The 2018 DAVIS Challenge on Video Object Segmentation - CVPR 2018 Workshops

Similarity Learning for Dense Label Transfer

Mohammad Najafi* Viveka Kulharia* Thalaiyasingam Ajanthan Philip H. S. Torr
University of Oxford, UK
{monaj, viveka, ajanthan, phst}@robots.ox.ac.uk

Abstract

In this work, we introduce a simple and flexible method for video object segmentation based on similarity learning. The proposed method can learn to perform dense label transfer from one image to the other. More specifically, the objective is to learn a similarity metric for dense pixel-wise correspondence between two images. This learned model can then be used in a label transfer framework to propagate object annotations from a reference frame to all the subsequent frames in a video. Unlike previous methods, our similarity learning approach works fairly well across various domains, even when no domain adaptation is involved. Using the proposed approach, we achieved the second place in the first DAVIS challenge for interactive video object segmentation, in both quality and speed tracks.

1. Introduction

Object segmentation in a video is an important problem in computer vision with many interesting real world applications in surveillance, robotics, and autonomous driving. Given one or more target objects, the task is to segment those objects in every frame of the video.

In this paper, we introduce a simple and flexible method for video object segmentation based on *similarity learning*. More specifically, we turn the problem of video object segmentation into a dense binary classification problem using a Siamese network, where the objective is to predict whether or not a pair of selected pixels from two frames belong to the same object (or instance of an object). Our goal here is to learn an embedding space in which, pixels of the same object are encouraged to be close to each other, whereas pixels of different objects are pushed away from each other. In short, we train a similarity metric for pixel correspondence between two images. We then use the trained model in a *label transfer* framework to propagate object annotations from a frame with ground-truth information (*e.g.*, first frame of the video in case of *one-shot* video object segmentation task) to other frames.

Note that, with this simple similarity learning approach, our method performs fairly well across various domains and clearly outperforms state-of-the-art methods when no domain adaptation is involved. Furthermore, unlike previous approaches [9], multi-target segmentation comes at virtually no cost in our approach. Figure 1 demonstrates qualitative results computed using our method for one-shot object segmentation on sample videos, with no test domain adaptation, online learning or human interaction involved.

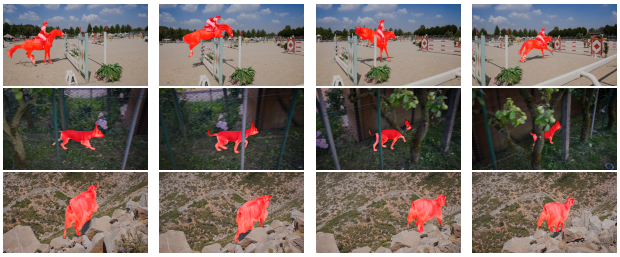

Figure 1: One-shot segmentation outputs for sample videos from DAVIS-2016, obtained using our model without any test domain adaptation, online learning or interaction. Each row illustrates outputs for different time frames of the video.

In this work, we consider the scenario of *interactive* video object segmentation with scribbles, as suggested by [1]. In this process, the segmentation system learns interactively by getting feedbacks from the evaluation system. Moreover, at each interaction, the method is fine-tuned using only a set of squiggles from one video frame, which are much easier to obtain compared to dense ground-truth object masks, but make the problem more challenging.

Following the above-mentioned protocol, we developed an interactive segmentation system based on our dense similarity learning and label transfer method, and achieved the second place in the first DAVIS challenge for interactive video object segmentation, in both quality and speed tracks.

2. Method

Our approach builds upon the standard Siamese network to learn the dense correspondence between the target objects in two given images. To this end, we first review the Siamese architecture and then introduce our idea for using it in a dense label transfer framework.

*Joint first author

2.1. Siamese Networks

The Siamese network is a special type of neural network, which is trained to make comparisons between pairs of input data. In other words, it learns whether two input items belong to the same class or not. Figure 2 illustrates an example of a Siamese network that is used in our framework.

Let $\{\mathbf{x}, \mathbf{y}\}$ be a pair of input data (usually a pair of images) which is fed into two identical deep neural networks with a shared set of parameters $\mathcal{W}$. These networks generate two embeddings $\{\Phi(\mathbf{x}; \mathcal{W}), \Phi(\mathbf{y}; \mathcal{W})\}$ at their outputs. The inner-product of these embeddings can be considered as the similarity between $\mathbf{x}$ and $\mathbf{y}$. Also let ℓ be the target of the Siamese network, which is defined as $\ell = \mathbb{1}[l_{\mathbf{x}} == l_{\mathbf{y}}]$, where $l_{\mathbf{x}}$ and $l_{\mathbf{y}}$ are the class labels of $\mathbf{x}$ and $\mathbf{y}$, respectively. In other words, ℓ is set to one if the input pair have the same class labels and zero otherwise. Now the objective of the standard Siamese network can be defined as:

