Supplementary Material

Learning Discriminative Model Prediction for Tracking

Goutam Bhat* Martin Danelljan* Luc Van Gool Radu Timofte
CVL, ETH Zürich, Switzerland

This supplementary material for [1] provides additional details and results. Section 1 derives the closed form expression of the filter gradient, employed in the optimizer module. In section 2 we derive the application of the Jacobian in order to compute the quantity h, employed in algorithm 1 in the paper. In section 3 we provide detailed results on the VOT2018 dataset, while in section 4, we provide detailed results on the LaSOT dataset. We also provide additional details on the NFS, OTB100 and UAV123 datasets in section 5. We analyze the impact when training with less data in section 6. Finally, we provide a 2d visualization of the learned functions parametrizing the discriminative loss in section 7.

1. Closed-Form Expression for ∇L

Here, we derive a closed-form expression for the gradient of the loss (1) in the main paper, also restated here,

$$L(f) = \frac{1}{|S_{\text{train}}|} \sum_{(x,c)\in S_{\text{train}}} \|r(s,c)\|^2 + \|\lambda f\|^2 \,. \quad (1)$$

Here, $s = x * f$ is the score map obtained after convolving the deep feature map x with the target model f. The training set is given by $S_{\text{train}} = \{(x_j, c_j)\}_{j=1}^{n}$. The residual function $r(s,c)$ is defined as (also eq. (2) in the paper),

$$r(s,c) = v_c \cdot (m_c s + (1-m_c)\max(0,s) - y_c) \,. \quad (2)$$

The gradient $\nabla L(f)$ of the loss (1) w.r.t. the filter coefficients f is then computed as,

$$\nabla L(f) = \frac{2}{|S_{\text{train}}|} \sum_{(x,c)\in S_{\text{train}}} \left(\frac{\partial r_{s,c}}{\partial f}\right)^{\mathrm{T}} r_{s,c} + 2\lambda^2 f \,. \quad (3)$$

Here, we have defined $r_{s,c} = r(s,c)$ and $\frac{\partial r_{s,c}}{\partial f}$ corresponds to the Jacobian of the residual function (2) w.r.t. the filter coefficients f. Using eq. (2) we obtain,

$$\begin{aligned}\frac{\partial r_{s,c}}{\partial f} &= \operatorname{diag}(v_c m_c)\frac{\partial s}{\partial f} + \operatorname{diag}\left((1-m_c)\cdot \mathbb{1}_{s>0}\right)\frac{\partial s}{\partial f} \\ &= \operatorname{diag}(q_c)\frac{\partial s}{\partial f} \,. \end{aligned} \quad (4)$$

Here, $\operatorname{diag}(q_c)$ denotes a diagonal matrix containing the elements in q_c. Further, $q_c = v_c m_c + (1-m_c)\cdot \mathbb{1}_{s>0}$ is computed using only point-wise operations, where $\mathbb{1}_{s>0}$ is 1 for positive s and 0 otherwise. Using eqs. (3) and (4) we finally obtain,

$$\nabla L(f) = \frac{2}{|S_{\text{train}}|} \sum_{(x,c)\in S_{\text{train}}} \left(\frac{\partial s}{\partial f}\right)^{\mathrm{T}} (q_c \cdot r_{s,c}) + 2\lambda^2 f \,. \quad (5)$$

Here, $\cdot$ denotes the element-wise product. The multiplication with the transposed Jacobian $\left(\frac{\partial s}{\partial f}\right)^{\mathrm{T}}$ corresponds to backpropagation of the input $q_c \cdot r_{s,c}$ through the convolution layer $f \mapsto x * f$. This is implemented as a transposed convolution with x. The closed-form expression (5) is thus easily implemented using standard operations in a deep learning library like PyTorch.

2. Calculation of h in Algorithm 1

In this section, we show the calculation of $h = J^{(i)}\nabla L(f^{(i)})$, used when determining the optimal step length α in Algorithm 1 in the main paper. Since we only need the squared L^2 norm of h in step length calculation, we will directly derive an expression for $\|h\|^2 = \|J^{(i)}\nabla L(f^{(i)})\|^2$. Here, $J^{(i)} = \frac{\partial \xi}{\partial f}\Big|_{f^{(i)}}$ is the Jacobian of the residual vector ξ of loss (1), evaluated at the filter estimate $f^{(i)}$. Not to be confused with the residual function (2), the residual vector ξ is obtained as the concatenation of individual residuals $\xi_j = r(x_j * f, c_j)/\sqrt{n}$ for $j \in \{1,\dots,n\}$ and $\xi_j = \lambda f$ for $j = n+1$. Here, $n = |S_{\text{train}}|$ is the number

*Both authors contributed equally.

