# Learning Dynamic Network Using a Reuse Gate Function in Semi-supervised Video Object Segmentation

**Hyojin Park**[1] **Jayeon Yoo**[1] * **Seohyeong Jeong**[1,3] *† **Ganesh Venkatesh**[2] **Nojun Kwak**[1]
[1]Seoul National University, [2]Facebook Inc., [3]AIRS Company, Hyundai Motor Group

## Abstract

*Current state-of-the-art approaches for Semi-supervised Video Object Segmentation (Semi-VOS) propagates information from previous frames to generate segmentation mask for the current frame. This results in high-quality segmentation across challenging scenarios such as changes in appearance and occlusion. But it also leads to unnecessary computations for stationary or slow-moving objects where the change across frames is minimal. In this work, we exploit this observation by using temporal information to quickly identify frames with minimal change and skip the heavyweight mask generation step. To realize this efficiency, we propose a novel dynamic network that estimates change across frames and decides which path – computing a full network or reusing previous frame's feature – to choose depending on the expected similarity. Experimental results show that our approach significantly improves inference speed without much accuracy degradation on challenging Semi-VOS datasets – DAVIS 16, DAVIS 17, and YouTube-VOS. Furthermore, our approach can be applied to multiple Semi-VOS methods demonstrating its generality. The code is available in https://github.com/HYOJINPARK/Reuse_VOS .*

## 1. Introduction

Semi-VOS tracks an object of interest across all the frames in a video given the ground truth mask of the initial frame. VOS classifies each pixel as belonging to background or a tracked object. This task has wide applicability to many real-world use cases including autonomous driving, surveillance, video editing as well as to the emerging class of augmented reality/mixed reality devices. VOS is a challenging task because it needs to distinguish the target object from other similar objects in the scene even as target's appearance changes over time as well as through occlusions.

A variety of methods have been proposed for solving video object segmentation including online learning [30, 19], mask propagation [17, 3], and template matching [40, 22]. A common theme across most of these previous methods is to use information from previous frames – either just the first frame, some of the previous frames (first and last being a popular option) or all the previous frames – to produce high quality segmentation mask.

In this work, we ask a different question

**Q: Can we use temporal information to identify when the object appearance and position has not changed across frames?**

The motivation for doing so would be to skip much of the expensive computation needed to produce a high-quality mask for the current frame if that mask is almost the same as the mask we computed in the previous frame. Instead, we can produce current frame's mask using a cheap model that just makes minor edits to the previous frame's feature. As

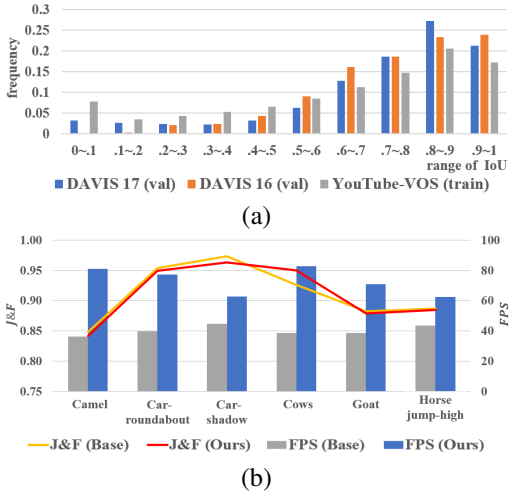



Figure 1. (a) Histogram of a range of IoU between the previous and the current ground truth masks. X-axis is a range of IoU and y-axis denotes frequency corresponding to the range of IoU. (b) FPS and accuracy ($J\&F$) comparison between the baseline model (FRTM [30]) and ours on videos having high IoU between the previous and current ground truth masks. The proposed method preserves the original accuracy while improving the speed a lot.

*Indicate same equal contribution as second authors
†This work was done when Seohyeong Jeong was with SNU

shown in Fig. 1(a), for a significant fraction of the frames in the popular DAVIS and YouTube-VOS dataset, object masks are very similar to their previous frame's masks (73.3% of consecutive frames in DAVIS 17 dataset have IoU greater than 0.7).

We build on the above observation by constructing a cheap temporal matching module to quickly quantify the similarity of the current frame with the previous frame. We use the similarity to gate the computation of high-quality mask for the current frame – if the similarity is high we reuse previous features with minor refinements and avoid the expensive mask generation step. This allows us to avoid majority of computations for the current frame without compromising on accuracy.

Our approach compliments the existing video object segmentation approaches and to demonstrate its generality, we integrate our proposal into multiple prior video object segmentation models – FRTM [30] and TTVOS [25]. To the best of our knowledge, we are the first to propose skipping computation of segmentation masks dynamically based on the object movement. We believe this is a significant contribution that will enable high-quality video object segmentation models to run on mobile devices in real time with minimal battery impact.

We make the following contributions in this paper:

- We make the case for exploiting temporal information to skip mask generation for frames with little or no movement.
- We develop a general framework to skip mask computation consisting of sub-networks to estimate movement across frames, dynamic selection between processing full-network or reusing previous frame's feature for generating mask and a novel loss function to train this dynamic architecture.
- We evaluate our approach on multiple video object segmentation models (FRTM, TTVOS) as well as multiple challenging datasets (DAVIS 16, DAVIS 17, Youtube-VOS) and demonstrate that we can save up to $47.5\%$ computation and speedup FPS by $1.45\times$ with minimal accuracy impact on DAVIS 16 (within around 0.4 $\%$ of baseline).

## 2. Related Work

**Online-learning:** Online-learning algorithms learn to update models from datastreams in sequential manners during the inference stage [31, 47, 13]. In the semi-VOS task, online learning takes place as fine-tuning the segmentation model during the inference stage given the image and the target mask of the first frame to inject the strong appearance of the mask to the model [19, 27, 1]. However, the fine-tuning step causes a significant bottleneck. FRTM [30] tackles this issue by splitting the model into two sub-networks: a light-weight target appearance model trained online and a segmentation network trained offline.

**Mask Propagation:** Mask propagation methods realigns the given segmentation mask or features. Optical flow is widely used to measure the changes in pixel-wise movements of objects in VOS [11, 4, 35, 32]. Segflow [3] designs two branches of image segmentation and optical flow, and bidirectionally combines both information into a unified framework to estimate target masks. Similarly, FAVOS [17] and CRN [7] utilize optical flow information to refine a coarse segmentation mask into an accurate mask.

