Learning What to Learn for Video Object Segmentation

Goutam Bhat[1(✉)], Felix Järemo Lawin[2], Martin Danelljan[1], Andreas Robinson[2], Michael Felsberg[2], Luc Van Gool[1], and Radu Timofte[1]

[1] CVL, ETH Zürich, Zürich, Switzerland
goutam.bhat@vision.ee.ethz.ch
[2] CVL, Linköping University, Linköping, Sweden

Abstract. Video object segmentation (VOS) is a highly challenging problem, since the target object is only defined by a first-frame reference mask during inference. The problem of how to capture and utilize this limited information to accurately segment the target remains a fundamental research question. We address this by introducing an end-to-end trainable VOS architecture that integrates a differentiable few-shot learner. Our learner is designed to predict a powerful parametric model of the target by minimizing a segmentation error in the first frame. We further go beyond the standard few-shot learning paradigm by learning what our target model should learn in order to maximize segmentation accuracy. We perform extensive experiments on standard benchmarks. Our approach sets a new state-of-the-art on the large-scale YouTube-VOS 2018 dataset by achieving an overall score of 81.5, corresponding to a 2.6% relative improvement over the previous best result. The code and models are available at https://github.com/visionml/pytracking.

1 Introduction

Semi-supervised Video Object Segmentation (VOS) is the problem of performing pixel-wise classification of a set of target objects in a video sequence. With numerous applications in e.g. autonomous driving [30,31], surveillance [7,9] and video editing [24], it has received significant attention in recent years. VOS is an extremely challenging problem, since the target objects are only defined by a reference segmentation in the first video frame, with no other prior information assumed. The VOS method therefore must utilize this very limited information about the target in order to perform segmentation in the subsequent frames.

While most state-of-the-art VOS methods employ similar image feature extractors and segmentation decoders, a number of approaches [15,25,29,33]

G. Bhat and F. J. Lawin—Both authors contributed equally.

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-58536-5_46) contains supplementary material, which is available to authorized users.

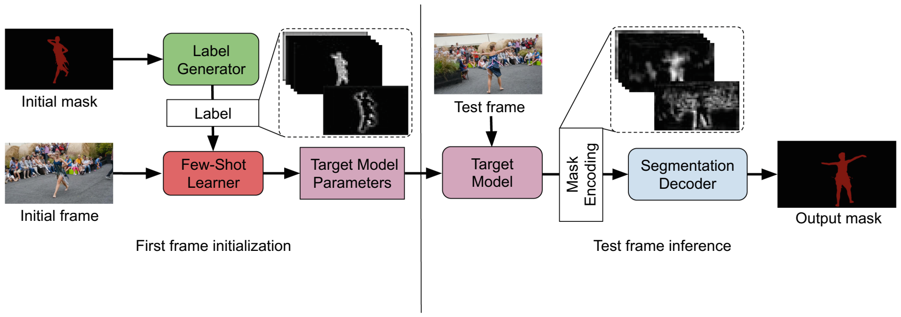

Fig. 1. An overview of our VOS approach. Given the annotated first frame, our few-shot learner optimizes the parameters of a target model which is tasked with predicting an encoding of the target mask (left). In subsequent test frames, the mask encoding output by the target model is utilized by the segmentation decoder to generate the final segmentation (right). Crucially, the label for the few-shot learner is generated by our label generator which is trained end-to-end jointly with the decoder. This allows us to learn what the target model should learn to output to the decoder in order to maximize segmentation accuracy.

have been proposed to utilize the reference frame annotation to perform segmentation. A promising direction is to employ feature matching techniques [14,15,25,33] in order to compare the reference frame regions with new images to segment. Such feature matching layers greatly benefit from their efficiency and differentiability. This allows the design of fully end-to-end trainable architectures, which has been shown to be important for segmentation performance [15,25,33]. However, in order to generalize to novel objects, feature matching techniques rely on a powerful and generic feature embedding, which can be difficult to learn. Instead of only relying on a pre-trained embedding, we investigate an alternative direction, where target-specific network parameters are learned during *inference*, in order to better integrate object appearance information.

We propose a novel VOS method, based on an effective few-shot learner that captures object information in a compact parametric target model. Given a test frame, our target model first predicts an intermediate representation of the target mask, which is then input to a segmentation decoder that generates the final prediction. To achieve a powerful model of the target object from the limited first frame annotation, our few-shot learner is designed to explicitly optimize an error between the target model prediction and a ground truth reference. Owing to the efficiency and differentiability of the few-shot learner, our approach can perform the inference-time learning without compromising the end-to-end training capability. Compared to the embedding based approaches, the inference-time learning in our approach provides greater adaptivity and generalizability to novel objects and scenarios.

We further address the problem of what intermediate mask representation the target model should be trained to predict in order to maximize segmentation accuracy. The standard optimization-based few-shot learning strategy forces the target model to learn to only generate an object mask output. However, directly learning to predict the segmentation mask from a single sample is difficult. More importantly, this approach limits the target-specific information sent to the segmentation decoder to be a single channel mask. To address this important issue, we further propose to *learn what to learn*. That is, our approach learns to generate a multi-channel mask encoding which is used by the few-shot learner as labels to train the target model. This enables our target model to provide richer target-specific information to the segmentation decoder in the test frames. Furthermore, in order to guide the learner to focus on the most crucial aspect of the target, we also learn to predict spatial importance weights for different elements in the few-shot learning objective. Since our optimization-based learner is differentiable, all modules in our architecture can be trained end-to-end by maximizing segmentation accuracy on annotated VOS videos. An overview of our video object segmentation approach is shown in Fig. 1.

Contributions: Our main contributions are listed as follows. **(i)** We propose a novel VOS architecture, based on an optimization-based few-shot learner. **(ii)** Our few-shot learner predicts the target model parameters in an efficient and differentiable manner, enabling end-to-end training. **(iii)** We go beyond standard few-shot learning approaches to further learn what the target model should learn in order to maximize segmentation accuracy. **(iv)** We utilize our learned mask encoding to design a light-weight bounding box initialization module, allowing our approach to generate segmentation masks using only a reference bounding box as input.

