Decoupling Features in Hierarchical Propagation for Video Object Segmentation

Zongxin Yang[1,2], **Yi Yang**[1†]

[1] CCAI, College of Computer Science and Technology, Zhejiang University [2] Baidu Research
{yangzongxin, yangyics}@zju.edu.cn

Abstract

This paper focuses on developing a more effective method of hierarchical propagation for semi-supervised Video Object Segmentation (VOS). Based on vision transformers, the recently-developed AOT approach introduces hierarchical propagation into VOS and has shown promising results. The hierarchical propagation can gradually propagate information from past frames to the current frame and transfer the current frame feature from object-agnostic to object-specific. However, the increase of object-specific information will inevitably lead to the loss of object-agnostic visual information in deep propagation layers. To solve such a problem and further facilitate the learning of visual embeddings, this paper proposes a Decoupling Features in Hierarchical Propagation (DeAOT) approach. Firstly, DeAOT decouples the hierarchical propagation of object-agnostic and object-specific embeddings by handling them in two independent branches. Secondly, to compensate for the additional computation from dual-branch propagation, we propose an efficient module for constructing hierarchical propagation, *i.e.*, Gated Propagation Module, which is carefully designed with single-head attention. Extensive experiments show that DeAOT significantly outperforms AOT in both accuracy and efficiency. On YouTube-VOS, DeAOT can achieve 86.0% at 22.4fps and 82.0% at 53.4fps. Without test-time augmentations, we achieve new state-of-the-art performance on four benchmarks, *i.e.*, YouTube-VOS (86.2%), DAVIS 2017 (86.2%), DAVIS 2016 (92.9%), and VOT 2020 (0.622). Project page: https://github.com/z-x-yang/AOT.

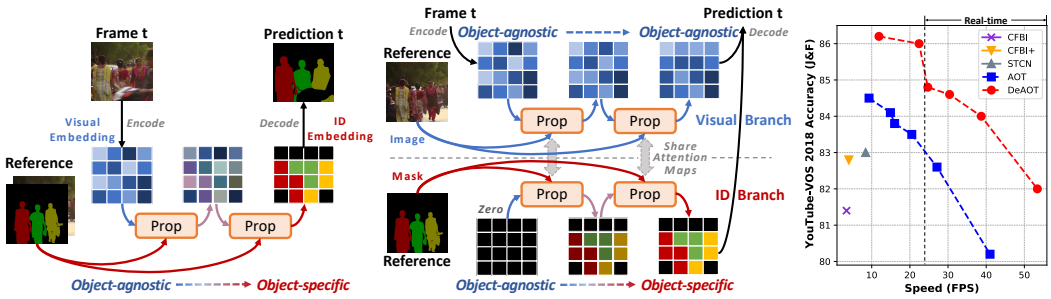
1 Introduction

Video Object Segmentation (VOS), which aims at recognizing and segmenting one or multiple objects of interest in a given video, has attracted much attention as a fundamental task of video understanding. This paper focuses on semi-supervised VOS, which requires algorithms to track and segment objects throughout a video sequence given objects' annotated masks at one or several frames.

Early VOS methods are mainly based on finetuning segmentation networks on the annotated frames [7, 32, 50] or constructing pixel-wise matching maps [10, 49]. Based on the advance of attention mechanisms [5, 47, 52], many attention-based VOS algorithms have been proposed in recent years and achieved significant improvement. STM [34] and the following works [11, 42, 43] leverage a memory network to store and read the target features of predicted past frames and apply a non-local attention mechanism to match the target in the current frame. Furthermore, AOT [61, 63] introduces hierarchical propagation into VOS based on transformers [8, 47] and can associate multiple objects

†: the corresponding author.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).



(a) AOT-like hierarchical propagation (b) Decoupling features (ours) (c) Comparison

Figure 1: (a) AOT [61] hierarchically propagates (Prop) *object-specific* information (*i.e.*, specific to the given object(s)) into the *object-agnostic* visual embedding. (b) By contrast, DeAOT decouples the propagation of visual and ID embeddings in two branches. (c) Speed-accuracy comparison. All the results were fairly recorded on the same device, 1 Tesla V100 GPU.

collaboratively by utilizing the IDentification (ID) mechanism [61]. The hierarchical propagation can gradually propagate ID information from past frames to the current frame and has shown promising VOS performance with remarkable scalability.

Fig. 1a shows that AOT's hierarchical propagation can transfer the current frame feature from an object-agnostic visual embedding to an object-specific ID embedding by hierarchically propagating the reference information into the current frame. The hierarchical structure enables AOT to be structurally scalable between state-of-the-art performance and real-time efficiency. Intuitively, the increase of ID information will inevitably lead to the loss of initial visual information since the dimension of features is limited. However, matching objects' visual features, the only clues provided by the current frame, is crucial for attention-based VOS solutions. To avoid the loss of visual information in deeper propagation layers and facilitate the learning of visual embeddings, a desirable manner (Fig. 1b) is to decouple object-agnostic and object-specific embeddings in the propagation.

Based on the above motivation, this paper proposes a novel hierarchical propagation approach for VOS, *i.e.*, Decoupling Features in Hierarchical Propagation (DeAOT). Unlike AOT, which shares the embedding space for visual (object-agnostic) and ID (object-specific) embeddings, DeAOT decouples them into different branches using individual propagation processes while sharing their attention maps. To compensate for the additional computation from the dual-branch propagation, we propose a more efficient module for constructing hierarchical propagation, *i.e.*, Gated Propagation Module (GPM). By carefully designing GPM for VOS, we are able to use single-head attention to match objects and propagate information instead of the stronger multi-head attention [47], which we found to be an efficiency bottleneck of AOT [61].

To evaluate the proposed DeAOT approach, a series of experiments are conducted on three VOS benchmarks (YouTube-VOS [56], DAVIS 2017 [38], and DAVIS 2016 [37]) and one Visual Object Tracking (VOT) benchmark (VOT 2020 [24]). On the large-scale VOS benchmark, YouTube-VOS, the DeAOT variant networks remarkably outperform AOT counterparts in both accuracy and run-time speed as shown in Fig. 1c. Particularly, our R50-DeAOT-L can achieve **86.0%** at a nearly real-time speed, **22.4fps**, and our DeAOT-T can achieve **82.0%** at **53.4fps**, which is superior compared to AOT-T [61] (80.2%, 41.0fps). Without any test-time augmentations, our SwinB-DeAOT-L achieves top-ranked performance on four VOS/VOT benchmarks, *i.e.*, YouTube-VOS 2018/2019 (**86.2%/86.1%**), DAVIS 2017 Val/Test (**86.2%/82.8%**), DAVIS 2016 (**92.9%**), and VOT 2020 (**0.622 EAO**).

Overall, our contributions are summarized below:

- We propose a highly-effective VOS framework, DeAOT, by decoupling object-agnostic and object-specific features in hierarchical propagation. DeAOT achieves top-ranked performance and efficiency on four VOS/VOT benchmarks [24, 37, 38, 56].
- We design an efficient module, GPM, for constructing hierarchical matching and propagation. By using GPM, DeAOT variants are consistently faster than AOT counterparts, although DeAOT's propagation processes are twice as AOT's.