**Template matching:** Template matching is one of the common approaches in the semi-VOS domain. Models generate a target template and calculate similarity between the template and given inputs. A-GAME [9] employs a mixture of Gaussians to learn the target and background feature distributions. RANet [41] integrates a ranking system to the template matching process and select feature maps according to their rank. FEELVOS [38] calculates a distance map by local and global matching mechanism for better robustness. Furthermore, SiamMask [40] exploits a depth-wise operation to make the matching operation faster. STM [22] and GC [15] integrate the memory network approach [12, 34, 42]. However, this approach requires lots of resources in maintaining the memory. TTVOS [25] proposes a light-weight template matching method for reducing the burden of computation and a temporal consistency loss for endowing a correction power about the incorrectly estimated mask without heavy optical flow.

**Dynamic network:** Dynamic networks perform efficient inference by dynamically choosing a subset of networks depending on the input. Constructing a dynamic inference path by dropping sub-layers of a network using gating modules has been widely studied in image-level tasks [48, 21, 36, 14]. This approach has been applied to image segmentation [45] and has been recently extended to the video domain as well [2, 6, 46, 16, 33]. [2, 46] adopt switching modules to Semi-VOS field, but they still requires the full computation on the feature extraction and the mask refinement for every frame.

We find two problems about applying dynamic network approaches to Semi-VOS field. First, the gate function outputs a discrete decision which are hard to be integrated with a convolutional network due to gradient calculation. Gumbel-Softmax trick [8] is generally used to resolve the problem by the softmax relaxation, but we empirically get unstable training problem. The second issue is how to penalize the model. Usually, there is a target rate for training a dynamic network. The target rate determines the fixed maximum number of layer blocks that can be used in the computation during training. We found that this constraint does not work well in this task. Since the target rate drives a model

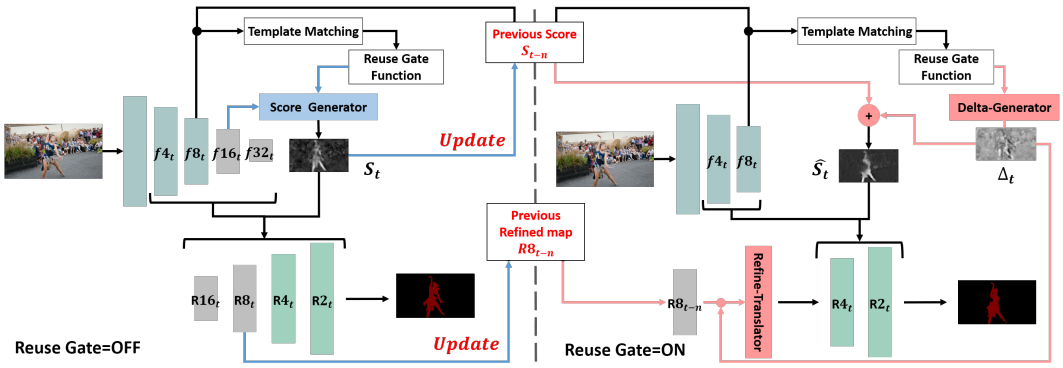



Figure 2. Overall architecture of our dynamic model. Some parts of feature extraction and segmentation network are skipped when the reuse gate is on. The delta-generator produces $\Delta_t$ to convert the previous feature information to that of the current feature. Refine-translator transforms the previous refined feature map into the current one with the help of $\Delta_t$ to make the final mask. When the reuse gate is off, the previous score map is updated to be used in next frames.

to concentrate on meeting the desired number of gates on, the model tends to treat preserving the original accuracy as a less important matter, resulting in a poor accuracy.

## 3. Method

We explain our dynamic architecture that estimates the movement across frames and skip mask generation for the VOS task. In Sec 3.1, we briefly summarize the baseline model, FRTM[30], and introduce our method of converting the baseline model into a dynamic architecture. Note that our proposed dynamic inference architecture can be applied to other VOS frameworks as well. In Sec 3.2, we explain our template matching method for measuring dissimilarity. The reuse gate function takes the dissimilarity information to quantify movement across frames and selects between different paths for generating mask (processing full-network when the reuse gate is off and reusing previous frame's feature when the reuse gate is on). In the case of the reuse gate being on, the model produces a difference map between adjacent frames using the delta-generator. In Sec 3.3, we show how the refine-translator adjusts the previous refined feature map, $R_{t-n}$, to the current refined feature map, $\hat{R}_t$. Finally, in Sec 3.4, we have empirically witnessed that when the common constraint for the gate function [14, 36] is used, the model experiences dramatic performance degradation. To resolve this problem, we introduce a new gating loss, called gating probability loss, that takes the IoU between the current and the previous masks into account as described.

### 3.1. Overall Architecture

**Previous work:** Online learning methods train models to learn target-specific appearance during the inference stage. This enhances the robustness of a model but it still suffers from extensive latency due to the fine-tuning step. FRTM [30] depicted in Fig. 2 (excluding the proposed *template matching* and *gate function* modules) resolves this chronic issue in the online-learning realm by splitting the model into a light-weight score generator and a segmentation network. The light-weight score generator simply consists of two layers for faster optimization during online learning, and it produces a coarse score map of an object. The segmentation network is much more complex than the light-weight module. Taking extracted feature maps as an extra input along with the score map, the network refines the coarse score map and generates high quality masks. The segmentation network is trained offline to reduce the burden of online-learning.

As shown in Fig. 2, a shared feature extractor produces feature maps $fN_t$ from the current frame, where $fN_t$ denotes a feature map at frame $t$ with an $1/N$-sized width and height of the input. $f16_t$ is forwarded to the light-weight score generator to generate a target score map. Then, the segmentation network gradually increases the spatial size of the feature map from $f32_t$ using the score map in a U-Net-like structure. $f32_t$, $f16_t$, $f8_t$ and $f4_t$ are enhanced for generating a more accurate mask by the score map. The final high resolution feature map is converted to a target segmentation mask. The second layer in the score generator is updated every eight frames to handle the changing of the target appearance.