We perform comprehensive experiments on the YouTube-VOS [39] and DAVIS [27] benchmarks. Our approach sets a new state-of-the-art on the large-scale YouTube-VOS 2018 dataset, achieving an overall score of 81.5. We further provide detailed ablative analyses, showing the impact of each component in the proposed method.

2 Related Work

In recent years, progress within video object segmentation has surged, leading to rapid performance improvements. Benchmarks such as DAVIS [27] and YouTube-VOS [39] have had a significant impact on this development.

Target Models in VOS: Early works mainly adapted semantic segmentation networks to the VOS task through online fine-tuning [5,13,22,28,38]. However, this strategy easily leads to overfitting to the initial target appearance and impractically long run-times. More recent methods [14,18,24,25,33,35,37] therefore integrate target-specific appearance models into the segmentation architecture. In addition to improved run-times, many of these methods can also benefit from full end-to-end learning, which has been shown to have a crucial impact on performance [15,25,33]. Generally, these works train a target-agnostic segmentation decoder that is conditioned on a target model. The latter integrates information about the target object, deduced from the initial image-mask pair. The target model predicts target-specific information for the test frame, which is then provided to the target-agnostic segmentation decoder to obtain the final prediction. Crucially, in order to achieve end-to-end training of the entire network, the target model needs to be differentiable.

While most VOS methods share similar feature extractors and segmentation decoders, several different strategies for encoding and exploiting the reference frame target information have been proposed. In RGMP [24], a representation of the target is generated by encoding the reference frame. This representation is then concatenated with the current-frame features, before being input to the segmentation decoder. Similarly, OSNM [40] predicts an attention vector from the reference frame and ground-truth target mask, which combined with a spatial guidance map is used to segment the target. The approach in [18] extends RGMP to jointly process multiple targets using an instance specific attention generator. In [15], a light-weight generative model is learned from embedded features corresponding to the initial target labels. The generative model is then used to classify features from the incoming frames. The target models in [14,33] directly store foreground features and classify pixels in the incoming frames through feature matching. The recent STM approach [25] performs feature matching within a space-time memory network. It implements a read operation, which retrieves information from the encoded memory through an attention mechanism. This information is then sent to the segmentation decoder to predict the target mask. The method [37] predicts template correlation filters given the input target mask. Target classification is then performed by applying the correlation filters on the test frame. Lastly, the recent method [29] trains a target model consisting of a two-layer neural network using the Conjugate Gradient method.

Meta-learning for VOS: Since the VOS task itself includes a few-shot learning problem, it can be addressed with techniques developed for meta-learning [3,10,17]. A few recent attempts [1,20] follow this direction. The method [1] learns a classifier using k-means clustering of segmentation features in the training frame. In [20], the final layer of a segmentation network is predicted by closed-form ridge regression [3], using the reference example pair. Meta-learning based techniques have been more commonly adopted in the related field of visual tracking [4,6,8,26]. The method in [26] performs gradient based adaptation to the current target, while [6] learns a target specific feature space online which is combined with a Siamese-based matching network. The recent work [4] proposes an optimization-based meta-learning strategy, where the target model directly generates the output classifications scores. In contrast to these previous approaches, we integrate a differentiable optimization-based few-shot learner to capture target information for the VOS problem. Furthermore, we go beyond standard few-shot and meta-learning techniques by learning what the target model should learn in order to generate accurate segmentations.

3 Method

In this section, we present our method for video object segmentation (VOS). First, we describe our few-shot learning formulation for VOS in Sect. 3.1. In Sect. 3.2 we then describe our approach to learn what the few-shot learner should learn. Section 3.3 details our target module and the internal few-shot learner. Our segmentation architecture is described next in Sect. 3.4. The inference and training procedures are detailed in Sects. 3.5 and 3.6, respectively. Finally, Sect. 3.7 describes how our approach can be easily extended to perform VOS with only a bounding box initialization.

3.1 Video Object Segmentation as Few-Shot Learning

In VOS, the target object is only defined by a reference target mask given in the first frame. No other prior information about the test object is assumed. The VOS method therefore needs to exploit the given first-frame annotation in order to segment the target in each subsequent frame. To address this core problem, we first consider a general class of VOS architectures formulated as $S_\theta(I, T_\tau(I))$, where θ denotes the learnable parameters. The network S_θ takes the current image I along with the output of a target model T_τ. While S_θ itself is target-agnostic, it is conditioned on T_τ, which integrates information about the target object, encoded in its parameters τ. The target model generates a target-aware output that is used by S_θ to predict the final segmentation. The target model parameters τ are obtained from the initial image I_0 and its given mask y_0, which defines the target object itself. We denote this as a function $\tau = A_\theta(I_0, y_0)$. The key challenge in this VOS formulation is in the design of T_τ and A_θ.

We note that the pair (I_0, y_0) in the above formulation constitutes a training sample for learning to segment the given target. However, this training sample is only given during inference. Hence, a few-shot learning problem naturally arises within VOS. We adopt this view to develop our approach. In relation to few-shot learning, A_θ constitutes the internal learning method, which generates the parameters τ of the target model T_τ from a single example pair (I_0, y_0). While there exist a diverse set of few-shot learning methodologies, we aim to find the target model parameters τ that minimizes a supervised learning objective ℓ,

$$\tau = A_\theta(x_0, y_0) = \arg\min_{\tau'} \ell(T_{\tau'}(x_0), y_0)\,. \tag{1}$$

Here, the target model T_τ is learned to output the segmentation of the target object in the initial frame. In general, we operate on a deep representation of the input image $x = F_\theta(I)$, generated by e.g. a ResNet architecture. Given a new frame I during inference, the object is segmented as $S_\theta(I, T_\tau(F_\theta(I))$. In other words, the target model is applied to the new frame to generate a first segmentation. This output is further refined by S_θ, which can additionally integrate powerful pre-learned knowledge from large VOS datasets.

