Hedging Deep Features for Visual Tracking

Yuankai Qi, Shengping Zhang, Lei Qin, Qingming Huang, Fellow, IEEE, Hongxun Yao, Jongwoo Lim, and Ming-Hsuan Yang, Senior Member, IEEE

Abstract—Convolutional Neural Networks (CNNs) have been applied to visual tracking with demonstrated success in recent years. Most CNN-based trackers utilize hierarchical features extracted from a certain layer to represent the target. However, features from a certain layer are not always effective for distinguishing the target object from the backgrounds especially in the presence of complicated interfering factors (*e.g.*, heavy occlusion, background clutter, illumination variation, and shape deformation). In this work, we propose a CNN-based tracking algorithm which hedges deep features from different CNN layers to better distinguish target objects and background clutters. Correlation filters are applied to feature maps of each CNN layer to construct a weak tracker, and all weak trackers are hedged into a strong one. For robust visual tracking, we propose a hedge method to adaptively determine weights of weak classifiers by considering both the difference between the historical as well as instantaneous performance, and the difference among all weak trackers over time. In addition, we design a Siamese network to define the loss of each weak tracker for the proposed hedge method. Extensive experiments on large benchmark datasets demonstrate the effectiveness of the proposed algorithm against the state-of-the-art tracking methods.

Index Terms—Visual tracking, convolutional neural network, adaptive hedge, Siamese network.

1 Introduction

Visual tracking is a fundamental problem in computer vision which has attracted increasing attention over the last decades [1], [2]. It is widely used in numerous applications such as surveillance [3], [4], human-computer interaction [5], autonomous driving [6], [7], and motion analysis [8], to name a few. Visual tracking aims to estimate the states (*e.g.*, location, scale, and motion) of a target object in a sequence of images after specifying the initial position and extent in the first frame. While significant efforts have been made in the past decades, developing a robust tracking algorithm for complicated scenarios is still a challenging task due to numerous factors such as heavy occlusion, pose changes, scale variations, shape deformations, camera motions, fast movements, and illumination conditions [9], [10].

Existing object tracking approaches mainly focus on either designing effective classification models [11]–[14] or extracting robust features [15]–[17]. Recently, numerous methods based on Convolutional Neural Networks (CNNs) have been proposed [18]–[21]. Empirical studies using a large object tracking benchmark [22] show that the CNN-based trackers perform well against methods using hand-crafted features such as SIFT [23], HOG [16], and color histogram [24].

Despite achieving state-of-the-art performance, existing CNN-based trackers typically represent target objects only using features from one layer, *e.g.*, the fully connected one, which are capable of capturing rich category-level semantic information but not always effective to accurately locate the target object due to low resolution feature maps. Figure 1 shows tracking results obtained by using CNN features extracted from different convolutional layers, and hedge features by the proposed algorithm. The red and green bounding boxes denote the tracking results and ground truth, respectively. The features from the last layer are not optimal for visual tracking as they do not capture spatial details of the target object. These details captured by the first few layers are crucial to visual tracking as they facilitate accurate localization of the target object [25]. On the other hand, as features from the first few layers are more generic rather than discriminative as those from latter layers, tracking methods based on features from the first few layers are likely to fail in challenging scenarios, as shown in the second and the third rows. Furthermore, in challenging scenarios such as heavy occlusions as shown in the last row of Figure 1, the target object (occluded by the person in the center) can be tracked by the proposed algorithm using a combination of features from different layers.

For visual tracking, it is of great interest to combine features from different layers to best represent and separate foreground objects from background clutter. In [25], features from different layers of a CNN are used for visual tracking. Nevertheless, features are combined with the same weights for all scenarios and thus it may not perform well for challenging scenarios as demonstrated in this work (see Section 4.5).

Corresponding author: Qingming Huang, Shengping Zhang.
Y. Qi is with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China (E-mail: yk.qi@hit.edu.cn).
S. Zhang is with the School of Computer Science and Technology, Harbin Institute of Technology, Weihai 264209, China (E-mail: s.zhang@hit.edu.cn).
L. Qin is with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology of Chinese Academy of Sciences, Beijing 100190, China (E-mail: qinlei@ict.ac.cn).
Q. Huang is with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China; with the School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 100049, China; and with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology of Chinese Academy of Sciences, Beijing 100190, China (E-mail: qmhuang@jdl.ac.cn).
H. Yao is with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China (E-mail: h.yao@hit.edu.cn).
J. Lim is with the Department of Computer Science, Hanyang University, Seoul, Seoul 133791, Republic of Korea (E-mail: jlim@hanyang.ac.kr).
M.-H. Yang is with the School of Engineering, University of California, Merced, CA 95344, USA (E-mail: mhyang@ucmerced.edu).

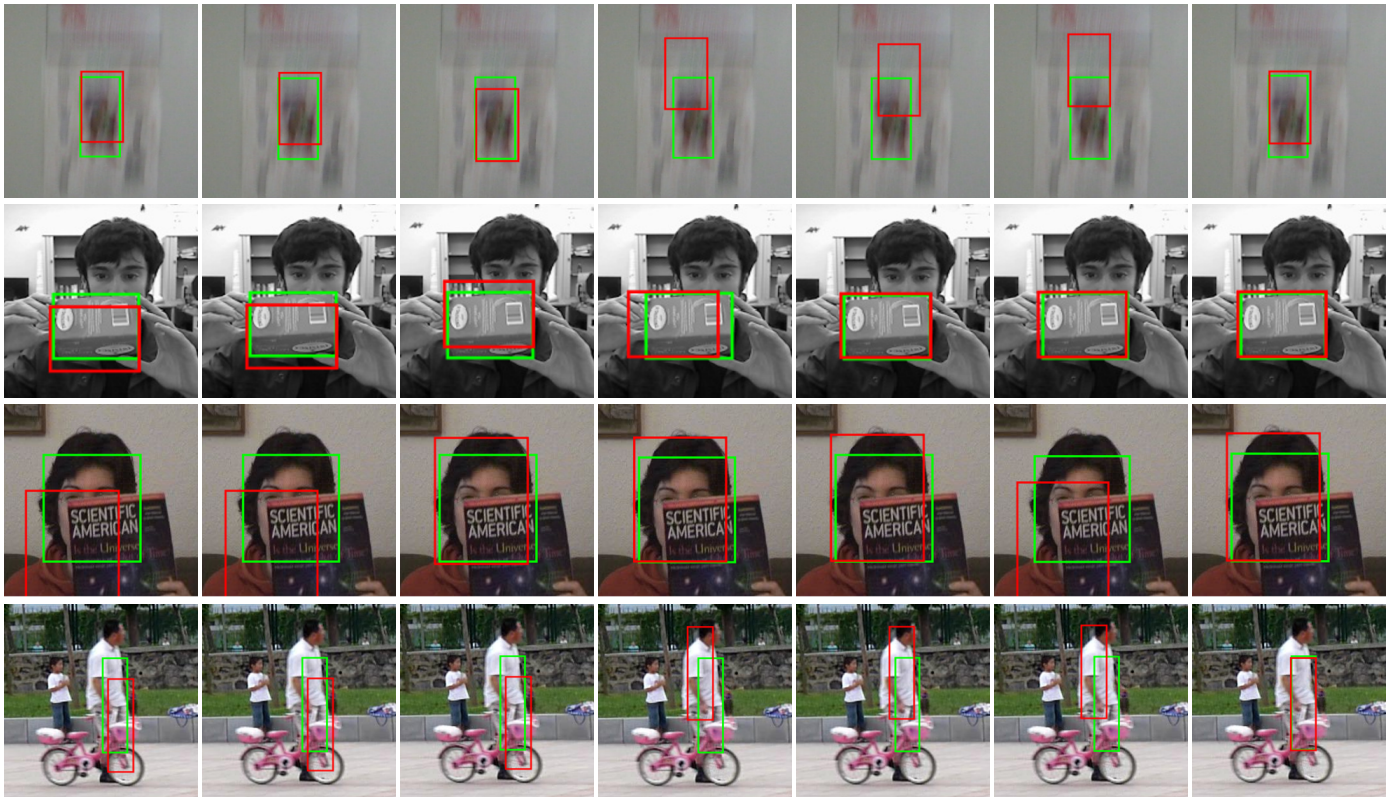


Fig. 1. Tracking results obtained by separately using CNN features extracted from six different convolutional layers of the VGGNet (with 19 layers) and by combining all features via the proposed adaptive hedge algorithm on representative frames of four sequences with different challenging factors. Red and green boxes denote the tracking results and ground truth, respectively. The tracking results by the proposed algorithm are more accurate than the ones obtained by using features only from one single layer.

In this paper, we propose a CNN-based tracking algorithm which first constructs weak trackers by applying correlation filters to the features from different layers, and then combines all the weak trackers into a stronger one using an online decision-theoretical hedge algorithm. The hedge algorithm [26] is developed for online decision-theoretic learning problems in a multi-expert multi-round setting, which defines the regret of an expert as the difference between the loss of this expert and the weighted average loss of all experts. In each round, it uses a regret to measure the performance of an expert and generates the corresponding weight of the expert based on its cumulative regret (accumulated from the first round to the current round). The final decision of the current round is the weighted predictions from all experts. While it performs well in numerous tasks, the existing hedge method is less effective for visual tracking where multiple weak trackers are used because it does not consider two crucial factors when computing the cumulative regret of weak classifiers. First, a target object usually undergoes large appearance changes throughout a video, which means that the historical regret should be taken into consideration with a varying proportion over time (rather than a static regret model for all experts). Second, since each weak tracker exploits CNN features extracted from different layers and thus represents a target object in different aspects, it is not effective to utilize the same proportion of the historical regret for all component trackers.

To address these issues, we propose an adaptive cumulative regret model for visual tracking. The proposed model determines the weight for each weak tracker by considering both the difference between the historical as well as instantaneous regrets, and the difference among all component trackers over time. In addition, a reliable loss measurement of each weak tracker plays an important role in the hedge algorithm. In this work, we design a Siamese network to measure the appearance similarity between a target template and each tracking result.

