Collaborative Video Object Segmentation by Multi-Scale Foreground-Background Integration

Zongxin Yang, Yunchao Wei, and Yi Yang

Abstract—This paper investigates the principles of embedding learning to tackle the challenging semi-supervised video object segmentation. Unlike previous practices that focus on exploring the embedding learning of foreground object (s), we consider background should be equally treated. Thus, we propose a Collaborative video object segmentation by Foreground-Background Integration (CFBI) approach. CFBI separates the feature embedding into the foreground object region and its corresponding background region, implicitly promoting them to be more contrastive and improving the segmentation results accordingly. Moreover, CFBI performs both pixel-level matching processes and instance-level attention mechanisms between the reference and the predicted sequence, making CFBI robust to various object scales. Based on CFBI, we introduce a multi-scale matching structure and propose an Atrous Matching strategy, resulting in a more robust and efficient framework, CFBI+. We conduct extensive experiments on two popular benchmarks, *i.e.*, DAVIS, and YouTube-VOS. Without applying any simulated data for pre-training, our CFBI+ achieves the performance ($\mathcal{J}\&\mathcal{F}$) of 82.9% and 82.8%, outperforming all the other state-of-the-art methods. Code: https://github.com/z-x-yang/CFBI.

Index Terms—Video Object Segmentation, Convolutional Neural Networks, Metric Learning

1 Introduction

Video Object Segmentation (VOS) is a fundamental task in computer vision with many potential applications, including augmented reality [1] and self-driving cars [2]. In this paper, we focus on semi-supervised VOS, which targets on segmenting a particular object across the entire video sequence based on the object mask given at the first frame. The development of semi-supervised VOS can benefit many related tasks, such as video instance segmentation [3], [4] and interactive video object segmentation [5], [6], [7].

Early VOS works([8], [9], [10]) rely on fine-tuning with the first frame in evaluation, which heavily slows down the inference speed. Recent works (*e.g.*, [11], [12], [13]) aim to avoid fine-tuning and achieve better run-time. In these works, STMVOS [13] introduces memory networks to learn to read sequence information and outperforms all the fine-tuning based methods. However, STMVOS relies on simulating extensive frame sequences using large image datasets [14], [15], [16], [17], [18] for training. The simulated data significantly boosts the performance of STMVOS but makes the training procedure elaborate. Without simulated data, FEELVOS [12] adopts a semantic pixel-wise embedding together with a global (between the first and current frames) and a local (between the previous and current frames) matching mechanism to guide the prediction. The matching mechanism is simple and fast, but the performance is not comparable with STMVOS.

Even though the efforts mentioned above have made significant progress, current state-of-the-art works pay little attention to the feature embedding of background region in videos and only focus on exploring robust matching strategies for the foreground object (s). Intuitively, it is easy to extract the foreground region from a video when precisely removing all the background. Moreover, modern video scenes commonly focus on many similar objects, such as the cars in car racing, the people in a conference, and the animals on a farm. For these cases, the contempt of integrating foreground and background embeddings traps VOS in an unexpected background confusion problem. As shown in Fig. 1, if we focus on only the foreground matching like FEELVOS, a similar and same kind of object (sheep here) in the background is easy to confuse the prediction of the foreground object. Such an observation motivates us that the background should be equally treated compared with the foreground so that better feature embedding can be learned to relieve the background confusion and promote the accuracy of VOS.

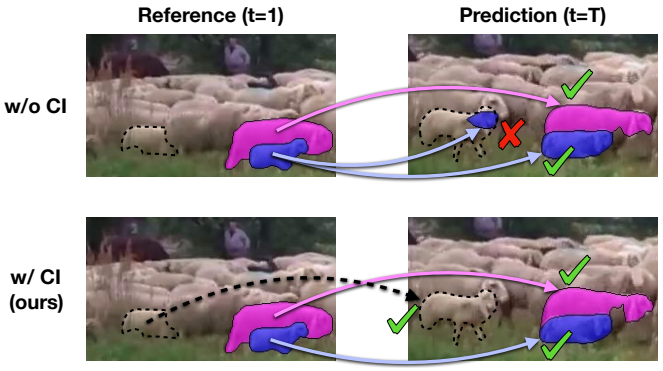


Fig. 1. CI: collaborative integration. There are two foreground sheep (pink and blue) in the sequence. In the top line, the contempt of background matching leads to a confusion of sheep's prediction. In the bottom line, we relieve the confusion problem by introducing background matching (dot-line arrow).

We propose a novel framework for Collaborative video object segmentation by Foreground-Background Integration (CFBI) based on the above motivation. Unlike the above methods, we not only extract the embedding and do match-

Z. Yang, Y. Wei and Y. Yang are with the ReLER Lab, Centre for Artificial Intelligence, University of Technology Sydney, Ultimo, NSW 2007, Australia. E-mail: zongxin.yang@student.uts.edu.au, {yunchao.wei, yi.yang}@uts.edu.au

ing for the foreground target in the reference frame, but also for the background region to relieve the background confusion. In particular, our framework extracts two types of embedding, pixel-level and instance-level, for each video frame to cover different scales of features. Like FEELVOS, we employ pixel-level embedding to match all the objects' details with the same global & local mechanism. However, the pixel-level matching is not sufficient and robust to match those objects with larger scales and may bring unexpected noises due to the pixel-wise diversity. Thus we introduce instance-level embedding to help the segmentation of large-scale objects by using attention mechanisms. For the training process, we propose a balanced random-crop scheme to avoid biasing learned attributes to background attributes. These proposed strategies can effectively improve the quality of the learned collaborative embeddings for conducting VOS while keeping the network simple yet effective simultaneously. Based on CFBI, we further introduce an efficient multi-scale matching structure, resulting in a more robust framework, CFBI+. Within CFBI+, we propose an Atrous Matching (AM) strategy, which can significantly save computation and memory usage of matching processes. The use of AM makes CFBI+ not only more robust but also more efficient than CFBI.

We perform extensive experiments on DAVIS [19], [20], and YouTube-VOS [21] to validate the effectiveness of the proposed CFBI and CFBI+. Without any bells and whistles (such as the use of simulated data, fine-tuning, or post-processing), CFBI+ outperforms all other state-of-the-art methods on the validation splits of DAVIS 2017 (ours, $\mathcal{J}\&\mathcal{F}$ $\mathbf{82.9}\%$) and YouTube-VOS ($\mathbf{82.8}\%$). Meanwhile, our multi-object inference speed is faster than previous state-of-the-art methods. We have made the code publicly available, and we hope our simple yet effective CFBI and CFBI+ will serve as two solid baselines and help ease the future research related to VOS.

This paper is an extension of our previous conference version [22]. The current work adds to the initial version in some significant aspects. First, we propose a plug-and-play Atrous Matching (AM) algorithm, which can significantly save computation and memory usage of matching processes. Second, based on the proposed AM, we design a multi-scale matching framework, resulting in a more strong and efficient VOS framework, CFBI+. Third, we incorporate considerable new experimental results, including ablation study, model setting, and visualization analysis.

2 Related Work

Semi-supervised Video Object Segmentation. Many previous methods for semi-supervised VOS rely on fine-tuning at test time. Among them, OSVOS [8] and MoNet [23] fine-tune the network on the first-frame ground-truth at test time. OnAVOS [9] extends the first-frame fine-tuning by an online adaptation mechanism, *i.e.*, online fine-tuning. MaskTrack [24] uses optical flow to propagate the segmentation mask from one frame to the next. PReMVOS [10] combines four different neural networks (including an optical flow network [25]) using extensive fine-tuning and a merging algorithm. Despite achieving promising results, all these methods are seriously slowed down by fine-tuning during inference.

Some other recent works (*e.g.*, [11], [26]) aim to avoid fine-tuning and achieve a better run-time. OSMN [11] employs two networks to extract the instance-level information and make segmentation predictions, respectively. PML [27] learns a pixel-wise embedding with the nearest neighbor classifier. Similar to PML, VideoMatch [28] uses a soft matching layer that maps the pixels of the current frame to the first frame in a learned embedding space. Following PML and VideoMatch, FEELVOS [12] extends the pixel-level matching mechanism by additionally matching between the current frame and the previous frame. Compared to fine-tuning methods, FEELVOS achieves a much higher speed, but there is still an accuracy gap. Like FEELVOS, RGMP [29] and STMVOS [13] does not require any fine-tuning. STMVOS, which leverages a memory network to store and read the information from past frames, outperforms all the previous methods. However, STMVOS and its following works (EGMN [30] and KMNVOS [31]) rely on an elaborate pre-training procedure using extensive simulated data generated from multiple datasets with pixel-level annotations. LWLVOS [32] proposes to use an online few-shot learner during both training and testing stages. Without simulated data, LWLVOS is comparable with KMNVOS on YouTube-VOS, but generalizes worse than the above methods using simulated data on DAVIS.

