Video Object Segmentation by Learning Location-Sensitive Embeddings

Hai Ci[1], Chunyu Wang[2], and Yizhou Wang[1,3]

[1] EECS, Peking University, Beijing, China
{cihai, yizhou.wang}@pku.edu.cn
[2] Microsoft Research Asia, Beijing, China
chnuwa@microsoft.com
[3] Deepwise AI Lab, Beijing, China

Abstract. We address the problem of video object segmentation which outputs the masks of a target object throughout a video given only a bounding box in the first frame. There are two main challenges to this task. First, the background may contain similar objects as the target. Second, the appearance of the target object may change drastically over time. To tackle these challenges, we propose an end-to-end training network which accomplishes foreground predictions by leveraging the location-sensitive embeddings which are capable to distinguish the pixels of similar objects. To deal with appearance changes, for a test video, we propose a robust model adaptation method which pre-scans the whole video, generates pseudo foreground/background labels and retrains the model based on the labels. Our method outperforms the state-of-the-art methods on the DAVIS and the SegTrack v2 datasets.

Keywords: Video object segmentation · Location-sensitive embeddings

1 Introduction

Video Object Segmentation (VOS) [3, 24, 31, 6, 17, 16] is an important problem in video understanding which estimates the masks of an object in a video given the masks in the first frame. The estimated masks can be used by higher level applications such as video editing. If only a bounding box of which object to segment is provided for the first frame, we denote the task as Box-based Video Object Segmentation (BVOS). Compared to VOS, BVOS provides an easier interface for users because it doesn't require cumbersome labelings.

BVOS is related to foreground extraction (FE) [2, 27] in static images which extracts the mask of the foreground object in a complex environment. The target of FE is to minimize the interactive efforts on the part of users. If only a bounding box of the foreground object is provided, it is similar to BVOS except that BVOS also needs to extract foreground masks for the consecutive frames without bounding boxes. BVOS is also related to object tracking [14, 1] which aims to localize the target object by bounding boxes in a video. In contrast, BVOS needs to generate sharp foreground masks.

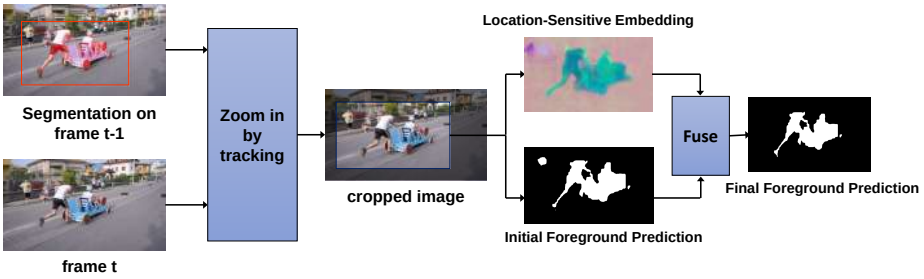

Fig. 1. The pipeline of the proposed method. (1) Based on the segmentation results of the previous frame, the method first zooms in to the probable foreground region by cropping and resizing in the current image. (2) The cropped image is fed to a network to predict the location-sensitive embeddings and initial foreground predictions. (3) The embeddings are fused with the foreground predictions to obtain final segmentation.

The FE methods such as [2, 27] model foreground and background distributions by low level features. For many cases, they can give satisfying results. But as shown in their work, failures may occur when the foreground and background have low contrast. This is because these methods use low level features and lack high level understandings of objects.

Recently, convolutional neural networks have demonstrated outstanding performance in many areas of computer vision. In particular, it is shown in [21, 5] that the pixel-wise semantic segmentation can be achieved by Fully Convolutional Networks (FCN) which are robust to appearance variations. The success inspires [12] to estimate the (dominant) foreground pixels directly from a detected image patch which tightly contains the object.

In VOS, the authors in [3] propose to estimate the foreground pixels in an image without first detecting the object. Since the target object may not be dominant in the image, this is more difficult than estimating from patches [12]. They [3] propose to retrain the model based on the first frame of the video, which helps to learn the specific features of the target object and alleviate this problem. However, this model has limited power to distinguish the objects of the same class especially when new objects enter the scene. The main reason is that the first frame lacks sufficient negative training data, i.e., the other objects of the same class.

Another challenge in VOS is that the appearance of the target object may vary drastically over time, for example, due to scale changes, occlusions and viewpoint changes. The authors in [30] propose to update their model in an online manner where the estimated foreground/background pixels in the current frame are used to retrain the network before applying to the following frame. This enables the network to adapt to the new features in the video sequence. However, it is a challenge for the method to choose the appropriate pixels and the predicted labels for finetuning the network as erroneous labels may mislead the network to learn wrong features.

