Learning Recurrent Memory Activation Networks for Visual Tracking

Shi Pu, *Member, IEEE*, Yibing Song, *Member, IEEE*, Chao Ma, *Member, IEEE*, Honggang Zhang, *Senior Member, IEEE*, and Ming-Hsuan Yang, *Fellow, IEEE*

***Abstract*— Facilitated by deep neural networks, numerous tracking methods have made significant advances. Existing deep trackers mainly utilize independent frames to model the target appearance, while paying less attention to its temporal coherence. In this paper, we propose a recurrent memory activation network (RMAN) to exploit the untapped temporal coherence of the target appearance for visual tracking. We build the RMAN on top of the long short-term memory network (LSTM) with an additional memory activation layer. Specifically, we first use the LSTM to model the temporal changes of the target appearance. Then we selectively activate the memory blocks via the activation layer to produce a temporally coherent representation. The recurrent memory activation layer enriches the target representations from independent frames and reduces the background interference through temporal consistency. The proposed RMAN is fully differentiable and can be optimized end-to-end. To facilitate network training, we propose a temporal coherence loss together with the original binary classification loss. Extensive experimental results on standard benchmarks demonstrate that our method performs favorably against the state-of-the-art approaches.**

***Index Terms*— Visual tracking, recurrent memory activation, temporal coherence, representation.**

I. Introduction

VISUAL tracking has received growing attention in recent years with numerous applications to analyzing and understanding video contents [9], [10], [18]. Existing tracking-by-detection methods based on deep learning have achieved the state-of-the-art performance on the standard benchmarks [35], [38], [68], [69]. This framework typically consists of two steps. First, a set of samples is drawn around the estimated target object location in the previous frame. Then, each sample is independently classified as the target or the background. Exiting deep trackers have demonstrated great success by exploiting discriminative features from convolutional neural networks (CNNs) [26] or classifiers [22], [46]. We note that these trackers often draw samples or perform classification solely in a frame at one time. Hence the temporal consistency of the target appearance is not exploited. A number of attempts have been made to incrementally collect samples from the previous frames for fine-tuning deep networks [22], [46], [60]. However, these samples are randomly collected and do not maintain an order to capture the temporal coherency. It is therefore of great importance to effectively model these samples in order to exploit the temporal consistency.

Intuitively, the Recurrent Neural Network (RNN) can be used to model object motions in the temporal domain in [47]. However, we observe that there is no significant performance

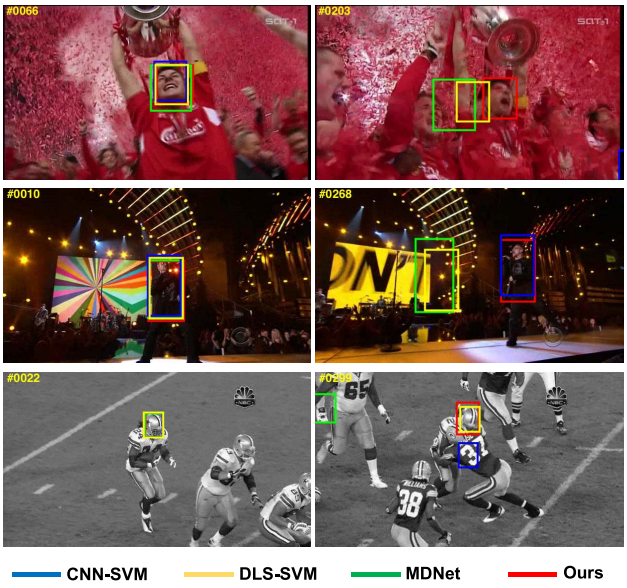

Fig. 1. Visual comparison with the tracking-by-detection methods (CNN-SVM [26], DLS-SVM [48], and MDNet [46]) on the *Soccer*, *Singer2*, and *Football* sequences [68]. Our recurrent memory activation network is integrated into the tracking-by-detection framework to exploit the temporal coherence. This network selectively organizes spatiotemporal features to model the target object as a coherent representation. The proposed method performs favorably against existing approaches using only spatial features.

Manuscript received July 18, 2019; revised September 7, 2020; accepted November 2, 2020. Date of publication November 24, 2020; date of current version December 4, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 62076034. The work of Chao Ma was supported in part by NSFC under Grant 60906119 and in part by the Shanghai Pujiang Program. The work of Ming-Hsuan Yang was supported by the National Science Foundation CAREER under Grant 1149783. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Riccardo Leonardi. *(Corresponding author: Ming-Hsuan Yang.)*

Shi Pu is with Tencent AI Lab, Beijing 100193, China (e-mail: pushi_519200@qq.com).

