# Targeted Test Time Adaptation of Memory Networks for Video Object Segmentation

Isidore Dubuisson[1], Damien Muselet[1], Christophe Ducottet[1] and Jochen Lang[2]
[1]*Université Jean Monnet Saint-Etienne, CNRS, Institut d Optique Graduate School, Laboratoire Hubert Curien UMR 5516, F-42023, Saint-Etienne, France*
[2]*School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, Canada*

Keywords: Video Segmentation, Test-Time Adaptation, Feature Matching Based, Localized Finetuning.

Abstract: Semi Automatic Video Object Segmentation (SVOS) aims to segment few objects in a video based on the annotation of these particular objects in the first frame only. State-of-the-art methods rely on offline training on a large dataset that may lack specific samples and details directly applicable to the current test video. Common solutions are to use test-time adaptation to finetune the offline model with the single annotated frame or by relying on complex semi-supervised strategies. In this paper, we introduce targeted test-time adaptation of memory-based SVOS providing the benefits of finetuning with much smaller learning effort. Our method targets specific parts of the model to ensure improved results while maintaining robustness of the offline training. We find that targeting the bottleneck features and the masks that are saved in memory provide substantial benefits. The evaluation of our method shows a significant improvement for video segmentation on DAVIS16 and DAVIS17 datasets.

## 1 INTRODUCTION

Semi Automatic Video Object Segmentation (SVOS), also referred to as One Shot Video Object Segmentation, is the task of segmenting an object, or a set of objects, in a video that are indicated by first frame annotations (mask, bounding boxes, scribbles). This task can be interpreted as propagating the annotation throughout the following frames of the video. There exist multiple approaches to address the SVOS task but we focus on state-of-the-art memory-based methods.

Starting with the first frame and its groundtruth annotation, memory-based methods periodically encode the features of previous frames along with information about their label (object vs. background) into the so-called memory. The segmentation of the current frame is then performed by first enriching features with label information extracted from the memory by a cross-attention mechanism. The enriched features are then passed through the decoder to predict the segmentation mask.

Memory-based models are pretrained on large scale datasets such as COCO or YT-VOS and finetuned on the target dataset (e.g. DAVIS17). Most methods assume that the training dataset is large enough to contain every video feature that could be evaluated during inference. Since SVOS is a class-agnostic task, the training should allow the model to segment images in any context. However, a limited dataset (even a very large one) cannot explicitly contain every possible situation.

We consider that the decision boundaries induced by the training pipeline have been fitted considering the large training dataset which trains the model with a wide range of features but the features are also sparse for any specific video. Because of the wide and sparse training, the decision boundary might not fit perfectly to a specific video of a limited subject and very specific information is needed for precise segmentation. To that end we propose to adapt the model to make it fit better to a specific video.

Our method is based on the observation that during inference, the first frame represents the rest of the video better than the videos that belong to the training dataset. Our goal is therefore to use specific information from the first frame in order to adapt the trained model to the specific video during inference. However, such adaptation must be done carefully, as we show in our experiments, because standard and even targeted fine-tuning may not be effective for a video during inference. One has to consider that fine-tuning



Dubuisson, I., Muselet, D., Ducottet, C. and Lang, J.
Targeted Test Time Adaptation of Memory Networks for Video Object Segmentation.
DOI: 10.5220/0013108500003912

In *Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2025) - Volume 3: VISAPP*, pages 30-38
ISBN: 978-989-758-728-3; ISSN: 2184-4321

with the first frame may lead to overfitting, which reduces the effectiveness of the model for subsequent frames. We also investigate how to use the remaining frames during inference without ground truth.

Our contributions are as follows: 1) We propose a very lightweight predecoder consisting of a single $1 \times 1$ layer that is targeted at the bottleneck of a segmentation network which can increase the performance of a memory-based SVOS model while requiring only minimal on-line training. We demonstrate that the predecoder in our targeted adaption outperforms even full fine-tuning of the decoder in the SVOS model but requires far less effort. Our targeted predecoder focuses the widely trained model on the specific context of the video given by the first frame and its annotation. 2) We demonstrate that our targeted test-time adaptation is compatible with on-line self-supervised improvement of features stored in the memory of the model. Our memory mask adaptation improves key frame segmentation results before the key frame features are stored in memory. We explain the success of our memory mask adaptation with the increase in the precision of the decision boundary between the segmented objects and their background. Memory mask adaption visually improves the border of the segmentation mask.