2 Related Work

Semi-supervised Video Object Segmentation. Given a video with one or several annotated frames (the first frame in general), semi-supervised VOS [51] requires algorithms to propagate the mask annotations to the entire video. Traditional methods often solve an optimization problem with an energy defined over a graph structure [2, 4, 48]. Based on deep neural networks (DNN), deep learning based VOS methods have achieved significant progress and dominated the field in recent years.

Finetuning-based Methods. Early DNN-based methods rely on fine-tuning pre-trained segmentation networks at test time to make the networks focus on the given object. Among them, OSVOS [7] and MoNet [55] propose to fine-tune pre-trained networks on the first-frame annotation. OnAVOS [50] extends the first-frame fine-tuning by introducing an online adaptation mechanism. Following these approaches, MaskTrack [36] and PReM [32] further utilize optical flow to help propagate the segmentation mask from one frame to the next.

Template-based Methods. To avoid using the test-time fine-tuning, many researchers regard the annotated frames as templates and investigate how to match with them. For example, OSMN [59] employs a network to extract object embedding and another one to predict segmentation based on the embedding. PML [10] learns pixel-wise embedding with the nearest neighbor classifier, and VideoMatch [22] uses a matching layer to map the pixels of the current frame to the annotated frame in a learned embedding space. Following these methods, FEELVOS [49] and CFBI(+) [60, 62] extend the pixel-level matching mechanism by additionally doing local matching with the previous frame, and RPCM [57] proposes a correction module to improve the reliability of pixel-level matching. Instead of using matching mechanisms, LWL [6] proposes to use an online few-shot learner to learn to decode object segmentation.

Attention-based Methods. Based on the advance of attention mechanisms [5, 47, 52], STM [34] and the following works (*e.g.*, KMN [42] and STCN [11]) leverage a memory network to embed past-frame predictions into memory and apply a non-local attention mechanism on the memory to propagate mask information to the current frame. Differently, SST [17] proposes to calculate pixel-level matching maps based on the attention maps of transformer blocks [47]. Recently, AOT [61, 63] introduces hierarchical propagation into VOS and can associate multiple objects collaboratively with the proposed ID mechanism.

Visual Transformers. Transformers [47] was initially proposed to build hierarchical attention-based networks for natural language processing (NLP). Compared to RNNs, transformer networks model global correlation or attention in parallel, leading to better memory efficiency, and thus have been widely used in NLP tasks [15, 39, 45]. Similar to Non-local Neural Networks [52], transformer blocks compute correlation with all the input elements and aggregate their information by using attention mechanisms [5]. Recently, transformer blocks were introduced to computer vision and have shown promising performance in many tasks, such as image classification [16, 30, 46], object detection [8]/segmentation [25, 53], image generation [35], and video understanding [1, 26, 31].

Based on transformers, AOT [61] proposes a Long Short-Term Transformer (LSTT) structure for constructing hierarchical propagation. By hierarchically propagating object information, AOT variants [61] have shown promising performance with remarkable scalability. Unlike AOT, which shares the embedding space for object-agnostic and object-specific embeddings, we propose to decouple them into different branches using individual propagation processes. Such a dual-branch paradigm avoids the loss of object-agnostic information and achieves significant improvement. Besides, a more efficient structure, GPM, is proposed for hierarchical propagation.

3 Rethinking Hierarchical Propagation for VOS

Attention-based VOS methods [11, 34, 42, 61] are dominating the field of VOS. In these methods, STM [34] and following algorithms [11, 42] uses a single attention layer to propagate mask information from memorized frames to the current frame. The use of only a single attention layer restricts the scalability of algorithms. Hence, AOT [61] introduces hierarchical propagation into VOS by proposing the Long Short-term Transformer (LSTT) structure, which can propagate the mask information in a hierarchical coarse-to-fine manner. By adjusting the layer number of LSTT, AOT variants can be ranged from state-of-the-art performance to real-time run-time speed.

Let $Q \in \mathbb{R}^{HW \times C}$ and $K, V \in \mathbb{R}^{THW \times C}$ denote the query embedding of the current frame, the key embedding, and the value embedding of the memorized frames respectively, where T, H, W, C represent the temporal, height, width, and channel dimensions. The formula of a common attention-based VOS propagation is,

$$Att(Q, K, V) = Corr(Q, K)V = softmax(\frac{QK^{tr}}{\sqrt{C}})V, \tag{1}$$

where the matching (or attention) map is calculated by the correlation function, $Corr(*, *)$.

To formulate a hierarchical propagation with L layers, we further define $X_l^t \in \mathbb{R}^{HW \times C}$ as the input feature embedding of l-th propagation layer ($l \in \{1, 2, ..., L\}$) at t frame. Moreover, $X_l^{\mathbf{m}} = Concat(X_l^{m_1}, ..., X_l^{m_T})$ and $Y^{\mathbf{m}} = Concat(Y^{m_1}, ..., Y^{m_T})$ stands for the feature embeddings and object masks in the memorized frames with indices $\mathbf{m} = \{m_1, ..., m_T\}$. Then, the formulation of l-th propagation layer in AOT's hierarchical propagation can be simplified as,

$$\widetilde{X}_l^t = Att(X_l^t W_l^K, X_l^{\mathbf{m}} W_l^K, X_l^{\mathbf{m}} W_l^V + ID(Y^{\mathbf{m}})), \tag{2}$$

where $ID(*)$ denotes the IDentification (ID) embedding [61] function used to encode masks. Besides, $W_l^K \in \mathbb{R}^{C \times C_k}$ and $W_l^V \in \mathbb{R}^{C \times C_v}$ are trainable parameters for projecting features into matching space and propagation space, respectively. For simplicity, the formulation keeps only the parts related to mask propagation in LSTT.

Obviously, before all the propagation layers, the current frame feature, X_1^t, is an object-agnostic feature extracted from an image encoder (*e.g.*, ResNet-50 [21]). Nevertheless, the mask information $ID(Y^{\mathbf{m}})$ will be gradually and hierarchically propagated into the current frame, and the output feature, $\widetilde{X}_L^t$, will become object-specific and can be decoded into the ID/mask prediction by a decoder network (*e.g.*, FPN [27]). In other words, step by step, the hierarchical propagation transfers the current frame feature, X_l^t, from an object-agnostic visual embedding to an object-specific ID embedding, as demonstrated in Fig. 1a.

Intuitively, the absorption of object-specific ID information will inevitably lead to the oblivion of object-agnostic visual information within X_1^t since the channel dimension of X_l^t is limited. Such a phenomenon can also be observed by increasing the ID information directly. As shown in Fig. 2, the performance of AOT heavily drops as we increase the information amount of $ID(Y^{\mathbf{m}})$ by containing more IDs inside. On the other hand, the significant progress of VOS in recent years is mainly based on matching object-agnostic visual embeddings (*e.g.*, pixel-level matching methods [57, 60, 62] and single-layer attention-based methods [11, 34, 42] mentioned above). Hence, we argue that the loss of visual information in deeper propagation layers limits the performance of hierarchical propagation.