**Our work:** We analyze that a substantial amount of frames in a video are similar to each others and for these redundant frames, the model can reuse previous information instead of full path calculation. Therefore, in our model, not every layer need to be fully-forwarded to extract and refine features. The template matching is applied to measure movement from dissimilarity between current and previous frames. The gate function decides whether to skip calculation or not from an output feature map of the template matching module. Details are described in Sec 3.2.

Our gate function consists of two convolution layers and two max-pooling layers as follows:

$$P_{gate} = \sigma(w_2 * f(w_1 * f(x))), \tag{1}$$

where $w$ and $f$ denotes convolution weights and max-pooling in a layer, respectively and $\sigma$ is a sigmoid function that returns the probability, $P_{gate}$, of the gate being on (reuse). If $P_{gate}$ is higher than a threshold $\tau$[1], which means the current and previous frames are similar enough, the reuse gate is on as follows:

$$g_t = \begin{cases} 1 \text{ (reuse)} & \text{if } P_{gate} \geq \tau \\ 0 \text{ (not reuse)} & \text{otherwise} \end{cases} \tag{2}$$

If the reuse gate is off ($g_t = 0$), the model generates a score map for the current frame following the original FRTM method and the generated score map is stored to be used as a previous score map for subsequent frames. On the other hand, if the reuse gate is on, the model makes a delta map, $\Delta_t$, through the delta-generator. The delta-map contains information on pixel-wise foreground-background conversion from the previous to the current frames. The model adds the delta-map into the previous score to estimate the current score map. Therefore, we can skip the remaining feature extraction process of calculating $f16_t$ and $f32_t$ which need to get score map in baseline. In the segmentation network, we cannot use the original network due to missing $f16_t$ and $f32_t$ as shown in Fig. 2. $RN_t$ denotes a refined feature map at frame $t$ with an $1/N$-sized width and height of the input, and is produced from $fN'_t$ and $S_t$, where $N' = N/2$. We estimate $\hat{R8}_t$ by using the refine-translator The previous refined feature map $R8_{t-n}$, and $\Delta_t$ are passed on to the refine-translator, and refine-translator consists of multiple size receptive fields to estimate $\hat{R8}_t$. Therefore, the model can also skip stages of making $R16_t$ and $R8_t$. Details are described in Sec 3.3.

### 3.2. Template Matching for Quantifying Movement

In order to quantify the movements across frames, we find dissimilarity information that is measured by utilizing a simple module introduced in TTVOS [25]. They proposed a light-weight template matching module, which generates a similarity map, as an output, to focus on the target from the input by comparing with a *template*. The template contains the target appearance. Inspired by this light-weight module, we integrate the template matching procedure into our framework to generate the dissimilarity feature, $D_t$ in Fig. 3, with some modifications.

In our work, we apply the property of template matching to focus on the dissimilarity, as described in Fig 4. To do this, we use current frame feature map, $f8_t$, as current

[1]In the training stage, we set $\tau = 0.5$.

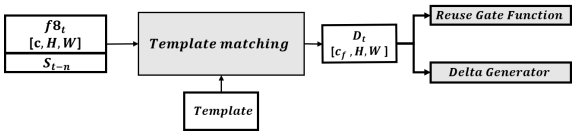
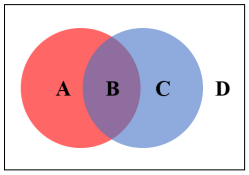



Figure 3. Process of the template matching. The output feature map of the template matching module, $D_t$, focuses on dissimilarities between the current and the previous frames. $D_t$ is forwarded to the gate function and the delta-generator.




Target location in frame $t-n$
Target location in frame $t$

| | Prev. Info. | Curr. Info. | Transition |
|---|---|---|---|
| A | O | X | FG -> BG |
| B | O | O | FG -> FG |
| C | X | O | BG -> FG |
| D | X | X | BG -> BG |

Figure 4. Example of template matching process. 'Prev. info.' is represented by $S_{t-n}$. 'Curr. info.' is represented by $f8_t$. FG and BG denotes foreground and background, respectively. Transition indicates change in class from $t-n$ to $t$. We focus to correctly identifying the dissimilar regions (A and C) between frames.

information, and the previous score map, $S_{t-n}$, which is produced from score generator for the previous information. Then, both feature maps are concatenated together and provided as an input to the template matching module to be compared with the *template*. The template matching module produces dissimilarity feature map, $D_t \in \mathbb{R}^{c_f,H,W}$, which has the same resolution as $f8_t$ with a channel size of $c_f$. The $D_t$ is forwarded to the reuse gate function for deciding whether to skip the computation or not. If the reuse gate is on, the delta-generator makes a delta map $\Delta_t$ from $D_t$, which represents which pixels are changed from background to foreground and vice versa. The delta-generator consists of a single convolution layer, so the computation is not heavy. The loss, $Loss_\Delta$, accomplishes the above mentioned process by reducing the gap between $y'_t - S_{t-n}$ and $\Delta_t$ as follows:

$$Loss_\Delta = L2(\Delta_t, y'_t - S_{t-n}). \tag{3}$$

where $y'_t$ denotes the reduced sized ground truth mask at frame $t$ and $S_{t-n}$ is the previous score map. L2 loss minimizes the pixel-wise difference between $\Delta_t$ and $y'_t - S_{t-n}$.

In the initialization stage, the *template* is generated with the given initial image $I_0$ and the corresponding mask $y_0$. We reduce the resolution of the $y_0$ and consider the downsampled mask as an initial score map $S_0$, i.e. $S_0 = y'_0$. $f8_0$ is produced from $I_0$ and concatenated with $S_0$ to construct the *template* as in TTVOS [25]. Unlike TTVOS where the *template* is updated every frame, our model skips the updating process to increase the speed of inference.

**Applying to general VOS frameworks:** In general VOS frameworks, the proposed method of template matching can

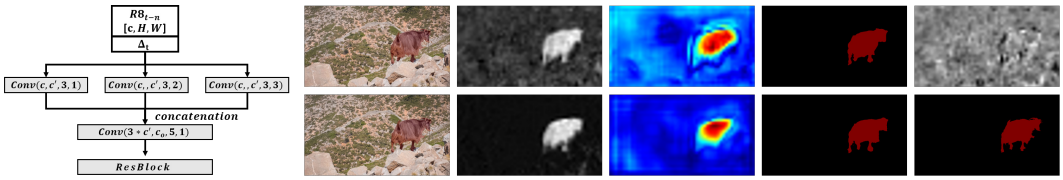



Figure 5. (a) Refine-translator. $Conv(c, c', k, d)$ denotes a $d$ dilated $k \times k$ convolution layer with an input channel of $c$ and an output channel of $c'$. (b)-(e) Example of *goat*. Top row : frame 24, Bottom row : frame 25. (b) Input frames overlapped with ground truth masks. (c) $S_{24}$ and $\hat{S}_{25}$. (d) Activation maps of $R8_{24}$ and $\hat{R}8_{25}$. (e) Estimated masks. (f) Top: $\Delta_{25}$, Bottom: ground truth mask of frame 25.

be applied by concatenating the current feature map and the estimated previous mask.