In previous practices, learning foreground feature embedding has been well explored. OSMN proposed to conduct an instance-level matching, but such a matching scheme fails to consider the feature diversity among the details of the target's appearance and results in coarse predictions. PML and FEELVOS alternatively adopt the pixel-level matching by matching each pixel of the target, which effectively takes the feature diversity into account and achieves promising performance. Nevertheless, performing pixel-level matching may bring unexpected noises in the case of some pixels from the background are with a similar appearance to the ones from the foreground (Fig. 1).

Thus, we propose a collaborative integration method by additionally learning background embedding. Furthermore, our CFBI utilizes both the pixel-level and instance-level embeddings to guide prediction.

Attention Mechanisms. Recent works introduce the attention mechanism into convolutional networks (*e.g.*, [33], [34]). Following them, SE-Nets [35] introduced a lightweight gating mechanism that focuses on enhancing the representational power of the convolutional network by modeling channel attention. Inspired by SE-Nets, CFBI uses an instance-level average pooling method to embed collaborative instance information from pixel-level embeddings. After that, we conduct a channel-wise attention mechanism to help guide prediction. Compared to OSMN, which employs an additional convolutional network to extract instance-level embedding, our instance-level attention method is more efficient and lightweight.

3 Methodology

The Overview of CFBI. To overcome or relieve the problems raised by previous methods and promote the foreground objects from the background, we present Collaborative video

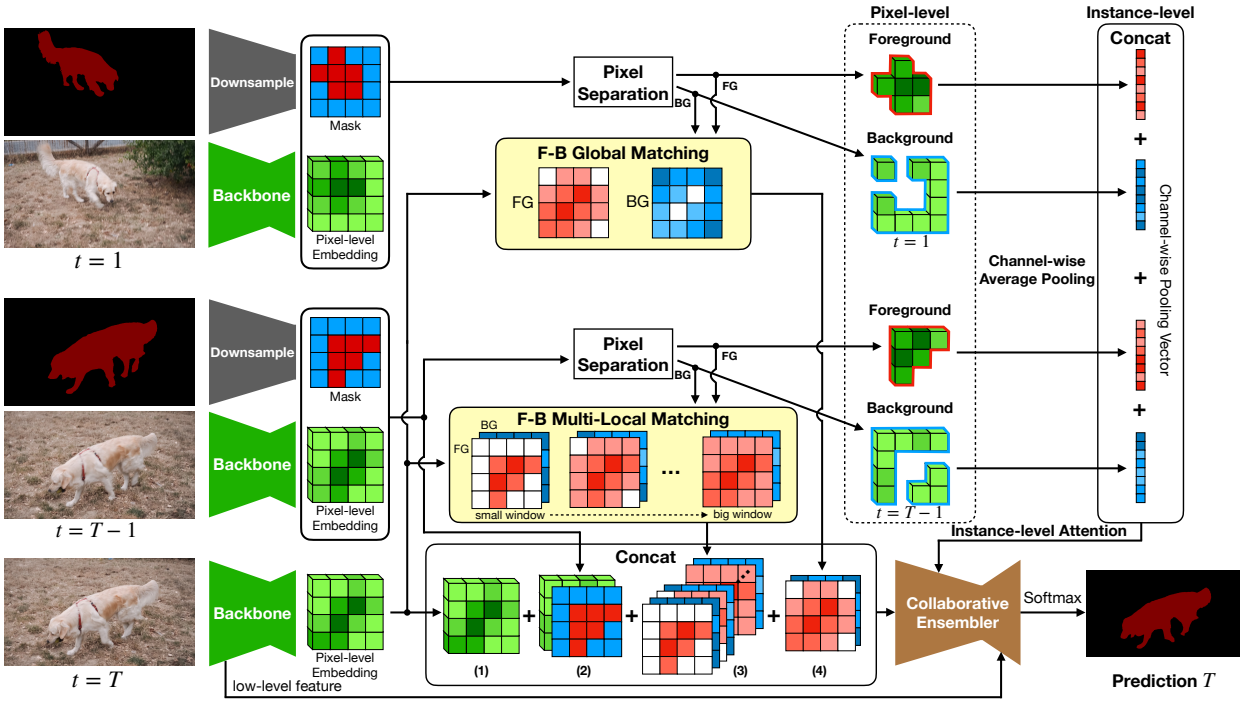


Fig. 2. An **overview** of CFBI. F-G denotes Foreground-Background. We use red and blue to indicate foreground and background separately. The deeper the red or blue color, the higher the confidence. Given the first frame ($t = 1$), previous frame ($t = T-1$), and current frame ($t = T$), we firstly extract their pixel-wise embedding by using a backbone network. Second, we separate the first and previous frame embeddings into the foreground and background pixels based on their masks. After that, we use F-G pixel-level matching and instance-level attention to guide our collaborative ensembler network to generate a prediction.

object segmentation by Foreground-Background Integration (CFBI), as shown in Figure 2. We use red and blue to indicate foreground and background separately. First, beyond learning feature embedding from foreground pixels, our CFBI also considers embedding learning from background pixels for collaboration. Such a learning scheme will encourage the feature embedding from the target object and its corresponding background to be contrastive, promoting the segmentation results accordingly. Second, we further conduct the embedding matching from both pixel-level and instance-level with the collaboration of pixels from the foreground and background. For the pixel-level matching, we improve the robustness of the local matching under various object moving rates. For the instance-level matching, we design an instance-level attention mechanism to augment the pixel-level matching efficiently. Moreover, to implicitly aggregate the learned foreground & background and pixel-level & instance-level information, we employ a collaborative ensembler to construct large receptive fields and make precise predictions.

3.1 Collaborative Pixel-level Matching

For the pixel-level matching, we adopt a global and local matching mechanism similar to FEELVOS for introducing the guided information from the first and previous frames, respectively. Unlike previous methods [12], [27], we additionally incorporate background information and apply multiple windows in the local matching, which is shown in the middle of Fig. 2.

For incorporating background information, we firstly redesign the pixel distance of [12] to further distinguish the foreground and background. Let B_t and F_t denote the pixel sets of background and all the foreground objects of frame t, respectively. We define a new distance between pixel p of the current frame T and pixel q of frame t in terms of their corresponding embedding, e_p and e_q, by

$$D(p,q) = \begin{cases} 1 - \frac{2}{1+exp(||e_p - e_q||^2 + b_B)} & \text{if } q \in B_t \\ 1 - \frac{2}{1+exp(||e_p - e_q||^2 + b_F)} & \text{if } q \in F_t \end{cases}, \quad (1)$$

where b_B and b_F are trainable background bias and foreground bias. We introduce these two biases to make our model be able further to learn the difference between foreground distance and background distance.

Foreground-Background Global Matching. Let $\mathcal{P}_t$ denote the set of all pixels (with a stride of 4) at time t and $\mathcal{P}_{t,o} \subseteq \mathcal{P}_t$ is the set of pixels at time t which belongs to the foreground object o. The global foreground matching between one pixel p of the current frame T and the pixels of the first reference frame (*i.e.*, $t = 1$) is,

$$G_o(p) = \min_{q \in \mathcal{P}_{1,o}} D(p,q). \quad (2)$$

Similarly, let $\overline{\mathcal{P}}_{t,o} = \mathcal{P}_t \backslash \mathcal{P}_{t,o}$ denote the set of relative background pixels of object o at time t, and the global background matching is,

$$\overline{G}_o(p) = \min_{q \in \overline{\mathcal{P}}_{1,o}} D(p,q). \quad (3)$$

Foreground-Background Multi-Local Matching. In FEELVOS, the local matching is limited in only one fixed extent of neighboring pixels, but the offset of objects across two adjacent frames in VOS is variable, as shown in Fig. 3.

(a) Slow moving rate (b) Fast moving rate

Fig. 3. The moving rate of objects across two adjacent frames is largely variable for different sequences. Examples are from YouTube-VOS [21].

Thus, we propose to apply the local matching mechanism on different scales and let the network learn how to select an appropriate local scale, which makes our framework more robust to various moving rates of objects. Notably, we use the intermediate results of the local matching with the largest window to calculate on other windows. Thus, the increase of computational resources of our multi-local matching is negligible.