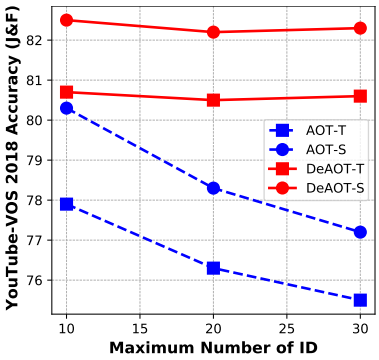
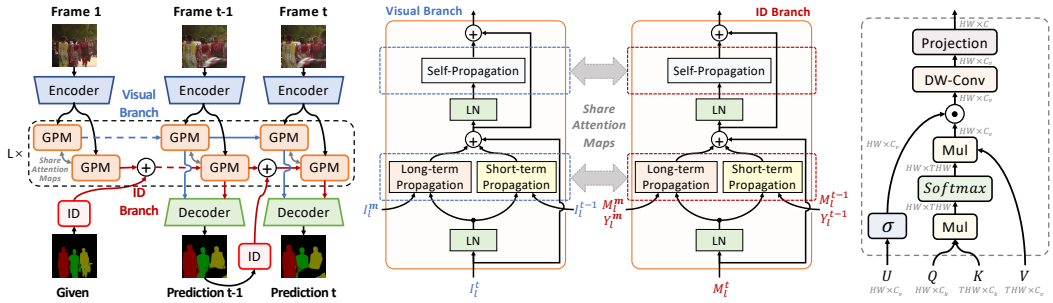


Figure 2: The performance of AOT [61] will be degraded by increasing ID's maximum number.

How to design a hierarchical propagation structure which can keep or even refine the initial object-agnostic visual information? Fig. 1b shows a simple, straightforward, and desirable approach, *i.e.*, propagating object-agnostic and object-specific information in two different branches (Visual Branch and ID Branch). The object-agnostic branch is responsible for gathering visual information, refining visual features, and matching objects. By contrast, the object-specific branch is responsible for absorbing ID information propagated from memorized frames. These two branches share the attention maps used to match objects and propagate features. Compared to the single-branch LSTT, our dual-branch approach can keep and further refine visual features in the hierarchical propagation and thus can further facilitate the learning of visual embeddings.

4 Decoupling Features in Hierarchical Propagation

This section will introduce a new framework, Decoupling Features in Hierarchical Propagation (DeAOT), for solving semi-supervised video object segmentation. We show an overview of DeAOT in Fig. 3a. Given a video with a reference frame annotation, DeAOT propagates the annotation to the entire video frame-by-frame. The multi-object annotation is encoded by the IDentification



(a) Overview (b) Gated Propagation Module (GPM) (c) GP function

Figure 3: (a) Overview. Decoupling Features in Hierarchical Propagation (DeAOT) decouples the propagation of visual embedding and IDentification (ID) embedding [61] in two branches, *i.e.*, Visual Branch and ID Branch. The propagation module is the proposed efficient GPM module. (b) A demonstration of the Gated Propagation Module (GPM) in both Visual and ID branches. LN: Layer Normalization [3]. (c) We propose to use the Gated Propagation (GP) function to construct GPM. DW-Conv: depth-wise convolution. Mul: matrix multiplication.

(ID) mechanism [61]. Different from AOT, DeAOT decouples the hierarchical propagation of visual embedding and ID embedding, *i.e.*, DeAOT propagates these two embeddings in two branches. Furthermore, DeAOT constructs the hierarchical propagation by using the proposed Gated Propagation Module (GPM), which is more efficient and effective than the LSTT block used in AOT.

4.1 Hierarchical Dual-branch Propagation

Different from the previous attention-based VOS methods [34,42,43,61], DeAOT propagates objects' visual features and mask features in two parallel branches. In detail, the visual branch is responsible for matching objects, gathering past visual information, and refining object features. To re-identify the objects, the ID branch reuses the matching maps (attention maps) calculated by the visual branch to propagate the ID embedding (encoded by the ID mechanism [61]) from past frames to the current frame. Both the branches share the same hierarchical structure with L propagation layers.

Visual Branch is responsible for matching objects by calculating attention maps on patch-wise visual embeddings. The visual embeddings in the memorized frames will be propagated to the current frame regarding the attention maps. Since the propagation is not directly related to the object-specific ID embedding, the visual branch can learn to refine visual embeddings to be more contrastive but avoid being biased toward the given object-specific information. Let I denote visual embeddings, we modify Eq. 2 into a layer of object-agnostic visual propagation,

$$\begin{aligned} \widetilde{I}_l^t &= Att(I_l^t W_l^K, I_l^{\mathbf{m}} W_l^K, I_l^{\mathbf{m}} W_l^V) \\ &= Corr(I_l^t W_l^K, I_l^{\mathbf{m}} W_l^K) I_l^{\mathbf{m}} W_l^I, \end{aligned} \tag{3}$$

which doesn't leverage the object-specific ID embedding, $ID(Y^{\mathbf{m}})$. Thus, the visual branch can learn to keep and refine the visual embedding in the hierarchical propagation.

ID Branch is designed for propagating the object-specific information from past frames to the current frame. The prediction of object-specific segmentation is essential for VOS and can not be processed by the above object-agnostic visual propagation branch. Let M denote the object-specific embeddings in our identification branch, the formulation of our object-specific ID propagation is,

$$\begin{aligned} \widetilde{M}_l^t &= Att(I_l^t W_l^K, I_l^{\mathbf{m}} W_l^K, M_l^{\mathbf{m}} W_l^{\overline{V}} + ID(Y^{\mathbf{m}})) \\ &= Corr(I_l^t W_l^K, I_l^{\mathbf{m}} W_l^K)(M_l^{\mathbf{m}} W_l^{\overline{V}} + ID(Y^{\mathbf{m}})), \end{aligned} \tag{4}$$

where $W_l^{\overline{V}} \in \mathbb{R}^{C \times C_v}$ is a trainable projection matrix for the identification propagation. Particularly, the identification propagation shares the same attention maps, $Corr(I_l^t W_l^K, I_l^{\mathbf{m}} W_l^K)$, from the visual branch, since the identification of objects is mainly based on objects' visual features instead of their ID indices. Without the visual information, the tracking of objects is inapplicable.

4.2 Gated Propagation Module

Instead of using the LSTT block [61], which employs multi-head attention in propagation, we stack the hierarchical propagation based on the proposed Gated Propagation Module (GPM), which is designed based on more efficient single-head attention.