We make the following contributions in the proposed hedge deep tracker (HDT*[1]) in this work:

- In contrast to existing CNN-based tracking methods which use features from either one or multiple layers with fixed weights, we propose an online tracking algorithm which adaptively combines weak CNN-based trackers from various convolutional layers.
- We develop a hedge algorithm for visual tracking by adaptively determining the proportion of instantaneous regret of each weak tracker over time.
- We design a Siamese network to measure the appearance similarity between a target template and each tracking result, which provides reliable input for the hedge algorithm.
- We carry out extensive experiments on large benchmark datasets to demonstrate the effectiveness of the

1. We use HDT to refer to the preliminary work [27].

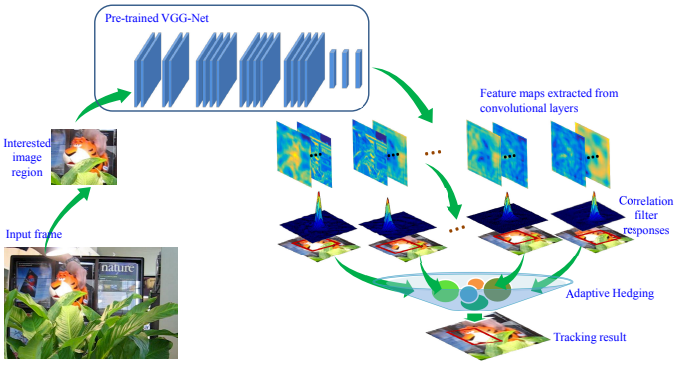


Fig. 2. Main steps of the proposed HDT*. When a new frame arrives, we first extract CNN features of the region of interest from different convolutional layers with the pre-trained VGGNet (19 layers), and then each weak tracker computes correlation filter responses using features from one specific layer (Section 3.2). Finally, these responses are combined together by the proposed adaptive hedge algorithm and produce the ultimate target location (Section 3.3 and 3.4).

proposed algorithm with comparisons to the state-of-the-art tracking methods.

Compared to the preliminary results [27], we make several extensions in this work. First, we design a loss function using a Siamese network to measure the appearance similarity between a target template and each tracking result, and consider the Euclidean distance between the target positions obtained by the hedged tracker and each weak one. Second, we propose a cumulative regret model, which adaptively determines the proportion of historical regrets by considering both the strength and trend of performance change over time. Third, we develop a one-degree potential function instead of the quadratic one, which leads to a smoother weight distribution over weak trackers. Fourth, we add a scale search step to handle size variations in the proposed tracking algorithm. Experimental results show that all these extensions contribute to the performance gain of over the HDT method in the preliminary work.

2 Related Work

In this section, we discuss the tracking methods closely related to this work in proper context. Comprehensive reviews on visual tracking approaches can be found in [1], [2].

Correlation filter based trackers. Correlation filters are introduced to visual tracking due to their computational efficiency in training and testing, with a ridge regression model in the case of large amounts of samples [13], [28], [29]. These methods approximate the dense sampling scheme by generating a circulant matrix, of which each row denotes a vectorized sample. With this representation, the regression model can be solved in the Fourier domain efficiently. Bolme *et al*. [28] develop the minimum output sum of squared error method to learn robust filters where intensity features are used for object representation. In [13], Henriques *et al*. propose a tracking algorithm based on correlation filters by introducing kernel methods and using ridge regression. Subsequently a method that extends the input features from a single channel to multiple channels (*e.g.*, HOG) is presented [29]. Danelljan *et al*. [30] present an approach that searches over the scale space for correlation filters to handle large variation in object size. In [31], Danelljan *et al*. improve the standard correlation filters by introducing the spatial regularization to mitigate the unwanted boundary effects. All the above-mentioned methods use only one correlation filter for visual tracking. In contrast, Ma *et al*. [25] utilize features from three convolutional layers to exploit both semantic information and spatial details for visual tracking. However, the weights of these correlation filters are fixed during tracking. In this work, we exploit the computational efficiency of correlation filters to construct component trackers using features from several convolutional layers. To integrate the component trackers into a stronger one, we propose a hedge algorithm to adaptively determine the decision weights for each component tracker.

CNN-based trackers. Hierarchical features learned from CNNs have been shown to be effective for numerous vision tasks, *e.g.*, image classification and object recognition [32]–[35] in recent years. Numerous methods have since been proposed to exploit CNN features [18]–[21], [36], [37] for visual tracking. Fan *et al*. [19] utilize a pre-trained deep network for human tracking whereas Wang and Yeung [18] design an autoencoder network to learn representative features for generic objects. In [36], Hong *et al*. construct a discriminative model with features from the first fully-connected layer of R-CNN [38] and a generative model with saliency maps for visual tracking. In [37], Nam and Han train a multi-domain network (MDNet) using a large set of videos, where each domain corresponds to an individual sequence. While these methods are effective for visual tracking, features from different layers are not analyzed or combined. In contrast, Wang *et al*. [21] design a CNN containing two subnetworks to exploit features from two different convolutional layers. One subnetwork is for capturing general target information and the other for capturing specific target information. Nevertheless, the computational load is high due to the complicated architecture. In this paper, we exploit the computational efficiency of correlation filters and representation strength of CNN features to construct an ensemble tracker. We regard each component tracker that uses CNN features from one layer as a weak expert and integrate them adaptively via the proposed hedge algorithm for visual tracking.

Ensemble trackers. Numerous approaches have been developed to combine multiple component trackers for visual tracking based on hand-craft features [11], [39]–[43]. Within the boosting framework [44], ensemble methods [11], [39], [41] incrementally train each component tracker to classify training samples that are not correctly classified in the previous iteration. Recently, Wang and Yeung [40] develop an ensemble tracking method based on a factorial Hidden Markov Model (HMM) where a conditional particle filter is used to infer the reliability of each component. In [43], Tomas *et al*. also utilize HMM to fuse multiple trackers, where the confidence of each tracker is estimated using a beta distribution with learned parameters. Bailer *et al*. [42] employ the trajectory optimization and distance minimization over both the positions and sizes of bounding boxes predicted by component trackers to filter out the final tracking result from the outputs of multiple trackers. Different

from prior works, we consider visual tracking as a decision-theoretic online learning task [26] to infer the tracked target using decisions from multiple expert trackers based on CNN features. In each round, an expert makes a decision and the final decision is determined by the weighted decisions from all components. We show in this work that the proposed adaptive hedge algorithm facilitates robust visual tracking.

3 Proposed Algorithm

3.1 Overview

We describe the main steps of the proposed approach in Figure 2. When a new frame arrives, the pre-trained VGGNet with 19 layers [33] is used to extract feature maps of convolutional layers from the region containing the target object, which encode visual information at different resolutions and semantic levels. Next, each feature map is convolved with a correlation filter to generate the response map, from which a weak tracker is constructed. All weak trackers are then combined into a stronger one using the proposed adaptive hedge algorithm, which exploits the representation strength of multiple CNN layers.

The proposed adaptive hedge algorithm determines decision weights for weak trackers based on the performance loss. The loss is computed based on appearance similarity and disagreement between weak trackers. We present a Siamese network to measure the appearance similarity between a target template and tracked results by weak trackers. In addition, we consider the disagreement in terms of central location between weak experts and the hedge tracker. To generate proper weights for weak trackers, it is crucial to vary the proportion of instantaneous regret of each tracker over time. In this work, we propose an adaptive cumulative regret model to adjust the relative proportion between historical and instantaneous regrets of each weak tracker.

3.2 CNN-Based Weak Trackers

In this work, a weak tracker is constructed based on a correlation filter and CNN features from one layer. Taking both representation strength and run-time performance into account, the VGGNet (19 weight layers) [33] is used to extract features as opposed to other CNNs such as AlexNet [32], CaffeNet [45], and GoogleNet [46]. Correlation filter based trackers [13], [28]–[31] exploit the circulant structure of training and test samples for significant speed-up with negligible loss of accuracy. Let $X^k \in \mathbb{R}^{P\times Q\times D}$ denote the extracted feature map of $P \times Q$ pixels and D channels from the k-th convolutional layer, and $Y \in \mathbb{R}^{P\times Q}$ represent the 2D Gaussian shape label matrix defined by

$$Y_{i,j} = \exp\left(-\frac{i^2+j^2}{2\sigma^2}\right), \tag{1}$$

where $\sigma = 0.025\sqrt{P\times Q}$ is proportional to the feature map size. Let $\mathcal{X}^k = \mathcal{F}(X^k)$ and $\mathcal{Y} = \mathcal{F}(Y)$, where $\mathcal{F}(\cdot)$ denotes the Discrete Fourier Transformation (DFT). The k-th filter can be modeled in the Fourier domain by

$$\mathcal{W}^k = \underset{\mathcal{W}}{\operatorname{argmin}} \|\mathcal{Y} - \mathcal{X}^k \cdot \mathcal{W}\|_F^2 + \lambda\|\mathcal{W}\|_F^2, \tag{2}$$

where

$$\mathcal{X}^k \cdot \mathcal{W} = \sum\nolimits_{d=1}^{D} \mathcal{X}^k_{*,*,d} \odot \mathcal{W}_{*,*,d}, \tag{3}$$

and the symbol $\odot$ denotes the element-wise product.

The optimization problem in (2) has a closed form solution, which can be efficiently computed in the Fourier domain by

$$\mathcal{W}^k_{*,*,d} = \frac{\mathcal{Y}}{\mathcal{X}^k \cdot \mathcal{X}^k + \lambda} \odot \mathcal{X}^k_{*,*,d}. \tag{4}$$

Given the test data T^k from the output of the k-th layer, we first transform it to the Fourier domain $\mathcal{T}^k = \mathcal{F}(T^k)$, and then compute the response by

$$S^k = \mathcal{F}^{-1}(\mathcal{T}^k \cdot \mathcal{W}^k), \tag{5}$$

where $\mathcal{F}^{-1}$ denotes the inverse of DFT.