of samples in S_{train}. Consequently, we get,

$$\begin{aligned}
\|h\|^2 &= \left\| J^{(i)} \nabla L(f^{(i)}) \right\|^2 \qquad (6) \\
&= \sum_{j=1}^{n+1} \left\| \left. \frac{\partial \xi_j}{\partial f} \right|_{f^{(i)}} \nabla L(f^{(i)}) \right\|^2 \\
&= \sum_{j=1}^{n} \left\| \frac{1}{\sqrt{n}} \left. \frac{\partial r(x_j * f, c_j)}{\partial f} \right|_{f^{(i)}} \nabla L(f^{(i)}) \right\|^2 + \left\| \lambda \nabla L(f^{(i)}) \right\|^2 \\
&= \frac{1}{n} \sum_{(x,c) \in S_{\text{train}}} \left\| \left. \frac{\partial r_{s,c}}{\partial f} \right|_{f^{(i)}} \nabla L(f^{(i)}) \right\|^2 + \left\| \lambda \nabla L(f^{(i)}) \right\|^2 .
\end{aligned}$$

Using eqs. (6) and (4) we finally obtain,

$$\begin{aligned}
\|h\|^2 &= \frac{1}{|S_{\text{train}}|} \sum_{(x,c) \in S_{\text{train}}} \left\| q_c \cdot \left(\left. \frac{\partial s}{\partial f} \right|_{f^{(i)}} \nabla L(f^{(i)}) \right) \right\|^2 + \\
&\qquad \|\lambda \nabla L(f^{(i)})\|^2 \\
&= \frac{1}{|S_{\text{train}}|} \sum_{(x,c) \in S_{\text{train}}} \left\| q_c \cdot \left(x * \nabla L(f^{(i)}) \right) \right\|^2 + \\
&\qquad \|\lambda \nabla L(f^{(i)})\|^2
\end{aligned}$$

As described in section 1, $\nabla L(f^{(i)})$ is computed using the closed-form expression (5). The term $\left. \frac{\partial s}{\partial f} \right|_{f^{(i)}} \nabla L(f^{(i)})$ corresponds to convolution of x with $\nabla L(f^{(i)})$, i.e. $\left. \frac{\partial s}{\partial f} \right|_{f^{(i)}} \nabla L(f^{(i)}) = x * \nabla L(f^{(i)})$. Thus, $\|h\|^2$ is computed easily using standard operations from deep learning libraries.

3. Detailed Results on VOT2018

In this section, we provide detailed results on the VOT2018 [7] dataset. The VOT protocol evaluates the expected average overlap (EAO) between the tracker predictions and the ground truth bounding boxes for different sequence lengths. The trackers are then ranked using the EAO measure, which computes the average of the expected average overlaps over typical sequence lengths. We refer to [8] for further details about the EAO computation. Figure 1 plots the expected average overlap for different sequence lengths on VOT2018 dataset. Our approach DiMP-50 achieves the best EAO score of 0.440.

4. Detailed Results on LaSOT

Here, we provide the normalized precision plots on the LaSOT [3] dataset. These are obtained in the following manner. First, the normalized precision score P_{norm} is computed as the percentage of frames in which the distance between the target location predicted by the tracker and the ground truth, relative to the target size, is less than a certain threshold. The normalized precision score over all the the videos are then plotted over a range of thresholds $[0, 0.5]$ to obtain the normalized precision plots. The trackers are ranked using the area under the resulting curve. Figure 2 shows the normalized precision plots over all 280 videos in the LaSOT dataset. Both our ResNet-18 (DiMP-18) and ResNet-50 (DiMP-50) versions outperform all previous methods, achieving relative gains of 5.9% and 12.8% over the previous best method, ATOM [2].

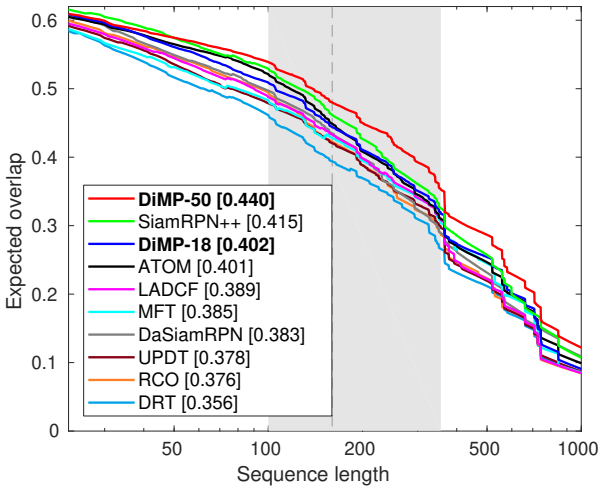


Figure 1. Expected average overlap curve on the VOT2018 dataset, showing the expected overlap between tracker prediction and ground truth for different sequence lengths. The EAO measure, computed as the average of the expected average overlap over typical sequence lengths (grey region in the plot), is shown in the legend. Our approach achieves the best EAO score, outperforming the previous best approach SiamRPN++ [9] with a relative gain of 6.3% in terms of EAO.

