VIDEO FRAME INTERPOLATION VIA LOCAL LIGHTWEIGHT BIDIRECTIONAL ENCODING WITH CHANNEL ATTENTION CASCADE

Xiangling Ding[†♮§♯], *Pu Huang*[‡], *Dengyong Zhang*[‡], *Xianfeng Zhao*[§♭]

[†] Hunan University of Science and Technology, xianglingding@163.com;
[♮] Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen Key Laboratory of Media Security
[‡] Changsha University of Science and Technology
[§]State Key Laboratory of Information Security, Chinese Academy of Sciences
[♭]School of Cyber Security, University of Chinese Academy of Sciences
[♯]Guangdong Provincial Key Laboratory of Information Security Technology

ABSTRACT

Deep Neural Networks based video frame interpolation, synthesizing in-between frames given two consecutive neighboring frames, typically depends on heavy model architectures, preventing them from being deployed on small terminals. When directly adopting the lightweight network architecture from these models, the synthesized frames may suffer from poor visual appearance. In this paper, a lightweight-driven video frame interpolation network (L^2BEC^2) is proposed. Concretely, we first improve the visual appearance by introducing the bidirectional encoding structure with channel attention cascade to better characterize the motion information; then we further adopt the local network lightweight idea into the aforementioned structure to significantly eliminate its redundant parts of the model parameters. As a result, our L^2BEC^2 performs favorably at the cost of only one third of the parameters compared with the state-of-the-art methods on public datasets. Our source code is available at `https://github.com/Pumpkin123709/LBEC.git`.

***Index Terms*—** Video frame interpolation, bidirectional encoder, channel attention cascade, local lightweight

1. INTRODUCTION

Video frame interpolation (VFI) is a lower level computer vision task, which is mainly adopted to improve video temporal super-resolution by synthesizing smooth transitions followed by the intermediate frames between two consecutive frames. It compensates for the motion information and then interpolates missing details for achieving a better visual appearance.

This work was supported in part by the National Natural Science Foundation of China under grant 62172059, Hunan Provincial Natural Science Foundations of China under Grant 2020JJ4626 and 2020JJ4029, the Opening Project of State Key Laboratory of Information Security under Grant 2021-ZD-07, the Opening Project of Guangdong Provincial Key Laboratory of Information Security Technology under Grant 2020B1212060078, National Key Technology R&D Program under 2019QY2202.

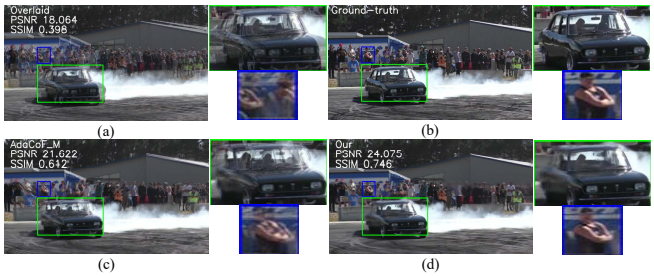


Fig. 1. A challenging example with complex motion. The frames by (a) the overlaid two inputs, (b) the ground-truth one, (c) lightweight AdaCoF[2], and (d) our method.

It plays an important role in wide applications such as frame-rate up-conversion [1, 2], slow-motion effects [3, 4], and view synthesis [5]. Though fundamental, the issue is also challenging in that the occlusion, complex motion, and feature change in real-world videos are hard to represent in a precise way.

Recently, deep neural network (DNN)-based VFI methods [2]-[18] have dominated in this field due to theirs strong representation ability. They learn the mapping from two continuous frames to the in-between frame by iteratively adjusting the parameters, and their key is to design novel network architecture and loss function. The designed network firstly predicts optical flow [6, 7, 8, 9], convolution kernel [2, 10, 11, 12], or feature [2, 13, 14] as precise as possible, and then jointly generates intermediate frames. The loss function conducts the synthesized frames to approximate ground-truth ones through back-propagation.

Despite the great success of DNN-based VFI methods, there are still a few limitations. 1) When directly adopting the lightweight network from these models, the synthesized frames suffer from poor visual appearance because of imprecise representation, shown in Fig. 1(c). 2) The DNN-based models are designed more and more complicated causing a heavy models size and high computational complexity. For

example, AdaCoF [2] consists of over 20 million parameters, of which 70%∼80% exists in the first stage to capture motion information. Thus it is difficult to deploy them into devices with less storage and low computation ability.