### 3.3. Estimation for Refined Feature Map

When the reuse gate is on, our model skips layers of a segmentation network. The segmentation network generates the final accurate mask with the coarse score map $S$ along with features from the feature extractor. Using the delta-generator, our model can skip layers of feature extraction step and generate $\hat{S}_t$ by adding $\Delta_t$ to the previous score map $S_{t-n}$. Therefore, our model is unable to use the same segmentation network as the original FRTM. Here, we explain how to estimate $\hat{R}_t$ using $R_{t-n}$ and $\Delta_t$ (See Fig. 2).

When the reuse gate is on, our model starts the segmentation network from $f8_t$. However, the original network increases the resolution starting from $f32_t$, which is a feature map 32 times smaller than the original input image size. Since we input different sized feature map to the network, we design a refine-translator to estimate $\hat{R}8_t$ using $R8_{t-n}$ and $\Delta_t$. Many segmentation networks used multi-sized receptive fields to improve the accuracy [20, 26, 24]. We adopt this method with different dilated ratios. As shown in Fig 5(a), $R8_{t-n}$ and $\Delta_t$ are concatenated and passed on to different dilated convolutions. Each output feature map has to embed features with different receptive fields to cope with various object sizes. Finally, the entire information is merged using a convolution layer and a ResBlock. After the refine-translator, the remaining process is the same as that of the original FRTM. Fig. 5(b)-(e) explains the overall process with two consecutive frames, 24 and 25, in a *goat* video. Fig. 5(d) shows $\hat{R}8_{25}$ in the first row and $R8_{24}$ in the second row. $\Delta_{24}$ is depicted in the first row of (f). The second row of (e) shows the generated mask produced using $\hat{S}_{25}$, $f8_{25}$, $f4_{25}$ and $\hat{R}8_{25}$.

### 3.4. Gate Probability Loss

This section explains how we train our dynamic network. The challenge is twofold: Firstly, we need to train multiple network paths as well as the gating logic (reuse gate). For example, the quantity of target's movement varies depending on consecutive input frames in the training stage. Therefore, the optimal number of selection for reuse gate is diverse for each interaction. We desire that if the quantity is significant, model learns not to select the reuse gate, while if the quantity is trivial, model learns to select the reuse gate. The second challenge is to encourage the network to achieve high segmentation accuracy while choosing the cheaper computation path as often as possible.

To accomplish this, we use the following training recipe – when the model training begins, we train the reuse gate function to predict the IoU of the current frame's ground truth mask with respect to that of the previous frame. Based on $P_{gate}$, as mentioned in Eq. (1) of Sec 3.1, we train the different paths for mask generation. As the training progresses, we want to avoid the situation where the network predicts a low value for $P_{gate}$ to select the more expensive full mask generation path for achieving a higher IoU score. We accomplish this by artificially boosting the IoU between the current and the previous frame's mask introducing a margin which is gradually increased until it reaches $m_1$ from 0. By doing so, we direct the network to select the skip path more often and achieve higher accuracy even with the reduced computation.

To realize the above training schedule, we propose a novel gate probability loss that is based on IoU as follows:

$$\begin{aligned} m &= m_1 * (ep_c / ep_T), \\ P_{target} &= Max(m,\ IoU_t^{t-n}), \end{aligned} \tag{4}$$

$$Loss_{gp} = Max(m_2,\ |P_{gate} - P_{target}|)^2 \tag{5}$$

where $IoU_t^{t-n}$ is IoU between the ground truth mask frames at time $t-n$ and $t$, and $ep_c$ is the current epoch and $ep_T$ is the target epoch at which time $m$ reaches to $m_1$. We set 120 for $ep_T$. A gate probability loss, denoted by $Loss_{gp}$, penalizes the model proportionally to the difference between $P_{gate}$ and $P_{target}$, when the difference is larger than the margin $m_2$. The final loss becomes:

$$Loss = Loss_{gp} + Loss_{\Delta} + BCE(y_t, \hat{y}_t), \tag{6}$$

where $BCE$ is the binary cross entropy loss between the pixel-wise ground truth $y_t$ at frame $t$ and its estimation $\hat{y}_t$.

## 4. Experiment

In this section, we prove the efficacy of the proposed method using the official benchmark code of DAVIS [28, 29] and the official evaluation server of YouTube-VOS 2018 [43]. DAVIS 16 is a single object task that consists of 30 training videos and 20 validation videos, while DAVIS 17 is a multiple object task with 60 training videos and 30 validation videos. The average of mean Jaccard index $J$ and F-measure $F$ are used to report the model accuracy in the DAVIS benchmark. $J$ index measures the overall similarity by comparing estimated masks with ground truth masks and $F$ score focuses on the boundary by delimiting the spatial extent of the mask. YouTube-VOS is the largest VOS dataset that consists of 3,471 training videos and 474 validation videos (in total 4,453 videos). The validation set is split into seen (65 categories) and unseen (26 categories) to evaluate the generalization ability.

We compare our model with other state-of-the-art models by extending it to two VOS models: FRTM and TTVOS. Our ablation study shows that the refine-translator and the gate probability loss are important factors in preserving the original accuracy and in activating gate properly. We measure detailed performance degradation with and without the refine-translator using different values of $\tau$ in Eq. (2). Furthermore, we report performance changes depending on the different setting of margins, $m_1$ and $m_2$, in Eq. (4).