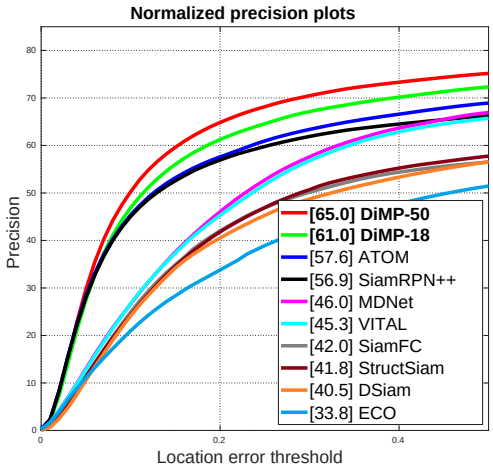
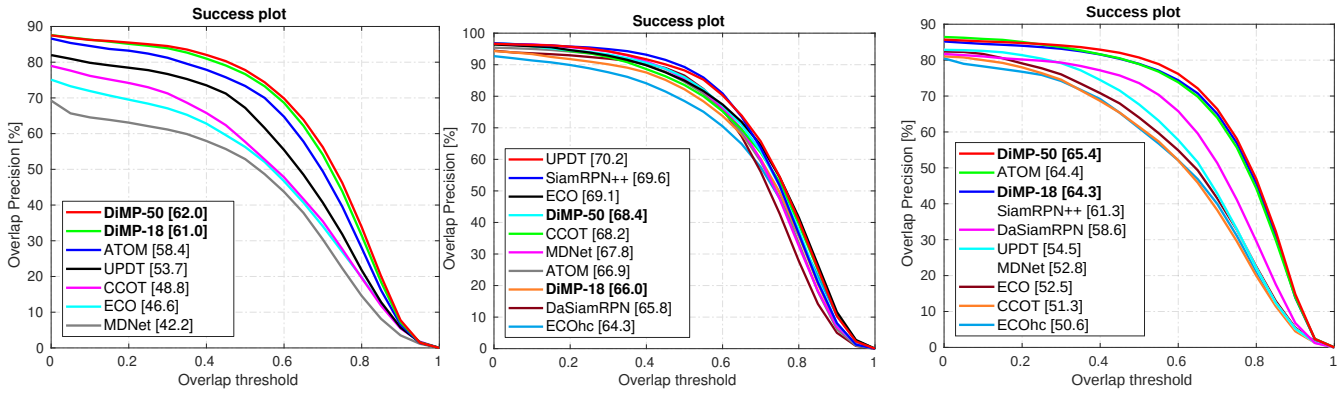


Figure 2. Normalized precision plot on the LaSOT dataset. Both our ResNet-18 and ResNet-50 versions outperform all previous methods by significant margins.



(a) NFS (b) OTB-100 (c) UAV123

Figure 3. Success plots on NFS (a), OTB-100 (b), and UAV123 (c) datasets. The area-under-the-curve (AUC) scores are shown in the legend. Note that the full raw results for SiamRPN++ and MDNet on the UAV123 dataset are unavailable. We therefore only show the final AUC scores of these trackers, as obtained from [9] and [6] respectively. Our approach achieves the best scores on both the NFS and UAV123 datasets.

5. Detailed Results on NFS, OTB-100, and UAV123

Here, we provide detailed results on NFS [4], OTB-100 [13], and UAV123 [11] datasets. We use the overlap precision (OP) metric for evaluating the trackers. The OP score denotes the percentage of frames in a video for which the intersection-over-union (IoU) overlap between the tracker prediction and the ground truth bounding box exceeds a certain threshold. The mean OP score over all the videos in a dataset are plotted over a range of thresholds $[0, 1]$ to obtain the success plot. The area under this plot provides the AUC score, which is used to rank the trackers. We refer to [13] for further details. The success plots over the entire NFS, OTB-100, and UAV123 datasets are shown in figure 3. Our tracker using ResNet-50 backbone, denoted DiMP-50, achieves the best results on both NFS and UAV123 datasets, while obtaining results competitive with the state-of-the-art on the, now saturated, OTB-100 dataset. On the challenging NFS dataset, our approach achieves an absolute gain of 3.6% AUC score over the previous best method ATOM [2].

