International Journal of Computer Vision
https://doi.org/10.1007/s11263-020-01357-4

# Unsupervised Deep Representation Learning for Real-Time Tracking

Ning Wang[1] · Wengang Zhou[1,2] · Yibing Song[3] · Chao Ma[4] · Wei Liu[3] · Houqiang Li[1,2]

Received: 17 December 2019 / Accepted: 9 July 2020


**Abstract**
The advancement of visual tracking has continuously been brought by deep learning models. Typically, supervised learning is employed to train these models with expensive labeled data. In order to reduce the workload of manual annotation and learn to track arbitrary objects, we propose an unsupervised learning method for visual tracking. The motivation of our unsupervised learning is that a robust tracker should be effective in bidirectional tracking. Specifically, the tracker is able to forward localize a target object in successive frames and backtrace to its initial position in the first frame. Based on such a motivation, in the training process, we measure the consistency between forward and backward trajectories to learn a robust tracker from scratch merely using unlabeled videos. We build our framework on a Siamese correlation filter network, and propose a multi-frame validation scheme and a cost-sensitive loss to facilitate unsupervised learning. Without bells and whistles, the proposed unsupervised tracker achieves the baseline accuracy of classic fully supervised trackers while achieving a real-time speed. Furthermore, our unsupervised framework exhibits a potential in leveraging more unlabeled or weakly labeled data to further improve the tracking accuracy.

**Keywords** Visual tracking · Unsupervised learning · Correlation filter · Siamese network

## 1 Introduction

Visual object tracking is a fundamental task in computer vision with numerous applications including video surveillance, autonomous driving, augmented reality, and human-computer interactions. It aims to localize a moving object annotated at the initial frame with a bounding box. Recently, deep models have improved the tracking accuracies by strengthening the feature representations (Ma et al. 2015; Danelljan et al. 2016, 2017) or optimizing networks end-to-end (Bertinetto et al. 2016; Li et al. 2018; Nam and Han 2016; Valmadre et al. 2017). These models are offline pretrained with full supervision, which requires a large number of annotated ground-truth labels during the training stage. Manual annotations are always expensive and time-consuming, whereas a huge number of unlabeled videos are readily available on the Internet. On the other hand, visual tracking differs from other recognition tasks (e.g., object detection, image classification) in the sense that object labels vary according to target initializations on the first frame. The extensive and uncertain labeling process for supervised learning raises our interest to develop an alternative learning scheme by using unlabeled video sequences in the wild.

Communicated by Mei Chen, Cha Zhang and Katsushi Ikeuchi.

✉ Wengang Zhou
zhwg@ustc.edu.cn

✉ Houqiang Li
lihq@ustc.edu.cn

Ning Wang
wn6149@mail.ustc.edu.cn

Yibing Song
yibingsong.cv@gmail.com

Chao Ma
chaoma@sjtu.edu.cn

Wei Liu
wl2223@columbia.edu

1 The CAS Key Laboratory of GIPAS, University of Science and Technology of China, Hefei, China

2 Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei, China

3 Tencent AI Lab, Shenzhen, China

4 The MOE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, Shanghai, China

Published online: 21 September 2020

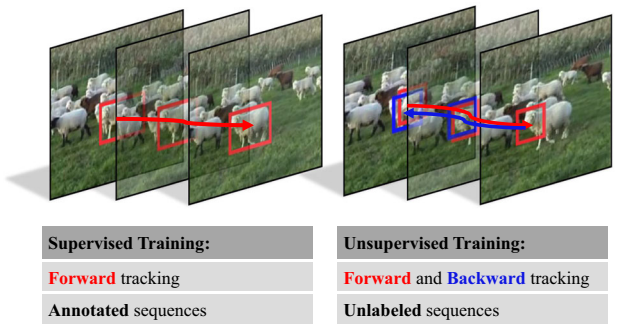



**Fig. 1** Visual tracking via supervised and unsupervised learnings. Supervised learning requires ground-truth labels for individual frames in the training videos, while our proposed unsupervised learning is free of any labels by measuring trajectory consistency in forward and backward trackings

In this paper, we propose an unsupervised learning approach for visual tracking. Instead of using off-the-shelf deep models, we train the visual tracking network from scratch. The intuition of unsupervised learning resides on the bidirectional motion analysis in video sequences. Tracking an object can be executed in both the forward and backward ways. Initially, given the bounding box annotation of a target object in the first frame, we can track the target object forward in the subsequent frames. When tracking backward, we use the predicted location in the last frame as the initial target bounding box, and track it backward towards the first frame. Ideally, the estimated bounding box location in the first frame is identical with the given one in the forward pass. In this work, we measure the difference between the forward and backward target trajectories and formulate it as a loss function. We use the computed loss to train our network in a self-supervised manner,[1] as shown in Fig. 1. By repeatedly tracking forward and backward, our model learns to locate target objects in consecutive frames without labeled supervision.

The proposed unsupervised training aims to learn a generic feature representation instead of strictly focusing on tracking a complete object. In the first frame, we initialize a bounding box that covers the informative local region with high image entropy. The bounding box may contain arbitrary image content and may not cover an entire object. Then, our tracking network learns to track the bounding box region in the training video sequences. Our unsupervised annotation shares similarity with the part-based (Liu et al. 2016) and edge-based (Li et al. 2017) tracking methods that track the subregions of a target object. We expect our tracker not only to concentrate on the shape of a complete object, but also to track any part of it. The bounding box initialization by image entropy gets rid of the manual annotation on the first frame and thus ensures the whole learning process unsupervised.

We employ unsupervised learning under the Siamese correlation filter framework. The training steps consist of forward tracking and backward verification. A limitation of the forward and backward consistency measurement is that the target trajectory in the forward pass may coincide with that in the backward pass although the tracker loses the target. The consistency loss function fails to penalize this situation because the predicted target region can still backtrace to the initial position on the first frame regardless of losing the target. In addition, challenges such as heavy occlusion or out-of-view in training videos will degrade the CNN feature representation capability. To tackle these issues, we introduce a multi-frame validation scheme and a cost-sensitive loss to facilitate unsupervised training. If the tracker loses the target, the trajectories predicted from the forward and backward directions are unlikely to be consistent when more frames are used in the training stage. Besides, we propose a new cost-sensitive loss to alleviate the impact of the noisy samples during unsupervised learning. The training samples containing background texture will be excluded by image entropy measurement. Based on the multi-frame validation and sample selection strategies discussed above, our network training is stabilized.

We evaluate our method on the challenging benchmark datasets including OTB-2013 Wu et al. (2013), OTB-2015 Wu et al. (2015), Temple-Color Liang et al. (2015), VOT2016 Kristan et al. (2016), VOT2017/2018 Kristan et al. (2018), LaSOT Fan et al. (2019), and TrackingNet Müller et al. (2018). Extensive experimental results indicate that without bells and whistles, the proposed unsupervised tracker is even comparable with the baseline configuration of fully supervised trackers (Bertinetto et al. 2016; Valmadre et al. 2017; Wang et al. 2017). When integrated with an adaptive online model update (Danelljan et al. 2016, 2017), the proposed tracker shows state-of-the-art performance. It is worth mentioning that our tracker trained via unsupervised learning achieves comparable performance with that via supervised learning when only limited or noisy labels are available. In addition, we demonstrate the potential of our tracker to further boost the accuracy by using more unlabeled data. A complete analysis of various training configurations is given in Sect. 4.2.

In summary, the contributions of this work are three-fold:

- We propose an unsupervised learning method on the Siamese correlation filter network. The unsupervised learning consists of forward and backward trackings to measure the trajectory consistency for network training.
- We propose a multi-frame validation scheme to enlarge the trajectory inconsistency when the tracker loses the target. In addition, we propose a cost-sensitive loss and

[1] In this paper, we do not distinguish between the terms *unsupervised* and *self-supervised*, as both refer to learning without ground-truth annotations.

an entropy selection metric to reduce the contributions from easy samples in the training process.
– The extensive experiments carried out on seven standard benchmarks show the favorable performance of the proposed tracker. We provide an in-depth analysis of our unsupervised representation and reveal the potential of unsupervised learning in visual tracking.

In the remainder of this paper, we describe the related work in Sect. 2, the proposed method in Sect. 3, and the experiments in Sect 4. Finally, we conclude the paper in Sect 5.

## 2 Related Work

In this section, we perform a literature review on deep tracking methods, forward–backward motion analysis, and unsupervised representation learning.

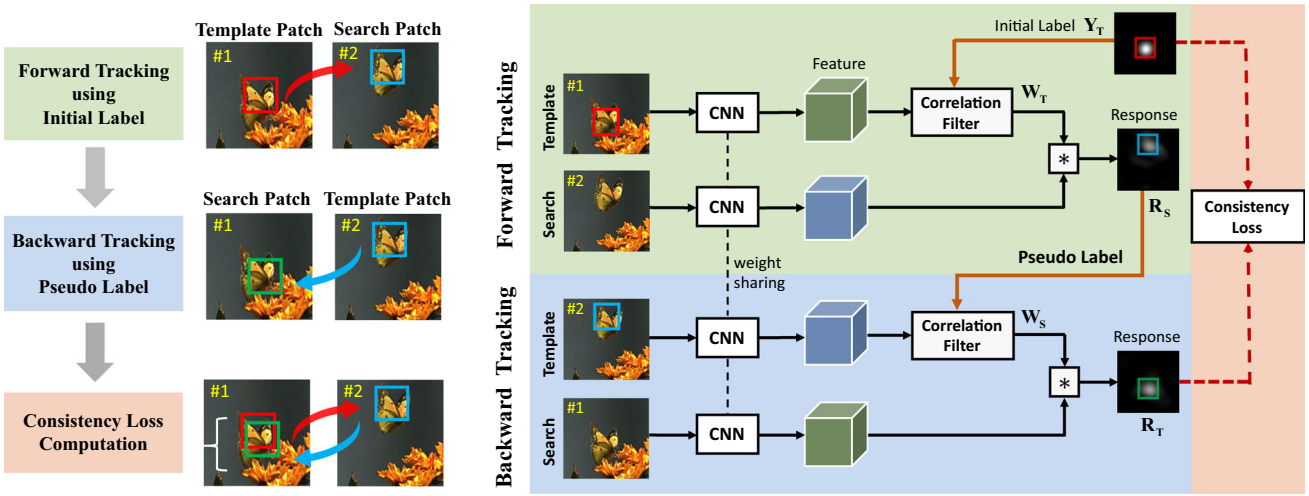
### 2.1 Deep Visual Tracking

Deep models have influenced visual tracking mainly from two perspectives. The first one is to provide a discriminative CNN feature representation by using off-the-shelf backbones (e.g., VGG Simonyan and Zisserman 2014; Chatfield et al. 2014), while the second one is to formulate a complete tracking network for end-to-end training and predictions. The discriminative correlation filters (DCFs) (Bolme et al. 2010; Henriques et al. 2015; Danelljan et al. 2014; Huang et al. 2017; Ma et al. 2018; Sui et al. 2019; Lukeźlăź et al. 2018) handle the visual tracking task by solving a ridge regression using densely sampled candidates. While being integrated with discriminative CNN features, the remaining operations (e.g., regression solver, online update) are kept still in the DCF trackers (Danelljan et al. 2016; Li et al. 2018; Wang et al. 2018; Danelljan et al. 2017). On the other hand, the end-to-end learning network can be categorized as classification and regression based networks. The classification networks (Nam and Han 2016; Song et al. 2018; Jung et al. 2018) incrementally train a binary classifier to differentiate the target and background distractors. The regression networks (Song et al. 2017; Lu et al. 2018) use CNN layers to regress CNN features of the search region to a response map for accurate localization. These end-to-end learning networks need online update and inevitably increase the computational burden.

Recently, the Siamese network has received huge investigations because of its efficiency in online prediction. The SiamFC tracker (Bertinetto et al. 2016) uses a cross-correlation layer to measure feature similarity between the template patch and search patches. The fully convolutional nature of SiamFC efficiently predicts the target response without redundancy. By incorporating the region proposal network (RPN) (Ren et al. 2016), the SiamRPN methods (Li et al. 2018; Zhu et al. 2018) achieve state-of-the-art performance while running at 160 FPS. Other improvements based on Siamese networks include ensemble learning (He et al. 2018), dynamic memory (Yang and Chan 2018), attention modulation (Wang et al. 2018), capacity increments (Zhipeng et al. 2019), and reinforcement learning (Huang et al. 2017; Dong et al. 2018). By integrating the correlation filter, the Siamese correlation filter network (Valmadre et al. 2017; Wang et al. 2017) achieves favorable performance even with an extremely lightweight model. Different from the above deep trackers that train a CNN model in a supervised manner or directly use off-the-shelf deep models, we propose to learn a tracking network from scratch using unlabeled videos via unsupervised learning.