The main advantage of the optimization-based formulation (1) is that the target model parameters are predicted by directly minimizing the segmentation error in the first frame. This ensures robust segmentation prediction in the coming frames, since consecutive video frames are highly-correlated. For practical purposes, however, the target model prediction (1) also needs to be efficient. Further, to enable end-to-end training of the entire VOS architecture, we wish the learner A_θ to be *differentiable*. While this is challenging in general, different strategies have been proposed in the literature [3,4,17], mostly in the context of meta-learning. We detail the employed approach in Sect. 3.3. In the next section, we first address another fundamental limitation of the formulation in Eq. (1).

3.2 Learning What to Learn

In the approach discussed in the previous section, the target model T_τ learns to predict an initial segmentation mask of the target object from the first frame. This mask is then refined by the network S_θ, which possesses strong learned segmentation priors. However, S_θ is not limited to operate on an approximate target mask in order to perform target-conditional segmentation. In contrast, any information that alleviates the task of the network S_θ to identify and accurately segment the target object is beneficial. Predicting only a single-channel mask thus severely limits the amount of target-specific information that can be passed to the network S_θ. Ideally, the target model should predict multi-channel activations which can provide strong target-aware cues in order to guide the network S_θ to generate accurate segmentations. However, this is not possible in the standard few-shot learning setting (1), since the output of the target model T_τ is directly defined by the available ground-truth mask y_0. In this work, we address this issue by learning what our internal few-shot learner should learn.

Instead of directly employing the first-frame mask y_0 as labels in our few-shot learner, we propose to *learn* these labels. To this end, we introduce a trainable label generator $E_\theta(y)$ that takes the ground-truth mask y as input and predicts the labels for the few-shot learner. The target model is thus predicted as,

$$\tau = A_\theta(x_0, y_0) = \underset{\tau'}{\arg\min}\ \ell\big(T_{\tau'}(x_0), E_\theta(y_0)\big)\,. \tag{2}$$

Unlike in (1), the labels $E_\theta(y_0)$ generated by encoding the ground-truth mask y_0 can be multi-dimensional, allowing the target model T_τ to predict a richer target mask representation in the test frames.

The formulation (2) assigns equal weight to all elements in the few-shot learning loss $\ell\big(T_\tau(x_0), E_\theta(y_0)\big)$. However, this might not be optimal for maximizing the final segmentation accuracy. For instance, it is often beneficial to assign higher weights to target regions in case of small objects, to account for an imbalanced training set. Similarly, it might be beneficial to assign lower weights to ambiguous regions such as object boundaries, and let the segmentation network S_θ handle them. We allow such flexibility in our loss by introducing a weight predictor module $W_\theta(y)$. Similar to E_θ, it takes the ground-truth mask y as input and predicts the importance weight for each element in the loss $\ell\big(T_\tau(x_0), E_\theta(y_0)\big)$. Thus, our weight predictor can guide the few-shot learner to focus on the most crucial aspects of the ground truth label $E_\theta(y)$.

We have not yet fully addressed the question of *how* to learn the label generator E_θ, and the weight predictor W_θ. Ideally, we wish to train all parameters θ in our segmentation architecture in an end-to-end manner on annotated VOS datasets. This requires back-propagating the error measured between the final segmentation output $\tilde{y}_t = S_\theta(I_t, T_\tau(F_\theta(I_t))$ and the ground truth y_t on a test frame I_t. However, this is feasible only if the internal learner (2) is efficient and differentiable w.r.t. both the underlying features x *and* the parameters of the label generator E_θ and weight predictor W_θ. We address these open questions in the next section, to achieve an efficient and end-to-end trainable VOS architecture.

3.3 Few-Shot Learner

In this section, we detail our target model T_τ and the internal few-shot learner A_θ. The target model $T_\tau : \mathbb{R}^{H\times W\times C} \to \mathbb{R}^{H\times W\times D}$ is trained to map C-dimensional deep features x to a D-dimensional encoding of the target mask with the same spatial resolution $H \times W$. We require T_τ to be efficient and differentiable. To ensure this, we employ a linear target model $T_\tau(x) = x * \tau$, where $\tau \in \mathbb{R}^{K\times K\times C\times D}$ constitutes the weights of a convolutional layer with kernel size K. While such a target model is simple, it operates on high dimensional deep feature maps. Consequently, it is capable of predicting a rich encoding of the target mask, leading to improved segmentation performance, as shown in our experiments (see Sect. 4). Moreover, while a more complex target module has larger capacity, it is also prone to overfitting and is computationally more costly to learn.

The parameters of the target model are obtained using our internal few-shot learner A_θ by minimizing the squared error between the output of the target model $T_\tau(x)$ and the generated ground-truth labels $E_\theta(y)$, weighted by the element-wise importance weights $W_\theta(y)$,

$$L(\tau) = \frac{1}{2} \sum_{(x_t, y_t)\in\mathcal{D}} \big\| W_\theta(y_t) \cdot \big(T_\tau(x_t) - E_\theta(y_t)\big) \big\|^2 + \frac{\lambda}{2} \|\tau\|^2 . \tag{3}$$

Here, $\mathcal{D} = \{(x_t, y_t)\}_{t=0}^{Q-1}$ is a set of feature-mask pairs $(x_t,\, y_t)$ of size Q. While it usually contains a single ground-truth annotated frame, it is often useful to include additional frames by, for instance, self-annotating new images in the video. The scalar λ is a learned regularization parameter.

As a next step, we design a differentiable and efficient few-shot learner that minimizes (3) as $\tau = A_\theta(\mathcal{D}) = \arg\min_{\tau'} L(\tau')$. We note that (3) is a convex quadratic objective in τ. Thus, it has a well-known closed-form solution, which can be expressed in either primal or dual form. However, both options lead to computations that are intractable when aiming for acceptable frame-rates, requiring extensive matrix multiplications and solutions to linear systems. Moreover, these methods cannot directly utilize the convolutional structure of the problem. Instead we find an approximate solution of (3) by applying steepest descent iterations, previously also used in [4]. Given a current estimate τ^i, it finds the step-length α^i that minimizes the loss in the gradient direction $\alpha^i = \arg\min_\alpha L(\tau^i - \alpha g^i)$. Here, $g^i = \nabla L(\tau^i)$ is the gradient of (3) at τ^i. The optimization iteration can then be expressed as,

$$\tau^{i+1} = \tau^i - \alpha^i g^i , \quad \alpha^i = \frac{\|g^i\|^2}{\sum_t \|W_\theta(y_t) \cdot (x_t * g^i)\|^2 + \lambda \|g^i\|^2} ,$$
$$g^i = \sum_t x_t *^{\mathrm{T}} \Big(W_\theta^2(y_t) \cdot \big(x_t * \tau^i - E_\theta(y_t) \big) \Big) + \lambda \tau^i . \qquad (4)$$

Here, $*^{\mathrm{T}}$ denotes the transposed convolution operation. A detailed derivation is provided in the supplementary material.