1.1 Method Overview

In this section, we present an overview of our approach. We first describe how we perform foreground predictions by leveraging the location-sensitive embeddings assuming that the model has already been learned or adapted. Then in the second part, we describe how to perform model adaptation.

Foreground Predictions We present an end-to-end training network for foreground predictions which leverages location-sensitive embeddings (LSEs) to distinguish the pixels of different objects. The framework is shown in Fig. 1. First, given a bounding box in the previous frame (obtained from foreground predictions), we first zoom in to the highly probable foreground region in the current frame in order to improve the predictions for small objects. Then the cropped image is fed to a fully convolutional network to generate foreground predictions and location-sensitive embeddings. Finally, the two branches are combined to generate the final foreground predictions.

The LSEs are targetted to distinguish objects with similar appearance. We learn the embeddings by an FCN network aiming that the pixels of the same object are pushed closer and those of different objects are pulled further. Since FCN is translation invariant, it is not trivial to learn distinct embeddings for very similar objects in different locations of an image. So we propose to jointly regress the center of the object to which the pixel belongs. By fusing the appearance and location features, we obtain LSEs with improved discriminative power. Different from foreground predictions, the LSEs can be learned on the large scale static image datasets and transferred to the video segmentation dataset.

Model Adaptation Adapting the model to the consecutive frames of a video is important. To robustly adapt the model, we propose an approach which scans a video twice and adapts the model once. In the first scan, we obtain coarse object segmentations which are used as pseudo-labels to retrain the network. The retrained model will be applied to each frame without adaptation any more. This alleviates the problem of error accumulation which is common for online update. Note that a very small portion of erroneous labels would not significantly degrade the performance of the network. The method is simple and has few hyperparameters to tune. But it outperforms the state-of-the-arts on two datasets.

2 Related Work

On Feature Embeddings Learning embeddings resembles a particular type of clustering methods where the pixels of the same label are pushed closer and those of different labels are pulled apart. This has been extensively explored in the area of clustering [28], human pose estimation [23], segmentation [11] and instance segmentation [8, 4].

In [28], the authors propose to embed a face image into a vector (on a unit sphere). The training data are in the form of triplets where the distance between a positive pair is smaller than that of a negative pair by a margin. Then the authors in [8, 20] extend the work to learn pixel-wise embeddings for instance segmentation. They use the embeddings to group the pixels into segments which are postprocessed for instance segmentation. The differences between our work and [8, 20] are two folds. First, our embeddings are location sensitive, which provide enhanced power to distinguish very similar objects in different locations. To the best of our knowledge, this hasn't been explored before. Second, we combine the embeddings with foreground predictions in an end-to-end way, rather than using the embeddings as postprocessing.

On Model Update Online model adaptation has been extensively studied for bounding box level tracking [9, 18, 26]. But the online adaptation for pixel level models has been rarely touched. In [30], the authors select the pixels for which the predicted foreground probability exceeds a certain threshold as positive examples. The negative examples are carefully selected to be those which are very far away (based on a threshold) from the last predicted object mask. The reason for applying such a complex rule-based selection method is that the erroneous supervisions may bias the network to learn wrong features. The online adaptation method differs from ours in that they update the model at each time step. In contrast, we update the model only once which is more robust to small portions of inaccurate supervisions and less dependent on parameter choices.

3 Location-Sensitive Embeddings

The LSE of a pixel not only relies on its appearance but also relies on its spatial location relative to the object it belongs to. This allows the embedding to distinguish very similar objects in different locations. We jointly learn the appearance embedding x_i and location embedding l_i for each pixel p_i which are fused to generate location-sensitive embeddings.

Denote the instance labels for the two pixels p_i and p_j as y_i and y_j, respectively. If p_i and p_j are from the same instance, then $y_i = y_j$. Denote the location embeddings for the two pixels as l_i and l_j, respectively. A location embedding l_i encodes the spatial offset between the pixel and the center of the object to which the pixel belongs. In addition, $c_i = l_i + p_i$ gives the object center where p_i presents the pixel location in the image in this context. We can imagine that c_i is the same as c_j if two pixels belong to the same object.

We propose to jointly learn the appearance and location embeddings, which are fused to generate LSEs. In the following subsections, we will introduce the details for learning appearance and location embeddings.