### 2.2 Forward-Backward Analysis

The forward and backward strategy has been investigated in motion analysis scenarios. Meister et al. (2018) combined the forward–backward consistency estimation and pixel construction to learn optical flows. Wang et al. (2019) leveraged the cycle-consistency across multiple steps temporally to learn feature representations for different tasks. The differentiable tracker in Wang et al. (2019) is deliberately designed to be weak for feature representation learning. In contrast, aiming at robust visual tracking, we adopt a strong tracking baseline (Siamese correlation filter network), which is not fully-differentiable in the trajectory loop due to the pseudo labeling. However, by repeating forward tracking and backward verification, we incrementally promote the tracking network by pseudo-labeling based self-training. The forward–backward consistency check is also applied in image alignment (Zhou et al. 2015, 2016) and depth estimation (Yin and Shi 2018; Zhou et al. 2017). In the visual tracking community, the forward–backward consistency is mainly used for the output reliability or uncertainty measurement. The tracking-learning-detection (TLD) Kalal et al. (2012) uses the Kanade–Lucas–Tomasi (KLT) tracker (Tomasi and Kanade 1991) to perform forward–backward matching to detect tracking failures. Lee et al. (2015) proposed to select the reliable base tracker by comparing the geometric similarity, cyclic weight, and appearance consistency between a pair of forward–backward trajectories. However, these methods rely on empirical metrics to identify the target trajectories. In addition, repeatedly performing forward and backward trackings brings in a heavy computational cost for online tracking and largely hurts the real-time performance. Differently, in TrackingNet Müller et al. (2018), forward–backward analysis is used for evaluating the tracking performance and annotating the sparsely labeled dataset such as Youtube-BoundingBox Real et al. (2017) to obtain the per-frame object bounding box labels. In this work, we target at visual tracking but revisit the forward–




**(a) Unsupervised Learning Motivation** **(b) Unsupervised Learning Pipeline using a Siamese Network**

**Fig. 2** An overview of unsupervised learning in deep tracking. We show our motivation in **a** that we track forward and backward to compute the consistency loss for network training. The detailed training procedure is shown in **b**, where unsupervised learning is integrated into a Siamese correlation filter network. In the testing stage, we only track forward to predict the target location

backward scheme from a different view, i.e., train a deep visual tracker in an unsupervised manner.

### 2.3 Unsupervised Representation Learning

Our tracking framework relates to unsupervised representation learning. Learning feature representations from raw videos in an unsupervised manner has gained increasing attention in recent years. These approaches typically design ingenious techniques to explore and utilize the free supervision in images or videos. In Lee et al. (2017), the feature representation is learned by shuffling the video frames and then sorting them again to achieve self-supervised training. The multi-layer autoencoder on large-scale unlabeled data has been explored in Le et al. (2011). Vondrick et al. (2016) proposed to anticipate the visual representation of frames in the future. In Vondrick et al. (2018), colorized gray-scale videos by copying colors from a reference frame to learn a CNN model. Wang and Gupta (2015) used the KCF tracker (Henriques et al. 2015) to pre-process the raw videos, and then selected a pair of tracked images together with another random patch for learning CNNs using a ranking loss. Our method differs from Wang and Gupta (2015) significantly in two aspects. First, we integrate the tracking algorithm into unsupervised training instead of merely utilizing an off-the-shelf tracker as the data pre-processing tool. Second, our unsupervised framework is coupled with a tracking objective function, thus the learned feature representation is effective in presenting the generic target objects.

In the visual tracking community, unsupervised learning has rarely been touched. To our knowledge, the only related but different approach is the auto-encoder based method (Wang and Yeung 2013). However, the encoder-decoder is a general unsupervised framework (Olshausen and Field 1997), whereas our unsupervised method is specially designed for the tracking task. Since the visual objects or scenes in videos typically change smoothly, the motion information of the objects in a forward–backward trajectory loop provides free yet informative self-supervision signals for unsupervised learning, which is naturally suitable for the motion-related visual tracking task.

## 3 Proposed Method

The motivation of our unsupervised learning is shown in Fig. 2a. We first select a content-rich local region as the target object. Given this initialized bounding box label, we track forward to predict its location in the subsequent frame. Then, we reverse the sequence and take the predicted bounding box in the last frame as the pseudo label for backward verification. The predicted bounding box in the first frame via backward tracking is ideally identical to the original bounding box. We measure the difference between the forward and backward trajectories using the consistency loss to train the network. Figure 2b shows an overview of our unsupervised Siamese correlation filter network.

In the following, we first revisit the correlation filter as well as the Siamese network. In Sect. 3.2, we present our unsupervised learning prototype for an intuitive understanding. In Sect. 3.3, we improve our prototype to facilitate

unsupervised training. Finally, training details and online tracking are elaborated in Sects. 3.4 and 3.5, respectively.

### 3.1 Revisiting Correlation Tracking

The Discriminative Correlation Filters (DCFs) Bolme et al. (2010), Henriques et al. (2015) regress the circularly shifted versions of the input features of a search patch to a soft target response map for target localization. When training a DCF, we select a template patch $\mathbf{X}$ with the corresponding ground-truth label $\mathbf{Y}$, which is Gaussian-shaped with the peak localized at the target position. The size of the template patch is usually larger than that of the target. Figure 2 shows an example of the template patch, where there are both target and background contents. The filter $\mathbf{W}$ can be learned by solving the following ridge regression problem:

$$\min_{\mathbf{W}} \|\mathbf{W} * \mathbf{X} - \mathbf{Y}\|_2^2 + \lambda \|\mathbf{W}\|_2^2, \tag{1}$$

where $\lambda$ is a regularization parameter and $*$ denotes the circular convolution. Equation 1 can be efficiently calculated in the Fourier domain (Bolme et al. 2010; Danelljan et al. 2014; Henriques et al. 2015) and the DCF can be computed by

$$\mathbf{W} = \mathscr{F}^{-1}\left(\frac{\mathscr{F}(\mathbf{X}) \odot \mathscr{F}^{\star}(Y)}{\mathscr{F}^{\star}(\mathbf{X}) \odot \mathscr{F}(X) + \lambda}\right), \tag{2}$$

where $\odot$ is the element-wise product, $\mathscr{F}(\cdot)$ is the Discrete Fourier Transform (DFT), $\mathscr{F}^{-1}(\cdot)$ is the inverse DFT, and $\star$ denotes the complex-conjugate operation. In each subsequent frame, given a search patch $\mathbf{Z}$, its corresponding response map $\mathbf{R}$ can be computed in the Fourier domain:

$$\mathbf{R} = \mathbf{W} * Z = \mathscr{F}^{-1}\left(\mathscr{F}^{\star}(\mathbf{W}) \odot \mathscr{F}(\mathbf{Z})\right). \tag{3}$$

The above DCF framework starts from learning the target template's correlation filter (i.e., $\mathbf{W}$) using the template patch and then convolves it with a search patch $\mathbf{Z}$ to generate the response. Recently, the Siamese correlation filter network (Valmadre et al. 2017; Wang et al. 2017) embeds the DCF in the Siamese framework and constructs two shared-weight branches to extract feature representations, as shown in Fig. 2b. The first one is the template branch which takes a template patch $\mathbf{X}$ as input and extracts its features to further generate a target template filter via DCF. The second one is the search branch which takes a search patch $\mathbf{Z}$ as input for feature extraction. The template filter is then convolved with the CNN features of the search patch to generate the response map. The advantage of Siamese DCF network is that both the feature extraction CNN and correlation filter are formulated into an end-to-end framework, so the learned features are more related to the visual tracking scenarios.

### 3.2 Unsupervised Learning Prototype

Given two consecutive frames $P_1$ and $P_2$, we crop the template and search patches from them, respectively. By conducting forward tracking and backward verification, the proposed framework does not require additional supervision. The location difference between the initial bounding box and the predicted bounding box in $P_1$ will formulate a consistency loss. We utilize this loss to train the network without ground-truth annotations.

#### 3.2.1 Forward Tracking

Following the previous approaches (Valmadre et al. 2017; Wang et al. 2017), we build a Siamese correlation filter network to track the initialized bounding box region in frame $P_1$. After generating the template patch $\mathbf{T}$ from the first frame $P_1$, we compute the corresponding template filter $\mathbf{W_T}$ as follows:

$$\mathbf{W_T} = \mathscr{F}^{-1}\left(\frac{\mathscr{F}(\varphi_\theta(\mathbf{T})) \odot \mathscr{F}^{\star}(\mathbf{Y_T})}{\mathscr{F}^{\star}(\varphi_\theta(\mathbf{T})) \odot \mathscr{F}(\varphi_\theta(\mathbf{T})) + \lambda}\right), \tag{4}$$

where $\varphi_\theta(\cdot)$ denotes the CNN feature extraction operation with trainable network parameters $\theta$, and $\mathbf{Y_T}$ is the label of the template patch $\mathbf{T}$. This label is a Gaussian response centered at the initialized bounding box center. Once we obtain the learned template filter $\mathbf{W_T}$, the response map of a search patch $\mathbf{S}$ from frame $P_2$ can be computed by

$$\mathbf{R_S} = \mathscr{F}^{-1}(\mathscr{F}^{\star}(\mathbf{W_T}) \odot \mathscr{F}(\varphi_\theta(\mathbf{S}))). \tag{5}$$

If the ground-truth Gaussian label of patch $\mathbf{S}$ is available, the network $\varphi_\theta(\cdot)$ can be trained by computing the $L_2$ distance between $\mathbf{R_S}$ and the ground-truth label. Different from the supervised framework, in the following, we present how to train the network without requiring labels by exploiting backward trajectory verification.

#### 3.2.2 Backward Tracking

After generating the response map $\mathbf{R_S}$ for frame $P_2$, we create a pseudo Gaussian label centered at its maximum value, which is denoted by $\mathbf{Y_S}$. In backward tracking, we switch the role between the search patch and the template patch. By treating $\mathbf{S}$ as the template patch, we generate a template filter $\mathbf{W_S}$ using the pseudo label $\mathbf{Y_S}$. The template filter $\mathbf{W_S}$ can be learned using Eq. 4 by replacing $\mathbf{T}$ with $\mathbf{S}$ and replacing $\mathbf{Y_T}$ with $\mathbf{Y_S}$, as follows:

$$\mathbf{W_S} = \mathscr{F}^{-1}\left(\frac{\mathscr{F}(\varphi_\theta(\mathbf{S})) \odot \mathscr{F}^{\star}(\mathbf{Y_S})}{\mathscr{F}^{\star}(\varphi_\theta(\mathbf{S})) \odot \mathscr{F}(\varphi_\theta(\mathbf{S})) + \lambda}\right). \tag{6}$$

Then, we generate the response map $\mathbf{R_T}$ of the template patch through Eq. 5 by replacing $\mathbf{W_T}$ with $\mathbf{W_S}$ and replacing $\mathbf{S}$ with

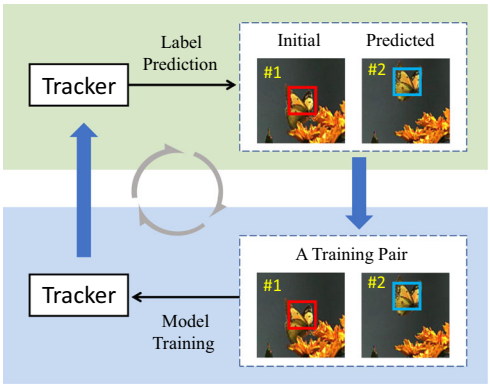



**Fig. 3** The intuition of pseudo-labeling based self-training. We use the same network for both forward and backward predictions. The forward stage generates a pseudo label for the search patch. The backward stage updates the tracking network using training pairs via loss back-propagation. During training iterations, the response map of the template gradually approaches the initial label via self supervision

$\mathbf{T}$, as shown in Eq. 7.

$$\mathbf{R_T} = \mathscr{F}^{-1}(\mathscr{F}^{\star}(\mathbf{W_S}) \odot \mathscr{F}(\varphi_\theta(\mathbf{T}))). \tag{7}$$

Note that we only use one Siamese correlation filter network for executing forward and backward trackings. The network parameters $\theta$ are fixed during the tracking steps.