# 2 RELATED WORK

Segmentation in video has been extensively studied in the last decade, with the emergence of new tasks, including Semi-Supervised Video Object Segmentation (SVOS), on which we will focus. SVOS requires the first frame to be annotated at test time, but is very flexible in terms of the object of interest, more than many other related tasks ((Gao et al., 2022; Zhou et al., 2022)). Other tasks segment the primary moving object (Liu et al., 2024; Zhou et al., 2021), any object belonging to a predefined semantic class (Yang et al., 2019; Lin et al., 2021), all objects (Cheng et al., 2023; Wang et al., 2023a), or according to a prompt (Ravi et al., 2024; Hu et al., 2024).

## 2.1 Semi-Automatic Video Object Segmentation

Several approaches to address SVOS have been explored. Some methods find the object in the current frame given the first frame annotation in a segmentation approach (Oh et al., 2018; Li et al., 2019), while other methods propagate the annotation from frame to frame using optical flow (Khoreva et al., 2019; Su et al., 2023) or similarity. State-of-the-art methods use a memory of past frames, either by storing key features (Oh et al., 2019; Wang et al., 2021; Cheng and Schwing, 2022; Cheng et al., 2021), or by generating a global representation of the object throughout the video (Chen et al., 2020), to achieve better consistency by considering the possible appearance changes that may occur in the video.

## 2.2 Test-Time Adaptation

In the context of video object segmentation, reducing the domain shift between the large training set and the current test video has been the main goal of many recent approaches (Liu et al., 2024; Colomer et al., 2023; Dubuisson et al., 2023; Bertrand et al., 2024). The most straightforward solution consists in applying classical unsupervised domain adaptation approaches (Colomer et al., 2023; Su et al., 2023) that require to feed the model with both source and target data. These solutions are not applicable when the source data is not available at test time. Some alternatives (Liu et al., 2024; Su et al., 2023) exploit the self-supervision framework to avoid using the source data but either they are based on a heavy adversarial student-teacher training (Su et al., 2023) or require to predict additional features such as depth (Liu et al., 2024).

Many methods fine-tune the whole model at test-time (Caelles et al., 2017; Paul and Leibe, 2017; Yuxi et al., 2020). The labels used for fine-tuning are either only the ground truth mask provided on the first frame (Caelles et al., 2017) or also the most confident predicted masks for the other frames (Paul and Leibe, 2017). In this latter case, the negative examples are selected in the areas that are far from the predicted mask areas in the image space. Obviously, finetuning the whole model at test time is not efficient and meta-learning solutions exist to provide the best initialization state and hyper-parameters for a given test video (Tim and Leal-Taixé, 2020). Other approaches rather propose to concentrate on the bottleneck of the encoder-decoder architecture for finetuning the network (Bhat et al., 2020; Robinson et al., 2020; Liu et al., 2021). Their idea is to insert a light target network before the decoder in order to provide adapted accurate features that help the decoder to reconstruct the segmentation mask. Unfortunately, since this target network modifies the dimension and structure of the features that feed the decoder, this solution does not allow to exploit the default pre-trained decoder and requires to re-train a new specific one. Nevertheless, convinced that the bottleneck is the correct place to fine-tune, we propose to build on this idea by adding a tiny predecoder that can be adapted very

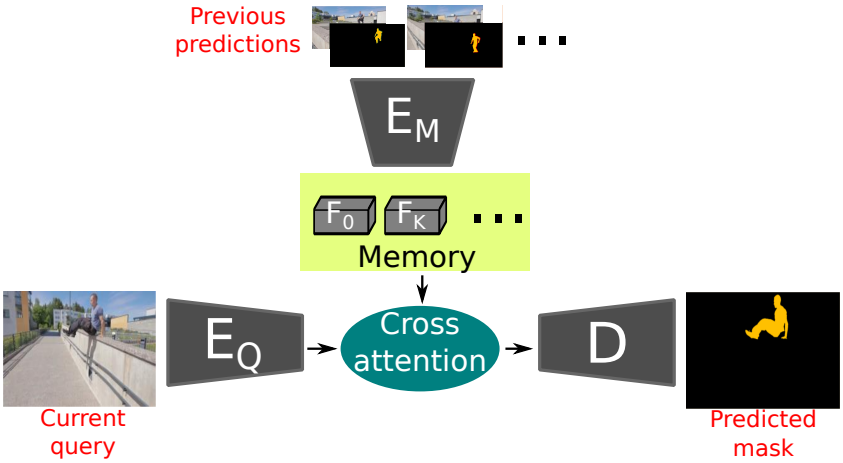


Figure 1: General memory-based pipeline. The query frame is encoded and concatenated with specific memory features thanks to a cross attention module. The memory update frequency is a fixed parameter $K$. Note that the mask information is provided by the previous memory features.

efficiently on the given test video without modifying the structure or dimension of the decoder input. Note that Dubuisson et al. propose to fine-tune the key features used for affinity computation between the current frame and the memory which also takes place at the bottleneck without modifying the decoder input dimensions (Dubuisson et al., 2023).