To conquer those above limitations, we propose a DNN-based VFI network (L^2BEC^2) that exploits local lightweight bidirectional encoding structure with channel attention cascade for interpolating accurate intermediate frames and reducing model size. We introduce the bidirectional encoding structure to combine the forward encoding and the backward encoding for extracting the bidirectional motion features. The learned features can better capture and characterize the motion details and changing content. Besides, channel attention cascade is introduced to adaptively obtain the weight of the bidirectional encoders to further retain the motion characteristics and prevent from the issue of two convergence in the training. Moreover, network lightweight idea has been directly utilized in VFI models [15, 16, 17, 18]. They either use an off-the-shelf optical flow model or predict their own task-specific optical flow as a guidance of pixel-level motion inference to achieve a smaller models. However, they either are bound to specific optical flow design or lightweight the entire network causing poor performance on public datasets. In this work, a local lightweight strategy is integrated into the stage of feature representation. Such operation not only reduces the size of the model, but also maintains a certain quality by being partially lightweight for high-level motion semantics to preserve the directionality of low-level motion semantics. The contributions are summarized as follows:

(1) We present an effective and efficient DNN-based VFI with a lightweight structure. The proposed approach achieves comparable performance to the state-of-the-art methods on public datasets with less than a third of the parameters.

(2) We design the bidirectional encoding structure to combine the forward encoding and the backward encoding. By involving the information of content change and motion details in the learned feature in an adaptive fashion, the unsatisfied artifacts are alleviated in the interpolated results.

(3) We propose to adopt a local lightweight strategy for feature extraction by using depth separable convolution [19, 20]. The lightweight of high-dimensional motion semantics reduces the number of parameters while the preservation of low-dimensional motion semantics ensures its directivity contributing to the improvement of the visual appearance.

2. THE PROPOSED METHOD

2.1. Problem Definition

Given two adjacent frames I_n, I_{n+1}, where n is a video temporal index, the aim of VFI is to produce an intermediate frame $\widehat{I}_t$, where t is the interpolated time interval. If its ground truth is defined as I_t, the lightweight DNN is designed to obtain intermediate frame $\widehat{I}_t$ in this work. Its ultimate goal

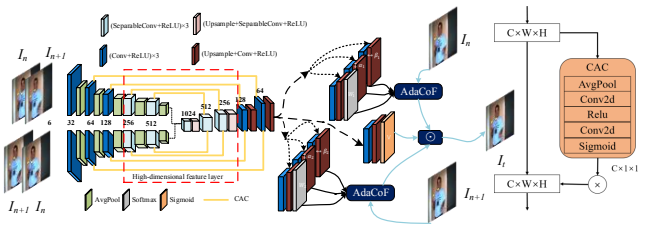


Fig. 2. Overview of the proposed L^2BEC^2, and the structure of channel self-attention.

is to train a lightweight network L_θ that is parameterized by ι, where ι is iteratively updated by minimizing a specific loss function ℓ, denoted as

$$\iota^* = \arg\min_{\iota} \frac{1}{N}\sum_{n=1}^{N} \ell(L_\iota(I_n, I_{n+1}, t), I_t) \qquad (1)$$

where $\{(I_n, I_{n+1}), I_t\}_{n=1}^{N}$ is the set of video frames.

2.2. The Proposed Model

In this section, we elaborate the proposed VFI method (L^2BEC^2) which exploits local lightweight bidirectional encoding structure with channel attention cascade. The overview of L^2BEC^2 is illustrated in Fig. 2. Next, we will discuss three main components of our network in details.

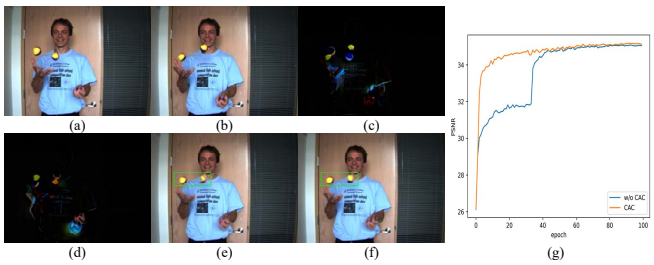


Fig. 3. (a) I_n; (b) I_{n+1}; the encoding feature of (c) forward, and (d) backward; the synthesized frames by (e) unidirectional, and (f) bidirectional;(g) the convergence for CAC.