**Implementation Details:** We implement our method with the FRTM official code. FRTM consists of two versions: FRTM and FRTM-fast. FRTM uses ResNet101 and FRTM-fast uses ResNet18 for feature extraction, and different numbers of iterations are used for fine-tuning the score generator. We follow their training scheme with the following modifications to better fit out dynamic architecture training. We change batch size from 16 to 8 and increase the number of sequences from 3 to 6 to train with greater temporal history within the same memory budget. The learning rate is decreased from $1e^{-3}$ to $5e^{-4}$ and we use the total training epoch of 260. Following the setting of AIG [36], we initialize the reuse gates to be on with the probability of 15%.

### 4.1. DAVIS Benchmark Result

We compare our method with other state-of-the-art models, as shown in Tab. 1 and Fig. 6. We report a model used for feature extraction and training datasets for clarification, since each model has a different setting. Furthermore, we also show additional results on TTVOS to claim the generality of our method. Our method improves the inference speed without any significant accuracy degradation. In Tab. 1, a prefix of **G-** denotes the implementation of the proposed method on the baseline models, FRTM and TTVOS. In the slowest case of $\tau = 1$, the model uses every layer of the network, which is equivalent to using the original baseline model. Our model reports slightly higher performance when applied to FRTM with $\tau = 1$ than the original FRTM. We assume that the difference comes from the different number of training sequences used, as in our implementation 6 is used instead of 3. In case of $\tau = 0.7$, the average reuse rate in the model is 0.402 for DAVIS 17 and 0.475 for DAVIS 16 with marginal performance degradation of 1.8 and 0.4. FPS improves from 24.6 to 37.8 and 37.6 to 58.0 on each dataset, which are 1.5 times faster than the slowest case of $\tau = 1$. We also apply our method into TTVOS to prove that our method can be used for other VOS models regardless

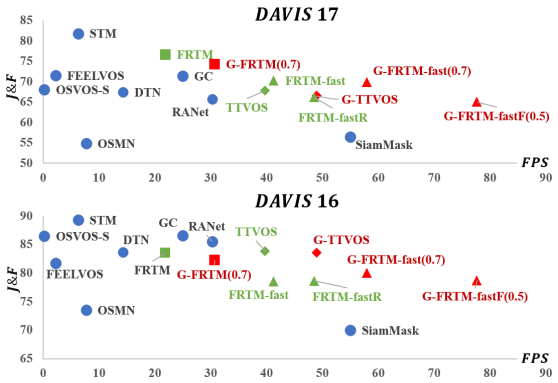



Figure 6. FPS vs $J\&F$ score on the DAVIS validation sets. △, □, and ◇ denotes experiments based on FRTM-fast, FRTM, and TTVOS, respectively. **G-** indicates using the proposed method and fastF denotes experiments based on FRTM-fast with the fusion method and FRTM-fastR is result of reducing a channel size from multiple layers in original FRTM-fast as shown on Tab. 2.

| Method | Feature | Train Dataset | | | DAVIS | | FPS |
|---|---|---|---|---|---|---|---|
| | | Ytb | Seg | Syn | 17 | 16 | |
| OnAVOS [39] | VGG16 | - | o | - | 67.9 | 85.5 | 0.08 |
| OSVOS-S [19] | VGG16 | - | o | - | 68.0 | 86.5 | 0.22 |
| STM [22] | RN50 | o | - | o | 81.8 | 89.3 | 6.25 |
| GC [15] | RN50 | o | - | o | 71.4 | 86.6 | 25.0 |
| OSMN [44] | VGG16 | - | o | - | 54.8 | 73.5 | 7.69 |
| RANet [41] | RN101 | - | - | o | 65.7 | 85.5 | 30.3 |
| A-GAME [9] | RN101 | o | - | o | 70 | 82.1 | 14.3 |
| FEELVOS [38] | XC65 | o | o | - | 71.5 | 81.7 | 2.22 |
| SiamMask [40] | RN50 | o | o | - | 56.4 | 69.8 | 55.0 |
| DTN [46] | RN50 | - | o | - | 67.4 | 83.6 | 14.3 |
| FRTM [30] | RN101 | o | - | - | 76.7 | 83.5 | 21.9 |
| FRTM-fast [30] | RN18 | o | - | - | 70.2 | 78.5 | 41.3 |
| TTVOS [25] | RN50 | o | - | o | 67.8 | 83.8 | 39.6 |
| G-FRTM ($\tau = 1$) | RN101 | o | - | - | 76.4 | 84.3 | 18.2 |
| G-FRTM ($\tau = 0.7$) | RN101 | o | - | - | 74.3 | 82.3 | 28.1 |
| G-FRTM-fast ($\tau = 1$) | RN18 | o | - | - | 71.7 | 80.9 | 37.6 |
| G-FRTM-fast ($\tau = 0.7$) | RN18 | o | - | - | 69.9 | 80.5 | 58.0 |
| G-TTVOS ($\tau = 0.7$) | RN50 | o | - | o | 66.5 | 83.5 | 49.1 |

Table 1. Quantitative comparison on the DAVIS benchmark validation set. Ytb represents using Youtube-VOS for training. Seg is a segmentation dataset for pre-training by Pascal [5] or COCO [18]. Syn is using a saliency dataset for making synthetic video clip by affine transformation. RN, and XC denotes ResNet and Xception for feature extraction, respectively. **G-** indicates using the proposed method based on FRTM and TTVOS. Similar to other works, we measure FPS on DAVIS 16. Note that performances and speed on baseline models are taken from original papers.

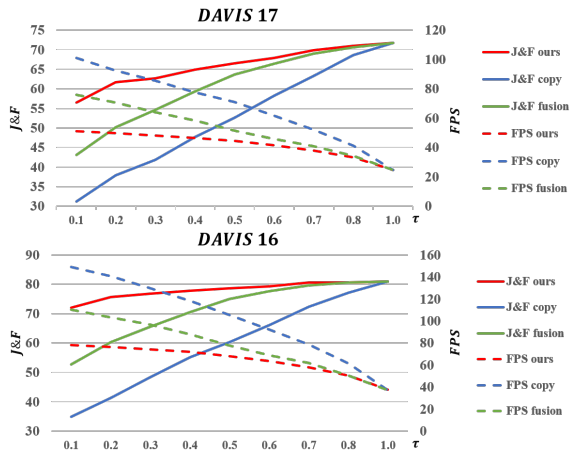



Figure 7. Ablation study about different method for reusing previous information, when the reuse gate is on by comparison of accuracy and FPS on DAVIS with different $\tau$. Ours is our method based on FRTM-fast. **Copy** simply copies the previous mask for the current frame, without using the refine-translator. **Fusion** copies the previous mask if the similarity of consecutive frames is extremely large. Otherwise, original method of the refine-translator is used.