Note that all computations in (4) are easily implemented using standard neural network operations. Since all operations are differentiable, the resulting target model parameters τ^i after i iterations are differentiable w.r.t. all network parameters θ. Our internal few-shot learner is implemented as a network module $A_\theta(\mathcal{D}, \tau^0) = \tau^N$, that performs N iterations of steepest descent (3), starting from a given initialization τ^0. Thanks to the rapid convergence of steepest descent, we only need to perform a handful of iterations during training and inference. Moreover, our optimization-based formulation allows the target model parameters τ to be efficiently updated with new samples by simply adding them to $\mathcal{D}$ and applying a few iterations (4), starting from the current parameters $\tau^0 = \tau$.

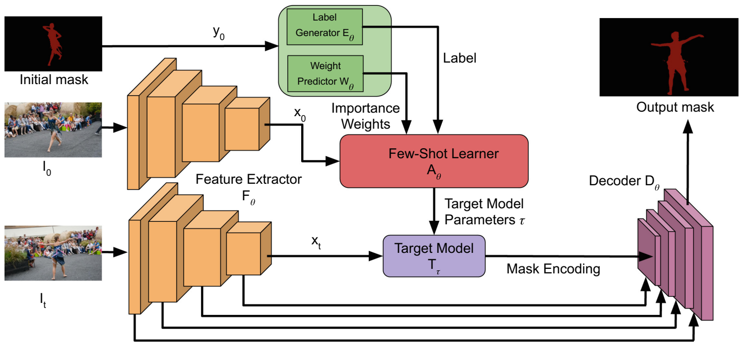

Fig. 2. An overview of our segmentation architecture. It contains a few-shot learner, which generates a parametric target model T_τ from the initial frame annotation. The parameters τ are computed by minimizing the loss (3), using the labels predicted by E_θ. The elements of the loss are weighted using the importance weights predicted by W_θ. In the incoming frames, the target model predicts the mask encoding, which is processed along with image features by our decoder D_θ to produce the final mask.

3.4 Video Object Segmentation Architecture

Our VOS method is implemented as a single end-to-end network, illustrated in Fig. 2. It is composed of a deep feature extractor F_θ, label generator E_θ, loss weight predictor W_θ, target model T_τ, few-shot learner A_θ and the segmentation decoder D_θ. As previously mentioned, θ denotes the network parameters learned during the offline training, while τ are the target model parameters that are *predicted* by the few-shot learner module during inference. The following sections describes the individual modules. More details are provided in the supplement.

Feature Extractor F_θ: We employ a ResNet-50 network as backbone feature extractor F_θ. Features from F_θ are input to both the decoder module D_θ and the target model T_τ. For the latter, we employ the third residual block, which has a spatial stride of $s = 16$. These features are first fed through an additional conv. layer that reduces the dimension to $C = 512$, before given to T_τ.

Few-Shot Label Generator E_θ: Our label generator E_θ predicts the ground truth label for the few-shot learner by encoding the input target mask. The latter is mapped to the resolution of the deep features as $E_\theta : \mathbb{R}^{sH \times sW \times 1} \rightarrow \mathbb{R}^{H \times W \times D}$, where H, W and D are the height, width and dimensionality of the target model features and s is the feature stride. We implement the proposed mask encoder E_θ as a conv-net, decomposed into a generic mask feature extractor for processing the input mask y and a prediction layer for generating the final label.

Weight Predictor W_θ: The weight predictor $W_\theta : \mathbb{R}^{sH \times sW \times 1} \rightarrow \mathbb{R}^{H \times W \times D}$ generates weights for the internal loss (3). It is implemented as a conv-net that takes the target mask y as input, similar to E_θ. In our implementation, W_θ shares the mask feature extractor with E_θ.

Target Model T_τ and Few-Shot Learner A_θ: We implement our target model T_τ as convolutional filter with a kernel size of $K = 3$. The number of output channels D is set to 16. Our few-shot learner A_θ (see Sect. 3.3) predicts the target model parameters τ in the first frame by applying steepest descent iterations (4). On subsequent test frames, we apply the predicted target model $T_\tau(x)$ to obtain target mask encodings, which are then provided to the segmentation decoder.

Segmentation Decoder D_θ: This module takes the output of the target model T_τ along with backbone features as input and predicts the final segmentation mask. Our approach can in principle be combined with any decoder architecture. For simplicity, we employ a decoder network similar to the one used in [29]. We adapt this network to process a multi-channel target mask encoding as input.

3.5 Inference

In this section, we describe our inference procedure. Given a test sequence $\mathcal{V} = \{I_t\}_{t=0}^{Q}$, along with the first frame annotation y_0, we first create an initial training set $\mathcal{D}_0 = \{(x_0, y_0)\}$ for the few-shot learner, consisting of the single sample pair. Here, $x_0 = F_\theta(I_0)$ is the feature map extracted from the first frame. The few-shot learner then predicts the parameters $\tau_0 = A_\theta(\mathcal{D}_0, \tau^0)$ of the target model by minimizing the internal loss (3). We set the initial estimate of the target model $\tau^0 = 0$ to all zeros. Note that the ground-truth $E_\theta(y_0)$ and importance weights $W_\theta(y_0)$ for the minimization problem (3) are predicted by our network.