Formally, let $K = \{k_1, k_2, ..., k_n\}$ denote all the neighborhood sizes and $H(p,k)$ denote the neighborhood set of pixels that are at most k pixels away from p in both x and y directions, our foreground multi-local matching between the current frame T and its previous frame $T-1$ is

$$ML_o(p,K) = \{L_o(p,k_1), L_o(p,k_2), ..., L_o(p,k_n)\}, \quad (4)$$

where

$$L_o(p,k) = \begin{cases} \min_{q \in \mathcal{P}^{p,k}_{T-1,o}} D(p,q) & \text{if } \mathcal{P}^{p,k}_{T-1,o} \neq \emptyset \\ 1 & \text{otherwise} \end{cases}. \quad (5)$$

Here, $\mathcal{P}^{p,k}_{T-1,o} := \mathcal{P}_{T-1,o} \cap H(p,k)$ denotes the pixels in the local window (or neighborhood). And our background multi-local matching is

$$\overline{ML}_o(p,K) = \{\overline{L}_o(p,k_1), \overline{L}_o(p,k_2), ..., \overline{L}_o(p,k_n)\}, \quad (6)$$

where

$$\overline{L}_o(p,k) = \begin{cases} \min_{q \in \overline{\mathcal{P}}^{p,k}_{T-1,o}} D(p,q) & \text{if } \overline{\mathcal{P}}^{p,k}_{T-1,o} \neq \emptyset \\ 1 & \text{otherwise} \end{cases}. \quad (7)$$

Here similarly, $\overline{\mathcal{P}}^{p,k}_{T-1,o} := \overline{\mathcal{P}}_{T-1,o} \cap H(p,k)$.

In addition to the global and multi-local matching maps, we concatenate the pixel-level embedding feature and mask of the previous frame with the current frame feature. FEELVOS demonstrates the effectiveness of concatenating the previous mask. Following this, we empirically find that introducing the previous embedding can further improve the performance ($\mathcal{J}\&\mathcal{F}$) by about 0.5%.

In summary, the output of our collaborative pixel-level matching is a concatenation of (1) the pixel-level embedding of the current frame, (2) the pixel-level embedding and mask of the previous frame, (3) the multi-local matching map and (4) the global matching map, as shown in the bottom box of Fig. 2.

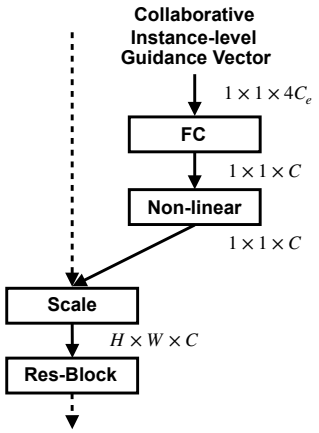


Fig. 4. The trainable part of the instance-level attention. C_e denotes the channel dimension of pixel-wise embedding. H, W, C denote the height, width, channel dimension of CE features.

3.2 Collaborative Instance-level Attention

As shown in the right of Fig 2, we further design a Collaborative instance-level attention mechanism to guide the segmentation for large-scale objects.

After getting the pixel-level embeddings of the first and previous frames, we separate them into foreground and background pixels (*i.e.*, $\mathcal{P}_{1,o}$, $\overline{\mathcal{P}}_{1,o}$, $\mathcal{P}_{T-1,o}$, and $\overline{\mathcal{P}}_{T-1,o}$) according to their masks. Then, we apply channel-wise average pooling on each group of pixels to generate a total of four instance-level embedding vectors and concatenate these vectors into one collaborative instance-level guidance vector. Thus, the guidance vector contains the information from both the first and previous frames, and both the foreground and background regions.

In order to efficiently utilize the instance-level information, we employ an attention mechanism to adjust our Collaborative Ensembler (CE). We show a detailed illustration in Fig. 4. Inspired by SE-Nets [35], we leverage a fully-connected (FC) layer (we found this setting is better than using two FC layers as adopted by SE-Net) and a non-linear activation function to construct a gate for the input of each Res-Block in the CE. The gate will adjust the scale of the input feature channel-wisely.

By introducing collaborative instance-level attention, we can leverage a full scale of foreground-background information to guide the prediction further. The information with a

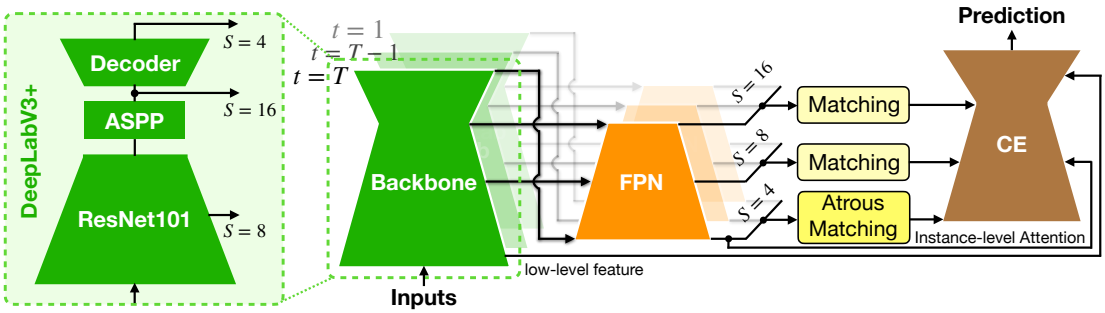


Fig. 5. An overview of CFBI+. S: the stride of feature maps. Firstly, CFBI+ extracts three features with different scales ($S = 4, 8, 16$) from backbone, ResNet101-DeepLabV3+ [36]. And then, we use the Feature Pyramid Network (FPN) [37] to fuse the information from small scales to large scales and reduce the channel dimensions of three features. After this, we do all the matching processes of CFBI on each scale. The output of each scale will be sent to the consistent stage of Collaborative Ensembler (CE).

large (instance-level) receptive field is useful to relieve local ambiguities [38], which is inevitable with a small (pixel-wise) receptive field.

3.3 Collaborative Ensembler (CE)

In the lower right of Fig. 2, we design a collaborative ensembler for making large receptive fields to aggregate pixel-level and instance-level information and implicitly learn the collaborative relationship between foreground and background.

Inspired by ResNets [39] and Deeplabs [36], [40], which both have shown significant representational power in image segmentation tasks, our CE uses a downsample-upsample structure, which contains three stages of Res-Blocks [39] and an Atrous Spatial Pyramid Pooling (ASPP) [36] module. The number of Res-Blocks in Stage 1, 2, and 3 are 2, 3, 3 in order. Besides, we employ dilated convolutional layers to improve the receptive fields efficiently. The dilated rates of the 3×3 convolutional layer of Res-Blocks in one stage are separately 1, 2, 4 (or 1, 2 for Stage 1). At the beginning of Stage 2 and Stage 3, the feature maps will be downsampled by the first Res-Block with a stride of 2. After these three stages, we employ an ASPP and a Decoder [36] module to increase the receptive fields further, upsample the scale of feature and fine-tune the prediction collaborated with the low-level backbone features.

3.4 CFBI+: Towards Efficient Multi-scale Matching

High-quality matching maps are essential for CFBI to generate accurate predictions with sharp object boundaries, which is one of the critic factors for improving VOS's performance. However, it is costly in terms of both GPU memory and time to achieve a delicate and high-resolution matching map (*e.g.*, with a stride of 4). Intuitively, there are two ways to accelerate matching processes. (1) Performing matching over low-resolution feature maps. Even the process will be much more light-weight, it is easy to miss many object details. (2) Reducing the channel dimensions while keeping using a high-resolution feature map. Even the amount of computation will decrease linearly with the channel dimensions, the accuracy of matching will also decrease.

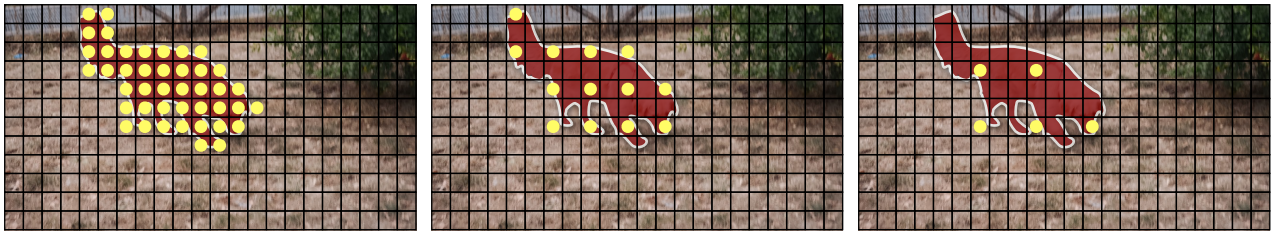
Some recent works (*e.g.*, [37], [41]) prove that multi-scale strategies can efficiently improve convolutional networks' performance. We consider that such strategies also benefit matching processes. Thus, we introduce an efficient multi-scale matching structure into CFBI, resulting in a more robust framework, *i.e.*, CFBI+. An overview of CFBI+ is shown in Fig. 5. Firstly, CFBI+ extracts three features with different scales ($S = 4, 8, 16$) from backbone. And then, we use the Feature Pyramid Network (FPN) [37] to further fuse the information from small scales to large scales. After this, we do all the matching processes of CFBI on every scale. The output of each scale will be sent to each corresponding stage of CE.