3.1 Appearance Embeddings

If two pixels are from the same object in an image, they form a positive training pair. Otherwise, they form a negative training pair. The appearance embedding

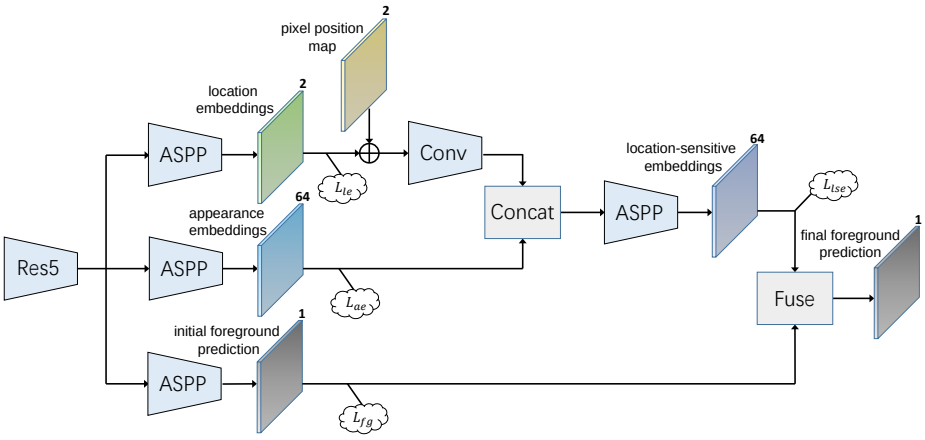

Fig. 2. The figure shows the overall network structure of the proposed method. *ASPP* represents the Atrous Spatial Pyramid Pooling layer introduced in [5].

branch is the same as [8]. We measure the similarity between two appearance embeddings x_i and x_j using the following similarity: $s_{i,j} = \frac{2}{1+\exp(\|x_i - x_j\|_2^2)}$.

The goal is to learn an embedding so that the pixels of the same object ($y_i = y_j$) have similar embeddings ($s_{i,j} = 1$). To avoid a trivial solution where all the embeddings are the same, we use an additional term which requires that the embeddings of different objects ($y_i \neq y_j$) are far apart. The total loss over all pairs and training images is:

$$L_{ae} = -\frac{1}{|S|} \sum_{i,j \in S} \omega_{ij} [1_{\{y_i = y_j\}} \log(s(i,j)) + 1_{\{y_i \neq y_j\}} \log(1 - s(i,j))] \quad (1)$$

where S is the set of the selected training positive and negative pairs. Involving all pixels of an image for loss computation is not practical due to the limitations in GPU memories. We randomly choose K pixels for each object instance in the image. ω_{ij} is the weight for that particular pair which is set to be a value inversely proportional to the size of the instance which the pixels belong to. This is to guarantee that the loss will not ignore the small instances.

3.2 Location Embeddings

We measure the distance between the predicted location embedding $\hat{l}_i$ and the groundtruth l_i using Euclidean distance: $d_i = \|l_i - \hat{l}_i\|_2$. Note that adding the location of each pixel p_i to the embedding $c_i = p_i + l_i$ gives the location of the object center. The pixels of the same object have the same c rather than l. But we don't directly estimate c because it is difficult to learn for a fully convolutional network which is translation invariant.

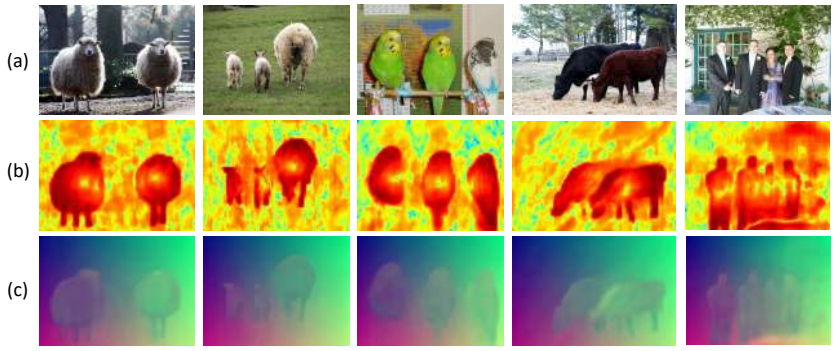

Fig. 3. Sample location embeddings on the PASCAL dataset. (a) shows five challenging test images. (b) shows the magnitude maps of the estimated spatial offsets ($\|l_i\|_2$). Ideally, the values in object centers are zero and increase gradually towards boundary. (c) shows the object center maps c_i. Pixels of the same object should have similar colors and those of different objects should have different colors.

Loss Function Different from appearance embeddings, we don't need to construct pairs of data for training because every foreground pixel has a fixed location embedding. The embeddings of background pixels are set to zero during training. The total loss over all training images is:

$$L_{le} = \frac{1}{M} \sum_{k=1}^{M} \frac{1}{U_k} \sum_{i=1}^{U_k} d_i, \tag{2}$$

where M is the number of training images and U_k is the number of non-background pixels in one image. Fig. 3 demonstrates several samples for location embeddings.