Bidirectional Encoding (BE) It is well known that unidirectional encoding from I_n to I_{n+1} cannot perfectly capture the relevant motion features between I_n and I_{n+1}. Besides, increasing the breadth of DNN in low-dimensional channels has more advantages than increasing its depth for the issue about the number of parameters of DNN. Thus, we design a **BE** for better inferring the motion information and maintaining fewer model parameters, denoted as

$$\begin{aligned}[F_{forward}, F_{backward}] &= L_{encoder}\{(I_n, I_{n+1}); (I_{n+1}, I_n)\} \\ O &= L_{decoder}([F_{forward}, F_{backward}])\end{aligned} \qquad (2)$$

Here, the same encoder is executed forward, and backward encoding, respectively to extract bidirectional motion features, and then a decoder is used to decode two different encoding features to characterize the motion details and content. The diagram is shown on the left of Fig. 2.

Further, we investigate the benefit of **BE** in inferring motion information, shown in Fig. 3 (a)-(f). From it, we can observe that the forward encoded motion feature and backward one have obvious differences, thus, the final prediction result by **BE** has a good interpolation effect compared with the one by unidirectional encoding due to the successful capture of the important motion information and changing content.

Channel Attention Cascade (CAC) To reduce the loss of information, we propose a channel self-attention cascade, shown in the right of Fig. 2, to adaptively obtain the weight map of the different layers of the two encoders to retain the information about the changing content, and occlusion. Besides, because our designed bidirectional encoding structure is not one-to-one between the encoder and the decoder, it will produce two convergence in the training causing the decrease of the overall convergence rate. Thus, CAC instead of skip connection is designed to connect the forward encoder or backward encoder with the decoder in a cascade manner. It can solve the problem of network convergence, the effect of which is shown in Fig. 3(g). From it, we can observe that the network with CAC shows a monotonous and rapid convergence trend while the one without CAC represents fluctuation and abrupt change.

Denoting the $i-th$ layer feature of encoder by $F^i \in \Re^{C\times W\times H}$, average pooling is used to aggregates the statistics of a channel; then two 1×1 convolution are expected to obtain the non-linear inter-channel relationships. Finally, the output feature of $i-th$ layer of decoder is computed as

$$\begin{aligned} O^i_{decoder} &= L^i_{decoder}([O^{i-1}_{decoder}, att(F^i_{forward}) \\ &\quad \otimes F^i_{forward}, att(F^i_{backward}) \otimes F^i_{backward}]) \\ att(F^i_x) &= \sigma(W_1 * (ReLU(W_0 * AVG(F^i_x)))), \\ &\quad x \in \{forward, backward\} \end{aligned} \tag{3}$$

where $\sigma(\cdot)$ and $*$ denote sigmoid function, and convolution, W_0 and W_1 are weights of two 1×1 convolution. $att(F^i_x)$ is the forward or backward attention adaptive weights.

Local Lightweight Strategy (LLs) To show the effect of different dimension levels on parameters of the model, we conduct an extensional experiment and report in the second and fourth row of Table 2. From it, we find that low-level motion semantics are more directional than high-level ones, and the parameters of 512 and 1024 dimensions are occupied close to 80% of the model, but the influence of the interpolation effect is only 0.03 dB. Therefore, we use deep separable convolution [19, 20] for high-level semantics to reduce the number of model parameters while the rest of the model holds unchanged. We can also infer that the LLs saves the 70% parameters of the original AdaCoF, i.e. 14.79 million. Denoting W^s is the deep separable convolution kernel, depth-wise convolution with one filter per input channel in the lightweight layer of encoder or decoder can be calculated as:

$$O_{k,l,m} = \sum_{i,j} W^s_{i,j,m} \cdot F_{k+i-1,l+j-1,m} \tag{4}$$

where the m_{th} filter in W^s is convolved with the m_{th} channel in F to output the m_{th} channel of the feature map O.