#### 3.2.3 Consistency Loss Computation

After forward and backward tracking, we obtain the response map $\mathbf{R_T}$. Ideally, $\mathbf{R_T}$ should be a Gaussian label with the peak located at the initialized target position. In other words, $\mathbf{R_T}$ should be as similar as the originally given label $\mathbf{Y_T}$. Therefore, the representation network $\varphi_\theta(\cdot)$ can be trained in an unsupervised manner by minimizing the reconstruction error as follows:

$$\mathcal{L}_{\text{un}} = \|\mathbf{R_T} - \mathbf{Y_T}\|_2^2. \tag{8}$$

Our unsupervised learning can be viewed as an incremental self-training process that iteratively predicts labels and updates the model to steadily improve the tracking capability. Figure 3 shows the intuition, where we use the same network for both forward and backward predictions. In the forward tracking, we generate a pseudo label $\mathbf{Y_S}$ for the search patch $\mathbf{S}$. Then we treat generated $\mathbf{Y_S}$ as the label of $\mathbf{S}$ and create a corresponding sample. Using these labeled training pairs (i.e., with initial or pseudo labels), we can update the Siamese correlation filter network in a similar way to supervised learning. During loss back-propagation, we follow the Siamese correlation filter methods (Wang et al. 2017; Zhang et al. 2018) to update the network:

$$\begin{aligned}
\frac{\partial \mathcal{L}_{\text{un}}}{\partial \varphi_\theta(\mathbf{T})} &= \mathscr{F}^{-1}\left(\frac{\partial \mathcal{L}_{\text{un}}}{\partial\left(\mathscr{F}\left(\varphi_\theta(\mathbf{T})\right)\right)^{\star}} + \left(\frac{\partial \mathcal{L}_{\text{un}}}{\partial\left(\mathscr{F}\left(\varphi_\theta(\mathbf{T})\right)\right)}\right)^{\star}\right), \\
\frac{\partial \mathcal{L}_{\text{un}}}{\partial \varphi_\theta(\mathbf{S})} &= \mathscr{F}^{-1}\left(\frac{\partial \mathcal{L}_{\text{un}}}{\partial\left(\mathscr{F}\left(\varphi_\theta(\mathbf{S})\right)\right)^{\star}}\right).
\end{aligned} \tag{9}$$

The above unsupervised training process is based on the forward–backward consistency between two frames, which is summarized by Algorithm 1. In the next section, we extend this prototype framework to consider multiple frames for better network training.

### 3.3 Enhancement for Unsupervised Learning

The proposed unsupervised learning method constructs the objective function based on the consistency between $\mathbf{R_T}$ and $\mathbf{Y_T}$. In practice, the tracker may deviate from the target in the forward tracking but still return to the original position during the backward process. However, the proposed loss function does not penalize this deviation because of the consistent trajectories. Meanwhile, the raw videos may contain textureless or occluded training samples that deteriorate the unsupervised learning process. In this section, we propose a multi-frame validation scheme and a cost-sensitive loss to tackle these two limitations.

#### 3.3.1 Multi-frame Validation

We propose a multi-frame validation approach to enlarge the trajectory inconsistency when the tracker loses the target. Our intuition is to incorporate more frames during training to reduce the limitation that the erroneous localization in the subsequent frame successfully backtraces to the initial position in the first frame. In this way, the reconstruction error in Eq. 8 will effectively capture the inconsistent trajectory. As shown in Fig. 3, adding more frames in the forward stage further challenges the model tracking capability.

Our unsupervised learning prototype can be easily extended to multiple frames. To build a trajectory cycle using three frames, we can involve another frame $P_3$ which is the subsequent frame after $P_2$. We crop a search patch $\mathbf{S}_1$ from $P_2$ and another search patch $\mathbf{S}_2$ from $P_3$. If the generated response map $\mathbf{R}_{\mathbf{S}_1}$ is different from its corresponding ground-truth response, the difference tends to become larger in the next frame $P_3$. As a result, the inconsistency is more likely to appear in backward tracking, and the generated response map $\mathbf{R_T}$ is more likely to differ from $\mathbf{Y_T}$, as shown in Fig. 4. By involving more search patches during forward and backward trackings, the proposed consistency loss will be more effective to penalize the inaccurate localizations.

We can further extend the number of frames utilized for multi-frame validation. The length of trajectory will increase as shown in Fig. 5. The limitation of consistent trajectory when losing the target is more unlikely to affect the training

**Algorithm 1:** Unsupervised training prototype

**Input**: Unlabeled videos.
**Output**: Pretrained tracking network $\varphi_\theta(\cdot)$.

```
Crop the patches (i.e., T and S) from the raw videos;
Initialize the CNN model φ_θ(·) with random weights θ;
for each training epoch do
    for each training pair do
        Obtain φ_θ(T) and φ_θ(S);
        // Forward Trajectory
        Construct W_T using φ_θ(T) and Y_T (Eq. 4);
        Compute R_S using W_T (Eq. 5) and obtain the pseudo
        label of S;
        // Backward Trajectory
        Construct W_S using φ_θ(S) and Y_S (Eq. 6);
        Compute the response map R_T of T (Eq. 7);
        // Calculate Consistency Loss
        Compute the consistency loss of Y_T and R_T (Eq. 8);
    end
    Update network φ_θ(·) using the computed loss;
end
```

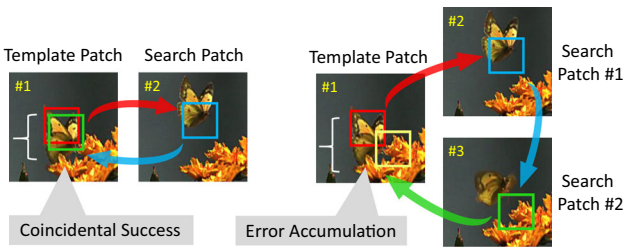



**Fig. 4** Single frame validation and multi-frame validation. The inaccurate localization in single frame validation may not be captured as shown on the left. By involving more frames as shown on the right, we accumulate the localization errors to break the prediction consistency during forward and backward trackings

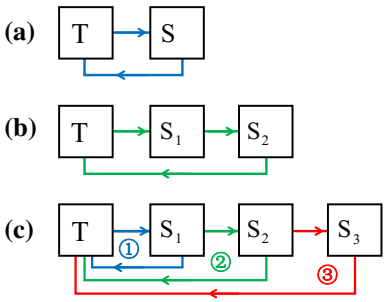



**Fig. 5** An overview of multi-frame trajectory consistency. We denote **T** as a template and **S** as a search patch, respectively. Our unsupervised training prototype is shown in **a**, where only two frames are involved. Using more frames as shown in **b** and **c**, we can gradually improve the training performance to overcome consistent trajectories when losing the target

process. Let $\mathbf{R}_{(\mathbf{S}_k \to \mathbf{T})}$ denote the response map of the template $\mathbf{T}$, which is generated (or tracked) by the DCF trained using the $k$th search patch $\mathbf{S}_k$. The corresponding consistency loss function can be computed as follows:

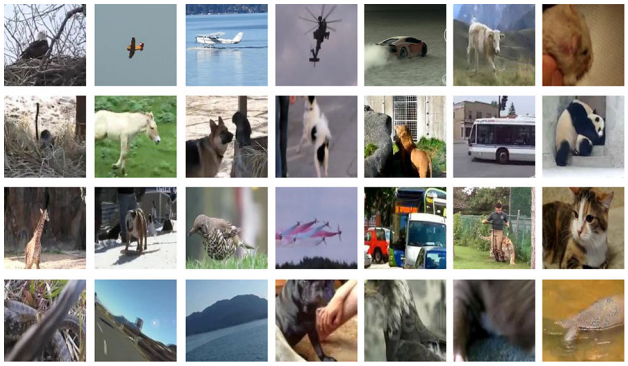


**Fig. 6** Examples of the cropped image patches from ILSVRC 2015 Russakovsky et al. (2015). Most of these samples contain meaningful objects, while some samples are less meaningful (e.g., last row)

$$\mathcal{L}_k = \left\| \mathbf{R}_{(\mathbf{S}_k \to \mathbf{T})} - \mathbf{Y}_\mathbf{T} \right\|_2^2 . \tag{10}$$

Considering different trajectory cycles, the multi-frame consistency loss can be computed by

$$\mathcal{L}_{\text{un}} = \sum_{k=1}^{M} \mathcal{L}_k , \tag{11}$$

where $k$ is the index of the search parch. Taking Fig. 5c as an example, the final consistency objective contains three losses (i.e., $M = 3$ in Eq. 11), which are denoted by the blue, green, and red cycles in Fig. 5c, respectively.

### 3.3.2 Cost-sensitive Loss

We initialize a bounding box region as a training sample in the first frame during unsupervised training. The image content within this bounding box region may contain arbitrary or partial objects. Figure 6 shows an overview of these regions. To alleviate the background interference, we propose a cost-sensitive loss to effectively exclude noisy samples for network training. For simplicity, we use three consecutive frames as an example to illustrate sample selection, which can be naturally extended to more frames. The pipeline of using three frames is shown in Fig. 5b.

During unsupervised learning, we construct multiple training triples from video sequences. For a trajectory containing three frames, each training triple consists of one initialized template patch $\mathbf{T}$ in frame $P_1$ and two search patches $\mathbf{S}_1$ and $\mathbf{S}_2$ in the subsequent frames $P_2$ and $P_3$, respectively. We use several triples to form a training batch for Siamese network learning. In practice, we find that some training triples with extremely high losses prevent network training from convergence. To reduce these outlier effects in pseudo-labeling based self-training, we exclude 10% of the whole training triples which contain the highest loss values. Their losses can be computed using Eq. 10. To this end, we

assign a binary weight $\mathbf{A}^i_{\text{drop}}$ to each training triple. All these weights constitute a vector $\mathbf{A}_{\text{drop}}$, where 10% of its elements are 0 and the others are 1.

In addition to the outlier training pairs, the raw videos include meaningless image patches, where there are textureless backgrounds or still objects. In these patches, the objects (e.g., sky, grass, or tree) do not contain big movements. We assign a motion weight vector $\mathbf{A}_{\text{motion}}$ to all the training pairs to increase the large motion effect for network learning. Each element $\mathbf{A}^i_{\text{motion}}$ within this vector can be computed by

$$\mathbf{A}^i_{\text{motion}} = \left\| \mathbf{R}^i_{\mathbf{S}_1} - \mathbf{Y}^i_{\mathbf{T}} \right\|_2^2 + \left\| \mathbf{R}^i_{\mathbf{S}_2} - \mathbf{Y}^i_{\mathbf{S}_1} \right\|_2^2, \tag{12}$$

where $\mathbf{R}^i_{\mathbf{S}_1}$ and $\mathbf{R}^i_{\mathbf{S}_2}$ are the response maps in the $i$-th training pair, and $\mathbf{Y}^i_{\mathbf{T}}$ and $\mathbf{Y}^i_{\mathbf{S}_1}$ are the corresponding initial (or pseudo) labels. Equation 12 calculates the target motion difference from frame $P_1$ to $P_2$ and $P_2$ to $P_3$. When the value of $\mathbf{A}^i_{\text{motion}}$ is large, the target object undergoes fast motion in this trajectory. On the other hand, the large value of $\mathbf{A}^i_{\text{motion}}$ represents the hard training pair which the network should pay more attention to. We normalize the motion weight and the binary weight as follows:

$$\mathbf{A}^i_{\text{norm}} = \frac{\mathbf{A}^i_{\text{drop}} \cdot \mathbf{A}^i_{\text{motion}}}{\sum_{i=1}^{N} \mathbf{A}^i_{\text{drop}} \cdot \mathbf{A}^i_{\text{motion}}}, \tag{13}$$

where $N$ is number of the training pairs in a mini-batch. The sample weight $\mathbf{A}^i_{\text{norm}}$ serves as a scalar that reweighs the training data without gradient back-propagation.

The final unsupervised loss for the case of Fig. 5b in a mini-batch is computed as:

$$\mathcal{L}_{\text{3-frame}} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{A}^i_{\text{norm}} \cdot \left\| \mathbf{R}^i_{(\mathbf{S}_2 \to \mathbf{T})} - \mathbf{Y}^i_{\mathbf{T}} \right\|_2^2. \tag{14}$$

We can naturally extend Eq. 14 to the following by using more frames to construct trajectories of different lengths, as illustrated by the toy example of Fig. 5c. Combining with Eq. 11, we compute the final unsupervised loss function using $M$ subsequent frames as:

$$\mathcal{L}_{\text{final}} = \frac{1}{N} \sum_{k=1}^{M} \sum_{i=1}^{N} \mathbf{A}^i_{\text{norm}} \cdot \mathcal{L}^i_k, \tag{15}$$

where $\mathcal{L}^i_k = \left\| \mathbf{R}^i_{(\mathbf{S}_k \to \mathbf{T})} - \mathbf{Y}^i_{\mathbf{T}} \right\|_2^2$ is similar to that in Eq. 10 but with the index $i$ for differentiating different samples in a mini-batch.