Since the only available ground truth in the context of SVOS is the mask of the first frame, few approaches propose to resort to time-backward predictions in order to check that the current fine-tuned network is still able to accurately predict the mask of the first frame (Bertrand et al., 2024; Yuxi et al., 2020). This solution prevents temporal drift of the segmentation and we also build on this idea when refining our predicted masks before storing them in the memory.

Finally, the above related works lead us to design our method starting from a strong pre-trained encoder-decoder and consider that:

- the bottleneck is a good place to adapt the features (from the large training set to the test video),
- the adaptation step should not modify the dimension and structure of the decoder inputs (otherwise, the decoder has to be retrained), and
- time-backward prediction is a nice way to control temporal drifts.

In the next section we detail how our proposed solution leverages these insights.

# 3 OUR APPROACH

## 3.1 Method Overview

We focus on memory-based models (Oh et al., 2019) which are currently the best performing SVOS models. These models are built upon a encoder-decoder segmentation pipeline with a memory read module at the bottleneck (see Fig. 1). This memory read module is based on a classical cross-attention step with queries, keys and values (Oh et al., 2019). The segmentation pipeline is solving a binary (object vs. background) segmentation task while the memory read module is continuously enriching the features from the query frame with object or background features thanks to this cross attention mechanism.

More precisely, the memory contains some previous frames and their corresponding masks encoded into feature maps ($F_0$, $F_K$, ...). The memory encoder $E_M$ takes as inputs the frames and their predicted masks and has been trained on the generic dataset. The memory is initialized with the encoding of the first frame (frame 0) and its corresponding mask. To segment the following frames, the memory read module first selects some relevant features stored in the memory (among $F_0$, $F_K$, ...) and concatenates them with the current features to enrich the representation. The memory is periodically updated to add segmentation information about selected previously segmented frames (referred to as key frames 0, $K$, $2K$, ..., where $K$ is a fixed hyperparameter).

Note that these key frames are stored with their mask prediction inferred by the model. So obviously, these predictions can be not accurate, although they

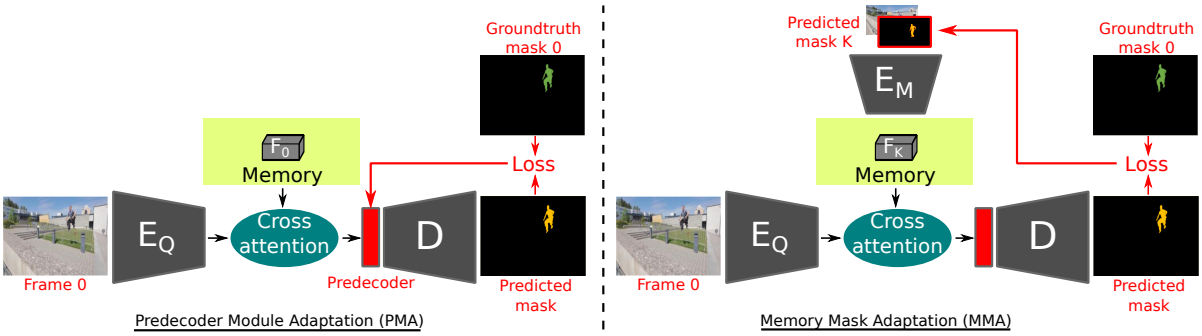


Figure 2: Our architecture for test-time finetuning. Left: We insert a small predecoder that is trained to predict a good mask for the frame 0, allowing to adapt the generic features to the current context. The predecoder is trained once on the frame 0 and frozen for all the other steps. Right: Each time a pair frame/mask is inserted in the memory, we adapt the mask so that the encoded pair promotes a good prediction of the mask of the frame 0, in order to avoid mask drift. The brown arrows represent gradient back-propagation.

are used to predict the next masks. Also note that the memory features transferred to the decoder ($D$) are selected by the cross-attention module and merged to the encoded query features (the query encoder is noted $E_Q$ in Fig. 1).

One of the great advantages of memory-based segmentation models is their ability to be class-agnostic. Thus they can segment the correct object across frames without having learned any representation of this object during the training of the model. The object representation is built online during inference by storing segmentation information from the first frame and from some selected previous frames into the memory. However, memory-based models are subject to two main problems: imprecise decision boundaries of the decoder and lack of consistency of the memory. Although the decoder has learned generic visual features from a large and diverse dataset, it is not perfectly adapted to focus on the current context of the video, leading to segmentation errors. The lack of consistency of the memory is due to the fact that new information added to the memory may contradict initial information and the latter are progressively masked by recent additions.