The k-th weak tracker predicts the target position with the largest response

$$(x^k, y^k) = \underset{x',y'}{\operatorname{argmax}}\, S^k(x', y'). \tag{6}$$

3.3 Loss of Each Component Tracker

The hedge algorithm [26] is proposed for decision-theoretic online learning problems in a multi-expert multi-round setting. Given the initial confidence weights of all experts in the current round, the final decision is made based on the weighted prediction of all experts. Based on the loss of each expert, the corresponding weight is updated accordingly.

For visual tracking, it is natural to treat each CNN-based tracker as an expert and predict the target position in the t-th frame by

$$(x^*_t, y^*_t) = \sum\nolimits_{k=1}^{K} w^k_t \cdot (x^k_t, y^k_t), \tag{7}$$

where w^k_t is the weight of k-th expert at time t such that $\sum_{k=1}^{K} w^k_t = 1$. Once the target position is predicted, the loss of each expert is computed and used by the adaptive hedge algorithm described in the next section to update the weights of all experts.

To exploit both appearance and spatial information, we use two metrics to compute the loss of each tracker. First, we use the appearance difference $\mathcal{A}(k,t)$ between a target template and the tracked target. Second, we use the center location difference $\mathcal{D}(k,t)$ between the estimated position (x^k_t, y^k_t) of an expert and the target position (x^*_t, y^*_t) via the proposed hedge algorithm.

The loss of the k-th component tracker at frame t is defined as

$$\ell^k_t = (1-\beta)\mathcal{A}(k,t), +\beta\mathcal{D}(k,t), \tag{8}$$

$$\mathcal{A}(k,t) = 1 - \mathcal{S}(T, R^k_t), \tag{9}$$

$$\mathcal{D}(k,t) = \frac{1}{\Gamma}\sqrt{(x^k_t - x^*_t)^2 + (y^k_t - y^*_t)^2}, \tag{10}$$

where $\mathcal{S}(\cdot,\cdot)$ measures the similarity between two images using the proposed similarity Siamese network (SSN). In addition, T is the target template specified in the first frame, R^k_t is the tracked target region in the t-th frame obtained by the k-th expert, and $\Gamma = \sum_{k=1}^{K} \mathcal{D}(k,t)$ is the normalization factor.

The architecture of the similarity Siamese network is shown in Figure 3, where local response normalization

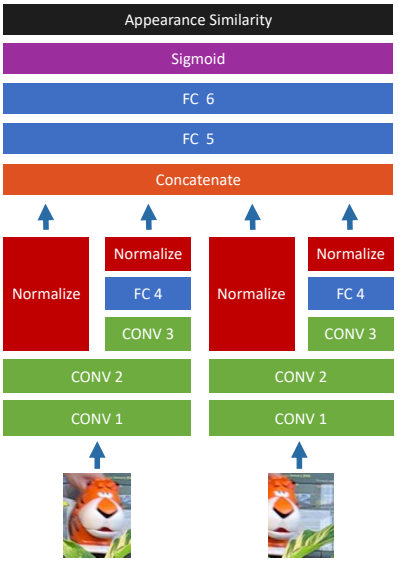


Fig. 3. The proposed similarity Siamese network takes two image and computes appearance similarity. The convolutional layers and the FC4 layer of these two streams share the same weights, respectively.

TABLE 1
Architecture of proposed similarity Siamese network.

	KernelSize	Stride	OutputNum.
CONV1	7x7	2	96
ReLU	–	-	–
LRN	LocalSize: 5, Alpha: 0.0005, Beta: 0.75		
Pooling	3x3	2	MAX
CONV2	5x5	2	256
ReLU	–	-	–
LRN	LocalSize: 5, Alpha: 0.0005, Beta: 0.75		
Pooling	3x3	2	MAX
CONV3	3x3	1	512
ReLU	–	-	–
LRN	LocalSize: 5, Alpha: 0.0005, Beta: 0.75		
FC4	3x3	1	512
FC5	1x1	1	512
FC6	1x1	1	1
Sigmoid	–	-	–
LOSS	EuclideanLoss		

(LRN), max pooling (MAXPooling), and rectifier linear unit (ReLu) layers are omitted for clarity. To reduce the computational load, we use a shallow network with three convolutional and three fully-connected layers. To obtain better representative features with such a shallow architecture, we concatenate features from two streams at two different levels: features from CONV2 and features from FC4. This design enables the input for the FC5 layer to encode both low-level texture and high-level semantic visual information. Similar to [37], we also use convolving operations to implement the fully-connected layers for robust performance. The network configuration is shown in Table 1.

3.4 Online Adaptive Hedge Algorithm

The hedge algorithm [26] generates a weight distribution $(w_t^1, \cdots, w_t^K)$ over all experts $1, 2, \cdots, K$ in round t. Each expert k incurs a loss ℓ_t^k, and the hedge algorithm incurs the expected loss $\bar{\ell}_t = \sum_{k=1}^{K} w_t^k \ell_t^k$. The instantaneous regret to an expert k is

$$r_t^k = \bar{\ell}_t - \ell_t^k. \tag{11}$$

The goal of the hedge algorithm is to minimize the cumulative regret

$$R_t^k = R_{t-1}^k + r_t^k, \tag{12}$$

to any expert k, for any round of t. Note that the historical and instantaneous regrets, R_{t-1}^k and r_t^k, contribute equally in the loss function as shown in (12).

Although the hedge algorithm [26] performs well on several problems, it is less effective for real-world tracking tasks since it does not consider two crucial factors. First, the appearance of a target object is likely to change significantly throughout a video, which means the historical regret should be taken into consideration with a varying proportion over time. Second, since each weak tracker exploits CNN features extracted from multiple layers and represents a target object in different aspects, it is not effective to utilize the same proportion of the historical regret for all component trackers.

To address these issues, we propose an adaptive cumulative regret model which considers both the difference between the historical as well as instantaneous regrets, and the difference among all component trackers over time. As the object appearance usually does not change significantly in a short time period, we model the loss of each expert ℓ^k during the recent time window Δt via a Gaussian distribution with mean μ_t^k and standard variance σ_t^k,

$$\mu_t^k = \frac{1}{\Delta t} \sum_{\tau=t-\Delta t+1}^{t} \ell_\tau^k, \tag{13}$$

$$\sigma_t^k = \sqrt{\frac{1}{\Delta t - 1} \sum_{\tau=t-\Delta t+1}^{t} (\ell_\tau^k - \mu_t^k)^2}. \tag{14}$$

The performance of the k-th expert at time t is measured by

$$s_t^k = \frac{\ell_t^k - \mu_t^k}{\sigma_t^k}. \tag{15}$$

A large positive s_t^k indicates that this expert tends to perform worse in the short duration. Therefore, we should compute its cumulative regret mainly based on its historical regret. In contrast, a small negative s_t^k means that this expert tends to perform well. Therefore, we should put a large weight on its instantaneous regret. Based on these observations, in this work we compute the adaptive cumulative regret for each expert as

$$R_t^k = R_{t-1}^k + \bar{\ell}_t - \alpha_t^k \ell_t^k, \tag{16}$$

$$\alpha_t^k = \tanh(\gamma s_t^k), \tag{17}$$

where γ is a scale factor that controls the shape of the hyperbolic tangent function (17) as shown in Figure 4. We demonstrate the effectiveness of the proposed adaptive cumulative regret model compared to the existing one (12) in Section 4.3.2.

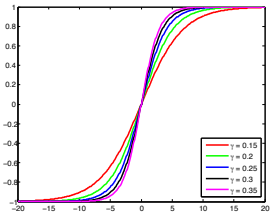


Fig. 4. Shape of the hyperbolic tangent function (17) is controlled by parameter γ. A smaller value of γ makes the function smoother. On the other hand, it becomes a step function when a large value of γ is used.

In [26], the potential function $\phi(R_t^k, c_t)$ is computed based on the quadratic regret,

$$\phi(R_t^k, c_t) = \exp\left(\frac{([R_t^k]_+)^2}{2c_t}\right), \tag{18}$$

where $[R_t^k]_+$ denotes $\max\{0, R_t^k\}$ and c_t is a scale parameter computed by solving $\frac{1}{K}\sum_{k=1}^{K}\exp\left(\frac{([R_t^k]_+)^2}{2c_t}\right) = e$. The new weights can be set proportional to the first-derivative of the potential function,

$$w_{t+1}^k \propto \frac{[R_t^k]_+}{c_t}\exp\frac{([R_t^k]_+)^2}{2c_t}. \tag{19}$$

As mentioned in [26], there are only a small portion of all experts which have important weights. When applying the potential function (18) to visual tracking, where only a total of six experts are used, we empirically observe that only one or two experts play important roles in the decision process. To let more experts play important roles during the decision process, we improve the potential function based on the one-degree regret,

$$\phi(R_t^k, c_t) = \exp\frac{[R_t^k]_+}{c_t}. \tag{20}$$

The new weights can be set proportional to its first-derivative,

$$w_{t+1}^k \propto \begin{cases} \frac{1}{c_t}\exp\frac{R_t^k}{c_t}, & R_t^k > 0, \\ 0, & R_t^k \leqslant 0, \end{cases} \tag{21}$$

where $\frac{1}{K}\sum_{k=1}^{K}\exp\left(\frac{[R_t^k]_+}{c_t}\right) = e$. The new weights generated by (21) are smoother than that generated by (19) as demonstrated in Section 4.3.3.

3.5 Model Update

Since the feature maps of the VGGNet have up to 512 channels, retraining the ridge regression models with the newly collected samples is impractical, especially when the amount of the training data becomes large over time. In practice, we use an incremental update strategy similar to that in [30] which only uses new samples $\bar{\mathcal{X}}^k$ collected in the current frame to partially update the previous models

$$\mathcal{Z}_{*,*,d}^k = \frac{\mathcal{Y}}{\bar{\mathcal{X}}^k \cdot \bar{\mathcal{X}}^k + \lambda} \odot \bar{\mathcal{X}}_{*,*,d}^k, \tag{22}$$

$$\mathcal{W}_t^k = (1-\eta)\mathcal{W}_{t-1}^k + \eta\mathcal{Z}_t^k. \tag{23}$$

For presentation clarity, we summarize the main steps of the proposed hedge deep tracking method in Algorithm 1.