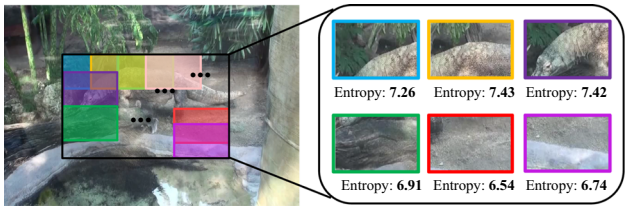



**Fig. 7** The illustration of training samples generation. The proposed method crops 5 × 5 patch candidates in the center region of the initial frame. Then we select the image patch with the highest image entropy. As shown in the right figure, the background patches (e.g., labeled by green and red boxes) have a small image entropy

### 3.4 Unsupervised Training Details

**Network Structure.** We follow the DCFNet Wang et al. (2017) to use a shallow Siamese network consisting of two convolutional layers for tracking. This shallow structure is demonstrated effective in CFNet Valmadre et al. (2017) to integrate DCF formulation. The filter sizes of these convolutional layers are $3 \times 3 \times 3 \times 32$ and $3 \times 3 \times 32 \times 32$, respectively. Besides, a local response normalization (LRN) layer is employed at the end of convolutional layers following Wang et al. (2017). This lightweight structure enables efficient forward inferences for online tracking.

**Training Data.** We choose ILSVRC 2015 Russakovsky et al. (2015) as our training data, which is the same dataset employed by existing supervised trackers. In the data preprocessing step, supervised approaches Bertinetto et al. (2016), Valmadre et al. (2017), Wang et al. (2017) require per-frame labels. Besides, the frames will be removed, where the target object is occluded, partially out-of-view, or in an irregular shape (e.g., snake). The data pre-precessing for the supervised approaches is time-consuming with human labor. In contrast, our method does not rely on manually annotated labels for data pre-processing.

In our approach, for the first frame in a raw video, we crop overlapped small patches (5 × 5 in total) by sliding windows as shown in Fig. 7. Then, we compute the image entropy of each image patch. Image entropy effectively measures the content variance of an image patch. When an image patch only contains the unitary texture (e.g., the sky), the entropy of this patch approaches 0. When an image patch contains textured content, the entropy will become higher. We select the cropped image patch containing the highest image entropy. This image patch initializes the KCF Henriques et al. (2015) tracker for localization in the subsequent frames. Then, we crop a larger image patch with a padding of 2 times of the target size following DCFNet Wang et al. (2017), which is further resized to 125 × 125 as the input of our network. Figure 6 exhibits some examples of the cropped patches. We randomly choose 4 cropped patches from the continuous 10 frames in a video to form a training trajectory, and one of

them is defined as the template and the rest as search patches. This is based on the assumption that the center located target objects are unlikely to move out of the cropped region in a short span of time. We track the content in the image patch regardless of specific object categories. Although this entropy-based method may not accurately select a target region and the KCF tracker is not robust enough to track the cropped region, this method can well alleviate the meaningless background regions.

### 3.5 Online Object Tracking

After offline unsupervised learning, we perform online tracking in the way of forward tracking as illustrated in Sect. 3.2. We online update the DCF to adapt to the target appearance changes. The DCF update follows a moving average operation shown as follows:

$$\mathbf{W}_t = (1 - \alpha_t)\mathbf{W}_{t-1} + \alpha_t \mathbf{W}, \tag{16}$$

where $\alpha_t \in [0, 1]$ is the linear interpolation coefficient. The target scale is estimated through a patch pyramid with scale factors $\{a^s | a = 1.015, s = \{-1, 0, 1\}\}$ (Danelljan et al. 2015). We name our Tracker as LUDT (i.e., Learning Unsupervised Deep Tracking). Besides, we update our model adaptively via $\alpha_t$ and follow the superior DCF formulation as that in ECO Danelljan et al. (2017). We name the improved tracker as LUDT+.

We keep the notation of our preliminary tracker UDT and UDT+ Wang et al. (2019) in the following experiment section. Our previous UDT uses a 3-frame cycle (Fig. 5b) and simply crops the center patch in raw videos. LUDT improves UDT in two aspects: (1) LUDT combines different trajectory cycles as shown in Fig. 5c and (2) LUDT utilizes image entropy to select the informative image patches instead of the center crop. The LUDT+ and UDT+ improve LUDT and UDT by adopting some online tracking techniques (e.g., adaptive update) proposed in Danelljan et al. (2017), respectively.

## 4 Experiments

In this section, we first analyze the effectiveness of our unsupervised training framework and discuss our network potentials. Then, we compare our tracker LUDT against state-of-the-art trackers on both standard and recently released large-scale benchmarks including OTB-2013 Wu et al. (2013), OTB-2015 Wu et al. (2015), Temple-Color Liang et al. (2015), VOT2016 Kristan et al. (2016), VOT2017/2018 Kristan et al. (2018), LaSOT Fan et al. (2019), and TrackingNet Müller et al. (2018).

### 4.1 Experimental Details

In our experiments, we use the stochastic gradient descent (SGD) with a momentum of 0.9 and a weight decay of 0.005 to train our model. Our unsupervised network is trained for 50 epoches with a learning rate exponentially decaying from $10^{-2}$ to $10^{-5}$ and a mini-batch size of 32. We set the trajectory length as 4. All the experiments are executed on a PC with 4.00GHz Intel Core I7-4790K and NVIDIA GTX 1080Ti GPU. On a single GPU, our LUDT and LUDT+ exhibit about 70 FPS and 55 FPS, respectively. The source code will be updated at https://github.com/594422814/UDT.

The proposed method is evaluated on seven benchmarks. On the OTB-2013/2015, TempleColor, LaSOT, and TrackingNet datasets, we use one-pass evaluation (OPE) with distance and overlap precision metrics. The distance precision threshold is set as 20 pixels. The overlap success plot uses thresholds ranging from 0 to 1, and the area-under-curve (AUC) is computed to evaluate the overall performance. On the VOT2016 and VOT2017/2018 datasets, we measure the performance using Expected Average Overlap (EAO).

### 4.2 Ablation Experiments and Discussions

#### 4.2.1 Improvements upon UDT

Our preliminary tracker UDT Wang et al. (2019) adopts a three-frame validation (i.e., Fig. 5b) and the center crop for sample generation. The improvement upon UDT is that we construct a multi-supervision consistency loss function using more frames. We denote this strategy as Trajectory Enlargement (TE) in Table 1. Meanwhile, we select the RoI (region of interest) from raw videos using image entropy and KCF tracker, while only the center region is utilized in UDT. We denote RoI Selection as RS in the table. Note that the performance of UDT has been close to that of its supervised configuration and exceeded several supervised trackers. Moreover, under the same training configuration, LUDT steadily improves UDT by using TE and RS during training. LUDT achieves 60.2% and 51.5% under the AUC on the OTB-2015 and Temple-Color benchmarks, respectively.

#### 4.2.2 Baseline Performance

To verify the effectiveness of the proposed unsupervised framework, we evaluate our tracker using different feature extractors. As shown in Table 2, without pre-training, the model still exhibits a weak tracking capability, which can be attributed to the discriminating power of the correlation filter. By adopting the empirical HOG representations, the performance is still significantly lower than ours. Furthermore, we leverage the auto-encoder framework (Wang and Yeung 2013) to train the backbone network in an unsuper-

**Table 1** Ablation study of Trajectory Enlargement (TE) and RoI Selection (RS). We denote UDT as our preliminary tracker (Wang et al. 2019). We integrate TE and RS into UDT during training and report the performance improvement. The evaluation metrics are DP and AUC scores on the OTB-2015 and Temple-Color datasets

| | OTB-2015 DP/AUC (%) | Temple-color DP/AUC (%) |
|---|---|---|
| UDT Wang et al. (2019) | 76.0/59.4 | 65.8/50.7 |
| UDT + TE | 76.5/59.8 | 66.7/51.2 |
| UDT + RS | 76.5/60.0 | 66.9/51.3 |
| UDT + TE + RS | 76.9/60.2 | 67.1/51.5 |

**Table 2** Comparison results of the DCFNet tracking framework with different feature extractors. Random: the randomly initialized feature extractor without pre-training

| | Random | HOG | ED | LUDT (Ours) |
|---|---|---|---|---|
| DP (%) | 59.1 | 69.2 | 71.6 | 76.9 |
| AUC (%) | 46.9 | 52.1 | 54.5 | 60.2 |

HOG: adopting HOG Dalal and Triggs (2005) without deep features. ED: the backbone network trained via encoder-decoder (Wang and Yeung 2013). The evaluation metrics are DP and AUC scores on OTB-2015

vised manner using the same training data. From Table 2, we can observe that our approach is superior to the encoder-decoder in this tracking scenario since our forward–backward based unsupervised training is tightly related to object tracking.

#### 4.2.3 Training Data

We evaluate the tracking performance using different data pre-processing strategies. The results are shown in Table 3. Our unsupervised LUDT method uses the last RoI selection strategy. During the evaluation, we keep the remaining modules fixed in LUDT.

**Comparison with Full Supervision**. Using the same videos (i.e., ILSVRC 2015 Russakovsky et al. 2015), we conduct the supervised training of our network. The supervised learning with ground-truth annotations can be regarded as the upper bound of our unsupervised learning. We observe that the performance gap is small (i.e., 2.6% AUC) between the trackers trained using unsupervised learning (60.0% AUC) and fully supervised learning (62.6% AUC).

**Comparison with Weak Supervision**. In ILSVRC 2015, we add deviations to the ground-truth boxes to crop the training samples. The deviations range from -20 pixels to 20 pixels randomly. The reason for setting sample deviations from the ground-truth bounding boxes is that we aim to simulate the inaccurate object localizations on in-the-wild videos using existing object detection or optical flow approaches. We assume that these deviated samples are predicted by existing methods and then utilized to train our unsupervised network. In Table 3, we observe that our tracker learned by these weakly labeled samples is comparable with the supervised results (61.4% vs. 62.6% AUC). Note that 20 pixels deviations can be achieved with many object localization methods. The comparable performance indicates that our method can be applied to raw videos with weakly or sparsely labeled annotations (e.g., the dataset Youtube-BB Real et al. 2017). On the other hand, existing object detectors and models are mostly trained by supervised learning. To ensure our method to be fully unsupervised, we use two unsupervised data pre-processing methods: center cropping and RoI selection based on entropy.

**Center Cropping**. In center cropping, we crop the center region of the video frame. Although we crop a fixed region of the image, the image content appears randomly in this region and we denote this operation as center cropping. There may be meaningless content (e.g., textureless sky, ocean) in this region to disturb our unsupervised learning. The tracker learned by center cropping achieves an AUC score of 59.4%.

**RoI Selection**. We use the entropy-based image patch selection as illustrated in Sect. 3.4. Compared to the center cropping, image-entropy based selection can suppress the meaningless background samples such as sky and grass, and the KCF tracker is able to capture the selected informative region in the subsequent frames. The RoI selection achieves better performance than center cropping with an AUC score of 60.0%.

#### 4.2.4 Trajectory Length

As discussed in Sect. 3.3.1, trajectory enlargement helps measure the consistency loss when the tracker loses RoI. In Table 4, we show the performance with different trajectory lengths on the OTB-2015 dataset. We use center cropping to generate training samples following UDT for comparison. The prototype of our unsupervised learning is denoted as 2 frames validation. By incorporating the third frame, the learned tracker achieves improvement (i.e., 2.8% DP and 2.0% AUC). The 4 frames validation proposed in this work not only extends the trajectory length but also combines multiple self-supervision constraints, which further improves the accuracy. However, the 5 frames validation seems to be less effective. It may be because the validation with 4 frames already contains adequate self-supervision and effectively measures the consistency loss.