We propose to add two online adaptations modules to overcome these problems:

1. **Predecoder Module** – a lightweight domain adaptation layer to slightly adjust the features output from the cross-attention module before transferring it into the decoder.
2. **Memory Mask Adaptation** – a back-propagation-based mask correction to enforce the consistency of information stored in the memory.

## 3.2 Predecoder Module Adaptation (PMA)

The common approach to adapt a model is to finetune the prediction head, i.e. the last layers of the network. The reason of this selection is that the last layers contain most of the semantic information. In the current SVOS architecture which is an encoder-decoder model, the semantic information is more likely to be present at the bottleneck, just before the decoder. Indeed, the features that feed the decoder (the output of the cross-attention module) contain the encoded query and memory features. It's worth remembering that only the memory features provide binary labels (background vs. object) for the current frame. These semantic features are fed to the decoder that reconstruct the mask prediction. Since the decoder has been pretrained on a large dataset, it appears promising to adapt these input features to the current context so that the decoder can accurately predict the current mask. The intuition is that the current decoder has been trained to reconstruct a wide diversity of features and we want to force it to concentrate on a small area of the embedding space, where the current video features are located. Thus, we propose to transform the input features with a simple $1 \times 1$ convolution layer. The benefit of choosing such a simple transform is to prevent overfitting on the first frame, since we want to avoid the decoder to forget interesting generic features. This layer is trained with a cross-entropy loss on the first frame for which we have the groundtruth mask (see Fig. 2, left). During training, all the layers are frozen except the predecoder.

Our predecoder module gives more attention to the channels that produce better results on the first frame. Thus, the boundary decision between object and background segmentation is adjusted with respect to the first frame. The following frames with close

context to the first frame benefit from this adaptation.

Note that the predecoder is trained only on the first frame (frame 0) and is frozen for the remainder. We consider that the first frame is providing enough information about the context of the video and the features selected by the predecoder during this short training are appropriate for the next frames. Nevertheless, to avoid overfitting we design a very light predecoder (one single layer) in order to preserve most of the generic features provided by the pretrained encoder model.

### 3.3 Memory Mask Adaptation (MMA)

As explained above, the memory masks given to the memory encoder are the only information that define the specific object to segment in the video. It can be seen as an instruction to the decoder. For the first mask in memory, the features rely on the first frame annotation which is exact. However, as the distance from the first frame in the video increases, the instruction become increasingly obsolete. For this reason, memory-based models update the memory with new masks predicted by the model itself. However, there is a risk that the previous predictions will be incorrect and cause the new predictions to drift, leading to an increase in error. Because of the prediction saved in memory, once an element is considered as a target object, the model tends to segment it as an object until the end of the process. To enhance the instructions stored in memory, we propose to use test-time adaptation to learn the logits, non-normalized predictions, that we use as input to the memory encoder. In order to exploit the single trustfully labels we have (i.e. those of the first frame 0), we use the segmentation process backwards in time, and fine-tune the current mask so that the decoder is able to predict the mask of the first frame reliably. As illustrated in Fig. 2 (right), given the current predicted mask and its respective key frame to be stored in memory, we predict the first frame mask. Since we have the ground truth for this first frame mask, we can compute the cross-entropy loss with the prediction and back-propagate the gradient to fine-tune the current key mask (see Fig. 2, right) in order to improve the prediction. During this adaptation, every layer is frozen except the memory mask that becomes a learnable parameter. This enables us to update the memory mask according to the error, or uncertainty, that could occur if this mask is used later as memory. We apply this adaptation at every memory update (every $K$ frame).

During this adaptation, temporal distance may cause the object to appear differently in the current than in the first frame. In that case, the memory module won't provide good features definition to reconstruct the object in the current frame. To ensure that the adaptation can learn properly, we propose to batch the query with previous memory frames. The memory mask to adapt will be optimized based on sparse subset of the past frames. By learning the memory mask considering a set of frames, we make the mask more temporally consistent and do not limit the mask to temporally close frames only.

Note that the memory read module is responsible of the selection of the memory values for segmentation. Only the most similar patches will be used and thus adapted. The backpropagation does not affect new appearances that haven't been seen before, and the model remains flexible with respect to appearance changes.