The learned model τ_0 is then applied on the subsequent test frame I_1 to obtain a mask encoding $T_{\tau_0}(x_1)$. This encoding is then processed by the decoder module, along with the image features, to generate the mask prediction $\tilde{y}_1 = D_\theta(x_1, T_{\tau_0}(x_1))$. In order to adapt to the changes in the scene, we further update our target model using the information from the processed frames. This is achieved by extending the training set $\mathcal{D}_0$ with the new sample $(x_1, \tilde{y}_1)$, where the predicted mask $\tilde{y}_1$ serves as the pseudo-label for the frame I_1. The extended training set $\mathcal{D}_1$ is then used to obtain new target model parameters $\tau_1 = A_\theta(\mathcal{D}_1, \tau_0)$. Note that instead of predicting τ_1 from scratch, our optimization-based learner allows us to efficiently update the previous target model τ_0. Specifically, we apply additional $N_{\text{update}}^{\text{inf}}$ steepest-descent iterations (4) with the new training set $\mathcal{D}_1$. The updated T_{τ_1} is then applied on the next frame I_2. This process is repeated till the end of the sequence.

Details: Our few-shot learner A_θ employs $N_{\text{init}}^{\text{inf}} = 20$ iterations in the first frame and $N_{\text{update}}^{\text{inf}} = 3$ iterations in each subsequent frame. Our few-shot learner formulation (3) allows an easy integration of a global importance weight for each frame in the training set $\mathcal{D}$. We exploit this flexibility to integrate an exponentially decaying weight η^{-t} to reduce the impact of older frames. We set $\eta = 0.9$ and ensure the weights sum to one. We ensure a maximum $K_{\max} = 32$ samples in the few-shot training set $\mathcal{D}$, by removing the oldest sample. We always keep the first frame since it has the reference target mask y_0. Each frame in the sequence is processed by first cropping a patch that is 5 times larger than the previous estimate of target, while ensuring the maximal size to be equal to the image itself. The cropped region is resized to 832×480 with preserved aspect ratio. If a sequence contains multiple targets, we independently process each in parallel and merge the predicted masks using the soft-aggregation operation [24].

3.6 Training

To train our end-to-end network architecture, we aim to simulate the inference procedure employed by our approach, described in Sect. 3.5. This is achieved by training the network on mini-sequences $\mathcal{V} = \{(I_t, y_t)\}_{t=0}^{Q-1}$ of length Q. These are constructed by sampling frames from annotated VOS sequences. In order to induce robustness to fast appearance changes, we randomly sample frames in temporal order from a larger window of Q' frames. As in inference, we create the initial few-shot training set from the first frame $\mathcal{D}_0 = \{(x_0, y_0)\}$. This is used to learn the initial target model parameters $\tau_0 = A_\theta(\mathcal{D}_0, 0)$ by performing $N_{\text{init}}^{\text{train}}$ steepest descent iterations. In subsequent frames, we use $N_{\text{update}}^{\text{train}}$ iterations to update the model as $\tau_t = A_\theta(\mathcal{D}_t, \tau_{t-1})$. The final prediction $\tilde{y}_t = D_\theta(x_t, T_{\tau_{t-1}}(x_t))$ in each frame is added to the few-shot train set $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(x_t, \tilde{y}_t)\}$. All network parameters θ are trained by minimizing the per-sequence loss,

$$\mathcal{L}_{\text{seq}}(\theta; \mathcal{V}) = \frac{1}{Q-1} \sum_{t=1}^{Q-1} \mathcal{L}\Big(D_\theta\big(F_\theta(I_t), T_{\tau_{t-1}}(F_\theta(I_t))\big),\, y_t\Big)\,. \tag{5}$$

Here, $\mathcal{L}(\tilde{y}, y)$ is the employed loss between the prediction $\tilde{y}$ and ground-truth y. We compute the gradient of the final loss (5) by averaging over multiple mini-sequences in each batch. The target model parameters τ_{t-1} in (5) are predicted by our few-shot learner A_θ, and therefore depend on the network parameters of the label generator E_θ, weight predictor W_θ, and feature extractor F_θ. These modules can therefore be trained end-to-end by minimizing the loss (5).

Details: Our network is trained using the YouTube-VOS [39] and DAVIS [27] datasets. We use mini-sequences of length $Q = 4$ frames, generated from video segments of length $Q' = 100$. We employ random flipping, rotation, and scaling for data augmentation. We then sample a random 832×480 crop from each frame. The number of steepest-descent iterations in the few-shot learner A_θ is set to $N_{\text{init}}^{train} = 5$ for the first frame and $N_{\text{update}}^{\text{train}} = 2$ in subsequent frames. We use the Lovasz [2] segmentation loss in (5). We initialize our backbone ResNet-50 with the Mask R-CNN [11] weights from [23] (see the supplementary for analysis). All other modules are initialized using [12]. Our network is trained using ADAM [16]. We first train our network for 70k iterations with the backbone weights fixed. The complete network, including the backbone feature extractor, is then trained for an additional 80k iterations. For evaluation on DAVIS, we further fine-tune the network using only the DAVIS training split. The entire training takes 48 hours on 4 Nvidia V100 GPUs. Further details are provided in the supplementary.

3.7 Bounding Box Initialization

In many practical applications, it is too costly to generate an accurate reference-frame annotation to perform VOS. We therefore follow the recent trend [34,36] of using weaker supervision by only assuming the target bounding box in the first frame. By exploiting our learned mask encoding, we show that our architecture can accommodate this setting with only a minimal addition. Analogously to the label generator E_θ, we introduce a bounding box encoder $B_\theta(b_0, x_0)$. It takes a mask-representation b_0 of the initial box along with backbone features x_0 as input and predicts a target mask encoding in the same D-dimensional output space of E_θ and T_τ. This allows us to exploit our existing decoder network in order to predict the target mask in the initial frame as $\tilde{y}_0 = D_\theta(x_0, B_\theta(b_0, x_0))$. VOS is then performed using the same procedure as described in Sect. 3.5, by simply replacing the ground-truth mask y_0 with the predicted mask $\tilde{y}_0$. Our box encoder B_θ consists of two residual blocks followed by a conv-layer and is easily trained by freezing the other parameters in the network. Thus, we only need to sample single frames during training and minimize the segmentation loss $\mathcal{L}(D_\theta(x_0, B_\theta(b_0, x_0)), y_0)$. As a result, we gain the ability to perform VOS with box-initialization without losing performance in the standard VOS setting.