LSTT Block [61] includes four parts, *i.e.*, a long-term attention responsible for propagating information from the memorized frames (in $\mathbf{m}$), a short-term attention responsible for propagating information from a spatial neighborhood in the previous $(t-1)$ frame, a self-attention module for associating objects in the current (t) frame, and a feed-forward module. The three kinds of attention modules are built on the multi-head [47] extension of Eq. 1 or Eq. 2. According to the experiments in Table 3b, reducing the head number from multiple heads (8 heads in default) to a single head will decrease the performance of AOT but can significantly improve the run-time speed, which means the multi-head attention is an efficiency bottleneck of LSTT. Concretely, the computational complexity of long-term attention is $\mathcal{O}(NTH^2W^2)$, which is proportional to the head number N since each head contains a correlation function, $Corr(Q, K)$.

Gated Propagation Function. To avoid using multiple attention heads but not decrease the network performance, we redesign the attention-based VOS propagation defined in Eq. 1 and propose a gated propagation function as demonstrated in Fig. 3c. Let $U \in \mathbb{R}^{HW \times C}$ denotes a gating embedding, the function is

$$GP(U, Q, K, V) = \mathcal{F}_{dw}(\sigma(U) \odot Corr(Q, K)V)W^O, \tag{5}$$

where σ is a non-linear gating function, $\odot$ denotes element-wise multiplication, $\mathcal{F}_{dw}(*)$ stands for a depth-wise 2D convolution layer [13], and $W^O \in \mathbb{R}^{C_v \times C}$ is the trainable weight of output projection. Firstly, we augment the attention-based propagation (Eq. 1) by using a conditional gate, $\sigma(U)$, which we empirically found to be effective in VOS. Notably, the presence of gating in weak attention mechanisms (*e.g.*, single-head attention) is also beneficial in some transformer-based methods [23, 29] for NLP. Moreover, we leverage a depth-wise convolution $\mathcal{F}_{dw}(*)$ to enhance the modeling of local spatial context in a lightweight manner.

Gated Propagation Module consists of three kinds of gated propagation, self-propagation, long-term propagation, and short-term propagation. Compared with LSTT, GPM removes the feed-forward module for further saving computation and parameters. All the propagation processes employ the gated propagation function defined in Eq. 5. In DeAOT, both the propagation branches (*i.e.*, visual branch and identification branch) are stacked by GPM as shown in Fig. 3b.

Based on the formulation of visual propagation (Eq. 3) and ID propagation (Eq. 4), the **Long-term Propagation** can be formulated as

$$GP^{vis}_{lt}(I^t_l, I^t_l, I^{\mathbf{m}}_l, I^{\mathbf{m}}_l) = GP(I^t_l W^U_l, I^t_l W^K_l, I^{\mathbf{m}}_l W^K_l, I^{\mathbf{m}}_l W^V_l), \tag{6}$$

$$GP^{id}_{lt}(M^t_l, I^t_l, I^{\mathbf{m}}_l, M^{\mathbf{m}}_l, Y^{\mathbf{m}}) = GP(M^t_l W^{\overline{U}}_l, I^t_l W^K_l, I^{\mathbf{m}}_l W^K_l, M^{\mathbf{m}}_l W^{\overline{V}}_l + ID(Y^{\mathbf{m}})) \tag{7}$$

for the visual branch and ID branch, respectively. The ID propagation reuses the attention maps of the visual propagation as discussed in Eq. 4. Based on the long-term propagation, we can formulate the **Short-term Propagation** at spatial location p to be

$$GP^{vis}_{st}(I^t_l, I^t_l, I^{t-1}_l, I^{t-1}_l|p) = GP^{vis}_{lt}(I^t_{l,p}, I^t_{l,p}, I^{t-1}_{l,\mathcal{N}(p)}, I^{t-1}_{l,\mathcal{N}(p)}), \tag{8}$$

$$GP^{id}_{st}(M^t_l, I^t_l, I^{t-1}_l, M^{t-1}_l, Y^{t-1}|p) = GP^{id}_{lt}(M^t_{l,p}, I^t_{l,p}, I^{t-1}_{l,\mathcal{N}(p)}, M^{t-1}_{l,\mathcal{N}(p)}|Y^{t-1}_{\mathcal{N}(p)}), \tag{9}$$

where $I^t_{l,p}, M^t_{l,p} \in \mathbb{R}^{1 \times C}$ are the feature of I^t_l, M^t_l at location p respectively, and $\mathcal{N}(p)$ stands for a $\lambda \times \lambda$ spatial neighbourhood centered at location p. The short-term propagation for each location p is restricted in its spatial neighbourhood ($I^{t-1}_{l,\mathcal{N}(p)}$ or $M^{t-1}_{l,\mathcal{N}(p)}$) of the previous $(t-1)$ frame. Since the object motions across several contiguous video frames are always smooth, non-local propagation processes becomes inefficient and not necessary in short-term information propagation [60].

Finally, the **Self-Propagation** can also be formulated similar to the long-term propagation, *i.e.*,

$$GP^{vis}_{self}(I^t_l|M^t_l) = GP(I^t_l W^U_l, (I^t_l \oplus M^t_l)W^K_l, (I^t_l \oplus M^t_l)W^K_l, I^t_l W^V_l), \tag{10}$$

$$GP^{id}_{self}(M^t_l|I^t_l) = GP(M^t_l W^{\overline{U}}_l, (I^t_l \oplus M^t_l)W^K_l, (I^t_l \oplus M^t_l)W^K_l, M^t_l W^{\overline{V}}_l), \tag{11}$$

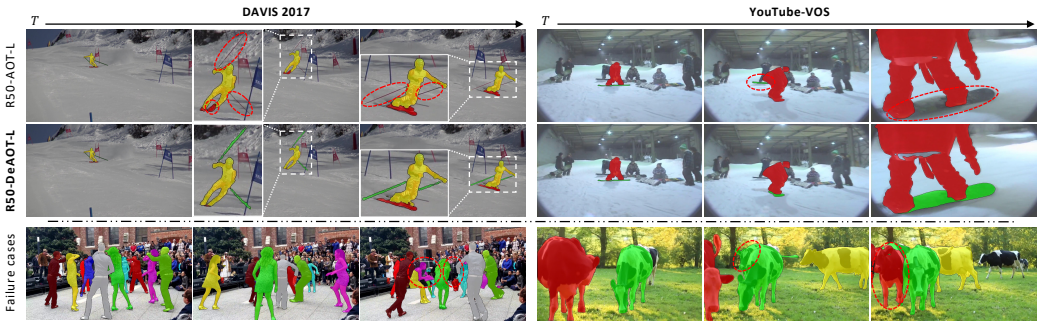


Figure 4: Qualitative results. (top) DeAOT performs better than AOT [61] on tiny or scale-changing objects. (bottom) DeAOT fails to track highly similar objects when serious occlusion happens.

where $\oplus$ is a concatenation process on the channel dimension. In the self-propagations, both the visual embedding I_l^t and ID embedding M_l^t are used in the calculation of attention maps (*i.e.*, $Corr(Q, K)$). Here, the object-specific M_l^t performs like a positional embedding [47] additional to the visual embedding I_l^t. We found that such a process can help associate the objects in the current frame more effectively. Apart from this, the current frame segmentation Y^t is unavailable before being decoded and is not used in the ID self-propagation GP_{self}^{id}. For simplicity, we reuse the parameter symbols in Eq. 6 and 7, but the trainable parameters are not shared with long-term propagation.