Concretely, to harness the merits of the two accelerate matching processes as well as alleviate their disadvantages, we adopt an adaptive matching strategy for feature maps of different scales. CFBI+ progressively and linearly increases the channel dimensions from larger scales to smaller scales, which reduces the amount of calculation for matching on larger scales. Meanwhile, richer semantic information in smaller scales can successfully make up for the performance drop due to the reduction of channel dimension for larger scales. In this way, various coarse-to-fine information can help CFBI+ achieve better segmentation results.

Besides, only progressively and linearly increasing channel dimensions is not enough to make the multi-scale matching more efficient than single-scale, because the complexity of matching processes increases exponentially with the resolution of feature maps. For example, the calculation of $G_{T,o}(p)$ on $S = 4$ scale is 256 times of $S = 16$. Thus, we additionally propose an Atrous Matching (AM) strategy to save computation and memory usage of matching processes further. Introducing AM helps CFBI+ to be more efficient than CFBI.

Atrous Matching (AM). *Algorithme àtrous* (or "atrous algorithm" in the following text), an algorithm for wavelet decomposition [42], played a key role in some recent convolutional networks [36], [43]. By adding a spatial interval during sampling, the atrous algorithm can reduce calculation while maintaining the same resolution. Intuitively, spatially close pixels always share similar semantic information. Hence, we argue that the atrous algorithm is also effective in matching processes. Removing part of similar pixels from referred pixels will not heavily drop performance but save much computation cost.

Let $q_{x,y}$ be the pixel at position (x, y) and let l be an atrous factor, we generalize the foreground global matching



(a) $l = 1$ (original matching) (b) $l = 2$ (c) $l = 3$

Fig. 6. An illustration of l-atrous object pixel set. Atrous matching improves computational efficiency by periodically filtering out referred object pixels without loss of resolution. (a) Yellow points indicate the referred object pixel set used in the original matching process. All the object (red dog) pixels are sampled. (b) (c) 2-atrous and 3-atrous object pixel set. Referred pixels are sampled out with a period of 2 or 3 pixels vertically and horizontally.

(Eq. 2) into an atrous form,

$$G_o^l(p) = \min_{q \in \mathcal{P}_{1,o}^l} D(p,q), \tag{8}$$

where

$$\mathcal{P}_{1,o}^l = \{q_{x,y} \in \mathcal{P}_{1,o}, \forall x, y \in \{l, 2l, 3l, ...\}\} \tag{9}$$

is a l-atrous object pixel set. We show an illustration in Fig. 6.

Let x_p and y_p denote the position of pixel p, the atrous form of the foreground local matching (Eq. 5) is

$$L_o^l(p,k) = \begin{cases} \min_{q \in \mathcal{P}_{T-1,o}^{l,p,k}} D(p,q) & \text{if } \mathcal{P}_{T-1,o}^{l,p,k} \neq \emptyset \\ 1 & \text{otherwise} \end{cases}, \tag{10}$$

where

$$\mathcal{P}_{T-1,o}^{l,p,k} := \mathcal{P}_{T-1,o} \cap H^l(p,k), \tag{11}$$

and

$$H^l(p,k) = \{q_{x,y} \in H(p,k) \forall x \in \{x_p, x_p \pm l, x_p \pm 2l, ...\}, \\ y \in \{y_p, y_p \pm l, y_p \pm 2l, ...\}\} \tag{12}$$

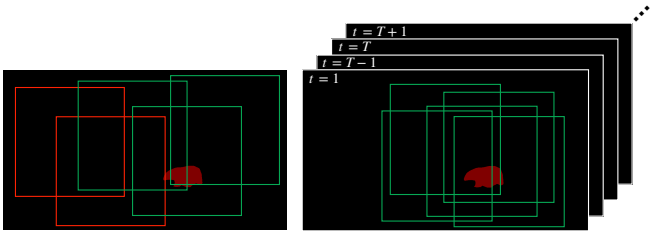
is a l-atrous neighborhood set.

In the same way, we can also generalize Eq. 3, Eq. 7 Eq. 4, and Eq. 6 into atrous forms, *i.e.*, $\overline{G}_o^l(p)$, $\overline{L}_o^l(p,k)$, $ML_o^l(p,K)$, and $\overline{ML}_o^l(p,K)$. Since the number of referred pixels is reduced l^2 times, AM's computational complexity is only $1/l^2$ of original matching. Notably, AM is equivalent to original matching when l is equal to 1, *i.e.*, $G_o^{l=1}(p) \equiv G_o(p)$ and $L_o^{l=1}(p,k) \equiv L_o(p,k)$.

On the largest matching scale ($S = 4$) of CFBI+, we apply 2-atrous matching processes, which significantly improve the efficiency of CFBI+. Notably, AM is a plug-and-play algorithm and can also improve the efficiency of CFBI during the testing stage.

4 Implementation Details

Following FEELVOS, we use the DeepLabv3+ [36] architecture as the backbone for our network. However, our backbone is based on the dilated ResNet-101 [36] instead of Xception-65 [44] for saving computational resources. We apply batch normalization (BN) [45] in our backbone and pre-train it on ImageNet [46] and COCO [15]. To make the training process more effective and consistent with the inference stage, we additionally adopt two tricks, *i.e.*, Balanced Random-Crop, and Sequential Training:



(a) Normal (b) Balanced

Fig. 7. When using normal random-crop, some red windows contain few or no foreground pixels. For reliving this problem, we propose balanced random-crop.

- **Balanced Random-Crop.** As shown in Fig. 7, there is an apparent imbalance between the foreground and the background pixel number on VOS datasets. Such an issue usually makes the models easier to be biased to background attributes. In order to relieve this problem, we take a balanced random-crop scheme, which crops a sequence of frames (*i.e.*, the first frame, the previous frame, and the current frame) by using a same cropped window and restricts the cropped region of the first frame to contain enough foreground information. The restriction method is simple yet effective. To be specific, the balanced random-crop will decide on whether the randomly cropped frame contains enough pixels from foreground objects or not. If not, the method will continually take the cropping operation until we obtain an expected one.
- **Sequential Training.** In the training stage, FEELVOS predicts only one step in one iteration, and the guidance masks come from the ground-truth data. RGMP and STMVOS use previous guidance information (mask or feature memory) in training, which is more consistent with the inference stage and performs better. The previous guidance masks are always generated by the network in the previous inference steps in the evaluation stage. Following RGMP, we train the network using a sequence of consecutive frames in each SGD iteration. In each iteration, we randomly sample a batch of video sequences. For each video sequence, we randomly sample a frame as the reference frame and a continuous $N + 1$ frames as the previous frame and current frame sequence (with N frames). When predicting the first frame, we use the ground-truth of the previous frame

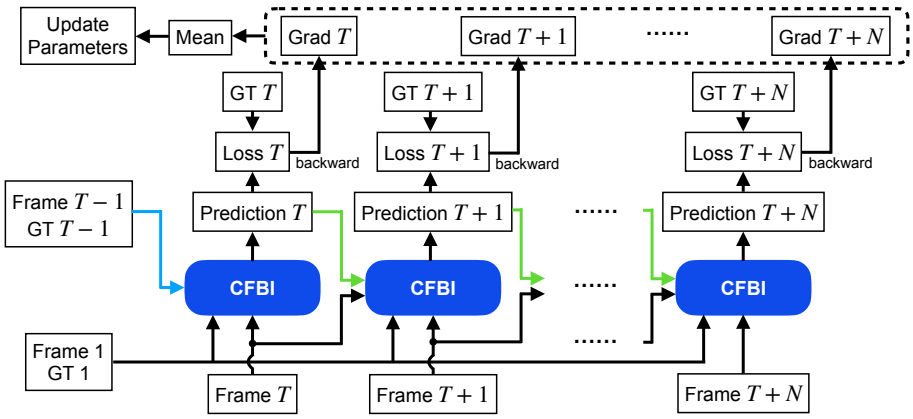


Fig. 8. An illustration of the sequential training. In each step, the previous mask comes from the previous prediction (the green lines) except for the first step, whose previous mask comes from the ground-truth (GT) mask (the blue line).

as the previous mask. When predicting the following frames, we use the latest prediction to be the previous mask. We show an illustration in Fig. 8.

In CFBI, the backbone is followed by one depth-wise separable convolution for extracting pixel-wise embedding (channels=100) with a stride of 4. We further downsample the embedding feature to a half size for the multi-local matching using bi-linear interpolation for saving GPU memory.

