and Previous* to check the effect of using additional memory frames. While both settings perform equally well on the successful range (over 30th percentile), the effect of additional memory frames becomes clearly visible for difficult cases (under 30th percentile). The huge accuracy gap between 10 and 30 percentile indicates that our network handles challenging cases better with additional memory frames. Comparing *First only* and *Previous only*, the previous frame looks more useful to handle failure cases.

Memory visualization. In Fig. 5, we visualize our memory read operation to validate the learned space-time matching. As can be observed in the visualization, our read operation accurately matches corresponding pixels between the query and the memory frames.

6. Conclusion

We have presented a novel space-time memory network for the semi-supervised video object segmentation. Our method performs the best among the existing methods in terms of both the accuracy and the speed. We believe the proposed space-time memory network has a great potential to become breakthroughs in many other pixel-level estimation problems. We are looking for other applications as future work that are suited for our framework including object tracking, interactive image/video segmentation, and inpainting.

Acknowledgment. This work is supported by the ICT R&D program of MSIT/IITP (2017-0-01772, Development of QA systems for Video Story Understanding to pass the Video Turing Test).

References

[1] Linchao Bao, Baoyuan Wu, and Wei Liu. Cnn in mrf: Video object segmentation via inference in a cnn-based higher-order spatio-temporal mrf. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 6

[2] Sergi Caelles, Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. One-shot video object segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017. 1, 2, 6

[3] Yuhua Chen, Jordi Pont-Tuset, Alberto Montes, and Luc Van Gool. Blazingly fast video object segmentation with pixel-wise metric learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[4] Jingchun Cheng, Yi-Hsuan Tsai, Wei-Chih Hung, Shengjin Wang, and Ming-Hsuan Yang. Fast and accurate online video object segmentation via tracking parts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 6

[5] Ming-Ming Cheng, Niloy J Mitra, Xiaolei Huang, Philip HS Torr, and Shi-Min Hu. Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):569–582, 2015. 5, 7

[6] Hai Ci, Chunyu Wang, and Yizhou Wang. Video object segmentation by learning location-sensitive embeddings. In *European Conference on Computer Vision (ECCV)*, 2018. 6

[7] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. 5, 7

[8] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *IEEE International Conference on Computer Vision (ICCV)*, pages 991–998. IEEE, 2011. 5, 7

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 4

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision (ECCV)*, pages 630–645. Springer, 2016. 4

[11] Yuan-Ting Hu, Jia-Bin Huang, and Alexander Schwing. Maskrnn: Instance level video object segmentation. In *Advances in Neural Information Processing Systems*, 2017. 1, 2, 6

[12] Yuan-Ting Hu, Jia-Bin Huang, and Alexander Schwing. Videomatch: Matching based video object segmentation. In *European Conference on Computer Vision (ECCV)*, 2018. 1, 2, 6

[13] Joakim Johnander, Martin Danelljan, Emil Brissman, Fahad Shahbaz Khan, and Michael Felsberg. A generative appearance model for end-to-end video object segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 5, 6

[14] Anna Khoreva, Rodrigo Benenson, Eddy Ilg, Thomas Brox, and Bernt Schiele. Lucid data dreaming for object tracking. *arXiv preprint arXiv:1703.09554*, 2017. 2

[15] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 5

[16] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. In *International Conference on Machine Learning*, pages 1378–1387, 2016. 2, 4

[17] Sangho Lee, Jinyoung Sung, Youngjae Yu, and Gunhee Kim. A memory network approach for story-based temporal summarization of 360 videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1410–1419, 2018. 2

[18] Xiaoxiao Li and Chen Change Loy. Video object segmentation with joint re-identification and attention-aware mask propagation. In *European Conference on Computer Vision (ECCV)*, 2018. 2, 6

[19] 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 (ECCV)*, pages 740–755. Springer, 2014. 5, 7

[20] Jonathon Luiten, Paul Voigtlaender, and Bastian Leibe. Premvos: Proposal-generation, refinement and merging for video object segmentation. In *Asian Conference on Computer Vision*, 2018. 6

[21] Kevis-Kokitsi Maninis, Sergi Caelles, Yuhua Chen, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. Video object segmentation without temporal information. *arXiv preprint arXiv:1709.06031*, 2017. 2, 6

[22] Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-value memory networks for directly reading documents. *arXiv preprint arXiv:1606.03126*, 2016. 2, 4

[23] Seil Na, Sangho Lee, Jisung Kim, and Gunhee Kim. A read-write memory network for movie story understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 677–685, 2017. 2

[24] Seoung Wug Oh, Joon-Young Lee, Kalyan Sunkavalli, and Seon Joo Kim. Fast video object segmentation by reference-guided mask propagation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 4, 5, 6

[25] Cesc Chunseong Park, Byeongchang Kim, and Gunhee Kim. Attend to you: Personalized image captioning with context sequence memory networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6432–6440. IEEE, 2017. 2

[26] Federico Perazzi, Anna Khoreva, Rodrigo Benenson, Bernt Schiele, and Alexander Sorkine-Hornung. Learning video object segmentation from static images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 5, 6

[27] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung.

A benchmark dataset and evaluation methodology for video object segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5, 6

[28] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017. 4, 5, 6, 7

[29] Jianping Shi, Qiong Yan, Li Xu, and Jiaya Jia. Hierarchical image saliency detection on extended cssd. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(4):717–729, 2016. 5, 7

[30] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448, 2015. 2, 4

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017. 4

[32] Carles Ventura, Miriam Bellver, Andreu Girbau, Amaia Salvador, Ferran Marques, and Xavier Giro-i Nieto. Rvos: End-to-end recurrent network for video object segmentation. In *CVPR*, 2019. 2, 6

[33] 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 *CVPR*, 2019. 2, 6

[34] Paul Voigtlaender and Bastian Leibe. Online adaptation of convolutional neural networks for video object segmentation. In *British Machine Vision Conference*, 2017. 2, 6

[35] Paul Voigtlaender, Jonathon Luiten, and Bastian Leibe. Boltvos: Box-level tracking for video object segmentation. *arXiv preprint arXiv:1904.04552*, 2019. 6

[36] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 4

[37] Ning Xu, Linjie Yang, Dingcheng Yue, Jianchao Yang, Yuchen Fan, Yuchen Liang, and Thomas Huang. Youtube-vos: A large-scale video object segmentation benchmark. In *arXiv preprint arXiv:1809.03327*, 2018. 5, 6

[38] Ning Xu, Linjie Yang, Dingcheng Yue, Jianchao Yang, Brian Price, Jimei Yang, Scott Cohen, Yuchen Fan, Yuchen Liang, and Thomas Huang. Youtube-vos: Sequence-to-sequence video object segmentation. In *European Conference on Computer Vision (ECCV)*, 2018. 2, 4, 5, 6, 7

[39] Jimei Yang, Scott E Reed, Ming-Hsuan Yang, and Honglak Lee. Weakly-supervised disentangling with recurrent transformations for 3d view synthesis. In *Advances in Neural Information Processing Systems*, pages 1099–1107, 2015. 5

[40] Linjie Yang, Yanran Wang, Xuehan Xiong, Jianchao Yang, and Aggelos K Katsaggelos. Efficient video object segmentation via network modulation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 5, 6

[41] Tianyu Yang and Antoni B Chan. Learning dynamic memory networks for object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 152–167, 2018. 2

[42] Jae Shin Yoon, Francois Rameau, Junsik Kim, Seokju Lee, Seunghak Shin, and In So Kweon. Pixel-level matching for video object segmentation using convolutional neural networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 1, 2