Fast Deep Tracking via Semi-Online Domain Adaptation

Xiaoping Li¹, a, Wenbing Luo¹,⁎, b, Yi Zhu¹, c, Hanxi Li¹, d and Mingwen Wang¹, e
¹Jiangxi Normal University, 99 Ziyang Road, Nanchang, Jiangxi, China
a xp.li@jxnu.edu.cn; b wb@jxnu.edu.cn; c deke.zhu@jxnu.edu.cn; d hanxi.li@jxnu.edu.cn; e mwwang@jxnu.edu.cn

Abstract. Deep tracking has been illustrating overwhelming superiorities over the shallow methods. Unfortunately, it also suffers from low FPS rates. To alleviate the problem, a number of real-time deep trackers have been proposed via removing the online updating procedure on the CNN model. However, the absent of the online update leads to a significant drop on tracking accuracy. In this work, we propose to perform the domain adaptation for visual tracking in two stages for transferring the information from the visual tracking domain and the instance domain respectively. In this way, the proposed visual tracker achieves comparable tracking accuracy to the state-of-the-art trackers and runs at real-time speed on an average consuming GPU.

1. Introduction

Visual tracking is one of the most fundamental computer vision tasks. The visual tracker is required to predict the motion state of the tracking target at each frame, given the target state at the first frame.

During the last decade, as the surge of deep learning, more and more tracking algorithms benefit from deep neural networks, e.g. Convolutional Neural Networks[1][2] and Recurrent Neural Networks [3][4]. Despite the well-admitted success, a dilemma still existing in the community is that, deep learning increases the tracking accuracy, while at the cost of high computational complexity. As a result, most well-performing deep trackers usually suffer from low efficiency[5][6]. Recently, some real-time deep trackers were proposed[7][8]. They achieved very fast tracking speed, but can not beat the shallow methods in some important evaluations, as we illustrate latter.

Wang et.al [9] illustrated an effective way to make the deep trackers real-time. They try to transfer the deep features from the classification domain to the tracking domain and meanwhile reduce the feature channels. The yielded tracking algorithm, termed MSDAT, achieves real-time tracking speed and still maintains the high tracking accuracy. Figure 1 compares the network structures and the learning strategy of MD-Net and the MSDAT method. Both of them successfully transfer the knowledge from the classification domain to the tracking domain and thus both of them achieve high tracking accuracies[6][9].

However, the 30 FPS speed of MSDAT is obtained on a high-end GPU. For normal computers in the market, the MSDAT can not achieve real-time. A straightforward way to accelerate the MSDAT further is to reduce the network size while this usually leads to serious performance drop. In this work, we propose to enhance the original MSDAT method via adding a semi-online learning procedure on MSDAT. The semi-online learning stage only covers the first 10 frames of the video sequence and the network is fixed after that. In this way, we further transfer the knowledge from the visual tracking domain to the current video sequence. The experiment part of this paper verifies the superiority of the
The proposed two-stage transfer learning which achieves 60 FPS on a high-end GPU and 23 FPS on an average consuming GPU.

![Flow-charts of the training process of MSDAT and MD-net](image)

**Figure 1.** The flow-charts of the training process of MSDAT and MD-net. Note that the network parts inside the dashed blocks are only used for training and will be abandoned in tracking. Better view in color.

2. The proposed method

The MSDAT algorithm performs well at a real-time speed. However, the 30 FPS speed is obtained on a high-end GPU (NVIDIA GTX1070 as described in [9]). For normal computers in the market, the MSDAT cannot achieve real-time. A straightforward way to accelerate the MSDAT further is to reduce the network size while this usually leads to serious performance drop. In this work, we propose to split the domain adaptation into two stages: an offline stage that transfers the knowledge (parameterized in the CNN model) from the image classification domain to the visual tracking domain; an online stage that further transfers the visual tracking knowledge to the current video sequence. In this way, we obtain a faster deep tracker compared with MSDAT while maintaining its high accuracy.