Details: We train the box encoder on images from MSCOCO[19] and YouTube-VOS for 50,000 iterations, while freezing the pre-trained components of the network. During inference we reduce the impact of the first frame annotation by setting $\eta = 0.8$ and remove it from the memory after $K_{\max}$ frames. For best performance, we only update the target model every fifth frame with $N_{\text{update}}^{\text{inf}} = 5$.

4 Experiments

We evaluate our approach on the two standard VOS benchmarks: YouTube-VOS and DAVIS 2017. Detailed results are provided in the supplementary material. Our approach operates at 14 FPS on single object sequences.

4.1 Ablative Analysis

Here, we analyze the impact of the key components in the proposed VOS architecture. Our analysis is performed on a validation set consisting of 300 sequences randomly sampled from the YouTube-VOS 2019 training set. For simplicity, we do not train the backbone ResNet-50 weights in the networks of this comparison. The networks are evaluated using the mean Jaccard $\mathcal{J}$ index (IoU). Results are shown in Table 1. Qualitative examples are visualized in Fig. 3.

Baseline: Our baseline constitutes a version where the target model is trained to directly predict an initial mask, which is subsequently refined by the decoder D_θ. That is, the ground-truth employed by the few-shot learner is set to the reference mask. Further, we do not back-propagate through the learning of the target model during offline training and instead only train the decoder module D_θ. Thus, we do not perform end-to-end training through the learner.

End-to-End Training: Here, we exploit the differentiablity of our few-shot learner to train the underlying features used by the target model in an end-to-end manner. That is, we train the conv. layer which processes the backbone features (see Sect. 3.4). Learning specialized features for the target model provides a substantial improvement of +3.0 in $\mathcal{J}$ score. This clearly demonstrates the importance of end-to-end learning capability provided by our few-shot learner.

Label Generator E_θ: Instead of training the target model to predict an initial segmentation mask, we here employ the proposed label generator E_θ to learn what the target model should learn. This allows training the target model to output richer representation of the target mask, leading to an improvement of +1.4 in $\mathcal{J}$ score over the version which does not employ the label generator.

Weight Predictor W_θ: In this version, we additionally include the proposed weight predictor W_θ to obtain the importance weights for the internal loss (3). Using the importance weights leads to an additional improvement of +0.9 in $\mathcal{J}$ score. This shows that our weight predictor learns to predict what the internal learner should focus on, in order to generate a robust target model.

Table 1. Ablative analysis on a validation set of 300 videos sampled from the YouTube-VOS 2019 training set. We analyze the impact of **end-to-end training**, the **label generator** and the **weight predictor** by incrementally adding them one at a time.

	Baseline	+End-to-end	+Label Generator E_θ	+Weight Predictor W_θ
$\mathcal{J}$ Score (%)	74.5	77.5	78.9	79.8

4.2 State-of-the-Art Comparison

In this section, we compare our method, denoted **LWL**, with state-of-the-art. Since many approaches employ additional segmentation datasets during training, we always indicate whether additional data is used. We report results for the standard version of our approach which employs additional data (as described in Sect. 3.6), and a second version that is only trained on the train split of the specific dataset. For the latter version, we initialize the backbone ResNet-50 with ImageNet pre-training instead of the MaskRCNN backbone weights.

Table 2. State-of-the-art comparison on the large-scale YouTube-VOS 2018 validation set. Our approach LWL outperforms all previous methods, both when using with additional training data and when training only on YouTube-VOS 2018 train split.

	Additional Training Data						**Only YouTube-VOS training**					
	LWL	STM [25]	SiamRCNN [34]	PreMVOS [21]	OnAVOS [32]	OSVOS [5]	**LWL**	STM [25]	FRTM [29]	AGAME [15]	AGSSVOS [18]	S2S [38]
$\mathcal{G}$ (overall)	**81.5**	79.4	73.2	66.9	55.2	58.8	**80.2**	68.2	72.1	66.1	71.3	64.4
$\mathcal{J}_{\text{seen}}$	**80.4**	79.7	73.5	71.4	60.1	59.8	**78.3**	–	72.3	67.8	71.3	71.0
$\mathcal{J}_{\text{unseen}}$	**76.4**	72.8	66.2	56.5	46.1	54.2	**75.6**	–	65.9	61.2	65.5	55.5
$\mathcal{F}_{\text{seen}}$	**84.9**	84.2	–	–	62.7	60.5	**82.3**	–	76.2	69.5	75.2	70.0
$\mathcal{F}_{\text{unseen}}$	**84.4**	80.9	–	–	51.4	60.7	**84.4**	–	74.1	66.2	73.1	61.2

YouTube-VOS [39]**:** We evaluate our approach on the YouTube-VOS 2018 validation set, containing 474 sequences and 91 object classes. Out of these, 26 classes are *unseen* in the training dataset. The benchmark reports Jaccard $\mathcal{J}$ and boundary $\mathcal{F}$ scores for *seen* and *unseen* categories. Methods are ranked by the overall $\mathcal{G}$-score, obtained as the average of all four scores.