Finally, we borrow the adaptive collaboration of flows [2] to produce the final interpolated result. It is computed as:

$$\begin{aligned} I_t &= V \odot \hat{I}_n + (J - V) \odot \hat{I}_{n+1} \\ \hat{I}_x &= \sum_{k=0}^{K-1} \sum_{l=0}^{K-1} W_{k,l}(i,j) I_x(i + dk + \alpha_{k,l}, \\ &\qquad j + dl + \beta_{k,l}), x \in \{n, n+1\} \end{aligned} \tag{5}$$

where $\odot$, J, and V are a pixel-wise multiplication, ones matrix, and occlusion map; K, $W_{k,l}$, $(\alpha_{k,l}, \beta_{k,l})$ and d are the kernel size, weights, the offset vectors and the dilation value.

3. EXPERIMENTS

3.1. Experiment Setups

Training Set: The Vimeo90K dataset [21] is adopted as the training set, which consists of 51,312 triplet frames with a resolution of 256×448. The first and third frames in the each triplet are utilized as the inputs, and the second one is set as the ground truth intermediate frame as $t = 0.5$. We randomly crop them to a size of 256×256, and augment the training data by flipping horizontally and vertically.

Loss Function: We combine the distortion loss and perception loss [2] to conduct the model training. For the former, the charbonnier penalty [4] is selected to reduce the difference between the synthesized frame and the ground truth one for its global smoothness and robustness to outliers. The perception loss contains distortion loss, perceptual loss, and adversarial loss. Perceptual loss utilizes the feature F_{conv4_3} from $conv4\_3$ of VGG16 network pretraied in ImageNet while adversarial loss utilizes a discriminator D, trained on the synthesized frames and the two adjacent input frames, to distinguish which one is the synthesized frames for the higher quality and sharpness output. The final loss is expressed as:

$$\begin{aligned} \ell_{all} &= \ell_d + \ell_p \\ \ell_d(\widehat{I}^t, I^t) &= Mean(\sqrt{(\widehat{I}^t - I^t)^2 + \varepsilon^2}) \\ \ell_p &= \lambda_d \ell_d + \lambda_{vgg} \ell_{vgg} + \lambda_{adv} \ell_{adv} \\ \ell_{vgg} &= \left\| F_{conv4\_3}(\widehat{I}^t) - F_{conv4\_3}(I^t) \right\|_2 \\ \ell_{adv} &= D([I^n, \widehat{I}^t]) \log(D([I^n, \widehat{I}^t])) \\ &\quad + D([\widehat{I}^t, I^{n+1}]) \log(D([\widehat{I}^t, I^{n+1}])) \end{aligned} \tag{6}$$

where $Mean(\cdot)$ and $[\cdot]$ are the mean and concatenation operation, respectively; $\varepsilon = 1\times10^{-3}$.

Training Process: The AdaMax optimizer[22] is used to train our network. The initial learning rate is set to 0.001, and

decays half every 20 epochs. The batch size is 12, the entire network is trained for 100 epochs.

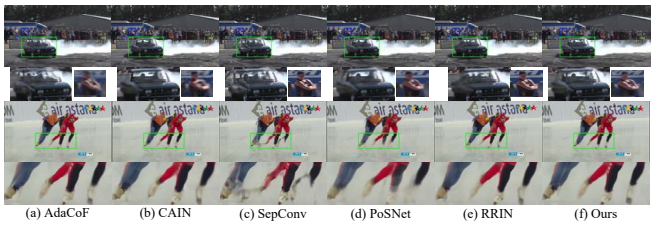
Evaluation settings: Two widely-used metrics: signal-to-noise ratio (PSNR) and structural similarity index (SSIM) are used to evaluated. Middlebury [23], UCF101 [24], and DAVIS [25] are adopted as testing sets to qualitatively compare the L^2BEC^2 with the previous approaches.

3.2. Main Results

We compare L^2BEC^2 with the most advanced VFI methods including PoSNet [6], AdaCoF [2], SepConv [11], CAIN [13] and RRIN [8]. PoSNet and RRIN are flow-based models which infer motions as pixel displacement. SepConv and AdaCoF are kernel-based algorithms that regard motions as local convolution. CAIN integrates pixelshuffle and channel attention for pixel-level frame synthesis.

Table 1. Quantitative comparisons on public datasets.