6. Impact of Training Data

Here, we investigate the impact of the number of videos used for training on the tracking performance. We train different versions of our tracker using the same datasets as in the main paper, i.e. TrackingNet [12], LaSOT [3], GOT10k [5], and COCO [10], but using only a sub-set of videos from each dataset. The results on the combined OTB-100, NFS, and UAV123 datasets are shown in figure 4. Observe that the performance degrades by only 1.5% when the model is trained with only 10% of the total videos. Even when using only 1% of videos, our approach still obtains a respectable AUC score of around 58%.

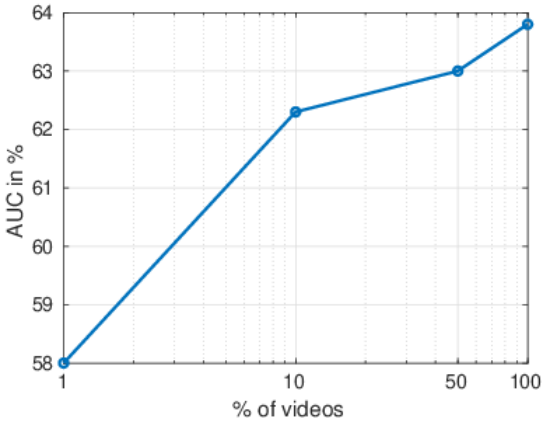


Figure 4. Impact of the percentage of total videos used for offline training (log x-axis). Results are shown on the combined OTB-100, NFS, and UAV123 datasets.

7. Visualizations of learned y_c, m_c, and v_c

A 2D visualization of the learned regression label (y_c), target mask (m_c), and spatial weight (v_c) is provided in figure 5. Note that each of these quantities are in fact continuous and are here sampled at the discrete feature grid points. In this example, that target (red box) is centered in the image patch. From the figure, we can see that the network learns to give the samples in the target-background transition region less weight due to their ambiguous nature.

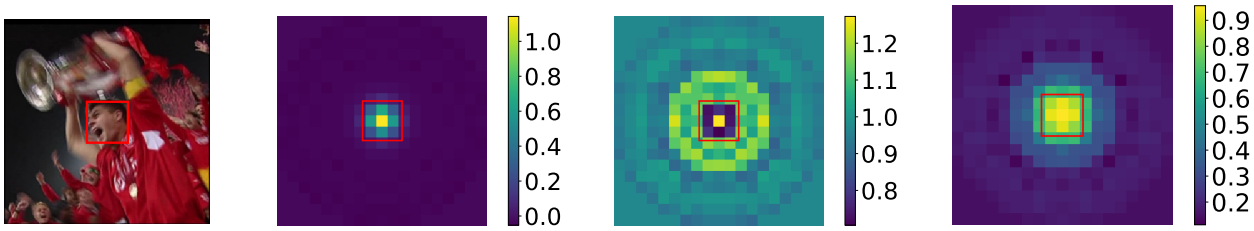


Image | Label y_c | Spatial Weight v_c | Target Mask m_c

Figure 5. Visualization of the learned label y_c, spatial weight v_c, and target mask m_c. The red box denotes the target object.

References

[1] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *ICCV*, 2019. 1

[2] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. ATOM: Accurate tracking by overlap maximization. In *CVPR*, 2019. 2, 3

[3] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. Lasot: A high-quality benchmark for large-scale single object tracking. *CoRR*, abs/1809.07845, 2018. 2, 3

[4] Hamed Kiani Galoogahi, Ashton Fagg, Chen Huang, Deva Ramanan, and Simon Lucey. Need for speed: A benchmark for higher frame rate object tracking. In *ICCV*, 2017. 3

[5] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *arXiv preprint arXiv:1810.11981*, 2018. 3

[6] Ilchae Jung, Jeany Son, Mooyeol Baek, and Bohyung Han. Real-time mdnet. In *ECCV*, 2018. 3

[7] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pfugfelder, Luka Cehovin Zajc, Tomas Vojir, Goutam Bhat, Alan Lukezic, Abdelrahman Eldesokey, Gustavo Fernandez, and et al. The sixth visual object tracking vot2018 challenge results. In *ECCV workshop*, 2018. 2

[8] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Čehovin, G. Fernández, T. Vojír, G. Nebehay, R. Pflugfelder, and G. Hger. The visual object tracking vot2015 challenge results. In *ICCV workshop*, 2015. 2

[9] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *CVPR*, 2019. 2, 3

[10] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, 2014. 3

[11] Matthias Mueller, Neil Smith, and Bernard Ghanem. A benchmark and simulator for uav tracking. In *ECCV*, 2016. 3

[12] Matthias Müller, Adel Bibi, Silvio Giancola, Salman Al-Subaihi, and Bernard Ghanem. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *ECCV*, 2018. 3

[13] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *TPAMI*, 37(9):1834–1848, 2015. 3