#### 4.2.5 Cost-sensitive Loss

On the OTB-2015 dataset, without hard sample reweighing (i.e., $\mathbf{A}_{\text{motion}}$ in Eq. 13), the performance of our LUDT tracker drops about 1.5% DP and 1% AUC score. We did not conduct

**Table 3** Evaluation results of our network trained using different data pre-processing strategies. Our LUDT tracker uses RoI selection via image entropy for unsupervised training. The evaluation metrics are DP and AUC scores on the OTB-2015 dataset

| | Groundtruth label<br>Full supervision | Groundtruth label with deviations<br>Weak supervision | Center cropping<br>Unsupervision | RoI selection via entropy<br>Unsupervision |
|---|---|---|---|---|
| DP (%) | 80.6 | 78.9 | 76.0 | 76.5 |
| AUC (%) | 62.6 | 61.4 | 59.4 | 60.0 |

**Table 4** Evaluation results of our unsupervised model trained using different trajectory lengths. Note that the 4 and 5 frames validations conduct multiple self-supervisions as illustrated in Fig. 5. The evaluation metrics are DP and AUC score on the OTB-2015 dataset. Compared with 3 frames validation, using more frames further improves the tracking accuracy

| Frame number | 2 frames<br>Figure 5a | 3 frames<br>Figure 5b | 4 frames<br>Figure 5c | 5 frames<br>akin to Figure 5c |
|---|---|---|---|---|
| DP (%) | 73.2 | 76.0 | 76.8 | 76.8 |
| AUC (%) | 57.4 | 59.4 | 59.8 | 59.7 |

the ablation study of the sample dropout because we observe that the unsupervised training cannot well converge without $\mathbf{A}_{\text{drop}}$ illustrated in Eq. 14.

#### 4.2.6 Unlabled Data Augmentation

**Few-shot Domain Adaptation**. To better fit a new domain such as OTB, we construct a small training set by collecting the first several frames (e.g., 5 frames in our experiment) from the videos in OTB-2015 with only the ground-truth bounding box in the first frame available. Using these limited samples, we fine-tune our network by 100 iterations using the forward–backward pipeline, which takes about 6 minutes. As our learning method is unsupervised, we can utilize the frames from test sequences to adapt our tracker. Table 5 shows that performance is further improved by using this strategy. Our offline unsupervised training learns general feature representation, which can be transferred to an interested domain (e.g., OTB videos) using few-shot domain adaptation. This domain adaptation is similar to that in MDNet Nam and Han (2016), while our network parameters are initially offline learned in an unsupervised manner.

**Additional Internet Videos**. We also utilize more unlabeled videos to train our network. These videos are from the OxUvA dataset Valmadre et al. (2018), where there are 337 videos in total. The OxUvA dataset is a subset of Youtube-BB Real et al. (2017) collected on YouTube. By adding these videos during training, our tracker improves the original one by 0.7% DP and 1.2% AUC as shown in Table 5. By leveraging another large-scale LaSOT dataset Fan et al. (2019) where there are 1200 videos collected on the Internet, the tracking performance is further improved. It indicates that unlabeled data advances the unsupervised training. As our framework is fully unsupervised, it has the potential to take advantage of the in-the-wild videos on the Internet to boost the performance.

#### 4.2.7 Empirical Features Embedding

As shown in Table 6, we train unsupervised LUDT+ using more unlabeled video sequences (both ILSVRC and LaSOT), which outperforms ECOhc Danelljan et al. (2017) leveraging hand-crafted features including HOG Dalal and Triggs (2005) and ColorName Weijer et al. (2009). In addition, we can combine the learned CNN features and empirical features to generate a more discriminative representation. We add the HOG feature to LUDT+ during tracking and evaluate its performance. Table 6 shows that this combination achieves a 65.7% AUC on OTB-2015. Moreover, embedding the HOG feature helps LUDT+ to outperform most state-of-the-art real-time trackers as shown in Table 7. Besides feature embedding and adaptive model update, there are still many improvements from Wang et al. (2018), Galoogahi et al. (2017), Mueller et al. (2017), Lukezic et al. (2017) available to benefit our tracker. However, adding more additional mechanisms is out the scope of this work. Following SiamFC and DCFNet, we currently use LUDT/LUDT+ trackers which are *only* trained on the ILSVRC dataset for *fair comparison* in the following evaluations.

### 4.3 Visualization of Unsupervised Representation

After learning the unsupervised Siamese tracking network, we visualize the network response to see how it differs from the same network trained using supervised learning. Figure 8 shows the visualization performance. The first column shows the input frames. The network responses from unsupervised learning and supervised learning are shown in the second and third columns, respectively. The remaining columns show the

**Table 5** Performance study by adding additional training data. Adding more unlabeled dataset steadily improves the tracking results. The evaluation metrics are DP and AUC scores on the OTB-2015 dataset

| | LUDT | Few-shot fine-tune OTB-2015 | More data OxUvA | More data LaSOT |
|---|---|---|---|---|
| DP (%) | 76.9 | 78.1 | 77.6 | 78.2 |
| AUC (%) | 60.2 | 61.5 | 61.4 | 62.0 |

**Table 6** Performance potential of our unsupervised tracker. When using more data (LaSOT) for network training, the performance further improves. By incorporating empirical features (HOG), our unsupervised tracker achieves superior results. The performance is evaluated on the OTB-2015 dataset using DP and AUC metrics

| | ECOhc | LUDT + only ILSVRC | LUDT + more data | LUDT + more data + HOG |
|---|---|---|---|---|
| DP (%) | 85.4 | 84.3 | 85.5 | 85.8 |
| AUC (%) | 64.1 | 63.9 | 64.9 | 65.7 |
| Speed (FPS) | 60 | 55 | 55 | 42 |

**Table 7** Evaluations with fully-supervised baseline (left) and state-of-the-art trackers (right) on the popular OTB-2015 benchmark Wu et al. (2015). The evaluation metrics are DP and AUC scores. Our unsupervised LUDT tracker performs favorably against popular baseline methods (left), while our LUDT+ tracker achieves comparable results with the recent state-of-the-art supervised trackers (right)

| Trackers | SiamFC | DCFNet | CFNet | LUDT | EAST | HP | SA-Siam | SiamRPN | RASNet | SACF | Siam-tri | RT-MDNet | MemTrack | StructSiam | LUDT+ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DP (%) | 77.1 | - | 74.8 | 76.9 | - | 79.6 | 86.5 | 85.1 | - | 83.9 | 78.1 | 88.5 | 82.0 | 85.1 | 84.3 |
| AUC (%) | 58.2 | 58.0 | 56.8 | 60.2 | 62.9 | 60.1 | 65.7 | 63.7 | 64.2 | 63.3 | 59.2 | 65.0 | 62.6 | 62.1 | 63.9 |
| FPS | 86 | 70 | 65 | 70 | 25 | 159 | 69 | 160 | 83 | 23 | 86 | 50 | 50 | 45 | 55 |

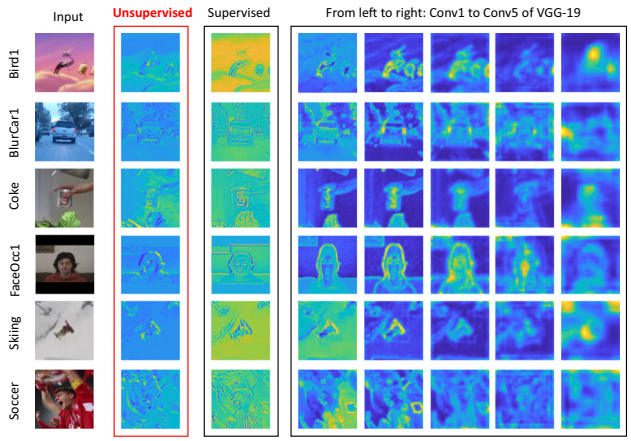



**Fig. 8** Visualization of feature representations. First column: input image patches. Second and third columns: feature representations of our unsupervised LUDT and fully-supervised LUDT. The rest columns: feature maps from VGG-19 (from left to right: Conv1-2, Conv2-2, Conv3-4, Conv4-4, and Conv5-4). The feature map is visualized by averaging all the channels. Best viewed in color and zoom in

feature responses from the off-the-shelf deep model VGG-19 Simonyan and Zisserman (2014). We observe that the responses from the unsupervised learning and supervised learning are similar with minor differences. Specifically, the boundary responses from supervised learning are higher than those of unsupervised learning. This is because of the strong supervisions brought by the ground-truth labels. The network has learned to differentiate the target and background distractors according to labels, which increases the network attention around the object boundaries. In comparison, our unsupervised learning does not employ this process for attention enhancement, while still focusing on the center region of the object responses. From the viewpoint of Siamese network, both unsupervised and supervised feature representations focus on the target appearances, which facilitate the template matching through the correlation operation. Compared with the empirical features (e.g., HOG), we will show in the following that our unsupervised feature representations achieve higher accuracy compared with hand-crafted features.

Our unsupervised representation is compared with the off-the-shelf deep model VGG-19. Note that the VGG model is trained under image classification task with supervised learning. We show the feature maps from different layers (i.e., Conv1-2, Conv2-2, Conv3-4, Conv4-4, and Conv5-4) of the VGG-19. From Fig. 8, we observe that our unsupervised feature representations share similarities with the low-level features (i.e., the first two layers) of VGG, which typically represents spatial details. It has been well studied in HCF Ma et al. (2015) and C-COT Danelljan et al. (2016) that only using the first or second layer of the VGG model for DCF tracking contains limitations. However, our unsupervised representation better suits the tracking scenario since we jointly combine the feature representation learning with

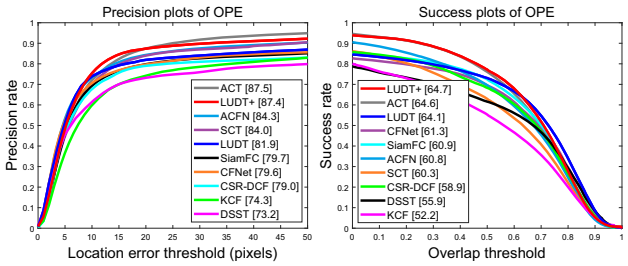



**Fig. 9** Precision and success plots on the OTB-2013 dataset Wu et al. (2013) for recent real-time trackers. The legend in each tracker shows the precision at 20 pixels of precision plot and AUC of success plot

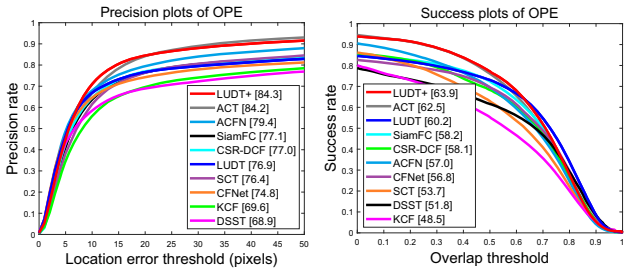



**Fig. 10** Precision and success plots on the OTB-2015 dataset Wu et al. (2015) for recent real-time trackers. The legend in each tracker shows the precision at 20 pixels of precision plot and AUC of success plot

the DCF formulation in an end-to-end fashion. In the deeper layers of VGG-19 such as Conv4-4 and Conv5-4, the feature representation gradually loses spatial details but increases semantics, which can be combined with the low-level features to further boost the tracking performance (Ma et al. 2015; Danelljan et al. 2016). The semantic representation capability is obtained by distinguishing different object categories (i.e., image classification), while our unsupervised learning process lacks such image labels. In the future, we will investigate how to learn rich multiple-level representations for visual tracking in an unsupervised manner.

### 4.4 Comparison with State-of-the-art Methods

**OTB-2013 Dataset.** The OTB-2013 dataset Wu et al. (2013) contains 50 challenging videos. On the OTB-2013 dataset, we evaluate our LUDT and LUDT+ trackers with state-of-the-art real-time trackers including ACT Chen et al. (2018), ACFN Choi et al. (2017), CFNet Valmadre et al. (2017), SiamFC Bertinetto et al. (2016), SCT Choi et al. (2016), CSR-DCF Lukezic et al. (2017), DSST Danelljan et al. (2014), and KCF Henriques et al. (2015) using precision and success plots.

As illustrated in Fig. 9, our unsupervised LUDT tracker outperforms CFNet and SiamFC in both distance precision and AUC score. It is worth mentioning that both LUDT and CFNet have similar network capability (network depth), leverage the same training data, and are not equipped with additional online improvements. Even though our approach is free of ground-truth supervision, it still achieves very competitive tracking accuracy. Our improved version, LUDT+, performs favorably against recent state-of-the-art real-time trackers such as ACT and ACFN. Besides, our LUDT and LUDT+ trackers also exhibit a real-time speed of about 70 FPS and 55 FPS, respectively.