	Training dataset	Middlebury		UCF101		DAVIS		Parameters (million)
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	
PoSNet[6]	pre-trained	33.449	0.930	33.932	0.962	27.180	0.834	22.54
SepConv[11]	pre-trained	35.163	0.953	34.258	0.962	26.218	0.819	21.68
RRIN[13]	pre-trained	35.778	0.960	34.968	0.966	27.365	0.842	19.19
AdaCoF[2]	Vimeo-90K	35.715	0.958	35.063	0.966	26.636	0.817	21.84
CAIN[8]	Vimeo-90K	35.100	0.950	34.964	0.965	27.458	0.832	42.78
ours	Vimeo-90K	35.841	0.961	35.120	0.966	26.833	0.825	7.05



(a) AdaCoF (b) CAIN (c) SepConv (d) PoSNet (e) RRIN (f) Ours

Fig. 4. Visual comparisons on the "car" and "speed-skating" of DAVIS dataset.

As demonstrated in Table 1, L^2BEC^2 achieves the best PSNR and SSIM on Middlebury, in which has about 0.74 dB gain in PSNR compared to CAIN [8] but with only one sixth parameters. For the results on UCF101, we also obtain the optimal PSNR and SSIM results. Moreover, our method is comparable to CAIN and RRIN with slight drop in performance on DAVIS. But CAIN and RRIN have much more parameters than our does and we only use 83%, and 63% fewer parameters, respectively. Moreover, different from RRIN, L^2BEC^2 does not need explicit optical flow estimation, which costs much more inference time. Besides, our method is still superior to other flow-based and kernel-based approaches on all testing datasets, especially AdaCoF. In a word, the parameters of L^2BEC^2 are only one third or even one sixth of the other models, which also indicates that our method is more appropriate for small terminals. Fig. 4 shows synthesized frames in the DAVIS dataset by different models.

3.3. Ablation Study

We conduct an ablation study to illustrate the effectiveness of **BE**, **CAC** and **LLs** on Middlebury dataset.We define "Base" network by replacing standard convolution with depth separable convolution for full lightweight without BE, CAC, and LLs. Then BEnet, and CACnet intergrade the cascade mode of double branch BE, and CAC into "Base". To reflect the effect of LLs, we conduct additional two experiments: BEnet without high dimensional feature layer that omits the red box in Fig. 2, and CACnet without lightweight and high dimensional feature layer, denoted as BEnet-w/o-HD and CACnet-w/o-LWHD, respectively. Besides, L^2BEC^2-w/o-LLs is validated with BE, CAC, and without LLs to illustrate the efficiency of our proposed method.

Table 2. Effectiveness of different components of our model

models	PSNR	SSIM	Size(MB)	Parameters (million)
"Base"	34.940	0.951	9.7	2.51
BEnet	35.107	0.953	15	3.85
CACnet	35.197	0.954	15	3.851
BEnet-w/o-HD	35.104	0.952	3	0.77
CACnet-w/o-LWHD	35.742	0.960	25	6.53
L^2BEC^2	35.841	0.961	27	7.05
L^2BEC^2-w/o-LLs	36.035	0.963	129	33.64

The results are reported in Table 2. From it and Table 1, we can observe that L^2BEC^2 outperforms top five models, which indicates the performance gain of BE and CAC, and the parameters reduction capacity of LLs. We can also deduce that L^2BEC^2-w/o-LLs obtains 0.257 dB gain against RRIN [13]. The results also show that BE significantly improves the quality of lightweight network, CAC enhances the feature transfer efficiency, maintains the network convergence and ultimately optimizes the interpolation quality, and LLs implies that low-dimensional features only occupy a small number of parameters, but have a greater impact on the interpolation effect, as a result, it can greatly reduce the parameters while achieving better performance.

4. CONCLUSION

In this paper, we propose a L^2BEC^2 model that uses BE, CAC and LLs for video frame interpolation. The BE combines the forward and the backward encoding in consideration of changing content and motion details and thus reduces the unsatisfied artifacts. The CAC fuses the learned features in an adaptive fashion, which guarantees the effective inference of motion information, and its convergence. Furthermore, LLs significantly decreases the size of the model while holding the visual quality. The proposed method achieved a comparable performance against the existing approaches while with only one third of the parameters. In the future, it will be of interest to package a generic lightweight modules into VFIs.