In CFBI+, the backbone is followed by FPN [37] for extracting three pixel-wise embeddings (channels=32, 64, and 128) with strides of 4, 8, and 16, respectively. The window sizes are $\{4, 8, 12, 16, 20, 24\}$, $\{2, 4, 6, 8, 10, 12\}$, and $\{4, 6, 8, 10\}$ for three scales (stride= 4, 8, and 16).

For the collaborative ensembler, we apply Group Normalization (GN) [47] and Gated Channel Transformation (GCT) [48] to improving training stability and performance when using a small batch size. We initialize b_B and b_F to 0. In CFBI+, each matching scale has individual b_B and b_F.

During training, we firstly downsample all the videos to $480p$ resolution, which is the same as the DAVIS default setting. We adopt SGD with a momentum of 0.9 and apply a bootstrapped cross-entropy loss, which only considers the 15% hardest pixels. In addition, we apply flipping, scaling, and balanced random-crop as data augmentations. The scaling range is from 1.0 to 1.3 times, and the cropped window size is 465×465. During the training stage, we freeze the parameters of BN in the backbone. In the testing stage, all the videos are resized to be no more than $1.3 \times 480p$ resolution, which is consistent with our training stage. For the multi-scale testing, we apply the scales of $\{1.0, 1.15, 1.3, 1.5\}$ and $\{1.5, 1.7, 1.9\}$ on YouTube-VOS, and DAVIS, respectively.

For YouTube-VOS experiments, we use a learning rate of 0.01 for $100,000$ steps with a batch size of 8 using 4 Tesla V100 GPUs. The current sequence's length is $N = 3$. The training time on YouTube-VOS is about 3 days. For training with only DAVIS, we use a learning rate of 0.006 for $50,000$ steps with a batch size of 6 videos using 2 GPUs. The current sequence's length is $N = 3$ as well. For training with both DAVIS and YouTube-VOS, we first train CFBI or CFBI+ on YouTube-VOS following the above setting. After that, we fine-tune the model on DAVIS. To avoid overfitting, we mix DAVIS videos with YouTube-VOS in a ratio of 1:2 during fine-tuning. Besides, we use a learning rate of 0.01 for $50,000$ steps with a batch size of 8 videos using 4 Tesla V100 GPUs. The current sequence's length is $N = 5$, which is slightly better than $N = 3$. We use PyTorch [53] to implement our method.

TABLE 1
The quantitative evaluation on YouTube-VOS [21]. F, S, and * denote online fine-tuning, using simulated data during training and performing model ensemble in evaluation, respectively. t/s: time per frame in seconds. $^{2\times}$: using double batch size and learning rate during training. ^{MS}: using a multi-scale and flip strategy in evaluation.

				Seen		Unseen		
Methods	F	S	Avg	$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	t/s
Validation 2018 Split								
A-GAME [CVPR19] [49]			66.1	67.8	-	60.8	-	-
PReMVOS [ACCV18] [10]	✓		66.9	71.4	75.9	56.5	63.7	6
BoLT [arXiv19] [50]	✓		71.1	71.6	-	64.3	-	1.36
STMVOS [ICCV19] [13]			68.2	-	-	-	-	-
STMVOS [ICCV19] [13]		✓	79.4	79.7	84.2	72.8	80.9	-
EGMN [ECCV20] [30]		✓	80.2	80.7	85.1	74.0	80.9	-
KMNVOS [ECCV20] [31]		✓	81.4	81.4	85.6	75.3	83.3	-
LWLVOS [ECCV20] [32]			81.5	80.4	84.9	76.4	84.4	-
CFBI			81.4	81.1	85.8	75.3	83.4	0.29
CFBI$^{2\times}$			81.8	**81.9**	86.3	75.6	83.4	0.29
CFBI+			82.0	81.2	86.0	76.2	84.6	**0.25**
CFBI+$^{2\times}$			**82.8**	81.8	**86.6**	**77.1**	**85.6**	**0.25**
CFBI^{MS}			82.7	82.2	86.8	76.9	85.0	2.51
CFBI+^{MS}			**83.3**	**82.8**	**87.5**	**77.3**	**85.7**	2.17
Testing 2019 Split								
MST* [ICCVW19] [51]		✓	81.7	80.0	83.3	77.9	85.5	-
EMN* [ICCVW19] [52]		✓	81.8	**80.7**	84.7	77.3	84.7	-
CFBI$^{2\times}$			81.6	80.2	84.6	77.2	84.5	0.30
CFBI+$^{2\times}$			**82.9**	80.6	**85.2**	**78.9**	**86.8**	**0.25**

5 Experiments

Following the previous state-of-the-art method [13], we evaluate our method on YouTube-VOS [21], DAVIS 2016 [19] and DAVIS 2017 [20]. For the evaluation on YouTube-VOS, we train our model on YouTube-VOS training split [21]. For DAVIS, we train our model on the DAVIS-2017 training split [20]. Furthermore, we provide DAVIS results using

TABLE 2
The quantitative evaluation on DAVIS 2017 [20]. ^{600p}: using 600p videos instead of 480p during inference. ‡: timing extrapolated from single-object speed assuming linear scaling in the number of objects.

Methods	F	S	Avg	$\mathcal{J}$	$\mathcal{F}$	t/s
Validation Split						
OSMN [CVPR18] [11]			54.8	52.5	57.1	0.28‡
VideoMatch [ECCV18] [28]			62.4	56.5	68.2	0.35
OnAVOS [BMVC17] [9]	✓		63.6	61.0	66.1	26
RGMP [CVPR18] [29]		✓	66.7	64.8	68.6	0.28‡
A-GAME [CVPR19] [49] (Y)			70.0	67.2	72.7	**0.14**‡
FEELVOS [CVPR19] [12] (Y)			71.5	69.1	74.0	0.51
PReMVOS [ACCV18] [10]	✓		77.8	73.9	81.7	37.6
LWLVOS [ECCV20] [32] (Y)			81.6	79.1	84.1	0.4‡
STMVOS [ICCV19] [13] (Y)		✓	81.8	79.2	84.3	0.32‡
EGMN [ECCV20] [30] (Y)		✓	82.8	**80.2**	85.2	0.4‡
KMNVOS [ECCV20] [31] (Y)		✓	82.8	80.0	85.6	0.24‡
CFBI			74.9	72.1	77.7	0.17
CFBI (**Y**)			81.9	79.3	84.5	0.17
CFBI+ (**Y**)			**82.9**	80.1	**85.7**	0.18
CFBI^{MS} (**Y**)			84.2	81.6	86.8	5.94
CFBI+^{MS} (**Y**)			**84.5**	**81.7**	**87.3**	3.94
Testing Split						
OSMN [CVPR18] [11]			41.3	37.7	44.9	0.42‡
OnAVOS [BMVC17] [9]	✓		56.5	53.4	59.6	39
RGMP [CVPR18] [29]		✓	52.9	51.3	54.4	0.42‡
FEELVOS [CVPR19] [12] (Y)			57.8	55.2	60.5	0.54
PReMVOS [ACCV18] [10]	✓		71.6	67.5	75.7	41.3
STMVOS^{600p} [ICCV19] [13] (Y)		✓	72.2	69.3	75.2	-
KMNVOS^{600p} [ECCV20] [31] (Y)		✓	77.2	74.1	80.3	-
CFBI (**Y**)			75.0	71.4	78.7	**0.19**
CFBI+ (**Y**)			75.6	71.6	79.6	**0.19**
CFBI^{600p} (**Y**)			76.6	73.0	80.1	0.35
CFBI+^{600p} (**Y**)			**78.0**	**74.4**	**81.6**	0.29

TABLE 3
The quantitative evaluation on the DAVIS-2016 validation set [19]. (**Y**) denotes additionally using YouTube-VOS for training.

Methods	F	S	Avg	$\mathcal{J}$	$\mathcal{F}$	t/s
OSMN [CVPR18] [11]			-	74.0		0.14
PML [CVPR18] [27]			77.4	75.5	79.3	0.28
VideoMatch [ECCV18] [28]			80.9	81.0	80.8	0.32
RGMP [CVPR18] [29]			68.8	68.6	68.9	0.14
RGMP [CVPR18] [29]		✓	81.8	81.5	82.0	0.14
A-GAME [CVPR19] [49] (Y)			82.1	82.2	82.0	**0.07**
FEELVOS [CVPR19] [12] (Y)			81.7	81.1	82.2	0.45
OnAVOS [BMVC17] [9]	✓		85.0	85.7	84.2	13
PReMVOS [ACCV18] [10]	✓		86.8	84.9	88.6	32.8
STMVOS [ICCV19] [13] (Y)		✓	89.3	88.7	89.9	0.16
KMNVOS [ECCV20] [31] (Y)		✓	**90.5**	**89.5**	**91.5**	0.12
CFBI			86.1	85.3	86.9	0.16
CFBI (**Y**)			89.4	88.3	90.5	0.16
CFBI+ (**Y**)			89.9	88.7	91.1	0.17

both DAVIS 2017 and YouTube-VOS for training following some latest works [12], [13].