5 Implementation Details

Network Details: Consistent with AOT [61], three kinds of encoders are used in our experiments, *i.e.*, MobileNet-V2 [41] (in default), ResNet-50 (R50) [21], and Swin-B [30]. The decoder is the same FPN [27] network. Besides, the spatial neighborhood size λ is set to 15, and the maximum object number within the ID embedding is 10. In our GPM module, the channel dimension C of visual and ID embeddings is 256, the matching features' dimension C_k is 128, and the propagation features' dimension C_v is 512. Moreover, the kernel size of $\mathcal{F}_{dw}$ is 5, and the gating function $\sigma(*)$ is SiLU/Swish [18, 40].

To make fair comparisons with AOT's variants [61], we build corresponding DeAOT variants with different GPM number L or long-term memory size $\mathbf{m}$. The hyper-parameters of these variants are: **DeAOT-T**: $L = 1$, $\mathbf{m} = \{1\}$; **DeAOT-S**: $L = 2$, $\mathbf{m} = \{1\}$; **DeAOT-B**: $L = 3$, $\mathbf{m} = \{1\}$; **DeAOT-L**: $L = 3$, $\mathbf{m} = \{1, 1+\delta, 1+2\delta, ...\}$. DeAOT-T/S/B considers only the reference frame as the long-term memory, leading to consistent run-time speeds. DeAOT-L updates the long-term memory per δ (set to 2/5 for training/testing) frames as AOT-L [61].

Training Details: Following [34, 42, 43, 54, 61], we first pre-train DeAOT on synthetic video sequence generated from static image datasets [12, 19, 20, 28, 44] by randomly applying multiple image augmentations [54]. Then, we do main training on the VOS benchmarks [38, 56] by randomly applying video augmentations [60, 61]. Besides, we keep our optimization strategies and related hyper-parameters the same as AOT. More details are supplied in Supplementary.

6 Experimental Results

We conduct experiments on three popular VOS benchmarks (YouTube-VOS [56], DAVIS 2017 [38], and DAVIS 2016 [37]) and one challenging Visual Object Tracking (VOT) benchmark (VOT 2020 [24]), which gives segmentation annotations and can be used to evaluate VOS algorithms.

To validate DeAOT's generalization ability, all the benchmarks share the same model parameters. When evaluating YouTube-VOS, we use the default 6fps videos, which are restricted to be smaller than $1.3 \times 480p$ resolution. On DAVIS, the default 480p 24fps videos are used. For evaluating VOT 2020, more details can be found in the supplementary material.

The evaluation metrics for VOS benchmarks include the $\mathcal{J}$ score (calculated as the average IoU score between the prediction and the ground truth mask), the $\mathcal{F}$ score (calculated as an average boundary

Table 1: The quantitative evaluation on multi-object benchmarks, YouTube-VOS [56] and DAVIS 2017 [38]. $\mathcal{J}_S/\mathcal{F}_S/\mathcal{J}_U/\mathcal{F}_U$: $\mathcal{J}/\mathcal{F}$ on seen/unseen classes. ‡: timing extrapolated from single-object speed assuming linear scaling in the number of objects. ⋆: recorded on our device.

	YouTube-VOS 2018 Val					YouTube-VOS 2019 Val						DAVIS-17 Val			DAVIS-17 Test			
Method	Avg	$\mathcal{J}_S$	$\mathcal{F}_S$	$\mathcal{J}_U$	$\mathcal{F}_U$	Avg	$\mathcal{J}_S$	$\mathcal{F}_S$	$\mathcal{J}_U$	$\mathcal{F}_U$	fps	Avg	$\mathcal{J}$	$\mathcal{F}$	Avg	$\mathcal{J}$	$\mathcal{F}$	fps
KMN[ECCV20] [42]	81.4	81.4	85.6	75.3	83.3	-	-	-	-	-	-	82.8	80.0	85.6	77.2	74.1	80.3	-
CFBI[ECCV20] [60]	81.4	81.1	85.8	75.3	83.4	81.0	80.6	85.1	75.2	83.0	3.4	81.9	79.3	84.5	76.6	73.0	80.1	2.9
SST[CVPR21] [17]	81.7	81.2	-	76.0	-	81.8	80.9	-	76.6	-	-	82.5	79.9	85.1	-	-	-	-
HMMN[ICCV21] [43]	82.6	82.1	87.0	76.8	84.6	82.5	81.7	86.1	77.3	85.0	-	84.7	81.9	87.5	78.6	74.7	82.5	3.4‡
CFBI+[TPAMI21] [62]	82.8	81.8	86.6	77.1	85.6	82.6	81.7	86.2	77.1	85.2	4.0	82.9	80.1	85.7	78.0	74.4	81.6	3.4
STCN[NeurIPS21] [11]	83.0	81.9	86.5	77.9	85.7	82.7	81.1	85.4	78.2	85.9	8.4⋆	85.4	82.2	88.6	76.1	72.7	79.6	19.5⋆
RPCM[AAAI22] [57]	84.0	83.1	87.7	78.5	86.7	83.9	82.6	86.9	79.1	87.1	-	83.7	81.3	86.0	79.2	75.8	82.6	-
AOT-T [61]	80.2	80.1	84.5	74.0	82.2	79.7	79.6	83.8	73.7	81.8	41.0	79.9	77.4	82.3	72.0	68.3	75.7	51.4
DeAOT-T	**82.0**	**81.6**	**86.3**	**75.8**	**84.2**	**82.0**	**81.2**	**85.6**	**76.4**	**84.7**	**53.4**	**80.5**	**77.7**	**83.3**	**73.7**	**70.0**	**77.3**	**63.5**
AOT-S [61]	82.6	82.0	86.7	76.6	85.0	82.2	81.3	85.9	76.6	84.9	27.1	**81.3**	**78.7**	**83.9**	73.9	70.3	77.5	40.0
DeAOT-S	**84.0**	**83.3**	**88.3**	**77.9**	**86.6**	**83.8**	**82.8**	**87.5**	**78.1**	**86.8**	**38.7**	80.8	77.8	83.8	**75.4**	**71.9**	**79.0**	**49.2**
AOT-B [61]	83.5	82.6	87.5	77.7	86.0	83.3	82.4	87.1	77.8	86.0	20.5	**82.5**	**79.7**	**85.2**	75.5	71.6	79.3	29.6
DeAOT-B	**84.6**	**83.9**	**88.9**	**78.5**	**87.0**	**84.6**	**83.5**	**88.3**	**79.1**	**87.5**	**30.4**	82.2	79.2	85.1	**76.2**	**72.5**	**79.9**	**40.9**
AOT-L [61]	83.8	82.9	87.9	77.7	86.5	83.7	82.8	87.5	78.0	**86.7**	16.0	83.8	**81.1**	86.4	**78.3**	**74.3**	**82.3**	18.7
DeAOT-L	**84.8**	**84.2**	**89.4**	**78.6**	**87.0**	**84.7**	**83.8**	**88.8**	**79.0**	**87.2**	**24.7**	**84.1**	81.0	**87.1**	77.9	74.1	81.7	**28.5**
R50-AOT-L [61]	84.1	83.7	88.5	78.1	86.1	84.1	83.5	88.1	**78.4**	86.3	14.9	84.9	**82.3**	87.5	79.6	75.9	83.3	18.0
R50-DeAOT-L	**86.0**	**84.9**	**89.9**	**80.4**	**88.7**	**85.9**	**84.6**	**89.4**	**80.8**	**88.9**	**22.4**	**85.2**	82.2	**88.2**	**80.7**	**76.9**	**84.5**	**27.0**
SwinB-AOT-L [61]	84.5	84.3	89.3	77.9	86.4	84.5	84.0	88.8	78.4	86.7	9.3	85.4	82.4	88.4	81.2	77.3	85.1	12.1
SwinB-DeAOT-L	**86.2**	**85.6**	**90.6**	**80.0**	**88.4**	**86.1**	**85.3**	**90.2**	**80.4**	**88.6**	**11.9**	**86.2**	**83.1**	**89.2**	**82.8**	**78.9**	**86.7**	**15.4**