3.3 Combine Appearance and Location Embeddings for LSEs

We first add a convolution layer to embed the object center map into a higher dimensional space which is then concatenated with the appearance embeddings. The concatenations are fed to an ASPP layer to generate the LSEs. See Fig. 2. We add a loss L_{lse} which is similar to L_{ae} after the outputs of LSEs.

We learn the embeddings by jointly optimizing $L_{em} = L_{ae} + L_{le} + L_{lse}$. We show several LSE examples in Fig. 4. For visualization purposes, we project the 64-dimensional embeddings to RGB space. Similar embeddings are mapped to have similar colors. We can see from the camel and car examples that the method is capable to distinguish the objects of the same class.

4 Video Object Segmentation Method

The framework of the proposed segmentation method is shown in Fig. 1. First, given the estimated bounding box in the previous frame, we first zoom in to the

Fig. 4. LSEs visualization by randomly projecting the 64-dimensional embeddings into RGB space. The embeddings can preserve the details around edges and can distinguish objects with similar appearance. See the camel and car examples.

probable foreground regions in the current frame. Second, the cropped image is fed to a network to generate initial foreground predictions and LSEs which are combined to obtain the foreground.

4.1 Zooming In to See Clearer

The benefits of zooming into a tight region surrounding the object are two folds. First, the target object becomes more dominant. Second, the size of the target object is increased and more details can be preserved which improves the sharpness of the predicted masks.

Since we don't have groundtruth bounding boxes (except for the first frame) for a video, we propose to take advantage of the foreground predictions of the previous frame to coarsely estimate a bounding box for the current frame. Suppose the foreground predictions for frame t are o_t. Then we can get a bounding box $b_t^{(i)}$ by identifying the left/right/top/bottom most foreground pixels. Fig. 7 shows some estimated bounding boxes. The estimations generally have high qualities. Then we propagate $b_t^{(i)}$ to frame $t+1$ as $b_{t+1}^{(p)}$. Considering the objects move smoothly in a video, we simply require $b_{t+1}^{(p)}$ is an enlarged version of $b_t^{(i)}$. It is worth noting that the estimated bounding box is not necessarily very accurate as long as all object pixels are within the box.

We crop frame $t+1$ according to $b_{t+1}^{(p)}$ and resize it to 321×321. After feeding it to the network, we obtain the foreground prediction from which we infer a tight box $b_{t+1}^{(i)}$ and propagate it to frame $t+2$. We repeat the procedure util reaching the last frame.

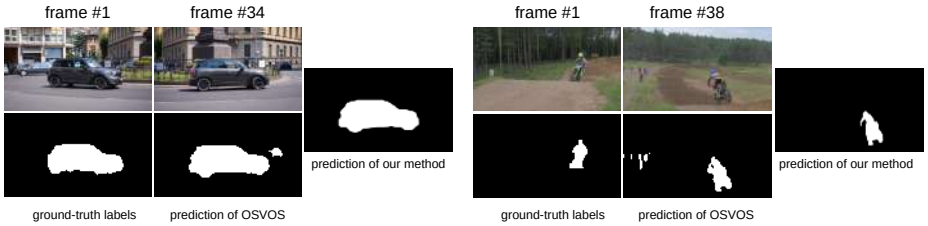

Fig. 5. Example foreground predictions without instance embeddings (OSVOS) and with instance embeddings (our method). Joint learning instance embeddings improves the model's capability to distinguish pixels of different objects.

4.2 Learning Foreground Predictions

Inspired by the previous work of [3] and [12], we propose to estimate the foreground pixels from an image using a network. Let o_i be the predicted probability or confidence of pixel i being the foreground and z_i be the binary groundtruth foreground label. We use the pixel-wise cross-entropy loss for binary classification which is defined as:

$$L_{fg} = -\sum_{k=1}^{M}\sum_{i=1}^{W_k} \frac{1}{M \times W_k} z_i \log(o_i) + (1 - z_i)\log(1 - o_i) \tag{3}$$

We jointly learn the foreground predictions and location-sensitive embeddings by summing the two loss functions $L_{em} + L_{fg}$. The joint training enables the network to focus more on the image details which are important to obtain sharp masks and distinguish objects of the same class. In particular, we will show in experiments that the joint training significantly improves the accuracy of the foreground prediction branch.

4.3 Combining Foreground Predictions and Embeddings

In this section, we present a fusion module to combine the LSEs and foreground predictions. We first obtain a set of foreground embeddings $\mathbb{F}$ according to the labels in the foreground predictions. Then we compute a representative r by identifying the median for each dimension of the embeddings in $\mathbb{F}$.