Algorithm 1: Hedged deep tracking

Input: initial weights $w_1^1, \cdots, w_1^K$; target position (x_1, y_1) in the 1st frame; VGGNet-19; $R_1^k = 0, \ell_1^k = 0$;
Crop interested image region;
Initiate K weak experts using (4);
for $t = 2, 3, \cdots$ **do**
  Exploit the VGGNet-19 to obtain K representations;
  Compute correlation filter responses using (5);
  Find target position predicted by each expert using (6);
  if $t \neq 2$ **then**
    Compute ultimate position using (7);
  else
    Set ultimate position as that in the 1st frame (this approximation operation has slight influence on the performance as the motion between the first two frames is generally negligible.);
  end
  Compute loss of each expert using (8);
  Update stability models using (13) and (14);
  Measure the tendency of each expert using (15);
  Compute adaptive proportion of historical regret for each expert using (17);
  Update cumulative regret of each expert using (16);
  Update decision weights for each expert using (21) and normalize them to have a sum of 1;
end

4 Experimental Results

In this section, we present extensive experimental evaluations of the proposed HDT* algorithm on the visual tracking OTB100 [9] and VOT2016 [47] benchmark datasets. We first describe the implementation details and the evaluation protocols. Then, we demonstrate the effectiveness of each component of the proposed hedge deep tracker with an ablation study. Next, we present the sensitivity analysis of two key parameters. Finally, we report experimental evaluations of the proposed algorithm against the state-of-the-art tracking methods. Preliminary results and source code of this work are presented in [27]. The source code and trained models of the HDT* will be made available to the public. Supplementary results including videos and figures can be found at https://github.com/YuankaiQi/Hedging-features-for-visual-tracking.

4.1 Implementation Details

Feature extraction for weak trackers. We use the VGGNet [33] with 19 layers (16 convolutional layers and 3 fully-connected layers) to extract features. The feature maps from six convolutional layers (10th-12th and 14th-16th) are used to represent objects. Given an image frame with a search window of $Z_h \times Z_w$ pixels (*e.g.*, 2 times of the target object size), we resize all the feature maps to a fixed spatial size of $\frac{Z_h}{4} \times \frac{Z_w}{4}$ pixels. These settings are set based on the consideration of feature diversities and computational load for visual tracking.

Training set for SSN. Similar to the settings of the MDNet [37], the proposed SSN is trained using 58 image sequences from the VOT2013 [48], VOT2014 [49], and VOT2015 [22] databases excluding the same ones in the OTB100 [9] dataset. In each frame, 128 image regions are

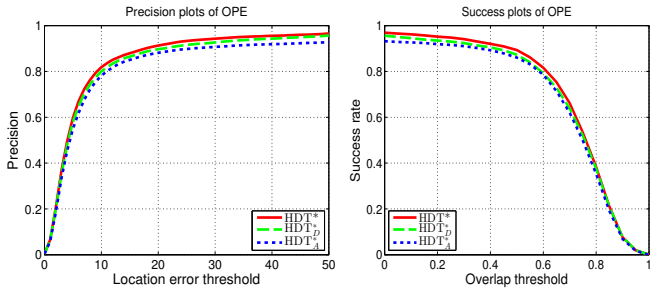


Fig. 5. Precision and success plots using OPE for the ablation analysis on the SSN appearance loss (denoted by A) and the Euclidean distance loss (denoted by D). In this plot, HDT$^*_D$ denotes HDT* without using the Euclidean loss, and likewise HDT$^*_A$ denotes HDT* without using the appearance loss.

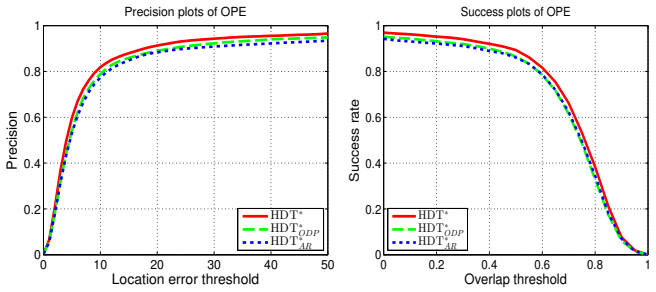


Fig. 6. Precision and success plots using OPE for the ablation analysis on the adaptive cumulative regret (denoted by AR) and one-degree potential function (denoted by ODP). In this plot, HDT$^*_{AR}$ is obtained by replacing AR with the original one (12), and HDT$^*_{ODP}$ is obtained by replacing ODP with the original one (18).

sampled with the same size as the ground truth while overlapping at least 60% with the ground truth. Each sampled image region and ground truth form one training pair. The Euclidean loss $\ell(y, v) = \|y - v\|^2$ is adopted, where v is the predicted real-valued score of the input training pair and y is the intersection over union of this pair of regions. As such, we obtain a total of 2,531,840 training pairs on the whole image sequences for regression. We randomly select 20% of all training pairs as the validation set and the rest as the training set.

Optimization of SSN. We implement the proposed SSN using the Caffe toolbox [50] and the stochastic gradient descent scheme with momentum. The weights of convolutional layers are initialized as in MDNet [37], which is trained using the same image sequences as HDT*. The weights of the fully-connected layers are randomly initialized using the Xavier algorithm [51]. The network converges after approximately 900K iterations with the initial learning rate 0.001. The learning rate is set to decrease in an order of magnitude every 80K iterations until reaching 10^{-8} and is then unchanged. The weight decay factor and momentum are set to 0.0005 and 0.9, respectively.

Scale search. Correlation filter responses are more sensitive to scale variation than spatial translation [31]. For robust visual tracking, we first search for the optimal scale and then locate the target. To reduce the computational load, the searching operation is performed on the scale set $\mathcal{E} = \{0.97, 0.98, 0.99, 1.00, 1.01, 1.02, 1.03\}$ using features extracted only from the 2nd and the 12th layers of the VGGNet. In practice we observe that HDT* performs well with this setting as a tradeoff between scale estimation precision and time cost. The optimal scale is obtained by $SC_t = \text{argmax}_{i \in \mathcal{E}} S^{2nd}_{i,t} + S^{12th}_{i,t}$, where $S^{2nd}_{i,t}$ and $S^{12th}_{i,t}$ denote the correlation filter responses computed with features from the 2nd and the 12th convolutional layers at frame t, respectively.

HDT* parameters. The tradeoff parameter λ in (2) is set to 10^{-4}; the time window Δt in (13) is set to 5; the scale factor γ in (17) is set to 0.25; the learning rate η in (23) is set to 0.01; and the initial weights of six component trackers are equally set to $1/6$. We demonstrate that the proposed HDT* is robust to the initial weights in Section 4.4.

All the experiments are carried out with the same parameters discussed above. Implemented in MATLAB, the HDT* runs at an average of 1.4 frames per second using the OTB100 dataset on a machine with an Intel I7-4790K CPU, 16GB RAM, and a GeForce GTX780Ti GPU. The state-of-the-art MDNet method runs at 1.1 frames per second on the same machine.

4.2 Evaluation Protocols

The tracking methods are evaluated by the success and precision plots in one-pass evaluation (OPE) [9]. In addition, we also use the spatial robustness evaluation (SRE), and temporal robustness evaluation (TRE) protocols on the OTB100 datasets to better assess tracking performance. We note existing methods are often evaluated only on the OTB50 or OTB100 datasets with the OPE protocol rather than using SRE or TRE metrics as these experiments require extensive computational loads. The success rate of a tracker is the proportion of the successful frames with an overlap rate larger than a given threshold. The trackers in success plots are ranked based on the area under the curve (AUC). The precision is an average of Euclidean distance in pixels between the center points of the tracked and the ground truth boxes. The trackers in precision plots are ranked based on the center location error at a threshold of 20 pixels. For the VOT2016 dataset, the expected average overlap (EAO) metric is used for performance evaluation.

4.3 Ablation Analysis

The adaptive hedge algorithm proposed in this paper is composed of three important parts including expert loss (8), adaptive cumulative regret (16), and one-degree potential function (20). To analyze each component, we conduct an ablation study on the OTB50 dataset [10]. When the adaptive hedge algorithm is applied to visual tracking, the effectiveness of each component tracker is also evaluated.

4.3.1 *Expert Loss*

The expert loss (8) consists of SSN appearance loss (9) and Euclidean distance loss (10). To evaluate the effectiveness of these two losses, we compare the proposed method with its two variants: HDT$^*_A$ and HDT$^*_D$, which are obtained by removing SSN appearance loss and Euclidean distance loss from the proposed method, respectively. The comparison

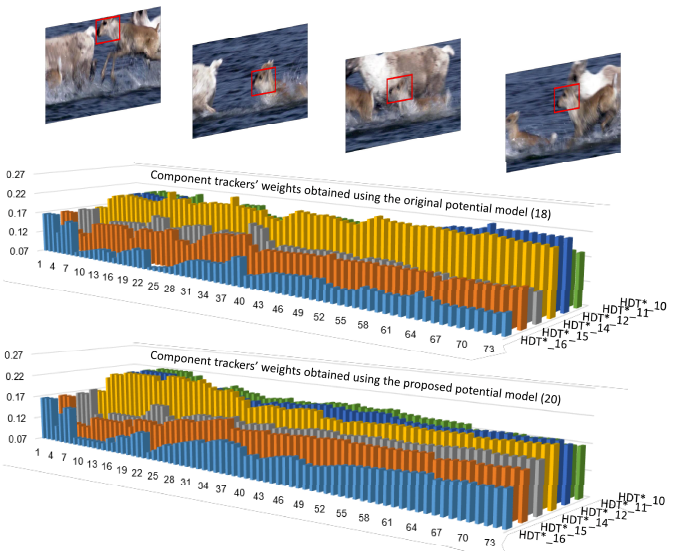


Fig. 7. Weight distributions on component trackers generated by the original potential function (18) and the proposed one (20), respectively, on the *Deer* sequence. The top images show that the target is challenged by scale variation, occlusion, and deformation. The middle figure shows the weights obtained using the original potential function (18). The bottom figure shows the weights obtained using the proposed potential function (20). The x-axis denotes frame indices, y-axis the names of component trackers, and z-axis the values of weights. Here, HDT*_16 denotes the component tracker using features extracted only from the 16th convolutional layer, and likewise for the others.