$$\min_{\mathcal{W}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(\langle \Phi(\mathbf{x}_i; \mathcal{W}), \Phi(\mathbf{y}_i; \mathcal{W}) \rangle, \ell_i) \,, \qquad (1)$$

where i ranges over the training set of data pairs of size N, $\mathcal{L}$ is a standard loss function, *e.g.*, binary cross entropy, and $\langle \cdot, \cdot \rangle$ denotes the inner-product operation. The hypothesis is that, the embedding space obtained by this optimization process corresponds to a similarity metric, in which, data of similar class are clustered together

2.2. Object Segmentation using Siamese Networks

In this section, we extend the idea of similarity learning to use it for dense video object segmentation. Figure 2 illustrates the proposed structure, where each branch of the Siamese network is a Deeplab-Resnet-101 [2] model with dilated convolutions to preserve spatial information, connected to a decoder network which upsamples the embeddings to a higher resolution. Note that, as shown in [2, 7], preserving spatial information is crucial for segmentation tasks. The outputs of the decoders are two pixel embedding maps $\{\Phi(\mathbf{x}; \mathcal{W}), \Phi(\mathbf{y}; \mathcal{W})\}$, where $\mathbf{x}$ and $\mathbf{y}$ are input images of size $[H, W]$ and Φ is a three-dimensional tensor of size $[h, w, d]$, where $h = H/2$, $w = W/2$ and d is the embedding size.

Next, we compute the similarities of the pixels between images $\mathbf{x}$ and $\mathbf{y}$ by taking the inner-product of their embeddings. The output of this step is a dense similarity map of size $[h \times w, h \times w]$, which indicates the similarity of every pixel embedding in $\Phi_{\mathbf{x}}$ with all the pixel embeddings in $\Phi_{\mathbf{y}}$. At this point, we apply batch-norm [4] to the output of inner-product to prevent the gradients from exploding. Finally, the sigmoid nonlinearity is applied to convert the similarity scores into probabilities. Furthermore, the target of the Siamese network is a matrix of size $[h \times w, h \times w]$ (denoted as $\mathbf{L}$), which is a binary map that shows if two pixels in the images belong to the same object class.

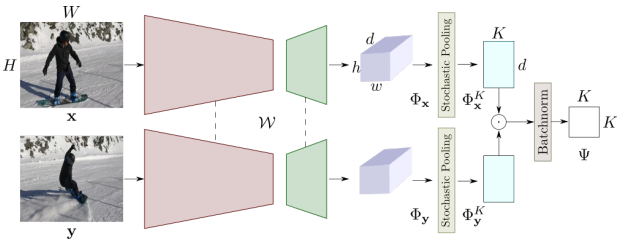

Figure 2: **The Siamese architecture used for learning dense correspondense.** The model takes in a pair of images and outputs the similarity of their pixels by computing the inner-product of their subsampled pixel embeddings ($\langle \Phi_{\mathbf{x}}^K, \Phi_{\mathbf{y}}^K \rangle$). During evaluation, stochastic pooling is not applied on the target image $\mathbf{y}$ and the inner-product output is $\langle \Phi_{\mathbf{x}}^K, \Phi_{\mathbf{y}} \rangle$.

The proposed network is trained according to the objective function in Eq. 1, but for an aggregate loss over $[h \times w, h \times w]$ data points for each pair of images. After training, the output similarity map can be used to predict the label correspondence between pixels of images $\mathbf{x}$ and $\mathbf{y}$. Then, during test time, we leverage this similarity map to predict the segmentation map of the second image $\mathbf{y}$, given a ground-truth segmentation map of the first image $\mathbf{x}$, via a *label transfer* approach, as described in Sec. 2.2.1.

The main challenge when using the Siamese architecture to learn dense correspondence is the high memory requirement. Note that, the output of the inner-product $\Psi = \langle \Phi_{\mathbf{x}}, \Phi_{\mathbf{y}} \rangle$ is of dimension $[h \times w, h \times w]$. In our case, $h = 240$ and $w = 427$, which leads to 1×10^{10} floating point values to be stored. This is infeasible to store on a standard GPU. To tackle this, we approximate the inner-product $\Psi = \langle \Phi_{\mathbf{x}}, \Phi_{\mathbf{y}} \rangle$ using only a subset of pixels.