similarity measure between the boundary of the prediction and the ground truth), and their mean value (denoted as $\mathcal{J}\&\mathcal{F}$). As to VOT 2020, we use the official EAO criteria [24]. We evaluate all the results on official evaluation servers or with official tools.

6.1 Compare with the State-of-the-art Methods

YouTube-VOS [56] is a large-scale multi-object VOS benchmark, which contains 3471 videos in the training split with 65 categories and 474/507 videos in the Validation 2018/2019 split with additional 26 unseen categories. Table 1 shows that DeAOT variants remarkably outperforms AOT counterparts in both accuracy and run-time speed on YouTube-VOS 2018/2019. For example, our R50-DeAOT-L achieves **86.0%/85.9%** ($\mathcal{J}\&\mathcal{F}$) at **22.4fps**, which is superior compared to R50-AOT-L [61] (84.1%/84.1% at 14.9fps). Particularly, our SwinB-DeAOT-L achieves new state-of-the-art performance (**86.2%/86.1%**), surpassing previous methods by more than 1.7%/1.6%. In addition, our smallest variant, DeAOT-T, precedes SST [17] (**82.0%/82.0%** *vs* 81.7%/81.8%) and runs about **15**× faster than CFBI [60] (**53.4fps** *vs* 3.4fps).

DAVIS 2017 [38] is a multi-object extension of DAVIS 2016. The training/validation split consists of 60/30 videos with 138/59 objects, and the test split contains 30 more challenging videos with 89 objects. As shown in Table 1, DeAOT variants can generalize to DAVIS 2017 well. R50-DeAOT-L achieves **85.2%/80.7%** on the validation/test split at a real-time speed (**27fps**), surpassing R50-AOT-L in accuracy and efficiency. Also, SwinB-DeAOT-L achieves the top-ranked performance on DAVIS 2017 (**86.2%/82.8%**).

DAVIS 2016 [37] is a single-object benchmark containing 20 videos in the validation split, and we show related experiments in Table 2. Although AOT-like methods focus on multi-object scenarios, our DeAOT-L is faster and more robust than STCN [11], whose architecture was designed for single-object VOS. Besides, SwinB-DeAOT-L achieves **92.9%** and outperforms all the VOS methods as well.

Table 2: The quantitative evaluation on the single-object benchmarks, DAVIS 2016 [37] and VOT 2020 [24]. EAO^{RT}: real-time EAO metric [24].

	DAVIS 2016				VOT 2020	
Method	Avg	$\mathcal{J}$	$\mathcal{F}$	fps	EAO	EAO^{RT}
CFBI+ [62]	89.9	88.7	91.1	5.9	-	-
RPCM [57]	90.6	87.1	94.0	5.8	-	-
HMMN [43]	90.8	89.6	92.0	10.0	-	-
STCN [11]	91.6	90.8	92.5	27.2⋆	-	-
AlphaRef [58]	-	-	-	-	0.482	0.486
RPT [33]	-	-	-	-	0.530	0.290
MixFormer-L [14]	-	-	-	-	0.555	-
AOT-T [61]	86.8	86.1	87.4	51.4	0.435	0.433
DeAOT-T	**88.9**	**87.8**	**89.9**	**63.5**	**0.472**	**0.463**
AOT-S [61]	**89.4**	**88.6**	90.2	40.0	0.512	0.499
DeAOT-S	89.3	87.6	**90.9**	**49.2**	**0.593**	**0.559**
AOT-B [61]	89.9	88.7	91.1	29.6	0.541	0.533
DeAOT-B	**91.0**	**89.4**	**92.5**	**40.9**	**0.571**	**0.542**
AOT-L [61]	90.4	89.6	91.1	18.7	0.574	0.560
DeAOT-L	**92.0**	**90.3**	**93.7**	**28.5**	**0.591**	**0.554**
R50-AOT-L [61]	91.1	90.1	92.1	18.0	0.569	0.540
R50-DeAOT-L	**92.3**	**90.5**	**94.0**	**27.0**	**0.613**	**0.571**
SwinB-AOT-L [61]	92.0	90.7	93.3	12.1	0.586	0.523
SwinB-DeAOT-L	**92.9**	**91.1**	**94.7**	**15.4**	**0.622**	**0.559**

Table 3: Ablation study. The experiments are conducted on YouTube-VOS 2018 [56] and based on DeAOT-S without pre-training on static images. De: decoupling features. C: the channel dimension. Prop: propagation type. LT/ST: long-term/short-term. ks: kernel size.

(a) Propagation module

Module	C	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}_S$	$\mathcal{J}_U$
GPM	**256**	**82.5**	**82.3**	**76.1**
w/o De	256	81.5	81.4	75.0
w/o De	512	82.0	82.1	75.4
LSTT	256	80.3	80.6	73.7

(b) Head number (N_h)

Model	N_h	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}_S$	$\mathcal{J}_U$	fps
DeAOT	**1**	**82.5**	**82.3**	**76.1**	38.7
DeAOT	8	**82.5**	**82.3**	75.8	24.7
AOT	1	79.6	80.1	72.6	**44.6**
AOT	8	80.3	80.6	73.7	27.1

(c) Attention map

Prop	Vis	ID	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}_S$	$\mathcal{J}_U$
LT/ST	✓		**82.5**	**82.3**	**76.1**
LT/ST	✓	✓	82.1	82.2	75.7
Self	✓	✓	**82.5**	**82.3**	**76.1**
Self	✓		82.2	82.1	75.7

(d) ks of $\mathcal{F}_{dw}$

ks	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}_S$	$\mathcal{J}_U$
5	**82.5**	**82.3**	**76.1**
0	81.1	81.5	74.2
3	82.2	82.2	76.1
9	82.4	82.2	75.8

VOT 2020 [24] consists of 60 single-object videos with challenging scenarios including fast motion, occlusion, etc. The average frame number of VOT 2020 is 327, which is much longer than the maximum video length of the above VOS benchmarks. DeAOT shows superior performance on VOT 2020 in Table 2. The DeAOT variants larger than DeAOT-T outperform MixFormer-L [14] (the state-of-the-art tracker), RPT [33] (VOT 2020 short-term challenge winner), and AlphaRef [58] (VOT 2020 real-time challenge winner) in both EAO and real-time EAO scores. Specifically, SwinB-DeAOT-L achieves **0.622** EAO, outstandingly exceeding MixFormer-L by **0.067**, and R50-DeAOT-L achieves **0.571** EAO under a **real-time** requirement, impressively overtaking AlphaRef by **0.085**.