## 4 EXPERIMENTS AND RESULTS

### 4.1 Model Baseline and Implementation Details

For the evaluation of our adaptations, we used STM (Oh et al., 2019) and STCN (Cheng et al., 2021) as two salient representatives of the memory based networks. We used the authors' respective model weights that had been obtained by pretraining on COCO (Lin et al., 2014) and finetuning on YOUTUBE-VOS (Xu et al., 2018) and DAVIS17 (Pont-Tuset et al., 2017). We also keep the memory update frequency suggested by the authors, every 8 and 5 frames respectively for STM and STCN. Additionally, only STM uses the previous frame in memory to predict the actual frame.

During inference every layer is frozen. Only parameters of interest are unfrozen during their respective adaptation. The learning of the predecoder is performed only at every first frame initialization, starting with the identity matrix. Once the adaptation is done, the segmentation process runs as the authors designed it to. For this first frame adaptation we use the SGD optimizer with a learning rate of 0.1 over 100 iterations.

In case of STCN, keys are shared between memory and query. In our predecoder adaptation configuration, keys from query and memory are exactly the same. Thus, the memory mask will be used directly in the decoder without complex combination of encoder and memory keys, which means the adaptation will just have to reconstruct the mask encoded in the memory value without considering the query. To add more robustness to the predecoder adaptation, we apply a random shift augmentation between 0 and 8 hor-

Table 1: Evaluation on DAVIS17 validation in single object configuration. The first line provides the results for the pretrained model without adaptation. PMA and MMA respectively refers to Predecoder Module Adaptation (Section 3.2) and Memory Mask Adaptation (Section 3.3).

| | | STM (Oh et al., 2019) | | | STCN (Cheng et al., 2021) | | |
|---|---|---|---|---|---|---|---|
| PMA | MMA | *J&F* | *J* | *F* | *J&F* | *J* | *F* |
| | | 78.97 | 76.30 | 81.64 | 83.60 | 80.50 | 86.70 |
| ✓ | | 80.46 | 78.03 | 82.90 | 83.73 | **80.80** | 86.66 |
| | ✓ | 79.15 | 76.48 | 81.83 | **83.77** | 80.43 | **87.12** |
| ✓ | ✓ | **80.52** | **78.08** | **82.95** | 83.68 | 80.44 | 86.93 |

Table 2: Study of different finetuning configurations with different number of iterations. Bold values represent the best results for each configuration. The used metric is *J&F* of DAVIS17 with each object separately segmented.

| iteration | full decoder | first layer | last layer | predecoder |
|---|---|---|---|---|
| no adaptation | 78.97 | 78.97 | 78.97 | 78.97 |
| 20 | **79.96** | 78.5 | 78.5 | 79.05 |
| 100 | 79.02 | **79.26** | **79.27** | 79.05 |
| 500 | 76.84 | **79.26** | 77.29 | 79.43 |
| 2000 | 76.29 | 78.2 | 78.25 | 79.63 |
| 10000 | 64.07 | 74.39 | 73.07 | **80.56** |
| 15000 | 59.92 | 74.12 | 71.33 | **80.56** |
| 20000 | 35.2 | 69.75 | 69.73 | 80.36 |

izontally and vertically. This allows to shift the patch from memory and query without interfering with the video context.

In contrast, the memory mask adaptation is performed every time a new frame is added to the memory. The adapted mask is used to encode the memory features but it is not used as prediction output of the current frame. Memory mask adaptation uses the cross-entropy loss with SGD optimizer and a learning rate of 10 over 100 iterations.

## 4.2 Results

The metric used is *J&F*, which is the average of Intersection over Union (*J*) with Boundary accuracy (*F*) as for the baseline methods. We evaluate the performance when a single object is selected by the user. In case of multiple objects in the video, we evaluate each object separately. This configuration is especially challenging when the scene contains many similar objects.

Evaluation of our both online adaptations are reported in Table 1. We compare the baseline methods with the different configuration of adaptation. Because they are active at different timestamp in the videos, we can enable them on the same video. Our predecoder performs improvement of $+1.49\%$ and $+0.13\%$, respectively on STM and STCN. As explained by the authors (Cheng et al., 2021), STCN allows more memory candidates to enrich the query frame with the memory read module, thus, the memory combination is more complex and this diversity improves the decoder's robustness on prediction. We believe this explains why our predecoder adaptation works better on STM than on STCN. The Memory mask adaptation reaches $+0.17\%$ improvement on both baselines, even though it is performed every 8 frames on STM, and every 5 frames on STCN. Combination of both adaptations results in slight improvements for both baselines. It is the most effective configuration for STM, while for STCN the adaptation used independently are slightly more effective. In future work, we would like to implement ways to handle long videos with possible additional updates of the predecoder or different management of batch training on memory mask adaptation.