Then for each embedding p, we compute a foreground probability by $\frac{2}{1+\exp^{\|r-p\|_2^2}}$. It is worth noting that the metric was previously used for computing the similarity between two embeddings when optimizing the embedding loss. So it is favorable to use the same metric here because it is suitable to the distributions of the learned embeddings. If an embedding has large distance with the representative, it is less likely to be a foreground pixel. Finally, after getting the foreground map from the embeddings, we compute the average of the two foreground predictions as the fused results. It is worth noting that, after introducing this fusion module, the overall network can still be trained end-to-end by passing the gradients directly to the two branches.

5 Experiments

We first evaluate the foreground extraction qualities of our approach by comparing it with GrabCut [27]. The experiment assumes the groundtruth bounding boxes are known for the target object on all frames. Second, we evaluate the proposed method for video object segmentation where only a bounding box in the first frame is known. We perform extensive ablation studies to evaluate the influences of each component and compare with the state-of-the-arts.

5.1 Dataset

The DAVIS dataset [25] consists of 50 high quality videos and 3455 frames. The dense foreground annotations are provided for each frame. We test our approach on the validation set as most of the previous work. The SegTrack-v2 dataset consists of 14 videos with 24 objects and 947 frames. The dense annotations are provided for the foreground object. Different from the DAVIS dataset, they provide instance annotations for images with multiple objects. Following [24], each object is treated as a separate target to segment.

On DAVIS, we use the officially provided measures to evaluate the segmentation result: region similarity $\mathcal{J}$ and contour accuracy ($\mathcal{F}$). The region similarity $\mathcal{J}$ is defined as the intersection-over-union of the estimated mask and the groundtruth mask. In particular, given an estimated mask M and the corresponding ground-truth mask G, it is defined as $\mathcal{J} = \frac{M \cap G}{M \cup G}$. The contour accuracy $\mathcal{F}$ interprets M as a set of closed contours $c(M)$ delimiting the spatial extent of the mask. The contour-based precision P_c and recall R_c between the contour points of $c(M)$ and $c(G)$ are computed by morphology operators. The contour accuracy reflects the sharpness of the extracted masks. Please refer to [25] for the precise definitions. On SegTrack-v2 we use $\mathcal{J}$ as our measure.

5.2 Implementation Details

Network Our model is based on the semantic segmentation network DeepLab [5] which is based on the ResNet-101 [13]. The atrous spatial pyramid pooling(ASPP) layer is extensively used in our model which shows better performance in capturing the details than using a single convolution. In particular, we use the ASPP layer to predict location embeddings, appearance embeddings and foreground probability, respectively, as shown in Fig. 2. To get the object center map, we add the pixel coordinates to the predicted location embedding vectors for each pixel. The center map is fed to a 1×1 convolution layer and then concatenated with the 64-channel appearance embeddings. The concatenations are fed to another ASPP layer to obtain the location sensitive embeddings.

Training We pretrain our network on the augmented PASCAL VOC2012 segmentation dataset [7, 10]. There are three training stages: (1) we first jointly train the location embedding and appearance embedding branches ($L_{ae} + L_{le}$)

for $20K$ iterations. The batchsize and learning rate are set to be 7 and $1.5e^{-4}$, respectively. We scale the location embedding with a factor of 1/321. (2) then we fix the two branches and train the following layers associated with LSE (L_{lse}). The parameters are set to be the same as step (1). (3) finally, we jointly train the whole network, including the foreground predictions and the LSEs, with a smaller learning rate of $1e^{-4}$.

For a test video, we also train our model on the first frame. Given the bounding box, we directly apply the pretrained model to the corresponding region and obtain a set of highly confident (≥ 0.6) foreground pixels. The pixels outside the bounding box are the background pixels. We retrain our model based on these pseudo-labels for 20 epochs with the learning rate of $2.5e^{-5}$.

5.3 Foreground Extraction

Since the core of our approach is to predict the foreground pixels from an image, it is natural to compare with the classic FE method GrabCut [27]. We assume the bounding boxes of the target in every frame are known and extract the foreground masks. The results are shown in Table 1.

Table 1. Foreground extraction results on the DAVIS dataset

	Measure	GrabCut [27]	FP	Ours
$\mathcal{J}$	Mean $M \uparrow$	61.5	80.4	80.9
	Recall $O \uparrow$	73.4	96.3	96.7
	Decay $D \downarrow$	1.3	3.1	2.1
$\mathcal{F}$	Mean $M \uparrow$	59.0	80.2	80.8
	Recall $O \uparrow$	72.8	90.6	90.2
	Decay $D \downarrow$	3.7	3.1	1.3

We propose a baseline *FP* which only learns the *F*oreground *P*rediction branch (by minimizing L_{fg} in Fig. 2). This can be regarded as a simplified version of OSVOS [3] without boundary snapping. From the table 1, we can see that FP significantly outperforms GrabCut in terms of both region similarity and contour accuracy. This is mainly because the FP is based on deep networks which are learned to extract higher level cues of objects which are robust to cluttered backgrounds. In addition, our approach which learns both foreground predictions and LSEs improves over FP by about 0.5. The improvement is marginal because the power of LSE is not fully revealed when groundtruth bounding boxes are known. This is because most background objects have already been suppressed and have negligible influence on the results.