An brief illustration of the proposed two-stage learning is shown in Figure 2.

2.1. Network structure

Considering that the semi-online learning is time consuming, especially for those median size network. In this work, we modify the PVA net, which was originally proposed for object detection for visual tracking. The network structure is shown in Figure 3.

As we can see in the figure, the network structure is mainly inherited from the PVA net [10] except the three “tails” mounted on the network, where the dimension of the deep feature is reduced. The branches are set following the convention of HCF[5] and MSDAT[9]. When tracking, the second stage of the domain adaptation, i.e., the knowledge transfer from the visual tracking domain to the object instance domain, is also conducted in those branches. That is to say, we update the CNN parameters of the branches online, namely, with the image samples captured in the current test sequence (see Figure 2).
2.2. **Gaussian response loss function**

As described in [9], the MSDAT algorithm modified the original loss function of the SSD algorithm[11] as

\[
L_{\text{loc}}^i(I, g) = \sum_{\mu \in \{x, y\}} m_{i, \mu} \cdot \text{smooth}_\mu(I_i^\mu - \hat{g}_\mu^\mu)
\]  

(1)
Where \( \hat{g}_u, u \in \{x, y, w, h\} \) is one of the geometry parameter of normalized ground-truth box, \( l^u \) is the prediction of one default box and \( m_{ij} \) indicates the relation between them. Readers are suggested to refer to [9] for more details.

In this work, we adopt the loss function of MSDAT for offline training as it has proved itself successfully for offline adaptation. However, we empirically find that loss function defined in Equation 1 is not suitable for online adaptation. We thus employ the Gaussian loss for the second stage of domain adaptation in this paper. The new loss is defined as

\[
L_{\text{gaussian}} = \|M - \hat{M}\|_2^2
\]

(2)

Where \( M \) is the response map predicted by one branch (note that here we directly predict the response map instead of predicting the bounding boxes) and \( \hat{M} \) is the corresponding ground-truth. To better illustrate the Gaussian response loss, we show the learning objective \( i.e. \), the ground-truth maps, the original response maps and the learned response maps in Figure 4.

In practice, when testing, we update the parameters in the three branches during the first 10 frames, with learning rate 5e-5 and decay ratio 1e-3. The KCF tracker is restarted when the online learning is finished.

![Figure 4](image)

**Figure 4.** Two examples of the proposed Gaussian response loss. From left to right: the initial response map before online learning; the response map after 10-frame update; the original image.

### 3. Experiment

**3.1. Dataset and competitors**

Similar to its prototype[12], the Object Tracking Benchmark 50 (OTB-50)[13] consists 50 video sequences and involves 51 tracking tasks. It is one of the most popular tracking benchmarks since the year 2013, The evaluation is based on two metrics: center location error and bounding box overlap ratio. The one-pass evaluation (OPE) is employed to compare our algorithm with some well-performing shallow visual trackers including the KCF tracker[14], TGPR[15], Struck[16], MIL[17], TLD[18] and SCM[19]. Also, some recently proposed deep trackers including MD-net[6], HCF[5], MSDAT[9], GOTURN[7] and the Siamese tracker[8] are also compared.

**3.2. Tracking results**

The comparison result curves are shown in Figure 5, note that we also report the running speeds (in FPS) of the comparing methods. From the curves we can see that the proposed method achieves comparable tracking accuracies to the state-of-the-art visual trackers while performs at 61 FPS on a high-end GPU (Nvidia GTX-1070). The number nearly double the speed of MSDAT. More importantly, on an average consuming GPU, Nvidia GTX-1050, our method performs at 27 FPS which is still real-time.
Figure 5. The location error plots and the overlapping accuracy plots of the involving trackers, tested on the OTB-50 dataset.