The evaluation metric is $\mathcal{J}$ score, calculated as the average IoU between the prediction and the ground truth mask, and $\mathcal{F}$ score, calculated as an average boundary similarity measure between the boundary of the prediction and the ground truth, and their average value ($\mathcal{J}\&\mathcal{F}$). We evaluate our results on the official evaluation server or use official tools.

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

YouTube-VOS [21] is the latest large-scale dataset for multi-object video segmentation. Compared to the popular DAVIS benchmark, which consists of 120 videos, YouTube-VOS is about 37 times larger. In detail, YouTube-VOS contains 3471 videos in training split (65 categories), 507 videos in validation split (additional 26 unseen categories), and 541 videos in testing split (additional 29 unseen categories). Due to the existence of unseen object categories, the YouTube-VOS validation split is much suitable for measuring the generalization ability of VOS methods.

As shown in Table 1, we compare our method with the latest VOS methods on both Validation 2018 and Testing 2019 splits. Without using any bells and whistles, like fine-tuning at test time [8], [9] or pre-training on larger augmented simulated data [13], [29], [30], [31], our CFBI+ achieves an average score of 82.0%, which significantly outperforms all other methods in every evaluation metric. We can improve the performance of CFBI+ to $\mathbf{82.8}\%$ (CFBI+$^{2\times}$) by using a stronger training schedule with a double batch size and learning rate. Especially, CFBI+'s multi-object inference speed is much faster than BoLT ($1.36s$).

Benefit from the multi-scale matching, our CFBI+ is robuster (82.0% *vs.*81.4%) and more efficient ($0.25s$ *vs.*$0.29s$) than CFBI. Especially, CFBI+ uses only half of the training batch size to exceed CFBI$^{2\times}$. Besides, the $\mathbf{82.8}\%$ result is significantly higher (1.4%) than KMNVOS, which follows STMVOS to use extensive simulated data for training. Without simulated data, the performance of STMVOS will drop a lot from 79.4% to 68.2%. Moreover, we can further boost the performance of CFBI+ to $\mathbf{83.3}\%$ by applying a multi-scale and flip strategy during the evaluation.

We also compare our method with two of the best results on Testing 2019 split, *i.e.*, *Rank 1* (EMN [52]) and *Rank 2* (MST [51]) results in the 2nd Large-scale Video Object Segmentation Challenge. Without using model ensemble, simulated data or testing-stage augmentation, our CFBI+ ($\mathbf{82.9}\%$) significantly outperforms the *Rank 1* result (81.8%) while maintaining an efficient multi-object speed of 4 FPS. Notably, the improvement of CFBI+ mainly comes from the unseen categories ($\mathbf{78.9}\%\mathcal{J}/\mathbf{86.8}\%\mathcal{F}$ *vs.*$77.3\%\mathcal{J}/84.7\%\mathcal{F}$) instead of seen. Such a strong result further demonstrates CFBI+'s generalization ability and effectiveness.

DAVIS 2017 [20] is a multi-object extension of DAVIS 2016. The validation split of DAVIS 2017 consists of 59 objects in 30 videos. And the training split contains 60 videos. Compared to YouTube-VOS, DAVIS is much smaller and easy to be over-fitted.

As shown in Table 2, CFBI+ exceeds KMNVOS and EGMN ($\mathbf{82.9}\%$ *vs.*82.8%) without using simulated data. Moreover, CFBI+ achieves a faster multi-object inference speed ($\mathbf{0.18}s$) than KMNVOS (0.24%). Different from KMNVOS and EGMN, the backbone features of CFBI+ or CFBI is shared for all the objects in each frame, which leads to a more efficient multi-object inference. The augmentation in evaluation can further boost CFBI+ to a higher score of $\mathbf{84.5}\%$.

We also evaluate our method on the DAVIS-2017 testing

TABLE 4
Ablation of background embedding on the DAVIS-2017 validation split. P and I denote the pixel-level matching and instance-level attention, respectively. *: removing the foreground and background bias.

P	I	Avg	$\mathcal{J}$	$\mathcal{F}$
✓	✓	74.9	72.1	77.7
✓*	✓	72.8	69.5	76.1
✓		73.0	69.9	76.0
	✓	72.3	69.1	75.4
		70.9	68.2	73.6

TABLE 5
Ablation of atrous matching. We evaluate the speed and performance of CFBI on the YouTube-VOS validation split using different atrous matching factors (l). $l = 1$ is equivalent to original matching.

l	1	2	3	4
Global Matching				
Avg	81.4	81.3	80.7	79.9
t/s	0.29	0.15	0.13	0.12
Multi-local Matching				
Avg	81.4	80.8	80.1	79.5
t/s	0.29	0.26	0.25	0.25

TABLE 6
Ablation of multi-scale matching. We evaluate the speed and performance of our methods on the YouTube-VOS validation split. S: the stride of feature maps. G: applying foreground-background global matching. L: applying foreground-background multi-local matching. l: atrous factor.

Name	S=4	S=8	S=16	Avg	t/s
CFBI-S4	$G\ L^{l=2}$			81.6	0.65
CFBI-S4-G2	$G^{l=2}\ L^{l=2}$			81.6	0.27
CFBI-S8		$G\ L$		80.9	0.13
CFBI-S16			$G\ L$	78.3	0.11
CFBI	G	L		81.4	0.29
CFBI-G2	$G^{l=2}$	L		81.3	0.15
CFBI+	$G^{l=2}\ L^{l=2}$	$G\ L$	$G\ L$	82.0	0.25

TABLE 7
Ablation of other components on the DAVIS-2017 validation split.

	Ablation	Avg	$\mathcal{J}$	$\mathcal{F}$
0	Ours (CFBI)	74.9	72.1	77.7
1	w/o multi-local windows	73.8	70.8	76.8
2	w/o sequential training	73.3	70.8	75.7
3	w/o collaborative ensembler	73.3	70.5	76.1
4	w/o balanced random-crop	72.8	69.8	75.8
5	w/o instance-level attention	72.7	69.8	75.5
6	baseline (reproduced FEELVOS)	68.3	65.6	70.9

split, which is much more challenging than the validation split. On the testing split, we outperform KMNVOS (77.2%) by **0.8%** under the setting proposed by STMVOS (*i.e.*, evaluating on $600p$ resolution). When evaluating on the default $480p$ resolution of DAVIS, CFBI+ or CFBI is much better than the STMVOS using $600p$ resolution (75.6% or 75.0% *vs.*72.2%). These strong results further prove the generalization ability of CFBI+ and CFBI.

The speed of CFBI+ is only comparable with CFBI when evaluating using a $480p$ resolution on DAVIS. The reason for this is that convolutional layers have a larger proportion of calculations when evaluating on a small resolution. And CFBI+ has more convolutional layers than CFBI because of introducing FPN. On larger resolution (*e.g.*, $600p$), CFBI+ is faster than CFBI ($0.29s$ *vs.*$0.35s$).

DAVIS 2016 [19] contains 20 videos annotated with high-quality masks each for a single target object. We compare our CFBI method with state-of-the-art methods in Table 3. On the DAVIS-2016 validation split, our CFBI+ trained with the additional YouTube-VOS training split achieves an average score of **89.9**%, which is slightly worse than KMNVOS 90.5%, a method using simulated data as mentioned before. Since the amount of data in DAVIS is tiny, using additional simulation data can help alleviate over-fitting. Compare to a much fair baseline (*i.e.*, FEELVOS) whose setting is closer to ours, the proposed CFBI+ not only achieves a much better accuracy (**89.9**% *vs.*81.7%) but also maintains a much faster inference speed ($0.17s$ *vs.*$0.45s$).

5.2 Ablation Study

We analyze the ablation effect of each component proposed in CFBI on the DAVIS-2017 validation split. Following FEELVOS, we use only the DAVIS-2017 training split as training data for these experiments.

Background Embedding. As shown in Table 4, we first analyze the influence of removing the background embedding while keeping the foreground only. Without any background mechanisms, the result of our method heavily drops from 74.9% to 70.9%. This result shows that it is significant to embed both foreground and background features collaboratively. Besides, the missing of background information in the pixel-level matching or the instance-level attention will decrease the result to 73.0% or 72.3% separately. Thus, compared to instance-level attention, the pixel-level matching performance is more sensitive to the effect of background embedding. A possible reason for this phenomenon is that the possibility of existing some background pixels similar to the foreground is higher than some background instances. Finally, we remove the foreground and background bias, b_F and b_B, from the distance metric, and the result drops to 72.8%, which further shows that the distance between foreground pixels and the distance between background pixels should be separately considered.