We observe that, in an image, a large portion of the pixels are highly correlated to each other. Hence, it is possible to estimate the embedding of the image pixels using only a small fraction of them. Based on this argument, we introduce a *stochastic pooling* layer to the network, that subsamples k pixels uniformly at random for each object class present in the image. Thus, a total of K pixels are chosen from each of the embeddings. Now, the inner-product can be approximated as:

$$\Psi = \langle \Phi_{\mathbf{x}}, \Phi_{\mathbf{y}} \rangle \approx \langle \Phi_{\mathbf{x}}^K, \Phi_{\mathbf{y}}^K \rangle \,, \qquad (2)$$

where $\Phi_{\mathbf{x}}^K$ and $\Phi_{\mathbf{y}}^K$ represent the embeddings corresponding to the chosen subsets of $\mathbf{x}$ and $\mathbf{y}$, respectively. Note that, Ψ is of size $[K, K]$, where K is the chosen subset size. Also note that, the same chosen subsets are used to create the target map $\mathbf{L}^K$ for the Siamese network. In our experiments, we observed that the performance plateaus for $k > 100$, which supports the above argument. This effectively reduces the memory requirement from 1×10^{10} to

In this section, for brevity, we call the pixel embedding as a pixel.

1×10^4 (assuming one object in each image), and also makes it faster in both training and inference phases.

2.2.1 Label Transfer

Once the network is trained, one can label a query image $\mathbf{y}$ by matching its pixels with the pixels of a reference image $\mathbf{x}$ (whose ground-truth annotation is available) using the learned similarity model. At test time, the reference image is fed into the top branch of the Siamese network, while the query image is given to the bottom branch. Then a representative set of pixel embeddings are subsampled from the reference image via the stochastic pooling layer, and their similarities with the embeddings of the query pixels are computed. Subsequently, the class labels of the selected reference pixels are transferred to all the query pixels based on their similarity scores. Note that, in this step, the stochastic pooling is applied only to the reference image and the similarity scores using the inner-product are computed for all the query pixels, *i.e.*, $\Psi \approx \langle \Phi_{\mathbf{x}}^K, \Phi_{\mathbf{y}} \rangle$.

Let $\mathcal{N}_{l'}$ be the selected subset of pixels from the reference image for class l', then the class label of pixel $i \in \mathcal{Q}$ within the query image $\mathcal{Q}$ is predicted using the following formulation:

$$l_i = \arg\max_{l'} \sum_{j \in \mathcal{N}_{l'}} \tilde{\Psi}_{ij} \ , \tag{3}$$

where $\tilde{\Psi}$ is the similarity score after the batch-norm layer.

3. Experiments

We evaluated our method interactively on DAVIS-2017 [8] validation (30 videos) and Test-dev (30 videos) sets. For validation, we used a model pre-trained on DAVIS-2017 training set (60 videos), however in the test phase, we pre-trained our model using the entire training+validation set (90 videos). All the experiments in this paper were performed on 480×854 resolution videos.

Interactive Segmentation The proposed approach was evaluated using the interactive toolkit released by [1]. Initially a set of squiggles representing the objects of interests in one of the video frames are provided as training examples. The segmentation system is fine-tuned based on this sparse set of ground-truth information and predicts pixel-level masks for targets over the entire video. The outputs are then analysed, and the video frame with the worst segmentation output is chosen to provide the user with a set of feedbacks on which regions the method has failed. Note that the feedbacks are also in the form of scribbles. This interaction improves the segmentation system, as it learns where it fails the most and updates its parameters such that those mistakes are resolved.

Evaluation Metrics Two evaluation metrics are designed for this interactive task by [1], which take into account both segmentation accuracy and speed. The first one is Area Under the Curve (AUC) of the plot Time vs Jaccard ($\mathcal{J}$) performance. Note that $\mathcal{J}$ is a standard metric which is defined as the intersection-over-union (IoU) of the predicted object mask and its respective ground-truth mask. Each point in the AUC-$\mathcal{J}$ plot is computed considering the average time and the average Jaccard, obtained for a certain interaction, for the whole dataset. The second metric is called $\mathcal{J}$@60 which evaluates what Jaccard performance a method can reach within 60 seconds.

Network Structure Each branch of our network (Figure 2) is a Deeplab-Resnet-101 model (pre-trained on ImageNet [3] and Microsoft-coco image dataset [6]), which produces feature maps whose resolutions are eight times smaller than the resolution of input images. These features maps are upsampled using the decoder network which yields high-resolution embeddings. The decoder consists of a transposed convolution layer (with stride $= 2$) and a non-parametric bilinear upsampling layer (with stride $= 2$).