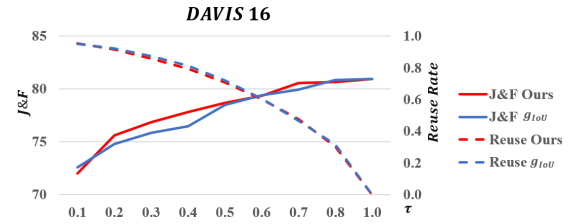



Figure 8. Ablation study about $P_{gate}$by comparison of accuracy and reuse rate on DAVIS 16 with different setting of $\tau$ . Ours estimates the similarity by gate function for deciding gate being on or off. $g_{IoU}$ is using ground truth IoU between adjacent frames as a similarity for deciding gate.

of whether a model produces the score map or not. Successfully, our method can bring improvement of speed over the baseline model, and details of the architecture are explained in the supplementary material. Moreover, our dynamic method is much faster than DTN [46], a dynamic network with switching modules, with better accuracy on DAVIS 17 and comparable accuracy on DAVIS 16.

## 4.2. Ablation Study

In this section, we analyze our modules to show the importance of using 1) the refine-translator, 2) the gate probability loss and 3) the margin in gate probability loss for preserving original accuracy.

Tab. 2 and Fig. 7 demonstrate the effect of the refine-translator. Originally, the refine-translator takes the previous refined feature map and estimates a feature map corresponding to the current stage. To validate the importance

| | $Loss_{gp}$ | D/R | dv17 | dv16 | FPS |
|---|---|---|---|---|---|
| FRTM-fast | x | x | 70.2 | 78.5 | 41.3 |
| FRTM-fast* | x | x | 71.7 | 81.3 | 40.1 |
| FRTM-fastR | x | x | 66.1 | 78.6 | 48.6 |
| $Loss_{Ngate}$ | x | o | 61.3 | 76.5 | 37.8 |
| Ours-copy ($\tau = 0.7$) | o | x | 63.3 | 72.4 | 75.6 |
| Ours-copy ($\tau = 0.5$) | o | x | 52.6 | 60.3 | 100.8 |
| Ours-copy ($\tau = 0.1$) | o | x | 31.2 | 34.8 | 150.2 |
| Ours-fusion ($\tau = 0.7$) | o | △ | 69.0 | 79.6 | 61.8 |
| Ours-fusion ($\tau = 0.5$) | o | △ | 63.7 | 75.0 | 77.7 |
| Ours ($\tau = 1$) | o | o | 71.7 | 80.9 | 37.8 |
| Ours ($\tau = 0.7$) | o | o | 69.6 | 80.5 | 58.0 |
| Ours ($\tau = 0.5$) | o | o | 66.5 | 78.7 | 68.2 |
| Ours ($\tau = 0.1$) | o | o | 56.5 | 72.0 | 78.1 |

Table 2. Ablation study on the proposed modules and the loss function. $Loss_{Ngate}$ means training the gate function using the constraint of the number of gates being on. $D/R$ means using the delta-generator and the refine-translator. △ indicates using one of **copy**, ours, and the full path calculation based on the value of the estimated similarity. FRTM-fast* is our implementation with the same training schemes as our gate function. FRTM-fastR is our implementation of reducing the number of channels in multiple layers to make its inference speed similar to ours.

of the refine-translator, we implement other methods to replace original method with **copy** and **fusion**. **copy** indicates that the model simply copies the previous mask as a result of the current frame when the reuse gate is set on. Therefore, these models do not use the refine-translator. **fusion** is a mixed method between the **copy** and the original method with additional threshold of $\tau_2$ which is greater than $\tau$. When the reuse gate is on and the probability value is greater than $\tau_2$, the model copies the previous mask for the current frame due to extreme similarity. As shown in Fig. 7, models that use **copy** method experience significant performance degradation, while models with our method manage to preserve the original accuracy. In our method, when $\tau = 0.1$, the reuse rate becomes 95.4% on DAVIS 16, with minimal 9% of performance degradation and the inference speed becomes twice faster than when $\tau = 1$. However, in case of the **copy** method, the accuracy decreases by 46%. The **fusion** method takes both advantages from the original and **copy** methods. The performance and speed of **fusion** are drawn in the middle of the our method and the **copy** method. It suggests an adequate alternative to the proposed method when greater increase in inference speed is needed.

Tab. 2 shows the importance of using the gate probability loss function for preserving original accuracy. The experiment of $Loss_{Ngate}$ is the result of using a fixed constraint on the number of gates that can be used in the computation as in [14, 36]. Using $Loss_{Ngate}$ shows huge degradation, even when the module does not reuse any of the previous features. Furthermore, our method with $\tau = 0.7, 0.5$ shows better performance and FPS compared to the case of FRTM-fastR, where channels of multiple layers in FRTM-fast, except a feature extractor, are reduced to meet the

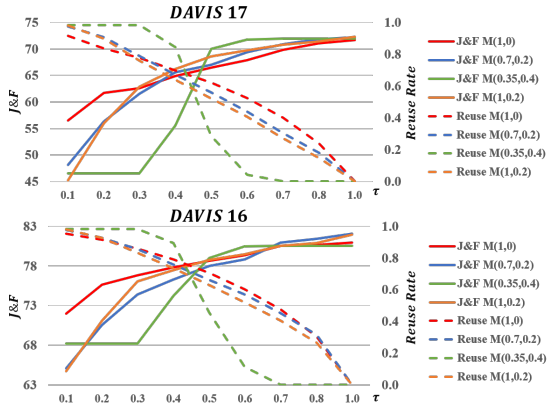



Figure 9. Comparison of accuracy and reuse rate on DAVIS with different settings of margin in the gate probability loss.

similar inference speed as ours. Fig. 8 demonstrates that our gate function is working properly following the ground truth similarity (IoU) between the current and the previous masks. $g_{iou}$ uses ground truth IoU instead of the estimated similarity probability from the gate function to decide whether to turn the reuse gate on or not. Our results on reuse rate and accuracy coincide with when $g_{iou}$ is used.