Atrous Matching. As shown in Table 5, the performance of CFBI will decrease, and the speed will increase as the atrous factor (l) increases. Compared to the original matching, 2-atrous matching will significantly accelerate inference speed, but the performance will only slightly decrease. The speed will no longer be fast improved by further increasing l, and the performance will be heavily decreased. Compared 2-atrous multi-local matching, 2-atrous global matching has a nearly identical performance as the original global matching (81.3%*vs.*81.4%) but greatly accelerates the speed by 93%. In short, Atrous Matching can significantly improve the efficiency of matching processes, especially for global matching.

Multi-scale Matching. Table 6 shows the ablation study of multi-scale matching. In CFBI experiments, the channel

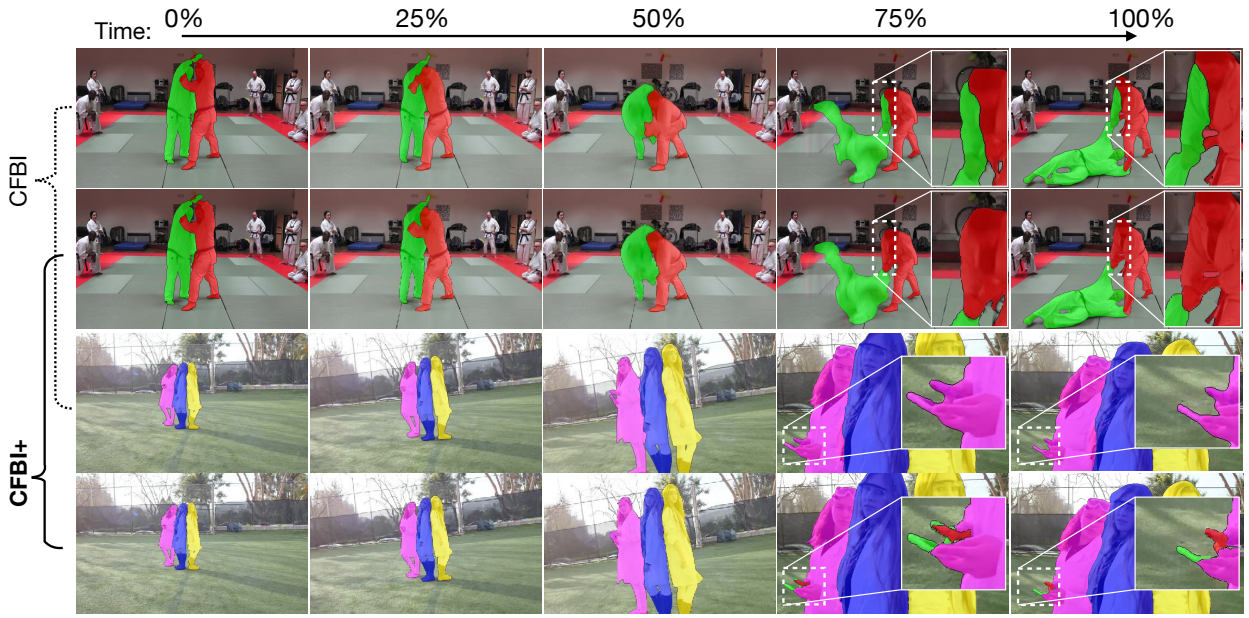


Fig. 9. Qualitative comparison between CFBI and CFBI+ on the DAVIS-2017 validation split. In the first video, CFBI fails to segment one hand of the right person (the white box), while CFBI+ generates an accurate boundary between two similar persons. In the second video, CFBI entirely loses two tiny objects (cellphones). In contrast, CFBI+ successfully predicts their masks.

dimension of pixel-wise features is 100. In CFBI+ experiments, the channel dimensions are 32, 64, and 128 for $S = 4$, $S = 8$, and $S = 16$, respectively. We first evaluate the difference between different matching scales. As shown, doing matching in larger scales leads to better performance but takes much more inference time. CFBI-S4 is much powerful than CFBI-S16 (81.6% *vs.*78.3%). However, CFBI-S16 is about 5 times faster than CFBI-S4. For better efficiency, CFBI brings the multi-local matching of CFBI-S4 from $S = 4$ to $S = 8$, which improves 124% speed and loses only 0.2% performance. If we want to do matching on larger scales, Atrous global matching ($l = 2$) is critic in saving computational resources (CFBI-G2 $0.15s$ *vs.*CFBI $0.29s$, CFBI-S4-G2 $0.27s$ *vs.*CFBI-S4 $0.65s$) while losing little performance (CFBI-G2 81.3% *vs.*CFBI 81.4%, CFBI-S4-G2 81.6% *vs.*CFBI-S4 81.6%). Finally, by combining all the matching processes on three scales and progressively increasing their channel dimensions, CFBI+ achieves better performance (82.0%) while proposed atrous matching helps guarantee an efficient speed ($0.25s$).

Qualitative Comparison. To further compare CFBI with CFBI+, we visualize some representative comparison results on the DAVIS-2017 validation split in Fig. 9. Benefit from the local matching on larger scales, CFBI+ can generate a more accurate boundary between similar targets. Moreover, CFBI+ is capable of predicting some tiny objects, which are difficult for CFBI. In addition, we show more results of CFBI+ under some of the hardest cases on the DAVIS-2017 testing split and YouTube-VOS in Fig. 10. CFBI generalize well on most of these cases, including similar objects, small objects, and occlusion.

Other Components. The ablation study of other proposed components is shown in Table 7. Line 0 (74.9%) is the result of proposed CFBI, and Line 6 (68.3%) is our baseline method reproduced by us. Under the same setting, our CFBI significantly outperforms the baseline.

In line 1, we use only one local neighborhood window to conduct the local matching following the setting of FEELVOS, which degrades the result from 74.9% to 73.8%. It demonstrates that our multi-local matching module is more robust and effective than the single-local matching module of FEELVOS. Notably, the computational complexity of multi-local matching dominantly depends on the biggest local window size because we use the intermediate results of the local matching of the biggest window to calculate on smaller windows.

In line 2, we replace our sequential training by using ground-truth masks instead of network predictions as the previous mask. By doing this, the performance of CFBI drops from 74.9% to 73.3%, which shows the effectiveness of our sequential training under the same setting.

In line 3, we replace our collaborative ensembler with 4 depth-wise separable convolutional layers (and we keep applying instance-level attention before each separable convolutional layer). This architecture is the same as the dynamic segmentation head of [12]. Compared to our collaborative ensembler, the dynamic segmentation head has much smaller receptive fields and performs 1.6% worse.

In line 4, we use normal random-crop instead of our balanced random-crop during the training process. In this situation, the performance drops by 2.1% to 72.8% as well. As expected, our balanced random-crop is successful in relieving the model form biasing to background attributes.

In line 5, we disable the use of instance-level attention as guidance information to the collaborative ensembler, which means we only use pixel-level information to guide the prediction. In this case, the result deteriorates even further to 72.7, which proves that instance-level information can further help the segmentation with pixel-level information.

In summary, we explain the effectiveness of each proposed component of CFBI and CFBI+. For VOS, it is necessary to embed both foreground and background features.

Fig. 10. Qualitative results of CFBI+ on the DAVIS-2017 testing split and YouTube-VOS 2018 validation split. These videos cover many of the most challenging VOS cases, including similar objects, small objects, occlusion, and blur. In the first four videos, CFBI+ performs well on similar objects, small objects, and occlusion. Especially for the third video, there are 10 similar targets (dancers) in total and many similar people in the background. Besides, all the targets continuously occlude each other. However, CFBI+ does not collapse under such a complicated case and correctly track every dancer. In the last video, we show one of the worst cases on the DAVIS-2017 testing split, where CFBI+ fails to segment all the motorbike parts. The blur caused by such a strong halo makes it difficult for CFBI+ to distinguish the motorbike's appearance.

Besides, the model will be more robust by combining pixel-level information and instance-level information. Notably, multi-scale pixel-level matching is much more potent than single-scale, and atrous matching is critical for improving matching efficiency. Apart from this, the proposed balanced random-crop and sequential training are useful but straightforward in improving training performance.

6 Conclusion

This paper proposes a novel framework for video object segmentation by introducing collaborative foreground-background integration and achieves new state-of-the-art results on three popular benchmarks. Specifically, we impose the feature embedding from the foreground target and its corresponding background to be contrastive. Moreover, we integrate both pixel-level and instance-level embeddings to make our framework robust to various object scales while keeping the network simple and fast. In particular, our design of multi-scale matching can further improve VOS's performance, and our atrous matching can greatly improve the efficiency of matching processes. We hope CFBI or CFBI+ will serve as a solid baseline and help ease the future research of VOS and related areas, such as video object tracking and interactive video editing.

Acknowledgements. This work is partly supported by ARC DP200100938 and ARC DECRA DE190101315.