**OTB-2015 Dataset.** The OTB-2015 dataset Wu et al. (2015) contains 100 challenging videos. On the OTB-2015 dataset Wu et al. (2015), we evaluate LUDT and LUDT+ trackers with state-of-the-art real-time algorithms as that in OTB-2013. In Table 7, we further compare our methods with more state-of-the-art real-time trackers such as StructSiam Zhang et al. (2018), MemTrack Yang and Chan (2018), RT-MDNet Jung et al. (2018), Siam-tri Dong and Shen (2018), SACF Zhang et al. (2018), RASNet Wang et al. (2018), SiamRPN Li et al. (2018), SA-Siam He et al. (2018), HP Dong et al. (2018), and EAST Huang et al. (2017).

From Fig. 10 and Table 7, we observe that our unsupervised LUDT tracker is comparable with supervised baseline methods (e.g., SiamFC and CFNet). On the OTB-2015 dataset, SiamFC achieves 77.1% DP and 58.2% AUC, while LUDT exhibits 76.9% DP and 60.2% AUC. Compared with CFNet, LUDT outperforms by 2.1% DP and 3.4% AUC. The DSST algorithm is a traditional DCF based tracker with accurate target scale estimation. LUDT significantly outperforms it by 8.0% DP and 8.4% AUC, which illustrates that our unsupervised feature representation is more robust than empirical features (e.g., HOG). With a better DCF formulation and more advanced online update strategies Danelljan et al. (2017), our LUDT+ tracker achieves comparable performance with the recent ACFN and ACT trackers. In Fig. 10 and Table 7, we do not compare with some remarkable non-realtime trackers. For example, MDNet Nam and Han (2016) and ECO Danelljan et al. (2017) can yield 67.8% and 69.4% AUC on the OTB-2015, but they are far from achieving a real-time speed.

Table 7 compares more recent supervised trackers. These latest approaches are mainly based on the Siamese network, which improve the baseline SiamFC method using various sophisticated techniques. Most trackers in Table 7 are trained using ILSVRC including LUDT+. However, it is worth mentioning that some algorithms (e.g., SA-Siam and RT-MDNet) adopt pre-trained CNN models (e.g., AlexNet Krizhevsky et al. 2012 and VGG-M Chatfield et al. 2014) for network initialization. The SiamRPN additionally uses more labeled training videos from the Youtube-BB dataset Real et al. (2017). Compared with them, LUDT+ does not require data labels or off-the-shelf deep models, while still achieving comparable performance and efficiency.

**Temple-Color Dataset.** Temple-Color Liang et al. (2015) is a more challenging benchmark with 128 color videos. In

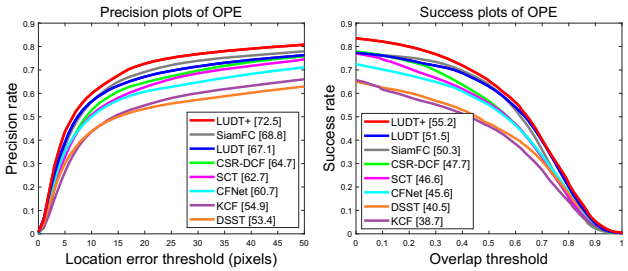



**Fig. 11** Precision and success plots on the Temple-Color dataset Liang et al. (2015) for recent real-time trackers. The legend in each tracker shows the precision at 20 pixels of precision plot and AUC of success plot

this dataset, we compare our trackers with some baseline and state-of-the-art trackers as on the OTB-2015 benchmark. Compared with the DCF trackers with empirical features (e.g., HOG feature), our tracker with unsupervised deep features exhibits a significant performance improvement as shown in Fig. 11. Specifically, SiamFC which is learned with full supervision achieves an AUC score of 50.3%, while LUDT exhibits 51.3% AUC score. Compared with another representative supervised method CFNet, LUDT exceeds its performance by 6.4% DP and 4.7% AUC. Furthermore, our LUDT+ tracker performs favorably against existing state-of-the-art trackers.

**VOT2016 Dataset.** We report the evaluation results on the VOT2016 benchmark Kristan et al. (2016), which contains 60 videos selected from more than 300 videos. Different from the OTB dataset, the VOT toolkit will reinitialize when the tracker fails. The expected average overlap (EAO) is the final metric for tracker ranking Kristan et al. (2016). In Fig. 12, we show the accuracy-robustness (AR) plot and EAO ranking plot on VOT2016 with some participant trackers. The VOT2016 champion C-COT uses the pre-trained VGG-M model for feature extraction while not achieving real-time performance. The proposed LUDT+ method performs slightly worse than C-COT but runs much faster. It is worth mentioning that our real-time LUDT+ tracker even performs favorably against remarkable non-realtime deep trackers such as MDNet. Our LUDT tracker, without bells and whistles, surpasses classic DCF trackers such as DSST and KCF by a considerable margin and is comparable with some DCF methods with an off-the-shelf deep model (e.g., DeepMKCF and HCF).

In Table 8, we include more state-of-the-art trackers including VITAL Song et al. (2018), DSLT Lu et al. (2018), RTINet Yao et al. (2018), ACT Chen et al. (2018), SA-Siam He et al. (2018), SiamRPN Li et al. (2018), SACF Zhang et al. (2018), StructSiam Zhang et al. (2018), and MemTrack Yang and Chan (2018) on the VOT2016 benchmark. Compared with the baseline SiamFC, our LUDT tracker yields favorable results. Compared with fully-supervised trackers,

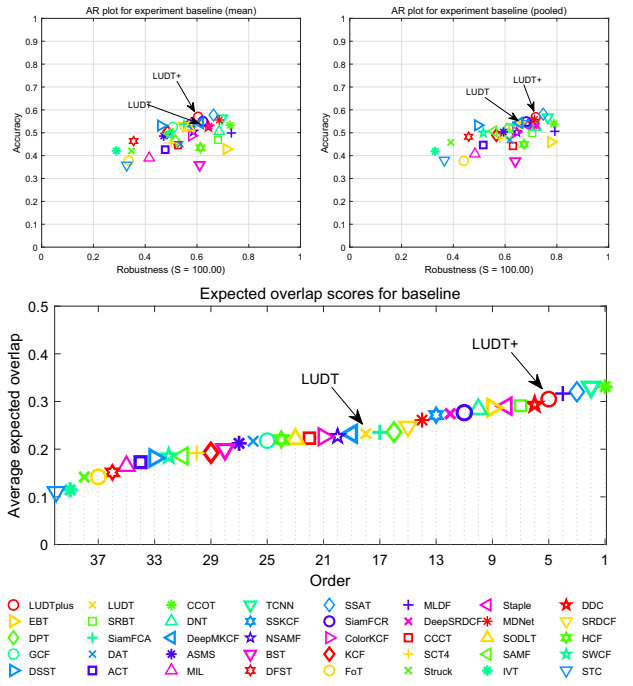



**Fig. 12** Top: Accuracy-Robustness (AR) ranking plots generated by sequence mean (left) and sequence pooling (right) on the VOT2016 dataset Kristan et al. (2016). Trackers closer to upper right corner perform better. Bottom: Expected Average Overlap (EAO) graph with trackers ranked from right to left evaluated on VOT2016

**Table 8** Comparison with state-of-the-art and baseline trackers on the VOT2016 benchmark Kristan et al. (2016). The evaluation metrics include Accuracy, Failures (over 60 sequences), and Expected Average Overlap (EAO). The up arrows indicate that higher values are better for the corresponding metric and vice versa

| Trackers | Accuracy (↑) | Failures (↓) | EAO (↑) | FPS (↑) |
|---|---|---|---|---|
| ECO | 0.54 | - | 0.374 | 6 |
| VITAL | - | - | 0.323 | 1 |
| DSLT | - | - | 0.332 | 6 |
| RTINet | 0.57 | - | 0.298 | 9 |
| C-COT | 0.52 | 51 | 0.331 | 0.3 |
| pyMDNet | - | - | 0.304 | 2 |
| HCF | 0.45 | 85 | 0.220 | 12 |
| ACT | - | - | 0.275 | 30 |
| SA-Siam | 0.53 | - | 0.291 | 50 |
| SiamRPN | 0.56 | - | 0.344 | 160 |
| SACF | - | - | 0.275 | 23 |
| StructSiam | - | - | 0.264 | 45 |
| MemTrack | 0.53 | - | 0.273 | 50 |
| SiamFC | 0.53 | 99 | 0.235 | 86 |
| SCT4 | 0.48 | 117 | 0.188 | 40 |
| DSST | 0.53 | 151 | 0.181 | 25 |
| KCF | 0.49 | 122 | 0.192 | 170 |
| LUDT (Ours) | 0.54 | 100 | 0.231 | 70 |
| LUDT+ (Ours) | 0.54 | 62 | 0.309 | 55 |

**Table 9** Comparison with state-of-the-art and baseline trackers on the VOT2017/2018 benchmark Kristan et al. (2016). The evaluation metrics include Accuracy, Failures (over 60 sequences), and Expected Average Overlap (EAO). The up arrows indicate that higher values are better for the corresponding metric and vice versa

| Trackers | Accuracy (↑) | Failures (↓) | EAO (↑) | FPS (↑) |
|---|---|---|---|---|
| ECO | 0.48 | 59 | 0.280 | 6 |
| C-COT | 0.49 | 68 | 0.267 | 0.3 |
| SA-Siam | 0.50 | - | 0.236 | 50 |
| SiamRPN | - | - | 0.243 | 160 |
| SiamFC | 0.50 | 125 | 0.188 | 86 |
| Staple | 0.53 | 147 | 0.169 | 70 |
| TRACA | 0.42 | 183 | 0.137 | 100 |
| SRDCF | 0.49 | 208 | 0.119 | 5 |
| DSST | 0.40 | 310 | 0.079 | 25 |
| KCF | 0.45 | 165 | 0.135 | 170 |
| LUDT (Ours) | 0.46 | 149 | 0.154 | 70 |
| LUDT+ (Ours) | 0.49 | 88 | 0.230 | 55 |

LUDT+ overall exhibits competitive performance as well as efficiency.

**VOT2017/2018 Dataset.** The VOT2017 Kristan et al. (2017) and VOT2018 Kristan et al. (2018) are the same benchmark with more challenging videos compared with those in VOT2016 dataset. In Table 9, we present the Accuracy, Failures, and EAO of the state-of-the-art trackers on VOT2017/VOT2018. The proposed LUDT tracker is still superior to the standard DCF trackers using hand-crafted features such as DSST and KCF. Our LUDT+ yields an EAO score of 0.230, which is comparable with the advanced Siamese trackers such as SA-Siam and SiamRPN that take advantage of additional backbone networks or training data.

**LaSOT Dataset.** We further evaluate our unsupervised approach on the large-scale LaSOT testing dataset Fan et al. (2019) with 280 videos. The videos in LaSOT are more challenging with an average length of about 2500 frames. As shown in Fig. 13, our LUDT tracker still outperforms hand-crafted feature based DCF trackers such as BACF Galoogahi et al. (2017), CSR-DCF LukeźIăź et al. (2018), DSST Danelljan et al. (2014), and SCT4 Choi et al. (2016). Furthermore, the proposed LUDT+ approach achieves an AUC score of 30.5%, which is even comparable with some state-of-the-art deep DCF trackers including ECO (32.4%) Danelljan et al. (2017), STRCF (30.8%) Li et al. (2018), and TRACA (25.7%) Choi et al. (2018) that leverage off-the-shelf deep models as feature extractors.