Fig. 6 shows some examples from the DAVIS dataset. GrabCut obtains coarse foreground extractions for most cases. This is because GrabCut lacks higher level cues for objects and the performance is degraded when the background is cluttered. In particular, for the dancing and camel examples, it cannot differentiate

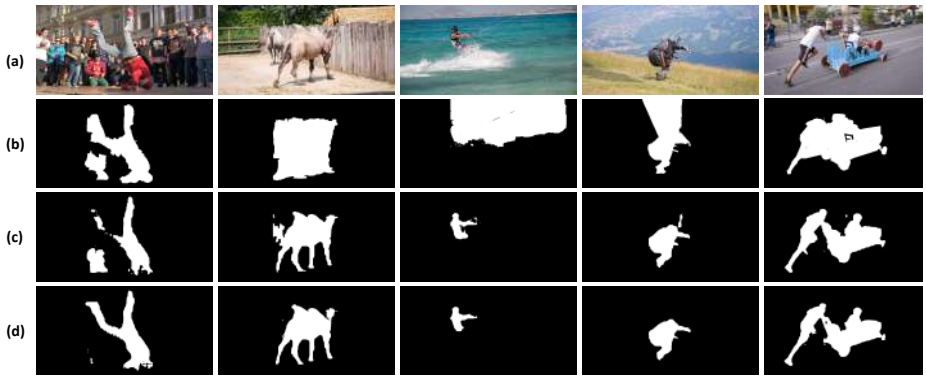

Fig. 6. Foreground extraction results on the DAVIS dataset. (a) shows five testing images. (b) shows the results of GrabCut. (c) shows the results of FP. (d) shows the results of our method.

the background pixels from the foreground objects. The FP method faces a similar problem. See Fig. 6 (c). Although it learns the concept of objects by using a deep network, it cannot distinguish objects of the same class with similar appearance. For example, see the background person in a red shirt and the small camel. In contrast, our approach successfully suppresses the background objects by learning the LSEs.

Table 2. Video object segmentation results on the DAVIS dataset

	Measure	FP(no-zoom)	FP	FPLSE	Ours	Ours+CRF	Ours(static)
$\mathcal{J}$	Mean $M\uparrow$	70.5	76.2	80.0	81.0	82.9	80.0
	Recall $O\uparrow$	83.5	90.0	95.6	96.4	96.7	95.6
	Decay $D\downarrow$	9.1	7.0	5.9	5.4	5.4	6.4
$\mathcal{F}$	Mean $M\uparrow$	69.1	75.0	79.1	80.3	80.1	79.6
	Recall $O\uparrow$	77.4	82.2	87.7	89.1	88.9	87.0
	Decay $D\downarrow$	8.2	7.3	5.9	5.7	5.4	6.2

5.4 Ablation Study on Video Object Segmentation

We investigate the influences of different components in our method on the video object segmentation task. We experiment on the DAVIS dataset and report the region similarity and contour accuracy.

Ablation Study on LSEs We first evaluate the influences of LSEs by designing three baselines. The first baseline FP only learns the foreground prediction branch, which actually is a simplified version of OSVOS [3] without boundary

snapping. The second baseline FPLSE jointly learns the foreground prediction branch and the location sensitive embedding branch. But it doesn't fuse the two branches and only uses the foreground prediction branch. The third baseline is our method which fuses the two branches for foreground prediction.

Table 2 shows the results. The baseline FP achieves the mIoU of 76.2 which is similar to the result in [3] without boundary snapping. FPLSE increases the mIoU to 80.0 although we don't fuse the two branches. The reason for the improvement may be because the joint training encourages the network to focus more on the appearance details (which are required for learning LSEs) and strikes a good balance between abstraction (high level cues of objects) and details (low level appearance cues). In addition, the boundary accuracy is improved from 75.0 to 79.1 which indicates the predictions are more precise. Fusing the two branches further improves the accuracy (from 80.0 to 81.0) which demonstrates the effectiveness of learning LSEs. It is worth noting that the fusion is trained end-to-end without manually setting the model parameters.

Table 3 shows an evaluation with respect to different attributes annotated in the DAVIS dataset. We can see that our method has the best performance on most attributes, and it has a significant resilience to these challenges.