References

[1] K. N. Ngan and H. Li, *Video segmentation and its applications*. Springer Science & Business Media, 2011. 1
[2] Z. Zhang, S. Fidler, and R. Urtasun, "Instance-level segmentation for autonomous driving with deep densely connected mrfs," in *CVPR*, 2016, pp. 669–677. 1
[3] L. Yang, Y. Fan, and N. Xu, "Video instance segmentation," in *ICCV*, 2019, pp. 5188–5197. 1
[4] Q. Feng, Z. Yang, P. Li, Y. Wei, and Y. Yang, "Dual embedding learning for video instance segmentation," in *ICCV Workshops*, 2019. 1
[5] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim, "Fast user-guided video object segmentation by interaction-and-propagation networks," in *CVPR*, 2019, pp. 5247–5256. 1
[6] J. Miao, Y. Wei, and Y. Yang, "Memory aggregation networks for efficient interactive video object segmentation," in *CVPR*, 2020. 1
[7] C. Liang, Z. Yang, J. Miao, Y. Wei, and Y. Yang, "Memory aggregated cfbi+ for interactive video object segmentation," in *CVPR Workshops*, 2020. 1
[8] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool, "One-shot video object segmentation," in *CVPR*, 2017, pp. 221–230. 1, 2, 8
[9] P. Voigtlaender and B. Leibe, "Online adaptation of convolutional neural networks for video object segmentation," in *BMVC*, 2017. 1, 2, 8
[10] J. Luiten, P. Voigtlaender, and B. Leibe, "Premvos: Proposal-generation, refinement and merging for video object segmentation," in *ACCV*, 2018, pp. 565–580. 1, 2, 7, 8
[11] L. Yang, Y. Wang, X. Xiong, J. Yang, and A. K. Katsaggelos, "Efficient video object segmentation via network modulation," in *CVPR*, 2018, pp. 6499–6507. 1, 2, 8
[12] P. Voigtlaender, Y. Chai, F. Schroff, H. Adam, B. Leibe, and L.-C. Chen, "Feelvos: Fast end-to-end embedding learning for video object segmentation," in *CVPR*, 2019, pp. 9481–9490. 1, 2, 3, 8, 10

[13] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim, "Video object segmentation using space-time memory networks," in *ICCV*, 2019. 1, 2, 7, 8

[14] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *IJCV*, vol. 88, no. 2, pp. 303–338, 2010. 1

[15] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*. Springer, 2014, pp. 740–755. 1, 6

[16] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. Torr, and S.-M. Hu, "Global contrast based salient region detection," *TPAMI*, vol. 37, no. 3, pp. 569–582, 2014. 1

[17] J. Shi, Q. Yan, L. Xu, and J. Jia, "Hierarchical image saliency detection on extended cssd," *TPAMI*, vol. 38, no. 4, pp. 717–729, 2015. 1

[18] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in *ICCV*. IEEE, 2011, pp. 991–998. 1

[19] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung, "A benchmark dataset and evaluation methodology for video object segmentation," in *CVPR*, 2016, pp. 724–732. 2, 7, 8, 9

[20] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool, "The 2017 davis challenge on video object segmentation," *arXiv preprint arXiv:1704.00675*, 2017. 2, 7, 8

[21] N. Xu, L. Yang, Y. Fan, D. Yue, Y. Liang, J. Yang, and T. Huang, "Youtube-vos: A large-scale video object segmentation benchmark," *arXiv preprint arXiv:1809.03327*, 2018. 2, 4, 7, 8

[22] Z. Yang, Y. Wei, and Y. Yang, "Collaborative video object segmentation by foreground-background integration," in *ECCV*, 2020. 2

[23] H. Xiao, J. Feng, G. Lin, Y. Liu, and M. Zhang, "Monet: Deep motion exploitation for video object segmentation," in *CVPR*, 2018, pp. 1140–1148. 2

[24] F. Perazzi, A. Khoreva, R. Benenson, B. Schiele, and A. Sorkine-Hornung, "Learning video object segmentation from static images," in *CVPR*, 2017, pp. 2663–2672. 2

[25] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *ICCV*, 2015, pp. 2758–2766. 2

[26] J. Cheng, Y.-H. Tsai, W.-C. Hung, S. Wang, and M.-H. Yang, "Fast and accurate online video object segmentation via tracking parts," in *CVPR*, 2018, pp. 7415–7424. 2

[27] Y. Chen, J. Pont-Tuset, A. Montes, and L. Van Gool, "Blazingly fast video object segmentation with pixel-wise metric learning," in *CVPR*, 2018, pp. 1189–1198. 2, 3, 8

[28] Y.-T. Hu, J.-B. Huang, and A. G. Schwing, "Videomatch: Matching based video object segmentation," in *ECCV*, 2018, pp. 54–70. 2, 8

[29] S. Wug Oh, J.-Y. Lee, K. Sunkavalli, and S. Joo Kim, "Fast video object segmentation by reference-guided mask propagation," in *CVPR*, 2018, pp. 7376–7385. 2, 8

[30] X. Lu, W. Wang, M. Danelljan, T. Zhou, J. Shen, and L. Van Gool, "Video object segmentation with episodic graph memory networks," in *ECCV*, 2020. 2, 7, 8

[31] H. Seong, J. Hyun, and E. Kim, "Kernelized memory network for video object segmentation," in *ECCV*, 2020. 2, 7, 8

[32] G. Bhat, F. J. Lawin, M. Danelljan, A. Robinson, M. Felsberg, L. Van Gool, and R. Timofte, "Learning what to learn for video object segmentation," in *ECCV*, 2020. 2, 7, 8

[33] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *ICML*, 2017. 2

[34] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *ICML*, 2017. 2

[35] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *CVPR*, 2018. 2, 4

[36] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *ECCV*, 2018, pp. 801–818. 5, 6

[37] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *CVPR*, 2017, pp. 2117–2125. 5, 7

[38] A. Torralba, "Contextual priming for object detection," *IJCV*, vol. 53, no. 2, pp. 169–191, 2003. 5

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016. 5

[40] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *TPAMI*, vol. 40, no. 4, pp. 834–848, 2017. 5

[41] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *CVPR*, 2017, pp. 2881–2890. 5

[42] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian, "A real-time algorithm for signal analysis with the help of the wavelet transform," in *Wavelets*. Springer, 1990, pp. 286–297. 5

[43] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015. 5

[44] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *CVPR*, 2017, pp. 1251–1258. 6

[45] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015. 6

[46] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*. Ieee, 2009, pp. 248–255. 6

[47] Y. Wu and K. He, "Group normalization," in *ECCV*, 2018, pp. 3–19. 7

[48] Z. Yang, L. Zhu, Y. Wu, and Y. Yang, "Gated channel transformation for visual recognition," 2020. 7

[49] J. Johnander, M. Danelljan, E. Brissman, F. S. Khan, and M. Felsberg, "A generative appearance model for end-to-end video object segmentation," in *CVPR*, 2019, pp. 8953–8962. 7, 8

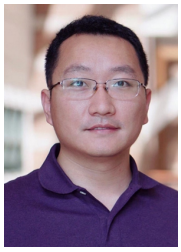
[50] P. Voigtlaender, J. Luiten, and B. Leibe, "Boltvos: Box-level tracking for video object segmentation," *arXiv preprint arXiv:1904.04552*, 2019. 7

[51] Q. Zhou, Z. Huang, L. Huang, Y. Gong, H. Shen, W. Liu, and X. Wang, "Motion-guided spatial time attention for video object segmentation," in *ICCV Workshops*, 2019. 7, 8

[52] Z. Zhou, L. Ren, P. Xiong, Y. Ji, P. Wang, H. Fan, and S. Liu, "Enhanced memory network for video segmentation," in *ICCV Workshops*, 2019. 7, 8

[53] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017. 7

Zongxin Yang is currently a Ph.D. student with the University of Technology Sydney. He received his Bachelor degree from University of Science and Technology of China, Hefei, China, in 2018. His current research interest is computer vision, including image recognition and video object segmentation.



Yunchao Wei is currently an Assistant Professor with the University of Technology Sydney. He received his Ph.D. degree from Beijing Jiaotong University, Beijing, China, in 2016. He was a Postdoctoral Researcher at Beckman Institute, UIUC, from 2017 to 2019. He is ARC Discovery Early Career Researcher Award Fellow from 2019 to 2021. His current research interests include computer vision and machine learning.

Yi Yang is currently a Professor with the University of Technology Sydney. He received his PhD degree in from Zhejiang University, Hangzhou, China, in 2010. He was a post-doctoral researcher in the School of Computer Science, Carnegie Mellon University. His current research interests include machine learning and its applications to multimedia content analysis and computer vision, such as multimedia retrieval and video content understanding.