Among previous approaches, STM [25] obtains the highest overall $\mathcal{G}$-score of 79.4 (see Table 2). Our approach LWL significantly outperforms STM with a relative improvement of over 2.6%, achieving an overall $\mathcal{G}$-score of 81.5. Without the use of additional training data, the performance of STM is notably reduced to an overall $\mathcal{G}$-score of 68.2. FRTM [29] and AGSS-VOS [18] achieve stronger performance of 72.1 and 71.3 respectively, when employing only YouTube-VOS data for training. Our approach outperforms all previous methods by a margin of over 8.1% in this setting. Remarkably, this version even outperforms all previous methods trained with additional data, achieving a $\mathcal{G}$-score of 80.2. This clearly demonstrates the strength of our few-shot learner. Furthermore, our approach achieves an improvement of 9.7% and 10.3% on the $\mathcal{J}_{\text{unseen}}$ and $\mathcal{F}_{\text{unseen}}$

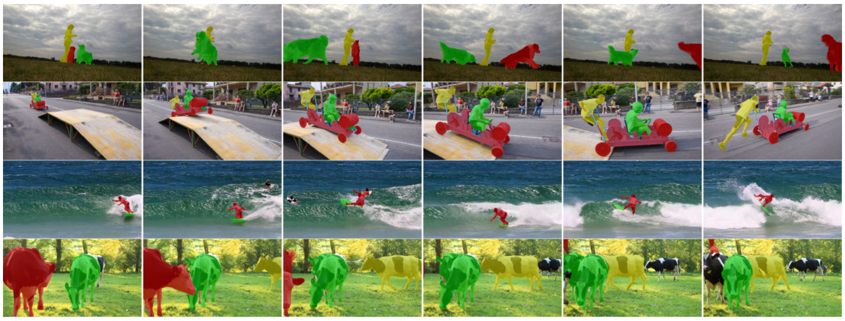

Fig. 3. Qualitative results of our VOS method. Our approach provides accurate segmentations in very challenging scenarios, including occlusions (row 1 and 3), distractor objects (row 1, and 2), and appearance changes (row 1, 2 and 3). Row 4 shows an example failure case, due to severe occlusions and very similar objects.

scores respectively, over FRTM. This demonstrates the superior generalization capability of our approach to classes that are unseen during training.

DAVIS 2017 [27]**:** The DAVIS 2017 validation set contains 30 videos. In addition to our standard training setting (see Sect. 3.6), we provide results of our approach when using only the DAVIS 2017 training set. Methods are evaluated in Table 3 in terms of mean Jaccard $\mathcal{J}$ and boundary $\mathcal{F}$ scores, along with the overall score $\mathcal{J}\&\mathcal{F}$. Our approach achieves similar performance to STM, with a marginal 0.2 lower overall score, when using additional training data. However, when employing only DAVIS 2017 training data, the performance of STM is significantly reduced. In contrast, our approach outperforms all previous methods in this setting, with an improvement of 5.5% over the second best method FRTM and 31.3% over STM in terms of $\mathcal{J}\&\mathcal{F}$.

Table 3. State-of-the-art comparison on the DAVIS 2017 validation dataset. Our approach LWL is on par with the best performing method STM, while significantly outperforming all previous methods with only the DAVIS 2017 training data.

	Additional Training Data								Only DAVIS 2017 training				
	LWL	STM [25]	SiamRCNN [34]	PreMVOS [21]	FRTM [29]	AGAME [15]	FEELVOS [33]	AGSSVOS [18]	**LWL**	STM [25]	FRTM [29]	AGAME [15]	AGSSVOS [18]
$\mathcal{J}\&\mathcal{F}$	81.6	**81.8**	74.8	77.8	76.7	70.0	71.5	67.4	**74.3**	43.0	68.8	63.2	66.6
$\mathcal{J}$	79.1	**79.2**	69.3	73.9	73.8	67.2	69.1	64.9	**72.2**	38.1	66.4	–	63.4
$\mathcal{F}$	84.1	**84.3**	80.2	81.7	79.6	72.7	74.0	69.9	**76.3**	47.9	71.2	–	69.8

Bounding Box Initialization: Finally, we evaluate our approach on VOS with bounding box initialization on YouTube-VOS 2018 and DAVIS 2017 validation sets. Results are reported in Table 4. We compare with the recent Siam-RCNN [34] and Siam-Mask [36]. Our approach LWL achieves a relative improvement of over 3% in terms of $\mathcal{G}$-score over the previous best method Siam-RCNN

Table 4. State-of-the-art comparison with box-initialization on YouTube-VOS 2018 and DAVIS 2017 validation sets. LWL outperforms existing methods on both datasets.

	YouTube-VOS 2018					**DAVIS 2017**		
Method	$\mathcal{G}$	$\mathcal{J}_{\text{seen}}$	$\mathcal{J}_{\text{unseen}}$	$\mathcal{F}_{\text{seen}}$	$\mathcal{F}_{\text{unseen}}$	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$
LWL	**70.4**	**73.0**	**63.0**	**75.9**	**70.0**	**70.8**	**68.2**	73.5
Siam-RCNN [34]	68.3	69.9	61.4	–	–	70.6	66.1	**75.0**
Siam-Mask [36]	52.8	62.2	45.1	58.2	47.7	56.4	54.3	58.5

on YouTube-VOS. Similarly, on DAVIS 2017, LWL outperforms Siam-RCNN with a $\mathcal{J}\&\mathcal{F}$-score of 70.8. Our approach remarkably outperforms several recent methods employing mask initialization in Table 2 and Table 3, demonstrating that it can readily generalize to the box-initialization setting.

5 Conclusions

We present a novel VOS approach that integrates an optimization-based few-shot learner. The learner predicts a compact target model by minimizing a few-shot objective in the first frame. Given a test frame, the target model outputs a mask encoding which is used by a decoder to predict the target mask. Our learner is differentiable, ensuring an end-to-end trainable VOS architecture We go beyond standard few-shot learning by also learning what the target model should learn in order to maximize segmentation accuracy. This is achieved by designing modules that predict the label and importance weights in the few-shot objective.

Acknowledgments. This work was partly supported by the ETH Zürich Fund (OK), a Huawei Technologies Oy (Finland) project, an Amazon AWS grant, Nvidia, ELLIIT Excellence Center, the Wallenberg AI, Autonomous Systems and Software Program (WASP) and the SSF project Symbicloud.