Finally, Fig. 9 describes the effect of different settings of margin $M(m_1, m_2)$ in the gate probability loss as mentioned in Sec. 3.4. We conduct experiments on various settings, $m_1 = 0.35, 0.7, 1.0$, and $m_2 = 0.0, 0.2, 0.4$. M(1,0) is used in our setting. When the value of $\tau$ is large, the performance of each experiment shows similar trend. However, when the value approaches to 0, which forces the model to reuse the previous information more, the accuracy of $M(1, 0)$ is much higher than others. $M(1, 0.2)$ and $M(0.7, 0.2)$ have similar accuracy and reuse rate. $M(0.35, 0.4)$ works improperly. When $\tau$ changes from 0.6 to 0.5, the model suddenly decides to reuse $80\%$ of the frames. Therefore, we think $m_2$, which is a margin for the gap between $P_{gate}$ and $P_{target}$, is a more important factor for preserving the accuracy.

### 4.3. YouTube-VOS Result

Tab. 3 shows our result on YouTube-VOS dataset. FRTM-fast* is a result of our implementation on baseline models, and we experience performance degradation compared to the original implementation due to difference in a training scheme. In our model, when $\tau = 0.6$ is used, the accuracy difference is $0.6\%$ compared to when $\tau = 1$, and the reuse rate is $25\%$. The reuse rate is lower than the rate in DAVIS datasets for the same $\tau$. We assume this is partially due to the fact that Youtube-VOS dataset contains faster moving objects compared to DAVIS datasets. As shown in Fig. 1, fewer consecutive frames are similar to each other than DAVIS datasets.

| | | Dataset | | $G$ | $J$ | | $F$ | |
|---|---|---|---|---|---|---|---|---|
| Method | Ft | seg | syn | All | S | Us | S | Us |
| onAVOS[39] | VG | o | - | 55.2 | 60.1 | 46.1 | 62.7 | 51.4 |
| OSVOS[30] | VG | o | - | 58.8 | 59.8 | 54.2 | 60.5 | 60.7 |
| S2S [43] | VG | - | - | 64.4 | 71.0 | 55.5 | 70.0 | 61.2 |
| PreMVOS [10] | RN* | o | o | 66.9 | 71.4 | 56.5 | - | - |
| STM [22] | R50 | - | o | 79.4 | 79.7 | 72.8 | 84.2 | 80.9 |
| GC [15] | R50 | - | o | 73.2 | 72.6 | 68.9 | 75.6 | 75.7 |
| A-GAME [9] | R101 | - | o | 66.1 | 67.8 | 60.8 | 69.5 | 66.2 |
| RVOS [37] | R101 | - | - | 56.8 | 63.6 | 45.5 | 67.2 | 51.0 |
| FRTM-fast [30] | R18 | - | - | 65.7 | 68.6 | 58.4 | 71.3 | 64.5 |
| FRTM-fast* | R18 | - | o | 61.9 | 67.0 | 52.6 | 69.5 | 58.6 |
| G-FRTM-fast ($\tau = 1$) | R18 | - | o | 60.9 | 65.1 | 53.0 | 66.7 | 58.8 |
| G-FRTM-fast ($\tau = 0.6$) | R18 | - | o | 60.3 | 64.3 | 53.1 | 65.2 | 58.6 |
| G-FRTM-fast* ($\tau = 0.6$) | R18 | - | o | 62.3 | 66.7 | 55.3 | 67.2 | 60.0 |

Table 3. Quantitative comparison on YouTube-VOS benchmark validation set. Seg is a segmentation dataset for pre-training by Pascal [5] or COCO [18]. Syn is saliency datasets for making synthetic video clips by affine transformation. S and Us are seen and unseen categories. VG is VGG16 and R is ResNet. RN* is a variation of ResNet proposed in [23]. FRTM-fast* is our implementation with the same training schemes as our gate function. **G-** indicates using proposed method based on FRTM-fast. G-FRTM-fast* is result of changed training schemes. The details are described in supplementary material. Note that performances on baseline models are taken from original papers.

## 5. Conclusion

Semi-VOS is a task where models generate a target mask for every single frame of videos given the ground truth mask of the first frame. Previous works on semi-VOS have treated every frame with the same importance, and this incurs redundant computation when the target object is slow-moving. In this paper, we propose a general dynamic network that skips sub-network by quantifying the movement of targets across frames. To do this, we estimate movement by calculating the dissimilarity between consecutive video frames using a template matching module. Then, we train the model to learn when to skip layers of the network using a reuse gate function. We also propose a novel gate probability loss that takes IoU into between the previous and the current ground truth masks into account. This loss forces the model to learn when to turn the reuse gate on, based on how similar the consecutive frames with preserving original accuracy. Our model achieves a boosted inference speed compared to multiple baseline architectures without significant accuracy degradation on standard semi-VOS benchmark datasets. We hope that this work casts a new perspective of applying dynamic inference on not only the semi-VOS task, but also on other video-level tasks.

## ACKNOWLEDGMENTS

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (2021R1A2C3006659).

## References


[1] S. Caelles, K.K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool. One-shot video object segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[2] Xi Chen, Zuoxin Li, Ye Yuan, Gang Yu, Jianxin Shen, and Donglian Qi. State-aware tracker for real-time video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9384–9393, 2020.

[3] Jingchun Cheng, Yi-Hsuan Tsai, Shengjin Wang, and Ming-Hsuan Yang. Segflow: Joint learning for video object segmentation and optical flow. In *Proceedings of the IEEE international conference on computer vision*, pages 686–695, 2017.

[4] Suyog Dutt Jain, Bo Xiong, and Kristen Grauman. Fusionseg: Learning to combine motion and appearance for fully automatic segmentation of generic objects in videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3664–3673, 2017.

[5] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1):98–136, 2015.

[6] Ping Hu, Fabian Caba, Oliver Wang, Zhe Lin, Stan Sclaroff, and Federico Perazzi. Temporally distributed networks for fast video semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8818–8827, 2020.

[7] Ping Hu, Gang Wang, Xiangfei Kong, Jason Kuen, and Yap-Peng Tan. Motion-guided cascaded refinement network for video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1400–1409, 2018.

[8] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[9] Joakim Johnander, Martin Danelljan, Emil Brissman, Fahad Shahbaz Khan, and Michael Felsberg. A generative appearance model for end-to-end video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8953–8962, 2019.