TABLE 2
AUC score and precision at a threshold of 20 pixels for the ablation analysis on the SSN appearance (denoted by A) and Euclidean distance (denoted by D) losses.

	HDT*	HDT*$_A$	HDT*$_D$
AUC	0.698	0.670	0.684
Precision	0.913	0.881	0.897

results are shown in Figure 5 and Table 2. As shown in Table 2, the success rate of the proposed HDT* is decreased by 3% without using the SSN appearance loss, and by 1% without using the Euclidean distance loss. These results demonstrate that both two losses facilitate the HDT* to perform better, and the appearance loss plays a more important role. Figure 5 shows comprehensive comparisons at all thresholds.

4.3.2 Adaptive Cumulative Regret

We evaluate the proposed tracking method against its variant HDT*$_{AR}$, which uses the original cumulative regret (12) to replace our adaptive cumulative regret (16). As shown in Figure 6 and Table 3, the proposed adaptive cumulative regret contributes to more accurate tracking results. Specifically, HDT* outperforms HDT*$_{AR}$ by about 4% in terms of both the AUC and precision metrics. These results demonstrate the effectiveness of the proposed adaptive cumulative regret.

4.3.3 One-degree Potential Function

We evaluate the proposed tracking method against its variant HDT*$_{ODP}$, which uses the original quadratic potential function (18) (instead of the proposed one-degree potential function (20)). Table 3 shows that HDT* achieves about a 3% improvements over HDT*$_{ODP}$ in terms of both precision and AUC metrics. This improvement demonstrates that the proposed one-degree potential function generates more effective weights for all component trackers than the quadratic one. We also present an intuitive weight distributions generated by these two potential functions in Figure 7 where the proposed potential helps achieve better tracking performance (0.57 vs. 0.52 in terms of average overlap rate, and 38.4 vs. 56.2 in terms of average center location error in pixels). This can be attributed to the smooth weight distributions, which enable HDT* to better exploit features from different CNN layers.

TABLE 3
AUC score and precision at a threshold of 20 pixels for the ablation analysis on the adaptive cumulative regret (AR) and one-degree potential (ODP) models.

	HDT*	HDT*$_{AR}$	HDT*$_{ODP}$
AUC	0.698	0.663	0.675
Precision	0.913	0.875	0.886

TABLE 4
Sensitivity analysis of γ in terms of AUC score and precision at a threshold of 20 pixels on the OTB100 and VOT2014 datasets.

	γ	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	Mean	StdDev
OTB100	AUC	0.671	0.668	0.669	0.675	0.687	0.674	0.667	0.674	0.667	0.6724	0.0063
	Precision	0.898	0.893	0.894	0.906	0.912	0.898	0.891	0.899	0.886	0.8974	0.0078
VOT2014	AUC	0.517	0.518	0.519	0.519	0.522	0.523	0.520	0.520	0.514	0.5191	0.0027
	Precision	0.682	0.691	0.694	0.683	0.690	0.695	0.693	0.688	0.676	0.6880	0.0064

4.3.4 Component Trackers

To demonstrate the effectiveness of the proposed tracking method based on hedging features, we compare HDT* against its component trackers which use features from different CNN layers. The component trackers are denoted by HDT*$_{10}$, HDT*$_{11}$, HDT*$_{12}$, HDT*$_{14}$, HDT*$_{15}$, and HDT*$_{16}$, where the number denotes the features from which convolutional layer are used.

Figure 8 shows the experimental results using the OTB50 dataset. The proposed HDT* outperforms any component tracker by 4% in terms of the success rate metric and 2% by the precision metric. These results demonstrate the effectiveness of tracking method which computes weights for weak trackers based on an adaptive cumulative regret model, a SSN appearance loss model, and a one-degree potential function.

4.4 Sensitivity Analysis

Scale factor γ for adaptive cumulative regrets. Table 4 shows the tracking results using different values of γ of (17) ranging from 0.05 to 0.45 on the OTB100 and VOT2014 datasets, respectively. We exclude values larger than 0.45 as otherwise the hyperbolic tangent function is close to a step function, which does not have smooth non-linear properties. Table 4 shows that as γ changes from 0.05 to 0.45, both

TABLE 5
Sensitivity analysis of the proposed HDT* to initial weights (0.1667, 0.1667, 0.1667, 0.1667, 0.1667, 0.1667) of six component trackers (HDT^*_{10}, HDT^*_{11}, HDT^*_{12}, HDT^*_{14}, HDT^*_{15}, and HDT^*_{16}) on the OTB100 and VOT2014 datasets in terms of AUC. The results in each row are obtained by disturbing the initial weight of the corresponding component tracker while unchanging the initial weights of all the remaining component trackers.

	Initial Weight	0.0667	0.0867	0.1067	0.1267	0.1467	0.1667	0.1867	0.2067	0.2267	0.2467	0.2667	Mean	Dev.
OTB100	HDT^*_{10}	0.675	0.675	0.678	0.675	0.681	0.687	0.683	0.677	0.678	0.672	0.675	0.6778	0.0041
	HDT^*_{11}	0.675	0.675	0.677	0.675	0.679	0.687	0.680	0.674	0.678	0.677	0.674	0.6774	0.0036
	HDT^*_{12}	0.676	0.678	0.678	0.674	0.682	0.687	0.681	0.674	0.677	0.675	0.676	0.6780	0.0038
	HDT^*_{14}	0.677	0.672	0.679	0.672	0.679	0.687	0.683	0.678	0.678	0.672	0.678	0.6777	0.0044
	HDT^*_{15}	0.677	0.677	0.676	0.675	0.682	0.687	0.684	0.675	0.680	0.672	0.678	0.6785	0.0042
	HDT^*_{16}	0.677	0.677	0.676	0.674	0.677	0.687	0.684	0.671	0.679	0.672	0.678	0.6775	0.0045
VOT2014	HDT^*_{10}	0.525	0.521	0.521	0.525	0.535	0.526	0.525	0.526	0.522	0.525	0.526	0.5252	0.0036
	HDT^*_{11}	0.525	0.521	0.521	0.526	0.521	0.526	0.526	0.526	0.522	0.526	0.527	0.5243	0.0023
	HDT^*_{12}	0.525	0.521	0.521	0.525	0.521	0.526	0.526	0.526	0.522	0.526	0.525	0.5240	0.0021
	HDT^*_{14}	0.519	0.525	0.526	0.521	0.519	0.526	0.525	0.526	0.523	0.519	0.523	0.5229	0.0028
	HDT^*_{15}	0.519	0.526	0.526	0.521	0.519	0.526	0.526	0.525	0.523	0.519	0.523	0.5230	0.0029
	HDT^*_{16}	0.519	0.526	0.526	0.521	0.519	0.526	0.526	0.525	0.523	0.519	0.523	0.5230	0.0029

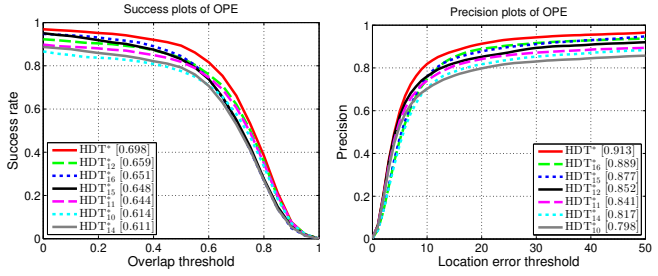


Fig. 8. Experimental results of HDT* and its component trackers on the OTB50 dataset.

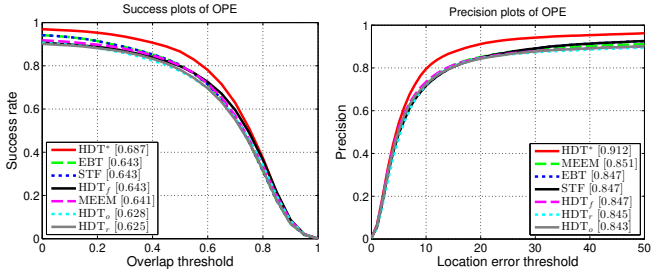


Fig. 9. Experimental results of HDT* and three ensemble methods as well as three baseline methods on the OTB100 dataset.

the AUC scores and the precisions on the OTB100 dataset perturb around 0.672 and 0.897 with small standard deviations 0.006 and 0.008, respectively. On the VOT2014 dataset, the AUC scores and the precisions perturb around 0.519 and 0.688 with small standard deviations 0.003 and 0.006, respectively. These results demonstrate that the proposed algorithm is not sensitive to γ within a wide range and across datasets.

Initial weights for component trackers. Based on grid search (i.e., parameter sweep), we determine that the proposed algorithm performs best when all the six component trackers have the same initial weight, i.e., 0.1667. For sensitivity analysis, we perturb the initial weight of a component tracker significantly while keeping initial weights of other component trackers unchanged. The perturbations are around the optimal value 0.1667, e.g., the initial weight is disturbed from 0.0667 to 0.2667 with a step size 0.02. The tracking results on the OTB100 and VOT2014 datasets are shown in Table 5 in terms of the AUC metric. When changing the initial weight of each component tracker, the AUC score varies around 0.678 with a small standard deviation 0.004 on the OTB100 dataset. On the VOT2014 dataset, the AUC score varies around 0.524 with a small standard deviation 0.003. These results demonstrate the proposed algorithm is insensitive to initial weights within a wide range and across datasets.

4.5 Comparisons to the State-of-the-art Trackers

4.5.1 Quantitative Evaluation

Comparison with ensemble methods. We evaluate the proposed HDT* against three recently published ensemble methods: STF [42], MEEM [14], and EBT [40] as well as three baselines: (1) HDT_f that combines component trackers using fixed weights learned in the second frame; (2) HDT_r that combines component trackers using random weights; and (3) HDT_o that learns a correlation filter tracker using the concatenated features of all six CNN layers. For fair comparisons, all of these ensemble methods uses the same component trackers as HDT*.