## 4.3 Ablation Study

The general approach to adapt a model is to finetune the whole prediction head or the final layer. In case of memory-based SVOS methods, the prediction head is the whole decoder and the final layer is a 2D convolution with a kernel of $3 \times 3$. To evaluate our predecoder, we compared it against the finetuning of the decoder and of the final layer on the first frame of the video, like the regular approach (see Table 2). To argue the need of an additional predecoder module ($1 \times 1$ - 2D convolution), we additionally study the finetuning of the layer located at the same place, which is the first layer of the decoder ($3 \times 3$ - 2D convolution). We finetune the STM model by using the first frame as both actual and memory frame and apply the cross-entropy loss with the SGD optimizer and

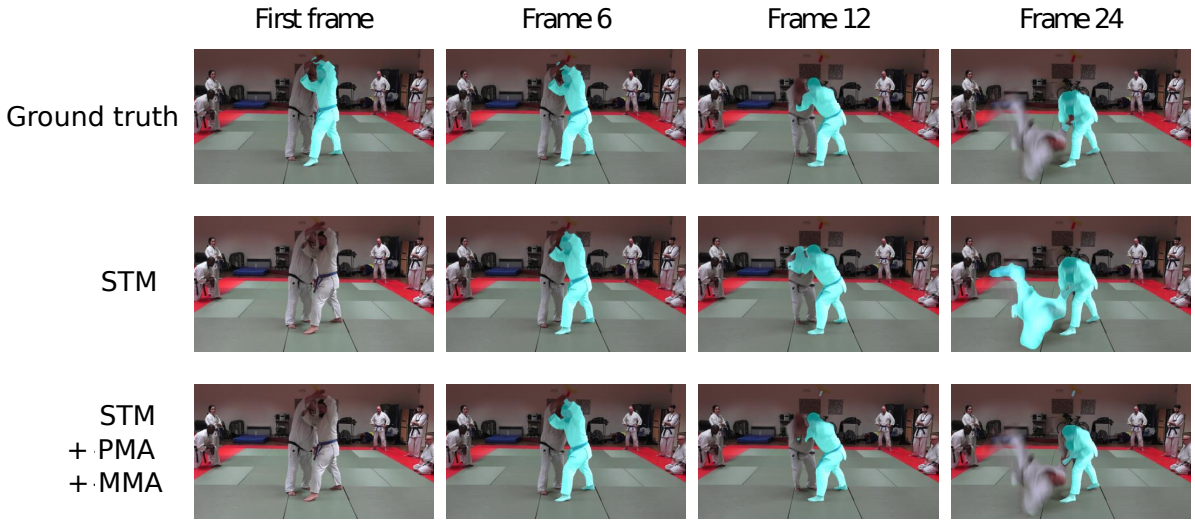


Figure 3: Qualitative result of Decoder adaptation module and Memory Mask Adaptation on STM.

a learning rate of $1e-3$ (see Fig.2, left).

As shown in Table 2, every finetuning configuration slightly improves the overall result. However, finetuning the first or the last layer of the decoder improves the result by only 0.29% and 0.3% respectively and starts to overfit at 100 iterations. Finetuning the full decoder leads to a 0.99% improvement but starts overfitting earlier at 20 iterations. Our predecoder module outperforms finetuning with 1.59% improvement and starts overfitting after 10000 iterations. In our experiment, the worst and best results, corresponding to respectively first layer and predecoder, are located at the same place in the decoder but have different behaviors. We do nbot use an activation layer between the predecode and decoder and hence they could fuse into one layer. However, the predecoder only adjusts the feature channels while the first layer has in addition a small receptive field. This behavior tends to demonstrate that our predecoder adaptation module is more related to a domain alignment than a regular finetuning problem. In order to make a trade off between quality and time efficiency we selected a learning rate of 0.1 for 100 iterations to reach similar result of 80.46% on STM in single object configuration.

### 4.4 Comparison to State of the Art

To compare our result with state of the art SVOS methods, we use the DAVIS16 dataset (Perazzi et al., 2016). Our methods have been implemented only on single object configuration, so we have to compare on a dataset that satisfies that constraint. We referenced best methods that are memory-based or use test-time adaptation to compare with.

As seen from Table 3, our two adaptations (PMA and MMA) on top of both STM and STCN provide very good results which are competitive with the transformer-based state-of-the-art.