4. Conclusion
We propose a novel two-stage learning strategy for visual tracking. Firstly, a lightened CNN network is adopted for transferring the knowledge from the image classification domain to the visual tracking domain. When testing, the Gaussian response loss is used to update the “tracking branches” of the CNN model for the current tracking target. In this way, we obtain a deep tracker which runs real-time on average consuming GPUs while still maintain high tracking accuracy.

Acknowledgments
The work described in this paper was fully supported by a grant from Jiangxi Province Science Foundation for Youths (No. 20151BAB217019) and National Natural Science Foundation of China for Youths (No. 61503168)

References
[1] Naiyan Wang and Dit-Yan Yeung, “Learning a deep compact image representation for visual tracking,” in NIPS, pp. 809–817. 2013. More references.
[2] Hanxi Li, Yi Li, and Fatih Porikli, “Deeptrack: Learning discriminative feature representations online for robust visual tracking,” IEEE Transactions on Image Processing (TIP), vol. 25, no. 4, pp. 1834–1848, 2016.
[3] Anton Milan, Seyed Hamid Rezatofighi, Anthony Dick, Konrad Schindler, and Ian Reid, “Online multi-target tracking using recurrent neural networks,” arXiv preprint arXiv:1604.03635, 2016.
[4] Guanghan Ning, Zhi Zhang, Chen Huang, Zhihai He, Xiaobo Ren, and Haoqiong Wang, “Spatially supervised recurrent convolutional neural networks for visual object tracking,” arXiv preprint arXiv:1607.05781, 2016.
[5] Chao Ma, Jia-Bin Huang, Xiaokang Yang, and Ming-Hsuan Yang, “Hierarchical convolutional features for visual tracking,” in ICCV, 2015, pp. 3074–3082.
[6] Hyeonseob Nam and Bohyung Han, “Learning multi-domain convolutional neural networks for visual tracking,” arXiv preprint arXiv:1510.07945, 2015.
[7] David Held, Sebastian Thrun, and Silvio Savarese, “Learning to track at 100 fps with deep regression networks,” in European Conference on Computer Vision, 2016, pp. 749–765.
[8] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr, “Fully-convolutional siamese networks for object tracking,” in ECCV, 2016, pp. 850–865.
[9] Xinyu Wang, Hanxi Li, Yi Li, Fumin Shen, and Fatih Porikli, “Robust and real-time deep tracking via multi-scale domain adaptation,” 2017.
[10] Kye Hyeon Kim, Sanghoon Hong, Byungseok Roh, Yeongjae Cheon, and Minje Park, “Pvanet: Deep but lightweight neural networks for real-time object detection,” 2016.
[11] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, and Scott Reed, “Ssd: Single shot multibox detector,” arXiv preprint arXiv:1512.02325, 2015.

[12] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang, “Online object tracking: A benchmark,” in CVPR, 2013, pp. 2411–2418.

[13] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang, “Object tracking benchmark,” IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 37, no. 9, pp. 1834–1848, 2015.

[14] Joao F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista, “High-speed tracking with kernelized correlation filters,” IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 37, no. 3, pp. 583–596, 2015.

[15] Jin Gao, Haibin Ling, Weiming Hu, and Junliang Xing, “Transfer learning based visual tracking with gaussian processes regression,” in ECCV, pp. 188–203. 2014.

[16] Sam Hare, Amir Saffari, and Philip HS Torr, “Struck: Structured output tracking with kernels,” in ICCV, 2011, pp. 263–270.

[17] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie, “Visual tracking with online multiple instance learning,” IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), pp. 1619–1632, 2011.

[18] Zdenek Kalal, Jiri Matas, and Krystian Mikolajczyk, “Pn learning: Bootstrapping binary classifiers by structural constraints,” in CVPR, 2010, pp. 49–56.

[19] Wei Zhong, Huchuan Lu, and Ming-Hsuan Yang, “Robust object tracking via sparsity-based collaborative model,” in CVPR, 2012, pp. 1838–1845.