Table 3. Attribute-based performance: quality of the methods on sequences with a certain attribute. See DAVIS [25] for the meanings of the acronyms

Attribute	OSVOS [3]	Lucid [19]	Ours	FP	FPLSE
AC	0.81	0.74	*0.86*	0.77	0.83
BC	*0.83*	*0.83*	*0.83*	0.78	0.81
DB	0.74	0.51	*0.75*	0.60	0.70
EA	0.77	0.71	*0.80*	0.76	0.78
FM	0.76	0.70	*0.82*	0.74	0.78
HO	0.75	0.75	*0.79*	0.71	0.76
IO	0.75	*0.80*	0.78	0.71	0.74
LR	0.77	0.64	*0.83*	0.79	0.81
OV	0.72	*0.83*	*0.83*	0.70	0.78
ROT	0.81	0.80	*0.87*	0.77	0.84
SC	0.71	*0.74*	*0.74*	0.68	0.70
SV	0.74	0.68	*0.80*	0.73	0.77

Ablation Study on Other Factors We first evaluate the proposed model adaptation method. We compare with a baseline *Static* which only trains the model on the first frame and doesn't perform adaptation on the video. The results are shown in Table 2. We can see that if we don't perform model adaptation, the mIoU drops from 81.0 to 80.0 due to the degraded power to handle appearance variations. In particular, the boundary recall decreases significantly from 89.1 to 87.0 indicating that, without adaptation, model has degraded capability to generate sharp and accurate masks for the video.

We also evaluate the influences of the zooming in operation. See Table 2. By comparing FP(no-zoom) and FP, we can see that the zooming in operation increases the mIoU by about 5.7. There are two reasons for the improvement. First, zooming in decreases the influence of background objects. Second, zooming in allows the method to capture the finer details of an object.

Table 4. The State-of-the-art Results on the DAVIS dataset

Measure	OnAVOS [30]	OSVOS [3]	Lucid Tracker [19]	Mask Track[24]	Mask RNN [15]	MaskTrack Box[24]	Ours
$\mathcal{J}_M \uparrow$	86.1	79.8	80.5	79.7	80.4	73.7	82.9
$\mathcal{J}_O \uparrow$	96.1	93.6	90.2	93.1	96.0	-	96.7
$\mathcal{J}_D \downarrow$	5.2	14.9	6.1	8.9	4.4	-	5.6
$\mathcal{F}_M \uparrow$	84.9	80.6	77.6	75.4	82.3	-	80.3
$\mathcal{F}_O \uparrow$	89.7	92.6	82	87.1	93.2	-	89.1
$\mathcal{F}_D \downarrow$	5.8	15.0	6.9	9.0	8.8	-	5.7

5.5 Comparison to the State-of-the-arts

Table 4 compares the proposed method to the state-of-the-arts on the DAVIS dataset. The methods [3, 24, 30, 19, 15] use the precise masks in the first frame as inputs. The MaskTrack-Box method uses only a bounding box in the first frame which is the same as our method.

We can see that our method outperforms most of the state-of-the-arts except [30]. But the method [30] uses additional training dataset from the DAVIS, which according to their experiments, increases the mIoU by about 3. In this case, our result is comparable with [30] although we only use a bounding box as inputs. MaskTrack-Box is one of the state-of-art methods which take a box as input.The result of it is obtained on the train-val dataset without CRF. For fair comparison, we also report the result on this dataset without CRF. The number is 80.7. Its mIoU is about 7.0 lower than that of our method.

Table 5 shows the results on the SegTrack v2 dataset. The methods [24, 19, 15] use the precise masks in the first frame as inputs. We can see that our method is comparable to the state-of-the-arts. The core contribution in [19] is the generation of new training data based on the first frame which can also be combined with our method to further improve the performance.

Table 5. The State-of-the-art Results on the SegTrack v2 dataset

Method	MaskTrack[24]	OBJFlow[29]	MaskRNN[15]	LucidTracker[19]	BVS[22]	MaskTrack Box[24]	Ours
mIoU	70.3	67.5	72.1	78.0	58.4	62.4	69.7

5.6 Qualitative Evaluation

Fig. 7 shows five sample results of our method from the DAVIS and the SegTrack v2 datasets. We can see that the method deals well with situations such as occlusions, scale changes, cluttered backgrounds, background with the same class of objects as the target. The green rectangles are the estimated bounding boxes for the current frame. We can see that in most cases, the bounding boxes are accurate and tight.

Fig. 7. Sample segmentation results on the DAVIS and the SegTrack v2 datasets.