## 5 CONCLUSION

In this paper, we have proposed a solution to efficiently finetune a pretrained model in the context of semi-supervised video object segmentation. First, we analyzed a generic representation of the memory-based architectures which outperforms the alternatives architectures for this task. Then, with a precise analysis of each block, we have identified two locations of the pipeline that should be adapted to the given context provided by the test video. The first one is the input of the decoder. The features at this bottleneck represent the most semantic information about the input frame and should be adapted to the current context before being decoded. Fortunately, we have noted that a very light module can adapt these features without risking any strong overfitting on the first frame. This is due to the appropriate position of this module in the architecture. The second crucial element that should be finetuned is the data that is stored in the memory. Instead of directly storing the consecutive mask predictions as it is usually done in this context, we propose to refine each mask with a time-backward prediction on the first frame. This allows to check that the data stored in memory are consistent with the past and avoids temporal drift. Experiments on different datasets and different memory-based architectures show that this approach improves the results over the baselines. Tuning the hyper-parameters

Table 3: Results on Single-Object Segmentation in DAVIS16. 'TTA' is for Test-Time Adaptation and 'MB' for Memory-Based networks. The results are extracted from the original papers.

| name | *tta* | *MB* | *J&F* | *J* | *F* |
|---|---|---|---|---|---|
| RGMP(Oh et al., 2018) | × | × | 81.8 | 81.5 | 82.0 |
| SAT(Chen et al., 2020) | × | × | 83.1 | 82.6 | 83.6 |
| OnAVOS(Paul and Leibe, 2017) | ✓ | × | 85.5 | 86.1 | 84.9 |
| OSVOS(Maninis et al., 2018) | ✓ | × | 86.0 | 85.6 | 86.4 |
| e-OSVOS(Tim and Leal-Taixé, 2020) | ✓ | × | 86.8 | 86.6 | 87.0 |
| LucidTracker(Khoreva et al., 2019) | ✓ | × | 85.7 | 86.6 | 84.8 |
| SWIFTNET(Wang et al., 2021) | × | ✓ | 90.4 | 90.5 | 90.3 |
| XMem(Cheng and Schwing, 2022) | × | ✓ | 92.0 | 90.7 | 93.2 |
| ISVOS(Wang et al., 2023b) | × | ✓ | 92.8 | 91.8 | 93.8 |
| SwinB-DeAOT-L(Yang and Yang, 2022) | × | ✓ | 92.9 | 91.1 | 94.7 |
| FRTM(Robinson et al., 2020) | ✓ | ✓ | 83.5 | – | – |
| FAMINet(Liu et al., 2021) | ✓ | ✓ | 82.9 | 82.4 | 83.4 |
| STM(Oh et al., 2019) | × | ✓ | 88.9 | 88.9 | 88.9 |
| STM w/ PMA | ✓ | ✓ | 89.5 | 89.3 | 89.7 |
| STM w/ MMA | ✓ | ✓ | 89.1 | 89.0 | 89.1 |
| STCN(Cheng et al., 2021) | × | ✓ | 91.7 | 90.4 | 93.0 |
| STCN w/ PMA | ✓ | ✓ | 92.4 | 91.2 | 93.5 |
| STCN w/ MMA | ✓ | ✓ | 92.6 | 91.2 | 94.0 |

(learning rate, iteration numbers) used for the finetuning step is not straightforward and we think that our solution could benefit from recent advances in meta-learning. This will be our future work.

# REFERENCES


Bertrand, J., Kordopatis Zilos, G., Kalantidis, Y., and Tolias, G. (2024). Test-time training for matching-based video object segmentation. *Advances in Neural Information Processing Systems*, 36.

Bhat, G., Lawin, F. J., Danelljan, M., Robinson, A., Felsberg, M., Van Gool, L., and Timofte, R. (2020). Learning what to learn for video object segmentation. In *Proceedings of the CVF European Conference on Computer Vision*.

Caelles, S., Maninis, K.-K., Pont-Tuset, J., Leal-Taixé, L., Cremers, D., , and Gool, L. V. (2017). One-shot video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Chen, X., Li, Z., Yuan, Y., Yu, G., Shen, J., and Qi, D. (2020). State-aware tracker for real-time video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Cheng, H. K., Oh, S. W., Price, B., Schwing, A., and Lee, J.-Y. (2023). Tracking anything with decoupled video segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1316–1326.

Cheng, H. K. and Schwing, A. G. (2022). Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *Proceedings of the CVF European Conference on Computer Vision*.

Cheng, H. K., Tai, Y.-W., and Tang, C.-K. (2021). Rethinking space-time networks with improved memory coverage for efficient video object segmentation. *Advances in Neural Information Processing Systems*, 34:11781–11794.

Colomer, M. B., Dovesi, P. L., Panagiotakopoulos, T., Carvalho, J. F., Härenstam-Nielsen, L., Azizpour, H., Kjellström, H., Cremers, D., and Poggi, M. (2023). To adapt or not to adapt? real-time adaptation for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16548–16559.