Figure 9 shows the OPE results on the OTB100 dataset, which indicate that the proposed HDT* outperforms MEEM, EBT and STF, by about 6% in terms of precision and 4% in terms of success rate. In addition, HDT* also outperforms the three baseline methods by about 7% and 4% in terms of the same metrics. Overall, these results demonstrate the effectiveness of the proposed HDT*.

Comparison with CNN-based methods. We evaluate the proposed HDT* against: 1) five state-of-the-art CNN-based trackers including MDNet [37], CFNet [52], SINT [53], CF2 [25], CNN-SVM [36]. 2) three modified CNN-based trackers: DeepMEEM, DeepKCF, and DeepSRDCF by substituting the state-of-the-art trackers MEEM [14], KCF [29], and SRDCF [31] with VGG19 features. For completeness, the results of HDT [27] are also presented. We note that MDNet, CFNet, and SINT use CNN features fine-tuned on videos. The remaining trackers including ours use CNN features pre-trained on the ImageNet dataset [54] without fine-tuning.

Figure 10 presents the OPE, SRE, and TRE results on the OTB100 dataset. In terms of OPE, HDT* performs favorably against all evaluated trackers. On the other hand, HDT* performs slightly worse than MDNet in terms of SRE

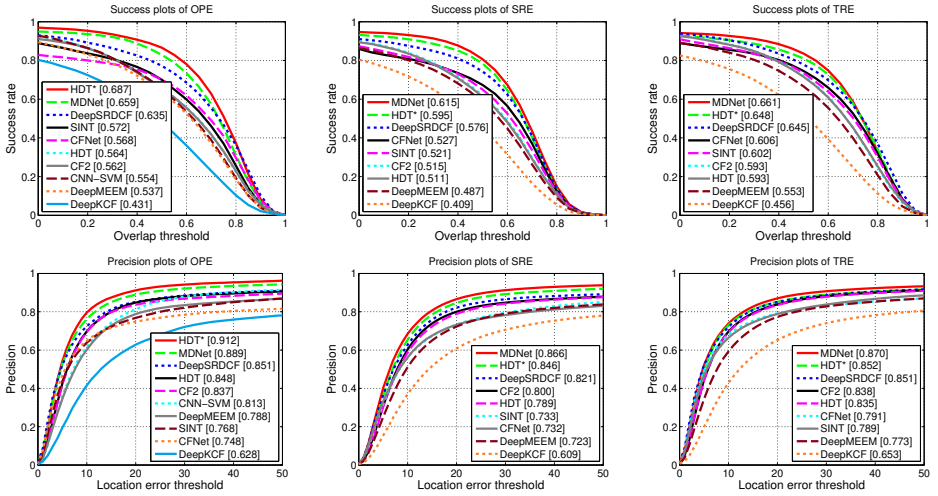


Fig. 10. One pass spatial robustness evaluation and temporal robustness evaluation results on the OTB100 dataset.

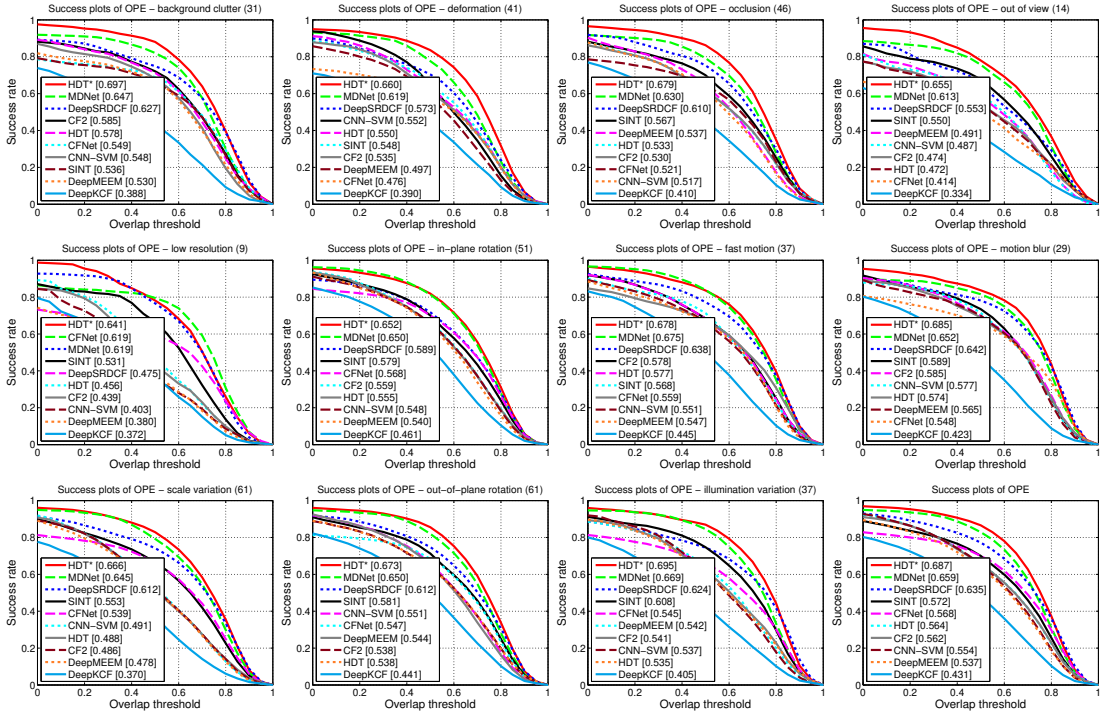


Fig. 11. Attribute-based evaluation on the OTB100 dataset. For completeness, we also include the overall results.

and TRE. In addition, the proposed HDT* outperforms the preliminary HDT by more than 5%.

We also present tracking results in terms of each attribute on the OTB100 dataset in Figure 11. Overall, HDT* performs well in challenges of background clutter, shape deformation, occlusion, low resolution, and out-of-view. This can be attributed to the proposed adaptive hedge algorithm, which adaptively weights component trackers to use CNN features with rich spatial details extracted from first few layers and high level semantics features extracted from last few layers. In contrast, other CNN-based trackers (*e.g.*, CNN-SVM and MDNet) only exploit features from the last few layers. In addition, we note that CF2 does not perform as well as MDNet and HDT* despite exploiting features from different convolutional layers. This can be explained by the fact that CF2 uses fixed weights for different convolutional layers and thus may not perform well in challenges with rapid appearance changes.

TABLE 6
Experimental results in terms of the expected average overlap metric on the VOT2016 dataset. The best three results are shown in red, blue, and green, respectively.

	ECO	CCOT	HDT*	CFNet	MDNet	CF2	Deep SRDCF	Deep KCF
Unsupervised	0.529	0.516	0.564	0.460	0.520	0.503	0.490	0.412
Baseline	0.357	0.335	0.275	0.201	0.257	0.230	0.256	0.177
Overall	0.443	0.426	0.420	0.331	0.389	0.367	0.373	0.295

Table 6 presents the tracking results on the VOT2016 dataset in terms of the EAO metric. The VOT2016 evaluation system conducts experiments on each image sequence in two settings: unsupervised and baseline. The first setting is

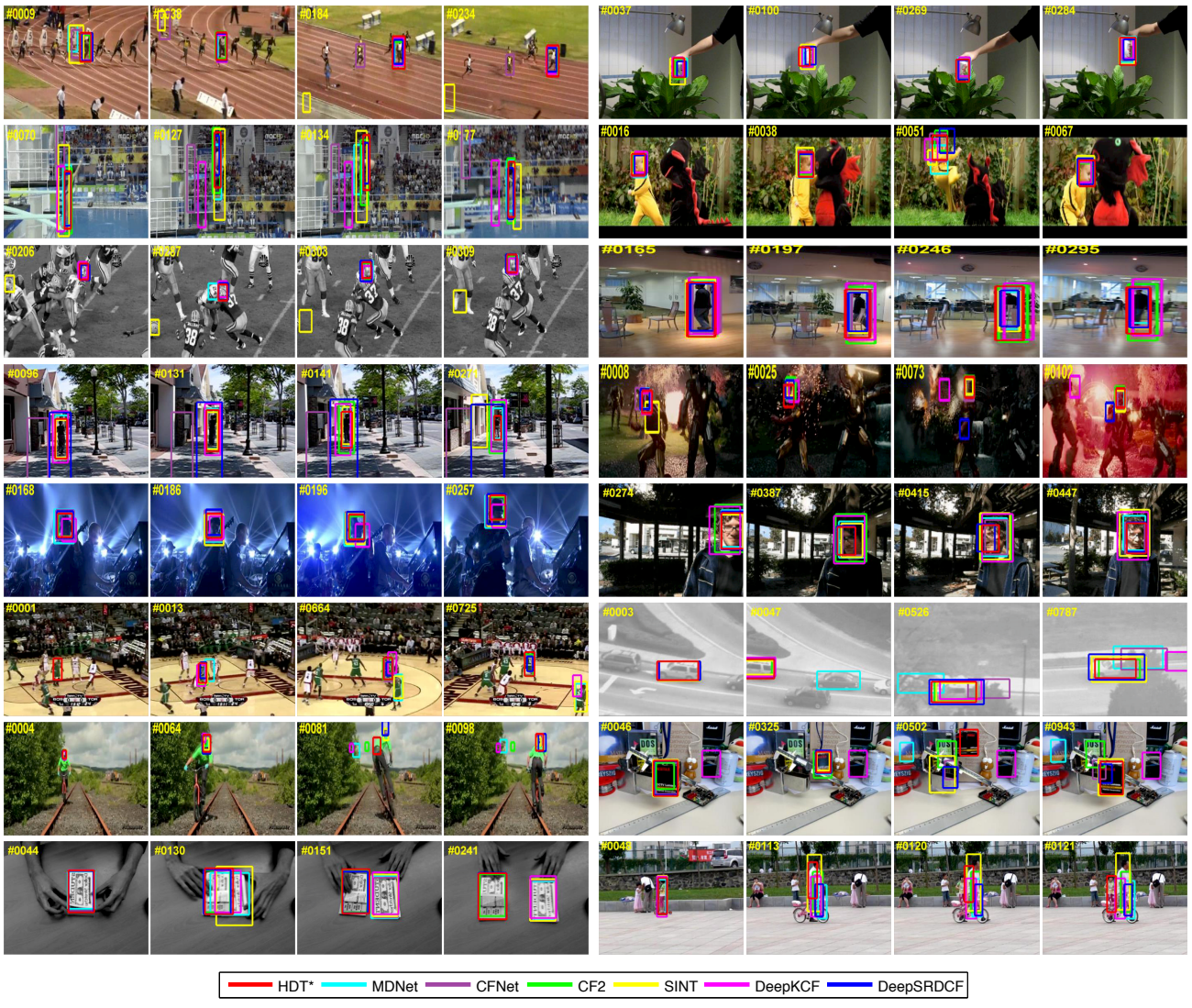


Fig. 12. Sample tracking results on challenging image sequences (from left to right and top to down are *bolt2*, *coke*, *diving*, *dragonBaby*, *football*, *human2*, *human9*, *ironman*, *shaking*, *trellis*, *basketball*, *suv*, *biker*, *box*, *coupon*, and *girl2*, respectively).