6 Conclusion

We present an end-to-end training method for video object segmentation. The main contribution of this work is to propose a method to learn location-sensitive embeddings which have enhanced power to distinguish very similar objects in different locations of an image. We validate the effectiveness of the approach by extensive numerical and qualitative experiments on the DAVIS and SegTrack v2 datasets. The method is simple and has fewer hyperparameters to tune, but it achieves the state-of-the-art performance on the datasets.

Acknowledgements This work is supported in part by NSFC-61625201, 61527804, 61650202.

References

1. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.: Fully-convolutional siamese networks for object tracking. In: ECCV. pp. 850–865. Springer (2016)
2. Boykov, Y.Y., Jolly, M.P.: Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In: ICCV. vol. 1, pp. 105–112 (2001)
3. Caelles, S., Maninis, K.K., Pont-Tuset, J., Leal-Taixé, L., Cremers, D., Van Gool, L.: One-shot video object segmentation. In: CVPR. IEEE (2017)
4. Chen, L.C., Hermans, A., Papandreou, G., Schroff, F., Wang, P., Adam, H.: Masklab: Instance segmentation by refining object detection with semantic and direction features. arXiv preprint arXiv:1712.04837 (2017)
5. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Semantic image segmentation with deep convolutional nets and fully connected crfs. ICLR (2015)
6. Cheng, J., Tsai, Y.H., Wang, S., Yang, M.H.: Segflow: Joint learning for video object segmentation and optical flow. In: ICCV. pp. 686–695. IEEE (2017)
7. Everingham, M., Eslami, S.M.A., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes challenge: A retrospective. IJCV **111**(1), 98–136 (Jan 2015)
8. Fathi, A., Wojna, Z., Rathod, V., Wang, P., Song, H.O., Guadarrama, S., Murphy, K.P.: Semantic instance segmentation via deep metric learning. arXiv preprint arXiv:1703.10277 (2017)
9. Grabner, H., Bischof, H.: On-line boosting and vision. In: CVPR. vol. 1, pp. 260–267 (2006)
10. Hariharan, B., Arbelaez, P., Bourdev, L., Maji, S., Malik, J.: Semantic contours from inverse detectors. In: ICCV (2011)
11. Harley, A.W., Derpanis, K.G., Kokkinos, I.: Segmentation-aware convolutional networks using local attention masks. In: ICCV. vol. 2, p. 7 (2017)
12. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV. pp. 2980–2988. IEEE (2017)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
14. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. TPAMI **37**(3), 583–596 (2015)
15. Hu, Y.T., Huang, J.B., Schwing, A.: Maskrnn: Instance level video object segmentation. In: NIPS. pp. 324–333 (2017)
16. Jampani, V., Gadde, R., Gehler, P.V.: Video propagation networks. In: Proc. CVPR. vol. 6, p. 7 (2017)
17. Jang, W.D., Kim, C.S.: Online video object segmentation via convolutional trident network. In: CVPR. vol. 1, p. 7 (2017)
18. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-learning-detection. TPAMI **34**(7), 1409–1422 (2012)
19. Khoreva, A., Benenson, R., Ilg, E., Brox, T., Schiele, B.: Lucid data dreaming for object tracking. arXiv preprint arXiv:1703.09554 (2017)
20. Li, S., Seybold, B., Vorobyov, A., Fathi, A., Huang, Q., Kuo, C.C.J.: Instance embedding transfer to unsupervised video object segmentation. arXiv preprint arXiv:1801.00908 (2018)
21. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR. pp. 3431–3440 (2015)

22. Märki, N., Perazzi, F., Wang, O., Sorkine-Hornung, A.: Bilateral space video segmentation. In: CVPR. pp. 743–751 (2016)
23. Newell, A., Huang, Z., Deng, J.: Associative embedding: End-to-end learning for joint detection and grouping. In: NIPS. pp. 2274–2284 (2017)
24. Perazzi, F., Khoreva, A., Benenson, R., Schiele, B., A.Sorkine-Hornung: Learning video object segmentation from static images. In: CVPR (2017)
25. 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)
26. Ren, X., Malik, J.: Tracking as repeated figure/ground segmentation. In: CVPR. pp. 1–8 (2007)
27. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: Interactive foreground extraction using iterated graph cuts. In: ACM transactions on graphics (TOG). vol. 23, pp. 309–314. ACM (2004)
28. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: CVPR. pp. 815–823 (2015)
29. Tsai, Y.H., Yang, M.H., Black, M.J.: Video segmentation via object flow. In: CVPR. pp. 3899–3908 (2016)
30. Voigtlaender, P., Leibe, B.: Online adaptation of convolutional neural networks for video object segmentation. BMVC (2017)
31. Xiao, F., Jae Lee, Y.: Track and segment: An iterative unsupervised approach for video object proposals. In: CVPR. pp. 933–942 (2016)