References

1. Behl, H.S., Najafi, M., Arnab, A., Torr, P.H.S.: Meta learning deep visual words for fast video object segmentation. In: NeurIPS 2019 Workshop on Machine Learning for Autonomous Driving (2018)
2. Berman, M., Rannen Triki, A., Blaschko, M.B.: The lovász-softmax loss: a tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4413–4421 (2018)
3. Bertinetto, L., Henriques, J.F., Torr, P., Vedaldi, A.: Meta-learning with differentiable closed-form solvers. In: International Conference on Learning Representations (2019)
4. Bhat, G., Danelljan, M., Van Gool, L., Timofte, R.: Learning discriminative model prediction for tracking. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 6182–6191 (2019)

5. Caelles, S., Maninis, K.K., Pont-Tuset, J., Leal-Taixé, L., Cremers, D., Van Gool, L.: One-shot video object segmentation. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5320–5329. IEEE (2017)
6. Choi, J., Kwon, J., Lee, K.M.: Deep meta learning for real-time target-aware visual tracking. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 911–920 (2019)
7. Cohen, I., Medioni, G.: Detecting and tracking moving objects for video surveillance. In: Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149), vol. 2, pp. 319–325. IEEE (1999)
8. Danelljan, M., Van Gool, L., Timofte, R.: Probabilistic regression for visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2020)
9. Erdélyi, A., Barát, T., Valet, P., Winkler, T., Rinner, B.: Adaptive cartooning for privacy protection in camera networks. In: 2014 11th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 44–49. IEEE (2014)
10. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70, pp. 1126–1135. JMLR. org (2017)
11. He, K., Gkioxari, G., Dollár, P., Girshick, R.B.: Mask r-cnn. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2980–2988 (2017)
12. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: ICCV (2015)
13. Hu, P., Wang, G., Kong, X., Kuen, J., Tan, Y.P.: Motion-guided cascaded refinement network for video object segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1400–1409 (2018)
14. Hu, Y.-T., Huang, J.-B., Schwing, A.G.: VideoMatch: matching based video object segmentation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11212, pp. 56–73. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01237-3_4
15. Johnander, J., Danelljan, M., Brissman, E., Khan, F.S., Felsberg, M.: A generative appearance model for end-to-end video object segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
16. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. In: International Conference on Learning Representations, December 2014
17. Lee, K., Maji, S., Ravichandran, A., Soatto, S.: Meta-learning with differentiable convex optimization. In: CVPR (2019)
18. Lin, H., Qi, X., Jia, J.: Agss-vos: attention guided single-shot video object segmentation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3949–3957 (2019)
19. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
20. Liu, Y., Liu, L., Zhang, H., Rezatofighi, H., Reid, I.: Meta learning with differentiable closed-form solver for fast video object segmentation. arXiv preprint arXiv:1909.13046 (2019)
21. Luiten, J., Voigtlaender, P., Leibe, B.: PReMVOS: proposal-generation, refinement and merging for video object segmentation. In: Jawahar, C.V., Li, H., Mori, G., Schindler, K. (eds.) ACCV 2018. LNCS, vol. 11364, pp. 565–580. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-20870-7_35

22. Maninis, K.K., et al.: Video object segmentation without temporal information. IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **41**(6), 1515–1530 (2018)
23. Massa, F., Girshick, R.: maskrcnn-benchmark: Fast, modular reference implementation of Instance Segmentation and Object Detection algorithms in PyTorch. https://github.com/facebookresearch/maskrcnn-benchmark (2018). Accessed 04 Sep 2019
24. Oh, S.W., Lee, J.Y., Sunkavalli, K., Kim, S.J.: Fast video object segmentation by reference-guided mask propagation. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7376–7385. IEEE (2018)
25. Oh, S.W., Lee, J.Y., Xu, N., Kim, S.J.: Video object segmentation using space-time memory networks. In: Proceedings of the IEEE International Conference on Computer Vision (2019)
26. Park, E., Berg, A.C.: Meta-tracker: fast and robust online adaptation for visual object trackers. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 569–585 (2018)
27. Perazzi, F., Pont-Tuset, J., McWilliams, B., Van Gool, L., Gross, M., Sorkine-Hornung, A.: A benchmark dataset and evaluation methodology for video object segmentation. In: Computer Vision and Pattern Recognition (2016)
28. Perazzi, F., Khoreva, A., Benenson, R., Schiele, B., Sorkine-Hornung, A.: Learning video object segmentation from static images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2663–2672 (2017)
29. Robinson, A., Lawin, F.J., Danelljan, M., Khan, F.S., Felsberg, M.: Learning fast and robust target models for video object segmentation (2020)
30. Ros, G., Ramos, S., Granados, M., Bakhtiary, A., Vazquez, D., Lopez, A.M.: Vision-based offline-online perception paradigm for autonomous driving. In: 2015 IEEE Winter Conference on Applications of Computer Vision, pp. 231–238. IEEE (2015)
31. Saleh, K., Hossny, M., Nahavandi, S.: Kangaroo vehicle collision detection using deep semantic segmentation convolutional neural network. In: 2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA), pp. 1–7. IEEE (2016)
32. Voigtlaender, P., Leibe, B.: Online adaptation of convolutional neural networks for video object segmentation. In: BMVC (2017)
33. Voigtlaender, P., Leibe, B.: Feelvos: fast end-to-end embedding learning for video object segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
34. Voigtlaender, P., Luiten, J., Torr, P.H., Leibe, B.: Siam r-cnn: visual tracking by re-detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
35. Vondrick, C., Shrivastava, A., Fathi, A., Guadarrama, S., Murphy, K.: Tracking emerges by colorizing videos. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11217, pp. 402–419. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01261-8_24
36. Wang, Q., Zhang, L., Bertinetto, L., Hu, W., Torr, P.H.: Fast online object tracking and segmentation: a unifying approach. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1328–1338 (2019)
37. Wang, Z., Xu, J., Liu, L., Zhu, F., Shao, L.: Ranet: ranking attention network for fast video object segmentation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3978–3987 (2019)

38. Xu, N., et al.: YouTube-VOS: sequence-to-sequence video object segmentation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11209, pp. 603–619. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01228-1_36
39. Xu, N., et al.: Youtube-vos: A large-scale video object segmentation benchmark. arXiv preprint arXiv:1809.03327 (2018)
40. Yang, L., Wang, Y., Xiong, X., Yang, J., Katsaggelos, A.K.: Efficient video object segmentation via network modulation. Algorithms **29**, 15 (2018)