**TrackingNet Dataset.** The recently released large-scale TrackingNet dataset Müller et al. (2018) contains more than 30K videos with more than 14 million dense bounding box annotations. The videos are collected on the Internet (YouTube), providing large-scale high-quality data for

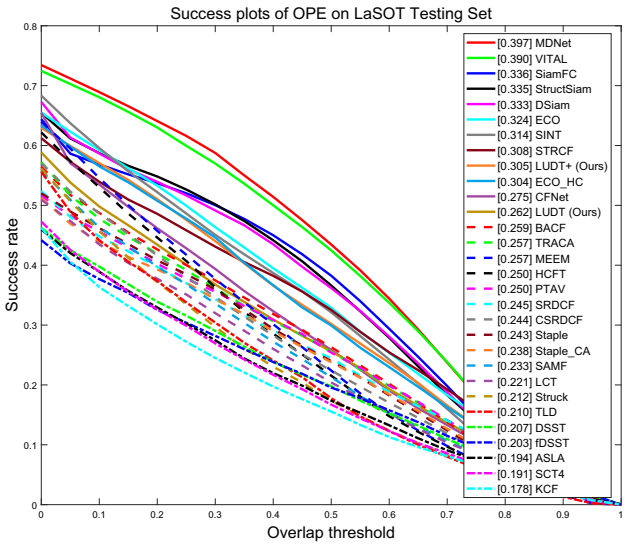



**Fig. 13** Success plots on the LaSOT testing set Fan et al. (2019). The legend in each tracker shows the AUC of the success plot. Best viewed in color and zoom in

**Table 10** Comparison with state-of-the-art and baseline trackers on the TrackingNet benchmark Müller et al. (2018). The evaluation metrics include Precision, Normalized Precision, and Success (AUC score)

| Trackers | Precision | Norm.prec | Success |
|---|---|---|---|
| MDNet | 0.565 | 0.705 | 0.606 |
| CFNet | 0.533 | 0.654 | 0.578 |
| SiamFC | 0.533 | 0.663 | 0.571 |
| ECO | 0.492 | 0.618 | 0.554 |
| ECOhc | 0.476 | 0.608 | 0.541 |
| CSRDCF | 0.480 | 0.622 | 0.534 |
| Staple_CA | 0.468 | 0.605 | 0.529 |
| Staple | 0.470 | 0.603 | 0.528 |
| BACF | 0.461 | 0.580 | 0.523 |
| SRDCF | 0.455 | 0.573 | 0.521 |
| SAMF | 0.477 | 0.598 | 0.504 |
| ASLA | 0.406 | 0.536 | 0.478 |
| SAMF_AT | 0.447 | 0.560 | 0.472 |
| DLSSVM | 0.418 | 0.562 | 0.470 |
| DSST | 0.460 | 0.588 | 0.464 |
| MEEM | 0.386 | 0.545 | 0.460 |
| Struck | 0.402 | 0.539 | 0.456 |
| DCF | 0.419 | 0.548 | 0.448 |
| KCF | 0.419 | 0.546 | 0.447 |
| CSK | 0.368 | 0.503 | 0.429 |
| TLD | 0.336 | 0.460 | 0.417 |
| TLD | 0.292 | 0.438 | 0.400 |
| MOSSE | 0.326 | 0.442 | 0.388 |
| LUDT (Ours) | 0.469 | 0.593 | 0.543 |
| LUDT+ (Ours) | 0.495 | 0.633 | 0.563 |

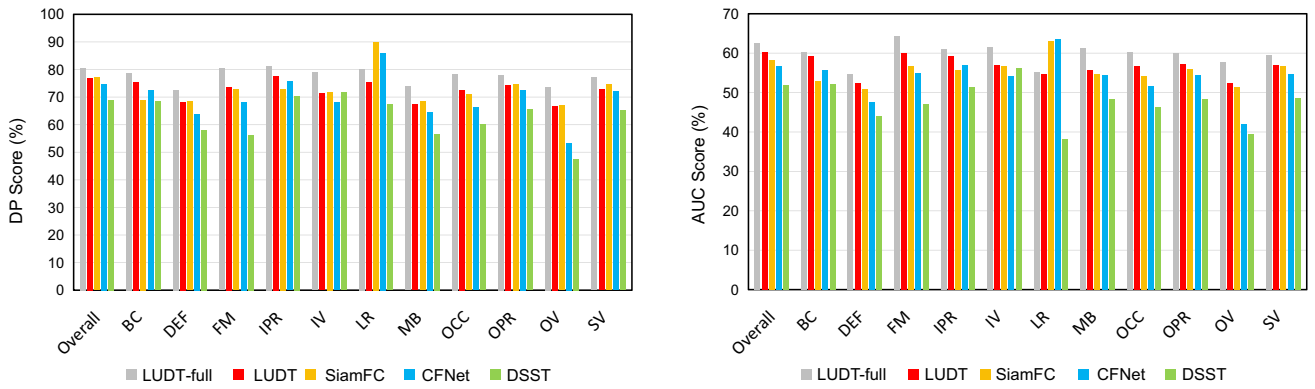



**Fig. 14** Attribute-based evaluation on the OTB-2015 dataset Wu et al. (2015). The 11 attributes are background clutter (BC), deformation (DEF), fast motion (FM), in-plane rotation (IPR), illumination variation (IV), low resolution (LR), motion blur (MB), occlusion (OCC), out-of-plane rotation (OPR), out-of-view (OV), and scale varition (SV), respectively

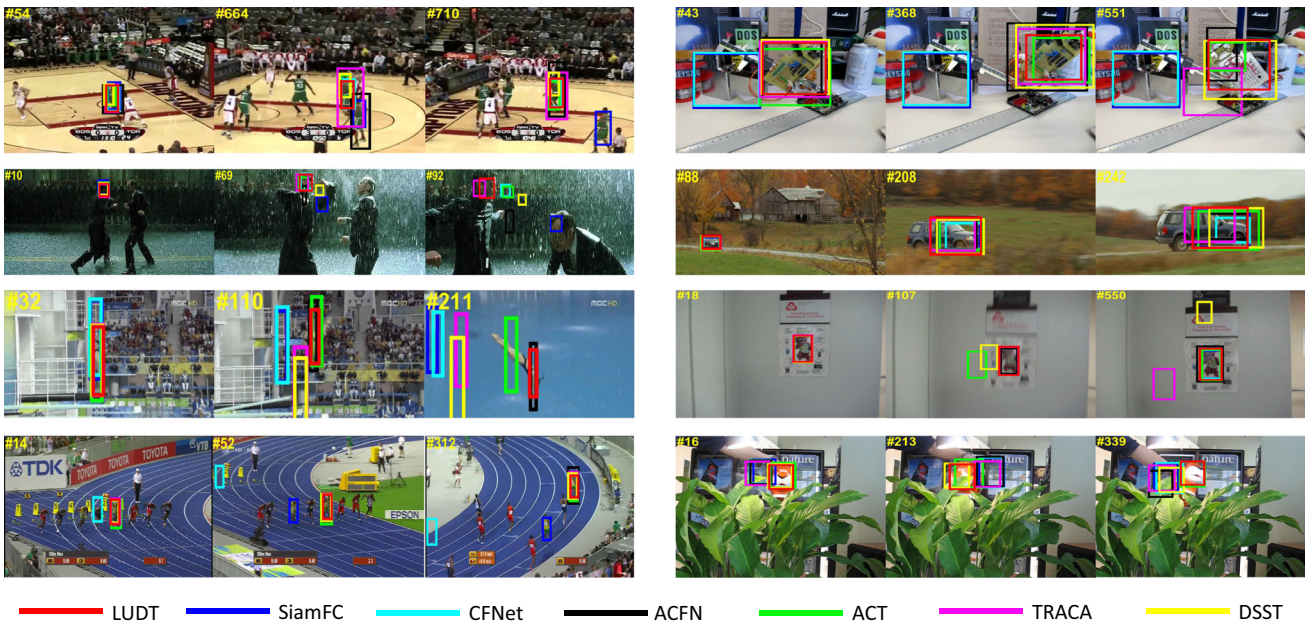



**Fig. 15** Qualitative evaluation of our proposed LUDT and other trackers including SiamFC Bertinetto et al. (2016), CFNet Valmadre et al. (2017), ACFN Choi et al. (2017), ACT Chen et al. (2018), TRACA Choi et al. (2018), and DSST Danelljan et al. (2014) on 8 challenging videos from OTB-2015. From left to right and top to down are *Basketball*, *Board*, *Matrix*, *CarScale*, *Diving*, *BlurOwl*, *Bolt*, and *Tiger1*, respectively. Best viewed in color

assessing trackers in the wild. We test our LUDT and LUDT+ on the testing set with 511 videos. Following Müller et al. (2018), we adopt three metrics including Precision, Normalized Precision, and Success (AUC) for performance evaluation. In Table 10, we exhibit the results of our methods and all the evaluated trackers on this benchmark. On this dataset, our LUDT achieves an AUC score of 54.3%, which obviously outperforms other hand-crafted feature based DCF trackers such as ECOhc, CSR-DCF, and BACF by 0.2%, 0.9%, and 2.0%. Note that the above DCF trackers are improved versions with additional regularization terms, while ours merely utilizes a standard DCF formulation. Our superior performance illustrates the representational power of our unsupervised features. Besides, it is worth mentioning that our LUDT, on this large-scale benchmark, is even comparable with the state-of-the-art ECO, which leverages both hand-crafted and off-the-shelf deep features. Without labeled data for model training, our improved LUDT+ achieves better performance and slightly outperforms ECO by 0.9% in terms of AUC.

**Attribute Analysis.** The videos on OTB-2015 Wu et al. (2015) are annotated with 11 different attributes, namely: background clutter (BC), deformation (DEF), out-of-plane rotation (OPR), scale variation (SV), occlusion (OCC), illumination variation (IV), motion blur (MB), in-plane rotation (IPR), out-of-view (OV), fast motion (FM), and low resolu-

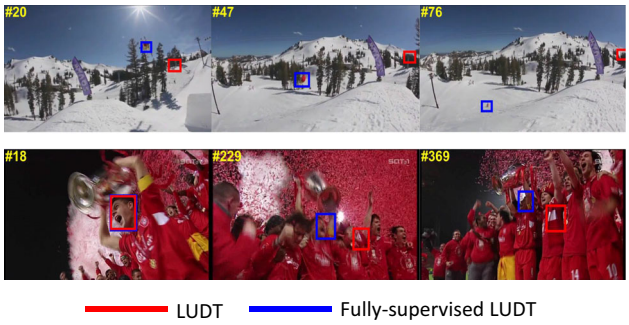



**Fig. 16** Failure cases of our LUDT tracker. The top and bottom videos are *Skiing* and *Soccer*, respectively. Compared to its fully-supervised version, our unsupervised method is not robust enough when target undergoes drastic appearance change and occlusion

tion (LR). On the OTB-2015 benchmark, we further analyze the performances over different challenges in Fig. 14. On the majority of challenging scenes, our LUDT tracker outperforms the popular SiamFC Bertinetto et al. (2016) and CFNet Valmadre et al. (2017) trackers. However, our performance advantage on DP metric is less obvious than that under the AUC metric. Compared with the fully-supervised LUDT tracker, the main performance gaps are from illumination variation (IV), occlusion (OCC), and fast motion (FM) attributes. Unsupervised learning can be further improved on these attributes.

**Qualitative Evaluation.** We evaluate LUDT with supervised trackers (e.g., ACT, ACFN, SiamFC, TRACA, and CFNet) and a baseline DCF tracker (DSST) on eight challenging videos, as shown in Fig. 15. On *Matrix* and *Tiger1* videos, the targets undergo partial occlusion and background clutter, while on *BlurOwl*, the target is extremely blurry due to the drastic camera shaking. In these videos, DSST based on empirical features fails to cope with the challenging factors while LUDT is able to handle. This illustrates the robustness of our unsupervised feature representation, which achieves favorable performance compared to the empirical features. The SiamFC and CFNet trackers tend to drift when the target and distractors are similar (e.g., *Bolt* and *Basketball* sequences) while LUDT is able to handle these challenging scenes because of the discrimination capability of the DCF and its online model update mechanism. Without online improvements, LUDT is still able to track the target accurately, especially on the challenging *Board* and *Diving* videos. It is worth mentioning that such a robust tracker is learned from raw videos in an unsupervised manner.

### 4.5 Limitations

Figure 16 shows the limitations of our unsupervised learning. First, compared with the fully supervised learning, our tracker trained via unsupervised learning tends to drift when occlusion or drastic appearance change occurs (e.g., the targets in *Skiing* and *Soccer* sequences). The semantic representations brought by ground-truth annotations are missing. Second, our unsupervised learning involves both forward and backward trackings. The computational load during the training phase is a potential drawback although the learning process is offline.

## 5 Conclusion

In this paper, we present how to train a visual tracker using unlabeled videos in the wild, which is rarely investigated in visual tracking. By designing an unsupervised Siamese correlation filter network, we verify the feasibility and effectiveness of our forward–backward based unsupervised training pipeline. To further facilitate the unsupervised training, we extend our framework to consider multiple frames and employ a cost-sensitive loss. Extensive experiments exhibit that the proposed unsupervised tracker, without bells and whistles, performs as a solid baseline and achieves comparable results with the classic fully-supervised trackers. Equipped with additional online improvements such as a sophisticated update scheme, our LUDT+ tracker performs favorably against the state-of-the-art tracking algorithms. Furthermore, we provide a deep analysis of our unsupervised representation by feature visualization and extensive ablation studies. Our unsupervised framework shows a promising potential in visual tracking, such as utilizing more unlabeled data or weakly labeled data to further improve the tracking accuracy.

**Acknowledgements** This work was supported in part to Dr. Houqiang Li by NSFC under contract No. 61836011, and in part to Dr. Wengang Zhou by NSFC under contract No. 61822208 & 61632019 and Youth Innovation Promotion Association CAS (No. 2018497). Dr. Chao Ma was supported by NSFC under contract No. 60906119 and Shanghai Pujiang Program.