based on one run of each tracker once. The second setting is based on 15 runs of each tracker with re-initializations using ground truth when tracking failure is detected. The overall results are the averages of the unsupervised and baseline scores. The results show that the HDT* method ranks the 1st, 3rd, and 3rd in the unsupervised, baseline, and overall scores, respectively. It should be noted that the unsupervised setting is closer to real applications, which indicates that the HDT* method is suitable for real applications. In terms of the baseline and overall scores, the HDT* method performs comparably with the state-of-the-art algorithms.

Table 7 presents the average run-time of the evaluated trackers on the OTB100 dataset. In addition, we also present the tracking results in terms of the AUC and precision metrics. As shown in Table 7, the proposed HDT* processes 1.4 frames per second, which is slightly faster than MDNet (1.1 frames per second). Although CFNet and DeepSRDCF run about 10 times faster than HDT*, they perform worse than HDT* with about 10% decrease.

TABLE 7
Run-time of the evaluated tracking methods in terms of frames per second on the OTB100 dataset. The tracking results in terms of the AUC and precision metrics are also presented.

	HDT*	MDNet	SINT	CFNet	CF2	Deep SRDCF	Deep KCF
FPS	1.4	1.1	7.4	17.3	0.2	10.4	7.5
AUC	0.687	0.659	0.572	0.568	0.635	0.562	0.431
Precision	0.912	0.889	0.768	0.748	0.851	0.837	0.628

4.5.2 *Qualitative Evaluation*

We present sample tracking results of the evaluated methods in Figure 12. For presentation clarity, only results from the top seven performing methods are shown. Overall, our tracker is able to locate the targets well in complicated scenes. In the *trellis* sequence, the target undergoes rotations in a cluttered background with varying illumination. From frame 274, all methods except the proposed tracker drift away to various certain extents. The tracking results by the MDNet are not precise as the bounding box extents are large. In contrast, the proposed HDT* algorithm is able to track the target object well.

In the *basketball* sequence, at frame 664 the CF2 tracker locates at the wrong object that is close to the target object. In contrast, the proposed HDT* algorithm is able to track the target object throughout the sequence. When the target is heavily occluded, as in the *suv* or *girl2* sequences, most evaluated methods lose track of the targets gradually whereas the proposed algorithm performs well.

5 Conclusions

In this paper, we propose a tracking algorithm based on an adaptive online decision learning algorithm to hedge weak trackers, constructed by correlation filters on CNN feature maps, into an effective one. The cumulative regret model of the adaptive hedge algorithm varies the proportion of current regret of each weak tracker over time. In addition, we develop a Siamese network to measure the appearance similarity between target templates and tracking results. Extensive experimental evaluations on large-scale benchmark datasets demonstrate the effectiveness of the proposed hedge deep tracking algorithm.

6 Acknowledgments

This work was supported in part by National Natural Science Foundation of China: 61620106009, 61332016, U1636214, 61650202, 61672188, 61572465, 61390510, 61732007, 61472103, 61772158, U1711265; in part by Key Research Program of Frontier Sciences, CAS: QYZDJ-SSW-SYS013; in part by the NRF grant funded by Ministry of Science, ICT Korea: NRF-2017R1A2B4011928 and NRF-2017M3C4A7069369; in part by the NSF CAREER Grant 1149783, and gifts from Adobe, Verisk, and Nvidia. S. Zhang was also supported by the Young Excellent Talent Program of Harbin Institute of Technology.

Appendix

Here we first present the upper bound of the proposed adaptive hedge as a theorem and then give its proof based on three lemmas. For the proofs of the lemmas, please see the supplementary material.

Theorem 1. *If our adaptive hedge has access to N experts, then for all loss sequences, for all t, for all $0 < \epsilon \leqslant 1$ and for all $0 < \delta \leqslant 1/2$, the regret of the algorithm to the top ϵ-quantile of the experts is at most*

$$[5(t-t_0)(1+\delta) + \frac{8\ln^3 N}{\delta} + \frac{81\ln^2 N}{\delta^3}](\ln\frac{1}{\epsilon}+1).$$

In particular, with $\epsilon = 1/N$, the regret to the best expert is at most

$$[5(t-t_0)(1+\delta) + \frac{8\ln^3 N}{\delta} + \frac{81\ln^2 N}{\delta^3}](\ln N+1).$$

The value δ in Theorem 1 appears to be an artifact of our analysis; we divide the sequences of rounds into two phases (the length of the first phase is controlled by the value of δ) and bound the behavior of the algorithm in each phase separately. The following corollary illustrates the performance of our algorithm for large values of t, in which case the effect of this phase (and the δ in the bound) essentially goes away.

Corollary 1. *If our adaptive hedge has access to N experts, then, as $t \to \infty$, the regret of our algorithm to the top ϵ-quantile of experts approaches an upper bound of*

$$5t(1+\ln\frac{1}{\epsilon}) + o(t).$$

In particular, the regret of our algorithm to the best expert approaches an upper bound of

$$5t(1+\ln N) + o(t).$$

The proof of Theorem 1 follows from a combination of Lemmas 1, 2, and 3, and is presented in detail at the end of the current section.

Lemma 1. *At any time t, the regret to the best expert can be bounded as:*

$$\max_i R_t^i \leqslant c_t(\ln N + 1).$$

Moreover, for any $0 < \epsilon \leqslant 1$ and any t, the regret to the top ϵ-quantile of experts is at most

$$c_t(\ln(1/\epsilon)+1). \tag{24}$$

Proof. We use E_t to denote the experts that have positive cumulative regret on iteration t. The first part of the lemma follows from the fact that, for any expert $i \in E_t$,

$$\exp(\frac{R_t^i}{c_t}) = \exp(\frac{[R_t^i]_+}{c_t}) \leqslant \sum_{i'=1}^{N}\exp(\frac{[R_t^{i'}]_+}{c_t}) \leqslant Ne$$

which implies $R_t^i \leqslant c_t(\ln N + 1)$. For the second part of the lemma, let R_t^i denote the regret of our algorithm to the expert with the ϵN-th highest regret. Then, the total potential of the experts with regrets greater than or equal to R_t^i is at least:

$$\epsilon N \exp(\frac{[R_t^i]_+}{c_t}) \leqslant Ne,$$

which implies $R_t^i \leqslant c_t(\ln(1/\epsilon)+1)$. □

Lemma 2. *For any time t,*

$$c_{t+1} \leqslant 2c_t(1+\ln N) + 3. \tag{25}$$

Lemma 3. *Suppose that at some time t_0, $c_{t_0} \geqslant \frac{4\ln^2 N}{\delta} + \frac{16\ln N}{\delta^3}$, where $0 < \delta \leqslant 1/2$ is a constant. Then, for any time $t \geqslant t_0$,*

$$c_{t+1} - c_t \leqslant 5(1+\delta). \tag{26}$$

In Lemmas 2 and 3 we bound the growth of the scale c_t as a function of the time t. The main outline of the proof of Theorem 1 is as follows. As c_t increases monotonically with t, we can divide the rounds t into two phases, $t < t_0$ and $t \geqslant t_0$, where t_0 is the first time such that

$$c_{t_0} \geqslant \frac{4\ln^2 N}{\delta} + \frac{16\ln N}{\delta^3},$$

for some fixed $\delta \in (0, 1/2)$. We then show bounds on the growth of c_t for each phase separately. Lemma 2 shows that c_t is not too large at the end of the first phase, while Lemma 3 bounds the per-round growth of c_t in the second phase.

We now combine Lemmas 2 and 3 together with Lemma 1 to prove the Theorem 1.

Proof of Theorem 1. Let t_0 be the first time at which $c_{t_0} \geqslant \frac{4\ln^2 N}{\delta} + \frac{16\ln N}{\delta^3}$. Then, from Lemma 2

$$c_{t_0} \leqslant 2c_{t_0-1}(1+\ln N)+3,$$

which is at most:

$$\frac{8\ln^3 N}{\delta} + \frac{34\ln^2 N}{\delta^3} + \frac{32\ln N}{\delta^3} + 3 \leqslant \frac{8\ln^3 N}{\delta} + \frac{81\ln^2 N}{\delta^3}.$$

The last inequality follows because $N \geqslant 2$ and $\delta \leqslant 1/2$. By Lemma 3, we have that for any $t \geqslant t_0$,

$$c_t \leqslant 5(t-t_0)(1+\delta) + c_{t_0}.$$

Combining these last two inequalities yields

$$c_t \leqslant 5(t-t_0)(1+\delta) + \frac{8\ln^3 N}{\delta} + \frac{81\ln^2 N}{\delta^3}.$$

Now the theorem follows by applying Lemma 1.