5. REFERENCES

[1] He J, Yang G, Liu X, Ding X. "Spatio-temporal saliency-based motion vector refinement for frame rate up-conversion," *ACM Trans. Multim. Comput.*, vol. 16, no. 2, pp. 1-18, 2020.

[2] Lee H, Kim T, Chung T Y, Pak D, Ban Y, and Lee S. "AdaCoF: adaptive collaboration of flows for video frame interpolation," in *CVPR*, 2020.

[3] Jiang H, Sun D, Jampani V, Yang M H, Learned-Miller E, and Kautz J. "Super SloMo: high quality estimation of multiple intermediate frames for video interpolation," in *CVPR*,2018.

[4] Bao W, Lai W, Ma Ch, Zhang X, Gao Zh, and Yang M. "Depth-aware video frame interpolation,"in *CVPR*, 2019.

[5] Flynn J, Neulander I, Philbin J, and Snavely N. "Deep stereo: learning to predict new views from the world's imagery," in *CVPR*,2016.

[6] Yu S, Park B, and Jeong J. "PoSNet: 4× video frame interpolation uUsing position-specific flow," in *ICCVW*, 2019.

[7] Long G, Kneip L, Alvarez J M, Li H, Zhang X, and Yu Q. "Learning image matching by simply watching video," in *ECCV*, 2016.

[8] Video frame synthesis using deep voxel flow Li H, Yuan Y, Wang Q."Video frame interpolation via residue refinement," in *ICASSP*, 2020.

[9] Bao W, Lai W S, Zhang X, et al. "MEMC-Net: motion estimation and motion compensation driven neural network for video interpolation and enhancement," *IEEE Trans. Pattern Anal.* vol. 43, pp. 933-948, 2021.

[10] Niklaus S, Mai L, Liu F. "Video frame interpolation via adaptive convolution," in *CVPR*, 2017.

[11] Niklaus S, Mai L, Liu F. "Video frame interpolation via adaptive separable convolution," in *ICCV*, 2017.

[12] Cheng X, and Chen Zh. "Video frame interpolation via deformable separable convolution," in *AAAI*, 2020.

[13] Choi M, Kim H, Han B, Xu N, and Lee K M. "Channel attention is all you need for video frame interpolation," in *AAAI*, 2020.

[14] Cheng X, and Chen Z. "Multiple video frame interpolation via enhanced deformable separable convolution," *IEEE Trans. Pattern Anal.*, 2021.

[15] Chi Z, Mohammadi Nasiri R, Liu Z, Lu J, Tang J, and Plataniotis K N. "All at once: temporally adaptive multi-frame interpolation with advanced motion modeling," in *ECCV*, 2020.

[16] Liu Z, Yeh R A, Tang X, Liu Y, and Agarwala A. "Video frame synthesis using deep voxel flow," in *ICCV*, 2017.

[17] Xue T, Chen B, Wu J, Wei D, and Freeman W T. "Video enhancement with task-oriented flow," *Int. J. Compt Vision*, vol. 127, no. 8, pp. 1106-1125, 2019.

[18] Yuan L, Chen Y, Liu H, Kong T, and Shi J. "Zoom-in-to-check: boosting video interpolation via instance-level discrimination," *CVPR*, 2019.

[19] Howard A G, Zhu M, Chen B, et al. "MobileNets: efficient convolutional neural networks for mobile vision applications," in *ArXiv* abs/1704.04861, 2017.

[20] Chollet F. "Xception: deep learning with depthwise separable convolutions," in *CVPR*, 2017.

[21] Xue T, Chen B, Wu J, et al. "Video enhancement with task-oriented flow," *Int. J. Compt Vision*, vol. 127, no. 8, pp. 1106-1125, 2019.

[22] Kingma D P, and Ba J. "Adam: a method for stochastic optimization," in *CoRR*, 2015.

[23] Baker S, Scharstein D, Lewis J P, Roth S, Black Michael J, and Szeliski R. "A database and evaluation methodology for optical flow," *Int. J. Compt Vision*, vol. 92, pp. 1-31, 2007.

[24] Soomro K, Zamir A, and Shah M. "UCF101: a dataset of 101 human actions classes from videos in the wild," *ArXiv* abs/1212.0402, 2012.

[25] Perazzi F, Pont-Tuset J, McWilliams B, et al."A benchmark dataset and evaluation methodology for video object segmentation," in *CVPR*, 2016.