Training and Label Transfer At each training iteration, a video is randomly selected from the training pool, and then a pair of frames are drawn randomly from the video to feed the network. We followed the pooling strategy explained in Sec. 2.2 and selected $k = 100$ pixel embeddings in our experiments. The network was trained for 100000 iterations, using dense ground-truth annotations in the training split, with the base learning rate of 2.5×10^{-4} for the branches and a learning rate factor of two for the decoder ($\text{lr}_{\text{decoder}} = 5 \times 10^{-4}$, with a momentum of 0.9 and weight decay of 0.0005. In addition, learning rates were decreased polynomially, with a power of 0.9.

At the validation/test time, the second branch of the network is fed by a query image, and we use the learned similarity model to transfer the ground-truth labels of the training input image in the top branch to the query image. Label transfer, as explained in Sec. 2.2.1, is merely a label matching operation, which does not leverage any prior information over the image pixels. Hence we can further improve the output of this step by passing it through a dense CRF [5].

The interactive process included a sequence of domain adaptation (fine-tuning based on the provided set of ground-truth scribbles), label transfer, and post-processing (CRF), which were designed differently at different interactions. We tailored the following strategy based on a set of experiments on the validation set. We initially fine-tuned our pre-trained model using the first set of scribbles for 5s, but extended this fine-tuning time to 20s for the second interaction and 60s for all remaining interactions. At each interaction, a new video frame together with a set of scribbles were provided by the evaluation toolkit. We augmented the frame by random horizontal flipping, scaling (0.5 to 1.5) and rotation (-30° to 30°). The augmented data (with different augmentation parameters) fed the branches of the network and fine-tuned the model using their scribbles, together with the images and scribbles supplied at previous interactions. In addition, we stopped fine-tuning when the maximum value

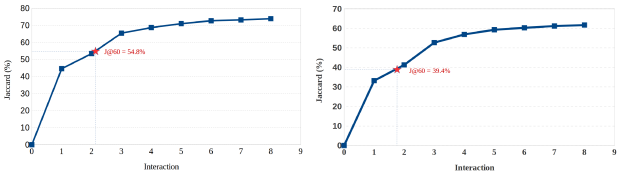

Figure 3: The performance of our method on DAVIS-2017 validation split (left) with $AUC = 70.2\%$ and $\mathcal{J}@60 = 54.8\%$, and Test-Dev split (right) with $AUC = 54.5\%$ and $\mathcal{J}@60 = 39.4\%$. Note that in the Test-Dev set, the $60s$ time-point occurs before the second interaction because more complex videos with larger number of objects are present and processed in this data split.

of loss dropped below 0.04 at each interaction. Also we skipped CRF for the first couple of interactions to speed up the evaluation. Furthermore, the provided scribbles at each interaction were dilated with a square kernel of size three, and we also denoted the image pixels that were far from all of the object scribbles, as background pixels. Note that whenever the feedbacks did not include any region from background during interactions, pixels which were the least similar to any of the initial ground-truth object scribbles were selected to represent the background.

Following the above strategy, we achieved the second place in both quality (AUC-$\mathcal{J}$) and speed ($\mathcal{J}@60$) tracks. The charts in Figure 3 show the results obtained using our model on the validation and test-dev splits of DAVIS-2017 dataset. As shown in the figure, the model has consistently improved by learning from the previous mistakes. Figure 4 demonstrates some examples, where the interactive process has incrementally improved the segmentation outputs.

4. Conclusion

In this work, we tackled the problem of video object segmentation using sparse ground-truth data, through similarity learning. In particular, we trained a Siamese network, whose objective is to predict whether or not two pixels from two video frames belong to the same object class. Next, a dense label transfer approach was proposed that used the learned similarity to propagate annotations from one video frame to the rest of the videos. The proposed method achieved the second place in the first DAVIS challenge for interactive video object segmentation, in both quality and speed tracks.

5. Acknowledgement

This work was supported by the EPSRC, ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1 and EPSRC/MURI grant EP/N019474/1. We also thank NVIDIA for donating a GPU for this project. Viveka Kulharia is wholly funded by TRI's grant.

Figure 4: **The impact of successive interactions on the quality of the segmentation outputs.** The top row illustrates the input images and the segmentation results for consecutive interactions are depicted in the 2nd to 8th rows.

References

[1] S. Caelles, A. Montes, K.-K. Maninis, Y. Chen, L. Van Gool, F. Perazzi, and J. Pont-Tuset. The 2018 davis challenge on video object segmentation. *arXiv:1803.00557*, 2018.

[2] 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. *IEEE T-PAMI*, 40(4):834–848, 2018.

[3] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

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

[5] P. Krähenbühl and V. Koltun. Parameter learning and convergent inference for dense random fields. In *ICML*, 2013.

[6] 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. 2014.

[7] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[8] 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:1704.00675*, 2017.

[9] P. Voigtlaender and B. Leibe. Online adaptation of convolutional neural networks for video object segmentation. In *BMVC*, 2017.