REFERENCES

[1] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1442–1468, 2014. 1, 3

[2] X. Li, W. Hu, C. Shen, Z. Zhang, A. R. Dick, and A. van den Hengel, "A survey of appearance models in visual object tracking," *ACM Transactions on Intelligent Systems and Technology*, vol. 4, no. 4, p. 58, 2013. 1, 3

[3] X. Song, X. Shao, Q. Zhang, R. Shibasaki, H. Zhao, J. Cui, and H. Zha, "A fully online and unsupervised system for large and high-density area surveillance: Tracking, semantic scene learning and abnormality detection," *ACM Transactions on Intelligent Systems and Technology*, vol. 4, no. 2, pp. 35:1–35:21, 2013. 1

[4] B. Benfold and I. D. Reid, "Stable multi-target tracking in real-time surveillance video," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 3457–3464. 1

[5] J. Yang, R. Stiefelhagen, U. Meier, and A. Waibel, "Visual tracking for multimodal human computer interaction," in *Conference on Human Factors in Computing Systems*, 1998. 1

[6] J. C. McCall and M. M. Trivedi, "Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 1, pp. 20–37, 2006. 1

[7] H. Schneiderman and M. Nashman, "A discriminating feature tracker for vision-based autonomous driving," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 6, pp. 769–775, 1994. 1

[8] P. J. Figueroa, N. J. Leite, and R. M. L. Barros, "Tracking soccer players aiming their kinematical motion analysis," *Computer Vision and Image Understanding*, vol. 101, no. 2, pp. 122–135, 2006. 1

[9] Y. Wu, J. Lim, and M.-H. Yang, "Object tracking benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, pp. 1834–1848, 2015. 1, 6, 7

[10] ——, "Online object tracking: A benchmark," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 1, 7

[11] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," in *Proceedings of the British Machine Vision Conference*, 2006. 1, 3

[12] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *Proceedings of European Conference on Computer Vision*, 2008. 1

[13] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proceedings of European Conference on Computer Vision*, 2012. 1, 3, 4

[14] J. Zhang, S. Ma, and S. Sclaroff, "MEEM: robust tracking via multiple experts using entropy minimization," in *Proceedings of European Conference on Computer Vision*, 2014. 1, 9

[15] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2006. 1

[16] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2005. 1

[17] W. Zhong, H. Lu, and M.-H. Yang, "Robust object tracking via sparsity-based collaborative model," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 1

[18] N. Wang and D.-Y. Yeung, "Learning a deep compact image representation for visual tracking," in *Advances in Neural Information Processing Systems*, 2013. 1, 3

[19] J. Fan, W. Xu, Y. Wu, and Y. Gong, "Human tracking using convolutional neural networks," *IEEE Transactions on Neural Networks*, vol. 21, no. 10, pp. 1610–1623, 2010. 1, 3

[20] S. Hong, T. You, S. Kwak, and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network," in *Proceedings of the International Conference on Machine Learning*, 2015. 1, 3

[21] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Visual tracking with fully convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015. 1, 3

[22] M. Kristan, J. Matas, A. Leonardis, J. Matas, and etal, "The visual object tracking VOT2015 challenge results," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015. 1, 6

[23] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the IEEE International Conference on Computer Vision*, 1999. 1

[24] G. Tian, R. Hu, Z. Wang, and Y. Fu, "Improved object tracking algorithm based on new HSV color probability model," in *International Symposium on Neural Networks*, 2009. 1

[25] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015. 1, 3, 9

[26] K. Chaudhuri, Y. Freund, and D. Hsu, "A parameter-free hedging algorithm," in *Advances in Neural Information Processing Systems*, 2009. 2, 4, 5, 6

[27] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, J. Lim, and M.-H. Yang, "Hedged deep tracking," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 2, 3, 6, 9

[28] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 3, 4

[29] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015. 3, 4, 9

[30] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *Proceedings of the British Machine Vision Conference*, 2014. 3, 4, 6

[31] ——, "Learning spatially regularized correlation filters for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015. 3, 4, 7, 9

[32] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012. 3, 4

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Computing Research Repository*, vol. abs/1409.1556, 2014. 3, 4, 6

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *Proceedings of European Conference on Computer Vision*, 2014. 3

[35] S. Gidaris and N. Komodakis, "Object detection via a multi-region and semantic segmentation-aware CNN model," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015. 3

[36] S. Hong, T. You, S. Kwak, and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network," in *Proceedings of the International Conference on Machine Learning*, 2015. 3, 9

[37] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 3, 5, 6, 7, 9

[38] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 3

[39] S. Avidan, "Ensemble tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 2, pp. 261–271, 2007. 3

[40] N. Wang and D.-Y. Yeung, "Ensemble-based tracking: Aggregating crowdsourced structured time series data," in *Proceedings of the International Conference on Machine Learning*, 2014. 3, 9

[41] Q. Bai, Z. Wu, S. Sclaroff, M. Betke, and C. Monnier, "Randomized ensemble tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013. 3
[42] C. Bailer, A. Pagani, and D. Stricker, "A superior tracking approach: Building a strong tracker through fusion," in *Proceedings of European Conference on Computer Vision*, 2014. 3, 9
[43] T. Vojir, J. Matas, and J. Noskova, "Online adaptive hidden markov model for multi-tracker fusion," *Computer Vision and Image Understanding*, vol. 153, pp. 109–119, 2016. 3
[44] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *Computational learning theory*, 1995. 3
[45] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *ACM Multimedia*, 2014. 4
[46] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 4
[47] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, and etal, "The visual object tracking VOT2016 challenge results," in *Proceedings of European Conference on Computer Vision Workshops*, 2016. 6
[48] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, and etal, "The visual object tracking vot2013 challenge results," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2013. 6
[49] M. Kristan, R. P. Pflugfelder, A. Leonardis, J. Matas, and etal, "The visual object tracking VOT2014 challenge results," in *Proceedings of European Conference on Computer Vision Workshops*, 2014. 6
[50] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014. 7
[51] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010. 7
[52] J. Valmadre, L. Bertinetto, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "End-to-end representation learning for correlation filter based tracking," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 9
[53] R. Tao, E. Gavves, and A. W. Smeulders, "Siamese instance search for tracking," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 9
[54] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, "Imagenet: A large-scale hierarchical image database," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 9

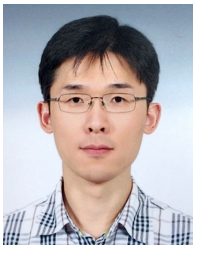
Yuankai Qi received the B.S. and M.S. degrees from Harbin Institute of Technology, China, in 2011 and 2013, respectively, and is currently working toward the Ph.D. degree in computer science and technology at Harbin Institute of Technology, China. His research interests include object tracking, sparse coding, and machine learning. He serves as a reviewer of top journals such as IEEE TIP, TMM, and TCSVT.

Shengping Zhang received the Ph.D. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 2013. He is currently a Professor with the School of Computer Science and Technology, Harbin Institute of Technology at Weihai. He had been a postdoctoral research associate with Brown University and with Hong Kong Baptist University, and a visiting student researcher with University of California at Berkeley. He has authored or coauthored over 50 research publications in refereed journals and conferences. His research interests include deep learning and its applications in computer vision. Dr. Zhang is also an Associate Editor of the Signal Image and Video Processing, and Journal of Electronic Imaging.

Lei Qin received the Ph.D. degree in computer science from the Institute of Computing Technology of the Chinese Academy of Sciences (ICTCAS), Beijing, China, in 2008. He is currently an associate professor with the Key Laboratory of Intelligent Information Processing of ICTCAS. His research interests include image/video processing, computer vision, and pattern recognition. He has authored or coauthored over 50 academic papers. He is a reviewer for IEEE TMM, IEEE TCSVT, and IEEE Trans. on Cybernetics. He has served as technical program committee member for various conferences, including ECCV, ICPR, ICME, PSIVT, ICIMCS, and PCM.

Qingming Huang (F'18) is a professor in University of Chinese Academy of Sciences (CAS) and an adjunct professor in both Harbin Institute of Technology (HIT) and Institute of Computing Technology of CAS. He received B.S. and Ph.D. degrees in 1988 and 1994 respectively, both from HIT, China. His research areas include multimedia video analysis, image processing, computer vision and pattern recognition. He has published more than 300 academic papers on top journals and conferences. He is the associate editor of Acta Automatica Sinica and Fellow of IEEE. He has served as program chair, track chair and technical program committee member for top conferences such as ACM Multimedia, CVPR, and ICCV.

Hongxun Yao received the B.S. and M.S. degrees in computer science from the Harbin Shipbuilding Engineering Institute, Harbin, China, in 1987 and in 1990, respectively, and received Ph.D. degree in computer science from Harbin Institute of Technology in 2003. Currently, she is a professor with School of Computer Science and Technology, Harbin Institute of Technology. Her research interests include computer vision, pattern recognition, multimedia computing, human-computer interaction technology. She has 6 books and over 200 scientific papers published, and won both the honor title of the new century excellent talent in China and enjoys special government allowances expert in Heilongjiang Province, China.



Jongwoo Lim received the B.S. degree from Seoul National University, Seoul, Korea, in 1997, and the M.S. and Ph.D. degrees from the University of Illinois at Urbana-Champaign, in 2003 and 2005, respectively. He was at Honda Research Institute USA Inc., Mountain View, CA, as a senior scientist from 2005 to 2011, and at Google Inc., Mountain View, CA, as a software engineer from 2011 to 2012. Currently, he is an associate professor in the Department of Computer Science at Hanyang University, Seoul, Korea. His research interests include computer vision, robotics, and machine learning.

Ming-Hsuan Yang (SM'06) is a professor of Electrical Engineering and Computer Science with the University of California, Merced, CA, USA. He received the Ph.D. degree in computer science from the University of Illinois at Urbana-Champaign, Urbana, IL, USA, in 2000. He served as an Associate Editor of the IEEE Transactions on Pattern Analysis and Machine Intelligence from 2007 to 2011, and is an Associate Editor of the International Journal of Computer Vision, Image and Vision Computing, and Journal of Artificial Intelligence Research. He received the NSF CAREER Award in 2012, and the Google Faculty Award in 2009. He is a senior member of the IEEE and ACM.