[10] Bastian Leibe Jonathon Luiten, Paul Voigtlaender. PReMVOS: Proposal-generation, Refinement and Merging for the YouTube-VOS Challenge on Video Object Segmentation 2018. *The 1st Large-scale Video Object Segmentation Challenge - ECCV Workshops*, 2018.

[11] Anna Khoreva, Rodrigo Benenson, Eddy Ilg, Thomas Brox, and Bernt Schiele. Lucid data dreaming for object tracking. In *The DAVIS Challenge on Video Object Segmentation*, 2017.

[12] Junyeong Kim, Minuk Ma, Kyungsu Kim, Sungjin Kim, and Chang D Yoo. Progressive attention memory network for movie story question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8337–8346, 2019.

[13] Jyrki Kivinen, Alexander J Smola, and Robert C Williamson. Online learning with kernels. *IEEE transactions on signal processing*, 52(8):2165–2176, 2004.

[14] Sang-ho Lee, Simyung Chang, and Nojun Kwak. Urnet: User-resizable residual networks with conditional gating module. In *AAAI*, pages 4569–4576, 2020.

[15] Yu Li, Zhuoran Shen, and Ying Shan. Fast video object segmentation using the global context module. In *The European Conference on Computer Vision (ECCV)*, 2020.

[16] Yule Li, Jianping Shi, and Dahua Lin. Low-latency video semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5997–6005, 2018.

[17] Fanqing Lin, Yao Chou, and Tony Martinez. Flow adaptive video object segmentation. *Image and Vision Computing*, 94:103864, 2020.

[18] 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*, pages 740–755. Springer, 2014.

[19] K-K Maninis, Sergi Caelles, Yuhua Chen, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. Video object segmentation without temporal information. *IEEE transactions on pattern analysis and machine intelligence*, 41(6):1515–1530, 2018.

[20] Sachin Mehta, Mohammad Rastegari, Anat Caspi, Linda Shapiro, and Hannaneh Hajishirzi. Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation. In *Proceedings of the european conference on computer vision (ECCV)*, pages 552–568, 2018.

[21] Ravi Teja Mullapudi, William R. Mark, Noam Shazeer, and Kayvon Fatahalian. Hydranets: Specialized dynamic architectures for efficient inference. In *CVPR*, 2018.

[22] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9226–9235, 2019.

[23] Aljoša Ošep, Paul Voigtlaender, Jonathon Luiten, Stefan Breuers, and Bastian Leibe. Large-scale object discovery and detector adaptation from unlabeled video. *arXiv preprint arXiv:1712.08832*, 2017.

[24] Hyojin Park, Lars Sjosund, YoungJoon Yoo, Nicolas Monet, Jihwan Bang, and Nojun Kwak. Sinet: Extreme lightweight portrait segmentation networks with spatial squeeze module and information blocking decoder. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 2066–2074, 2020.

[25] Hyojin Park, Ganesh Venkatesh, and Nojun Kwak. Ttvos: Lightweight video object segmentation with adaptive template attention module and temporal consistency loss, 2020.

[26] Hyojin Park, Youngjoon Yoo, Geonseok Seo, Dongyoon Han, Sangdoo Yun, and Nojun Kwak. C3: Concentrated-comprehensive convolution and its application to semantic segmentation. *arXiv preprint arXiv:1812.04920*, 2018.

[27] Federico Perazzi, Anna Khoreva, Rodrigo Benenson, Bernt Schiele, and Alexander Sorkine-Hornung. Learning video

object segmentation from static images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2663–2672, 2017.

[28] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 724–732, 2016.

[29] 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.

[30] Andreas Robinson, Felix Jaremo Lawin, Martin Danelljan, Fahad Shahbaz Khan, and Michael Felsberg. Learning fast and robust target models for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7406–7415, 2020.

[31] Doyen Sahoo, Quang Pham, Jing Lu, and Steven C. H. Hoi. Online deep learning: Learning deep neural networks on the fly. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 2660–2666. International Joint Conferences on Artificial Intelligence Organization, 7 2018.

[32] Laura Sevilla-Lara, Deqing Sun, Varun Jampani, and Michael J Black. Optical flow with semantic segmentation and localized layers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3889–3898, 2016.

[33] Evan Shelhamer, Kate Rakelly, Judy Hoffman, and Trevor Darrell. Clockwork convnets for video semantic segmentation. In *European Conference on Computer Vision*, pages 852–868. Springer, 2016.

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

[35] Yi-Hsuan Tsai, Ming-Hsuan Yang, and Michael J Black. Video segmentation via object flow. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3899–3908, 2016.

[36] Andreas Veit and Serge Belongie. Convolutional networks with adaptive inference graphs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–18, 2018.

[37] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[38] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9481–9490, 2019.

[39] Paul Voigtlaender and Bastian Leibe. Online adaptation of convolutional neural networks for video object segmentation. In *British Machine Vision Conference 2017, BMVC 2017, London, UK, September 4-7, 2017*. BMVA Press, 2017.

[40] Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip HS Torr. Fast online object tracking and segmentation: A unifying approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1328–1338, 2019.

[41] Ziqin Wang, Jun Xu, Li Liu, Fan Zhu, and Ling Shao. Ranet: Ranking attention network for fast video object segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 3978–3987, 2019.

[42] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

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

[44] Linjie Yang, Yanran Wang, Xuehan Xiong, Jianchao Yang, and Aggelos K Katsaggelos. Efficient video object segmentation via network modulation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6499–6507, 2018.

[45] Li Yanwei, Song Lin, Chen Yukang, Li Zeming, and Zhang Xiangyu. Learning dynamic routing for semantic segmentation. In *CVPR*, 2020.

[46] Lu Zhang, Zhe Lin, Jianming Zhang, Huchuan Lu, and You He. Fast video object segmentation via dynamic targeting network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[47] Guanyu Zhou, Kihyuk Sohn, and Honglak Lee. Online incremental feature learning with denoising autoencoders. In *Artificial intelligence and statistics*, pages 1453–1461, 2012.

[48] Wu Zuxuan, Nagarajan Tushar, Kumar Abhishek, Rennie Steven, Davis Larry S., Grauman Kristen, and Feris Rogerio. Blockdrop: Dynamic inference paths in residual networks. In *CVPR*, 2018.