Dubuisson, I., Muselet, D., Ducottet, C., and Lang, J. (2023). Fast context adaptation for video object segmentation. In *International Conference on Computer Analysis of Images and Patterns*, pages 273–283. Springer.

Gao, M., Zheng, F., Yu, J. J., Shan, C., Ding, G., and Han, J. (2022). Deep learning for video object segmentation: A review. *Artificial Intelligence Review*.

Hu, X., Hampiholi, B., Neumann, H., and Lang, J. (2024). Temporal context enhanced referring video object segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 5574–5583.

Khoreva, A., Benenson, R., Ilg, E., Brox, T., and Schiele, B. (2019). Lucid data dreaming for video object segmentation. *International Journal of Computer Vision*, 127(9):1175–1197.

Li, X., Ma, C., Wu, B., He, Z., and Yang, M.-H. (2019). Target-aware deep tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Lin, H., Wu, R., Liu, S., Lu, J., and Jia, J. (2021). Video instance segmentation with a propose-reduce paradigm. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1739–1748.

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., and Dollár, P. (2014). Microsoft coco: Common objects in context. In *Proceedings of the CVF European Conference on Computer Vision*.

Liu, W., Shen, X., Li, H., Bi, X., Liu, B., Pun, C.-M., and Cun, X. (2024). Depth-aware test-time training for zero-shot video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19218–19227.

Liu, Z., Liu, J., Chen, W., Wu, X., and Li, Z. (2021). Faminet: Learning real-time semisupervised video object segmentation with steepest optimized optical flow. *IEEE Transactions on Instrumentation and Measurement*, 71:1–16.

Maninis, K.-K., Caelles, S., Chen, Y., Pont-Tuset, J., Leal-Taixé, L., Cremers, D., and Gool, L. V. (2018). Video object segmentation without temporal information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Oh, S. W., Lee, J.-Y., Sunkavalli, K., and Kim, S. J. (2018). Fast video object segmentation by reference-guided mask propagation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Oh, S. W., Lee, J.-Y., Xu, N., and Kim, S. J. (2019). Video object segmentation using space-time memory networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Paul, V. and Leibe, B. (2017). Online adaptation of convolutional neural networks for video object segmentation. In *Proceedings of the British Machine Vision Conference*.

Perazzi, F., Pont-Tuset, J., McWilliams, B., Van Gool, L., Gross, M., and Sorkine-Hornung, A. (2016). A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*.

Pont-Tuset, J., Perazzi, F., Caelles, S., Arbeláez, P., Sorkine-Hornung, A., and Gool, L. V. (2017). The 2017 davis challenge on video object segmentation. In *arXiv preprint arXiv:1704.00675*.

Ravi, N., Gabeur, V., Hu, Y.-T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., et al. (2024). Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*.

Robinson, A., Lawin, F. J., Danelljan, M., Khan, F. S., and Felsberg, M. (2020). Learning fast and robust target models for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Su, T., Song, H., Liu, D., Liu, B., and Liu, Q. (2023). Unsupervised video object segmentation with online adversarial self-tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 688–698.

Tim, M. and Leal-Taixé, L. (2020). Make one-shot video object segmentation efficient again. In *Proceedings of Advances in Neural Information Processing Systems*.

Wang, H., Jiang, X., Ren, H., Hu, Y., and Bai, S. (2021). Swiftnet: Real-time video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Wang, H., Yan, C., Wang, S., Jiang, X., Tang, X., Hu, Y., Xie, W., and Gavves, E. (2023a). Towards open-vocabulary video instance segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4057–4066.

Wang, J., Chen, D., Wu, Z., Luo, C., Tang, C., Dai, X., Zhao, Y., Xie, Y., Yuan, L., and Jiang, Y.-G. (2023b). Look before you match: Instance understanding matters in video object segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2268–2278.

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

Yang, L., Fan, Y., and Xu, N. (2019). Video instance segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5188–5197.

Yang, Z. and Yang, Y. (2022). Decoupling features in hierarchical propagation for video object segmentation. *Advances in Neural Information Processing Systems*, 35:36324–36336.

Yuxi, L., Ning, X., Jinlong, P., John, S., and Weiyao, L. (2020). Delving into the cyclic mechanism in semi-supervised video object segmentation. In *Proceedings of Advances in Neural Information Processing Systems*.

Zhou, T., Li, J., Li, X., and Shao, L. (2021). Target-aware object discovery and association for unsupervised video multi-object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Zhou, T., Porikli, F., Crandall, D. J., Van Gool, L., and Wang, W. (2022). A survey on deep learning technique for video segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 45(6):7099–7122.