## References

Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., & Torr, P.H. (2016). Fully-convolutional siamese networks for object tracking. In *Proceedings of the European conference on computer vision workshops (ECCV workshop)*.

Bolme, D.S., Beveridge, J.R., Draper, B.A., & Lui, Y.M. (2010). Visual object tracking using adaptive correlation filters. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*

Chatfield, K., Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Return of the devil in the details: Delving deep into convolutional nets. In *British machine vision conference (BMVC)*.

Chen, B., Wang, D., Li, P., Wang, S., & Lu, H. (2018). Real-time'actor-critic'tracking. In: *Proceedings of the European conference on computer vision (ECCV)*.

Choi, J., Jin Chang, H., Fischer, T., Yun, S., Lee, K., Jeong, J., Demiris, Y., & Young Choi, J. (2018). Context-aware deep feature compres-

sion for high-speed visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Choi, J., Jin Chang, H., Jeong, J., Demiris, Y., & Young Choi, J. (2016). Visual tracking using attention-modulated disintegration and integration. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Choi, J., Jin Chang, H., Yun, S., Fischer, T., Demiris, Y., & Young Choi, J. (2017). Attentional correlation filter network for adaptive visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Danelljan, M., Bhat, G., Shahbaz Khan, F., & Felsberg, M. (2017). Eco: Efficient convolution operators for tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Danelljan, M., Häger, G., Khan, F., & Felsberg, M. (2014). Accurate scale estimation for robust visual tracking. In *British machine vision conference (BMVC)*.

Danelljan, M., Häger, G., Khan, F.S., & Felsberg, M. (2016) Adaptive decontamination of the training set: A unified formulation for discriminative visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Danelljan, M., Hager, G., Shahbaz Khan, F., & Felsberg, M. (2015). Learning spatially regularized correlation filters for visual tracking. In *Proceedings of the IEEE international conference on computer vision (ICCV)*.

Danelljan, M., Robinson, A., Khan, F.S., & Felsberg, M. (2016). Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *Proceedings of the european conference on computer vision (ECCV)*.

Dong, X., & Shen, J. (2018). Triplet loss in siamese network for object tracking. In *Proceedings of the European conference on computer vision (ECCV)*.

Dong, X., Shen, J., Wang, W., Liu, Y., Shao, L., & Porikli, F. (2018). Hyperparameter optimization for tracking with continuous deep q-learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*

Fan, H., Lin, L., Yang, F., Chu, P., Deng, G., Yu, S., Bai, H., Xu, Y., Liao, C., & Ling, H. (2019). Lasot: A high-quality benchmark for large-scale single object tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*

Galoogahi, H.K., Fagg, A., & Lucey, S. (2017). Learning background-aware correlation filters for visual tracking. In *Proceedings of the IEEE international conference on computer vision (ICCV)*.

He, A., Luo, C., Tian, X., & Zeng, W. (2018). A twofold siamese network for real-time object tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Henriques, J. F., Caseiro, R., Martins, P., & Batista, J. (2015). High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *37*(3), 583–596.

Huang, C., Lucey, S., & Ramanan, D. (2017). Learning policies for adaptive tracking with deep feature cascades. In *Proceedings of the IEEE international conference on computer vision (ICCV)*.

Huang, D., Luo, L., Chen, Z., Wen, M., & Zhang, C. (2017). Applying detection proposals to visual tracking for scale and aspect ratio adaptability. *International Journal of Computer Vision (IJCV)*, *122*(3), 524–541.

Jung, I., Son, J., Baek, M., & Han, B. (2018). Real-time mdnet. In *Proceedings of the European conference on computer vision (ECCV)*.

Kalal, Z., Mikolajczyk, K., & Matas, J. (2012). Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *34*(7), 1409–1422.

Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., & Cehovin Zajc, L., et al. (2018) The sixth visual object tracking vot2018 challenge results. In *Proceedings of the European conference on computer vision workshops (ECCV Workshop)*.

Kristan, M., Matas, J., Leonardis, A., Felsberg, M., Cehovin, L., Fernández, G., Vojir T., & Hager, et al. (2016). The visual object tracking vot2016 challenge results. In *Proceedings of the European conference on computer vision workshops (ECCV Workshop)*.

Kristan, M., Matas, J., Leonardis, A., Felsberg, M., Cehovin, L., Fernández, G., Vojir, T., & Hager, et al. (2017). The visual object tracking vot2017 challenge results. In *Proceedings of the IEEE international conference on computer vision workshops (ICCV Workshop)*.

Kristan, M., Matas, J., Leonardis, A., Vojíř, T., Pflugfelder, R., Fernandez, G., et al. (2016). A novel performance evaluation methodology for single-target trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *38*(11), 2137–2155.

Krizhevsky, A., Sutskever, I., & Hinton, G.E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems (NeurIPS)*.

Le, Q.V., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G.S., Dean, J. & Ng, A.Y. (2011). Building high-level features using large scale unsupervised learning. arXiv:1112.6209.

Lee, D.Y., Sim, J.Y., & Kim, C.S. (2015). Multihypothesis trajectory analysis for robust visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Lee, H.Y., Huang, J.B., Singh, M., & Yang, M.H. (2017). Unsupervised representation learning by sorting sequences. In *Proceedings of the IEEE international conference on computer vision (ICCV)*

Li, B., Yan, J., Wu, W., Zhu, Z. & Hu, X. (2018). High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Li, F., Tian, C., Zuo, W., Zhang, L., Yang, M.H. (2018). Learning spatial-temporal regularized correlation filters for visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Li, F., Yao, Y., Li, P., Zhang, D., Zuo, W., & Yang, M.H. (2017). Integrating boundary and center correlation filters for visual tracking with aspect ratio variation. In *Proceedings of the IEEE international conference on computer vision workshops (ICCV Workshop)*.

Liang, P., Blasch, E., & Ling, H. (2015). Encoding color information for visual tracking: algorithms and benchmark. *IEEE Transactions on Image Processing (TIP)*, *24*(12), 5630–5644.

Liu, S., Zhang, T., Cao, X., & Xu, C. (2016). Structural correlation filter for robust visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Lu, X., Ma, C., Ni, B., Yang, X., Reid, I., & Yang, M.H. (2018). Deep regression tracking with shrinkage loss. In *Proceedings of the European conference on computer vision (ECCV)*.

LukeźIăź, A., Vojíř, T., Čehovin Zajc, L., Matas, J., & Kristan, M. (2018). Discriminative correlation filter tracker with channel and spatial reliability. *International Journal of Computer Vision (IJCV)*, *126*(7), 671–688.

Lukezic, A., Vojir, T., Cehovin Zajc, L., Matas, J., & Kristan, M. (2017). Discriminative correlation filter with channel and spatial reliability. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Ma, C., Huang, J.B., Yang, X. & Yang, M.H. (2015). Hierarchical convolutional features for visual tracking. In *Proceedings of the IEEE international conference on computer vision (ICCV)*.

Ma, C., Huang, J. B., Yang, X., & Yang, M. H. (2018). Adaptive correlation filters with long-term and short-term memory for object tracking. *International Journal of Computer Vision (IJCV)*, *126*(8), 771–796.

Meister, S., Hur, J., & Roth, S. (2018). Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI conference on artificial intelligence (AAAI)*.

Mueller, M., Smith, N., & Ghanem, B. (2017). Context-aware correlation filter tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Müller, M., Bibi, A., Giancola, S., Al-Subaihi, S., & Ghanem, B. (2018). Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *Proceedings of the European conference on computer vision (ECCV)*.

Nam, H., & Han, B. (2016). Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Olshausen, B. A., & Field, D. J. (1997). Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research*, *37*(23), 3311–3325.

Real, E., Shlens, J., Mazzocchi, S., Pan, X., & Vanhoucke, V. (2017). Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *39*(6), 1137–1149.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, *115*(3), 211–252.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556.

Song, Y., Ma, C., Gong, L., Zhang, J., Lau, R., & Yang, M.H. (2017). Crest: Convolutional residual learning for visual tracking. In *Proceedings of the IEEE international conference on computer vision (ICCV)*.

Song, Y., Ma, C., Wu, X., Gong, L., Bao, L., Zuo, W., Shen, C., Lau, R.W., & Yang, M.H. (2018). Vital: Visual tracking via adversarial learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Sui, Y., Zhang, Z., Wang, G., Tang, Y., & Zhang, L. (2019). Exploiting the anisotropy of correlation filter learning for visual tracking. *International Journal of Computer Vision (IJCV)*, *127*, 1–22. Please confirm the inserted volume number is correct in Ref. Sui et al. (2019).

Tomasi, C., & Kanade, T. (1991). *Detection and tracking of point features*. Pittsburgh: Carnegie Mellon University.

Valmadre, J., Bertinetto, L., Henriques, J.F., Tao, R., Vedaldi, A., Smeulders, A., Torr, P., & Gavves, E. (2018). Long-term tracking in the wild: A benchmark. In *Proceedings of the European conference on computer vision (ECCV)*.

Valmadre, J., Bertinetto, L., Henriques, J.F., Vedaldi, A., & Torr, P.H. (2017). End-to-end representation learning for correlation filter based tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Vondrick, C., Pirsiavash, H., & Torralba, A. (2016). Anticipating visual representations from unlabeled video. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Vondrick, C., Shrivastava, A., Fathi, A., Guadarrama, S., & Murphy, K. (2018). Tracking emerges by colorizing videos. In *Proceedings of the European conference on computer vision (ECCV)*.

Wang, N., Song, Y., Ma, C., Zhou, W., Liu, W., & Li, H. (2019). Unsupervised deep tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Wang, N., & Yeung, D.Y. (2013). Learning a deep compact image representation for visual tracking. In *Advances in neural information processing systems (NeurIPS)*.

Wang, N., Zhou, W., Tian, Q., Hong, R., Wang, M., & Li, H. (2018). Multi-cue correlation filters for robust visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Wang, Q., Gao, J., Xing, J., Zhang, M., & Hu, W. (2017). Dcfnet: Discriminant correlation filters network for visual tracking. arXiv:1704.04057

Wang, Q., Teng, Z., Xing, J., Gao, J., Hu, W., & Maybank, S. (2018). Learning attentions: Residual attentional siamese network for high performance online visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Wang, X., & Gupta, A. (2015). Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE international conference on computer vision (ICCV)*.

Wang, X., Jabri, A., & Efros, A.A. (2019). Learning correspondence from the cycle-consistency of time. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Weijer, J. V. D., Schmid, C., Verbeek, J., & Larlus, D. (2009). Learning color names for real-world applications. *IEEE Transactions on Image Processing (TIP)*, *18*(7), 1512–1523.

Wu, Y., Lim, J., & Yang, M.H. (2013). Online object tracking: A benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Wu, Y., Lim, J., & Yang, M. H. (2015). Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *37*(9), 1834–1848.

Yang, T., & Chan, A.B. (2018). Learning dynamic memory networks for object tracking. In *Proceedings of the European conference on computer vision (ECCV)*

Yao, Y., Wu, X., Zhang, L., Shan, S., & Zuo, W. (2018). Joint representation and truncated inference learning for correlation filter based tracking. In *Proceedings of the European conference on computer vision (ECCV)*

Yin, Z., & Shi, J. (2018). Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Zhang, M., Wang, Q., Xing, J., Gao, J., Peng, P., Hu, W., & Maybank, S. (2018). Visual tracking via spatially aligned correlation filters network. In *Proceedings of the European conference on computer vision (ECCV)*.

Zhang, Y., Wang, L., Qi, J., Wang, D., Feng, M., & Lu, H. (2018). Structured siamese network for real-time visual tracking. In *Proceedings of the European conference on computer vision (ECCV)*.

Zhipeng, Z., Houwen, P., & Qiang, W. (2019). Deeper and wider siamese networks for real-time visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Zhou, T., Brown, M., Snavely, N., Lowe, D.G.: Unsupervised learning of depth and ego-motion from video. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (2017)

Zhou, T., Krahenbuhl, P., Aubry, M., Huang, Q., & Efros, A.A. (2016). Learning dense correspondence via 3d-guided cycle consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Zhou, X., Zhu, M., & Daniilidis, K. (2015). Multi-image matching via fast alternating minimization. In *Proceedings of the IEEE international conference on computer vision (ICCV)*.

Zhu, Z., Wang, Q., Li, B., Wu, W., Yan, J., & Hu, W. (2018). Distractor-aware siamese networks for visual object tracking. In *Proceedings of the European conference on computer vision (ECCV)*.