Qualitative results: Fig. 4 give qualitative comparisons to AOT. By introducing the dual-branch propagation, R50-DeAOT-L performs better than R50-AOT-L on tiny or scale-changing objects (*ski poles* or *ski board*). Nevertheless, R50-DeAOT-L still may fails to track multiple highly similar objects (*dancer* and *cow*) when serious occlusion happens.

6.2 Ablation Study

This section analyzes the necessity of dual-branch propagation and GPM of DeAOT in Table 3.

Propagation module: Table 3a shows that the performance of DeAOT drops from 82.5% to 81.5% by coupling the propagation of visual and ID embeddings (*w/o* De) like AOT. Furthermore, doubling the channel dimensions only partially relieves the performance loss. Moreover, the performance will be seriously degraded to 80.3% by replacing our GPM with the LSTT module of AOT. In conclusion, the dual-branch propagation approach and the GPM module are crucial in improving VOS performance.

Head number: According to the results in Table 3b, the head number (N_h) of attention-based modules is negatively correlated with the efficiency of AOT/De-AOT. The single-head AOT (44.6fps) runs much faster than the default AOT (N_h=8, 27.1fps) but loses 0.7% accuracy. By contrast, DeAOT is robust to the head number by using our proposed GPM module.

Attention map: Our DeAOT shares the attention maps between two propagation branches. Table 3c shows the study of different kinds of attention maps. Concretely, visual embeddings are essential in building attention maps in the long-term/short-term propagation, whose attention maps are used to match objects. Introducing ID embeddings does not help learn better visual embeddings and will decrease the performance (82.5% *vs* 82.1%). In the self-propagation, however, utilizing the ID embedding as a positional embedding will facilitate the association of objects (82.2% *vs* 82.5%) in the current frame.

Kernel size of $\mathcal{F}_{dw}$: Large receptive fields have been proved to be critical in segmentation-related tasks [9]. The depth-wise convolution, $\mathcal{F}_{dw}$, is an important part of GPM for enlarging the receptive fields. Without $\mathcal{F}_{dw}$, the performance of DeAOT drops from 82.5% to 81.1%, as shown in Table 3d. We empirically found the best kernel size of $\mathcal{F}_{dw}$ is 5 among $\{3, 5, 9\}$.

7 Conclusion

This paper proposes a highly effective and efficient framework, Decoupling Features in Hierarchical Propagation (DeAOT), for video object segmentation. Based on the rethinking of AOT-like hierarchical propagation, we propose to decouple the propagation of visual and ID embeddings into two network branches and thus avoid the loss of visual information in deep propagation layers. Besides,

we propose the Gated Propagation Module (GPM), an efficient module for constructing hierarchical VOS propagation. Applying GPM to the dual-branch propagation, our DeAOT variant networks achieve new state-of-the-art performance on four VOS/VOT benchmarks with superior run-time speed compared to previous solutions.

Acknowledgements. This work is partly supported by the Fundamental Research Funds for the Central Universities (No. 226-2022-00051).

References

[1] Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6836–6846 (2021)

[2] Avinash Ramakanth, S., Venkatesh Babu, R.: Seamseg: Video object segmentation using patch seams. In: CVPR. pp. 376–383 (2014)

[3] Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. In: NIPS Workshops (2016)

[4] Badrinarayanan, V., Galasso, F., Cipolla, R.: Label propagation in video sequences. In: CVPR. pp. 3265–3272. IEEE (2010)

[5] Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: ICLR (2015)

[6] Bhat, G., Lawin, F.J., Danelljan, M., Robinson, A., Felsberg, M., Van Gool, L., Timofte, R.: Learning what to learn for video object segmentation. In: ECCV (2020)

[7] Caelles, S., Maninis, K.K., Pont-Tuset, J., Leal-Taixé, L., Cremers, D., Van Gool, L.: One-shot video object segmentation. In: CVPR. pp. 221–230 (2017)

[8] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: ECCV. pp. 213–229. Springer (2020)

[9] Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: ECCV. pp. 801–818 (2018)

[10] Chen, Y., Pont-Tuset, J., Montes, A., Van Gool, L.: Blazingly fast video object segmentation with pixel-wise metric learning. In: CVPR. pp. 1189–1198 (2018)

[11] Cheng, H.K., Tai, Y.W., Tang, C.K.: Rethinking space-time networks with improved memory coverage for efficient video object segmentation. In: NeurIPS (2021)

[12] Cheng, M.M., Mitra, N.J., Huang, X., Torr, P.H., Hu, S.M.: Global contrast based salient region detection. TPAMI **37**(3), 569–582 (2014)

[13] Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: CVPR. pp. 1251–1258 (2017)

[14] Cui, Y., Cheng, J., Wang, L., Wu, G.: Mixformer: End-to-end tracking with iterative mixed attention. In: CVPR (2022)

[15] Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: NAACL. pp. 4171—-4186 (2019)

[16] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021)

[17] Duke, B., Ahmed, A., Wolf, C., Aarabi, P., Taylor, G.W.: Sstvos: Sparse spatiotemporal transformers for video object segmentation. In: CVPR (2021)

[18] Elfwing, S., Uchibe, E., Doya, K.: Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. Neural Networks **107**, 3–11 (2018)

[19] Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. IJCV **88**(2), 303–338 (2010)

[20] Hariharan, B., Arbeláez, P., Bourdev, L., Maji, S., Malik, J.: Semantic contours from inverse detectors. In: ICCV. pp. 991–998. IEEE (2011)

[21] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)

[22] Hu, Y.T., Huang, J.B., Schwing, A.G.: Videomatch: Matching based video object segmentation. In: ECCV. pp. 54–70 (2018)

[23] Hua, W., Dai, Z., Liu, H., Le, Q.V.: Transformer quality in linear time. arXiv preprint arXiv:2202.10447 (2022)

[24] Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., Kämäräinen, J.K., Danelljan, M., Zajc, L.Č., Lukežič, A., Drbohlav, O., et al.: The eighth visual object tracking vot2020 challenge results. In: ECCV. pp. 547–601. Springer (2020)

[25] Liang, C., Wang, W., Zhou, T., Miao, J., Luo, Y., Yang, Y.: Local-global context aware transformer for language-guided video segmentation. arXiv preprint arXiv:2203.09773 (2022)

[26] Liang, C., Wang, W., Zhou, T., Yang, Y.: Visual abductive reasoning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 15565–15575 (June 2022)

[27] Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR. pp. 2117–2125 (2017)

[28] Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV. pp. 740–755. Springer (2014)

[29] Liu, H., Dai, Z., So, D., Le, Q.V.: Pay attention to mlps. In: NeurIPS. vol. 34, pp. 9204–9215 (2021)

[30] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: ICCV (2021)

[31] Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., Hu, H.: Video swin transformer. arXiv preprint arXiv:2106.13230 (2021)

[32] Luiten, J., Voigtlaender, P., Leibe, B.: Premvos: Proposal-generation, refinement and merging for video object segmentation. In: ACCV. pp. 565–580 (2018)

[33] Ma, Z., Wang, L., Zhang, H., Lu, W., Yin, J.: Rpt: Learning point set representation for siamese visual tracking. In: ECCV. pp. 653–665. Springer (2020)

[34] Oh, S.W., Lee, J.Y., Xu, N., Kim, S.J.: Video object segmentation using space-time memory networks. In: ICCV (2019)

[35] Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., Tran, D.: Image transformer. In: ICCV. pp. 4055–4064. PMLR (2018)

[36] Perazzi, F., Khoreva, A., Benenson, R., Schiele, B., Sorkine-Hornung, A.: Learning video object segmentation from static images. In: CVPR. pp. 2663–2672 (2017)

[37] 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: CVPR. pp. 724–732 (2016)

[38] Pont-Tuset, J., Perazzi, F., Caelles, S., Arbeláez, P., Sorkine-Hornung, A., Van Gool, L.: The 2017 davis challenge on video object segmentation. arXiv preprint arXiv:1704.00675 (2017)

[39] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)

[40] Ramachandran, P., Zoph, B., Le, Q.V.: Searching for activation functions. arXiv preprint arXiv:1710.05941 (2017)

[41] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: CVPR. pp. 4510–4520 (2018)

[42] Seong, H., Hyun, J., Kim, E.: Kernelized memory network for video object segmentation. In: ECCV (2020)

[43] Seong, H., Oh, S.W., Lee, J.Y., Lee, S., Lee, S., Kim, E.: Hierarchical memory matching network for video object segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12889–12898 (2021)

[44] Shi, J., Yan, Q., Xu, L., Jia, J.: Hierarchical image saliency detection on extended cssd. TPAMI **38**(4), 717–729 (2015)

[45] Synnaeve, G., Xu, Q., Kahn, J., Likhomanenko, T., Grave, E., Pratap, V., Sriram, A., Liptchinsky, V., Collobert, R.: End-to-end asr: from supervised to semi-supervised learning with modern architectures. In: ICML Workshops (2020)

[46] Vaswani, A., Ramachandran, P., Srinivas, A., Parmar, N., Hechtman, B., Shlens, J.: Scaling local self-attention for parameter efficient visual backbones. In: CVPR. pp. 12894–12904 (2021)

[47] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NIPS (2017)

[48] Vijayanarasimhan, S., Grauman, K.: Active frame selection for label propagation in videos. In: ECCV. pp. 496–509. Springer (2012)

[49] Voigtlaender, P., Chai, Y., Schroff, F., Adam, H., Leibe, B., Chen, L.C.: Feelvos: Fast end-to-end embedding learning for video object segmentation. In: CVPR. pp. 9481–9490 (2019)

[50] Voigtlaender, P., Leibe, B.: Online adaptation of convolutional neural networks for video object segmentation. In: BMVC (2017)

[51] Wang, W., Zhou, T., Porikli, F., Crandall, D., Van Gool, L.: A survey on deep learning technique for video segmentation. arXiv preprint arXiv:2107.01153 (2021)

[52] Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: CVPR. pp. 7794–7803 (2018)

[53] Wang, Y., Xu, Z., Wang, X., Shen, C., Cheng, B., Shen, H., Xia, H.: End-to-end video instance segmentation with transformers. In: CVPR. pp. 8741–8750 (2021)

[54] Wug Oh, S., Lee, J.Y., Sunkavalli, K., Joo Kim, S.: Fast video object segmentation by reference-guided mask propagation. In: CVPR. pp. 7376–7385 (2018)

[55] Xiao, H., Feng, J., Lin, G., Liu, Y., Zhang, M.: Monet: Deep motion exploitation for video object segmentation. In: CVPR. pp. 1140–1148 (2018)

[56] Xu, N., Yang, L., Fan, Y., Yue, D., Liang, Y., Yang, J., Huang, T.: Youtube-vos: A large-scale video object segmentation benchmark. arXiv preprint arXiv:1809.03327 (2018)

[57] Xu, X., Wang, J., Li, X., Lu, Y.: Reliable propagation-correction modulation for video object segmentation. In: AAAI (2022)

[58] Yan, B., Zhang, X., Wang, D., Lu, H., Yang, X.: Alpha-refine: Boosting tracking performance by precise bounding box estimation. In: CVPR. pp. 5289–5298 (2021)

[59] Yang, L., Wang, Y., Xiong, X., Yang, J., Katsaggelos, A.K.: Efficient video object segmentation via network modulation. In: CVPR. pp. 6499–6507 (2018)

[60] Yang, Z., Wei, Y., Yang, Y.: Collaborative video object segmentation by foreground-background integration. In: ECCV (2020)

[61] Yang, Z., Wei, Y., Yang, Y.: Associating objects with transformers for video object segmentation. In: NeurIPS (2021)

[62] Yang, Z., Wei, Y., Yang, Y.: Collaborative video object segmentation by multi-scale foreground-background integration. TPAMI (2021)

[63] Yang, Z., Zhang, J., Wang, W., Han, W., Yu, Y., Li, Y., Wang, J., Wei, Y., Sun, Y., Yang, Y.: Towards multi-object association from foreground-background integration. In: CVPR Workshops (2021)

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] see the end of Sec. 1
   (b) Did you describe the limitations of your work? [Yes] we discuss the failure cases in Sec. 6, and demonstrate them in Fig. 4.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] see the supplementary material.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A] the paper does not contain any theoretical assumptions.
   (b) Did you include complete proofs of all theoretical results? [N/A] the paper does not contain any theoretical proofs.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] the instructions and details needed to reproduce the main results are supplied in Sec. 5 and the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] see Sec. 5 and the supplementary materials.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] we follow the usual format used in previous state-of-the-art methods [6, 11, 34, 42, 60, 61] to report and compare the results. Besides, all the networks are simultaneously evaluated on four benchmarks without re-training or checkpoint selection.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Sec. 5 and the supplementary materials.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] we cite the creators or original papers of all the related code, data, and models used in this paper.
   (b) Did you mention the license of the assets? [No] all the assets are free for research study and widely used in previous related works.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No] but the code of our proposed approach will be made publicly available as soon as the paper is accepted.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] all the datasets are publicly available, free for research study, and commonly used in previous related works.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] all the datasets are commonly used in previous research works.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] we didn't use crowdsourcing or conduct research with human subjects.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] we didn't use crowdsourcing or conduct research with human subjects.
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A] we didn't use crowdsourcing or conduct research with human subjects.