Yibing Song is with Tencent AI Lab, Shenzhen 518057, China (e-mail: yibingsong.cv@gmail.com).

Chao Ma is with AI Institute, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: chaoma@sjtu.edu.cn).

Honggang Zhang is with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: zhhg@bupt.edu.cn).

Ming-Hsuan Yang is with the Computer Science and Engineering, University of California Merced, Merced, CA 95343 USA (e-mail: mhyang@ucmerced.edu).

Digital Object Identifier 10.1109/TIP.2020.3038356

gain brought by the RNN to tracking-by-detection methods. This can be attributed to that the temporal modeling space in the RNN is different from the binary classification feature space in visual tracking. The appearance variations of the targets, which are small and effectively organized in the RNN, may become severe and unordered in the binary classification process. The representation discrepancies between the RNN and fully connected binary classification limit the use of temporal modeling in visual tracking.

In this paper, we propose a Recurrent Memory Activation Network (RMAN) to narrow the gap between the RNN model and the tracking-by-detection framework. In the RMAN, we use an LSTM to generate a serial of memory states as the spatiotemporal representations of target objects from input sequences. We then selectively activate the memory states using the proposed recurrent memory activation layer. The layer reorganizes the memory states according to their similarity with that of the current bounding box sample. When a bounding box contains the target, its historical appearances modeled by the LSTM will be activated by this layer to enrich its feature representation. This coherent spatiotemporal target modeling facilitates the binary classification and improves the tracking accuracy. Our recurrent memory activation layer is integrated into the RMAN for end-to-end training and prediction. In addition to the original classification objectives, we propose a temporal coherence loss to increase the discriminative margin between the target and background. The temporal coherence loss brings memory generation and binary classification together to ensure historical memory states activated by the activation layer facilitate the classification process. We demonstrate that the proposed method performs favorably against the state-of-the-art approaches on the standard benchmarks.

The main contributions of this work are:

- We propose a recurrent memory activation network to generate memory states of target objects from input sequences. We selectively activate these memory states via a recurrent memory activation layer for target feature representation enrichment.
- We present a temporal coherence loss to train RMAN together with the original classification loss. The proposed loss function brings sequential memory generation and binary classification together to ensure the coherent modeling to benefit visual tracking.
- We conduct extensive experiments on the benchmark datasets with large-scale sequences. The results show that the proposed method performs favorably against the state-of-the-art approaches.

II. Related Work

Visual tracking has been extensively studied [54], [58], [72] over the last decade. In this section, we discuss the visual trackers and recurrent neural networks most relevant to this work.

A. Visual Tracking

Numerous trackers proposed in recent years can be broadly categorized by the formulation based on correlation filters, tracking-by-detection, and Siamese networks. The trackers based on correlation filters regress all the circular-shifted versions of the input features to soft labels generated by a Gaussian function. As the correlation operation can be efficiently computed as an element-wise product in the Fourier domain, these schemes are widely used in the tracking literature. Since the MOSSE tracker [6], numerous methods have been developed to improve the correlation filter for visual tracking. These trackers include kernelized correlation filters [24], [25], spatial regularization and multi-scale fusion [13], [14], CNN feature integrations [34], [42], [53], [75], and end-to-end predictions [5], [59], [63], [65]. Existing correlation filter trackers mainly focus on appearance modeling in independent frames and pay less attention to exploit temporal coherence as an additional clues to handle appearance variations including deformation, illumination variation, rotation, and occlusion.

On the other hand, the tracking-by-detection framework formulates visual tracking as a binary classification problem. First, the trackers draw a set of samples around the target object location and then classify each sample as either the target or the background. A large number of efforts have been made to advance the tracking-by-detection framework for robust tracking including online boosting [2], [21], P-N learning [33], structured SVM [23], [48], reciprocative learning [52], and multiple domain learning [46]. Existing tracking-by-detection methods use bounding box samples from single frames for classification. Although these approaches collect historical samples for online training, these samples are randomly collected and do not maintain an order to capture the temporal coherency. Different from existing tracking-by-detection trackers which ignore the temporal coherence in videos, our method exploits temporal coherence to generate discriminative features to facilitate classification.

The Siamese based framework has attracted a lot of attention in visual tracking. Since the SiamFC tracker [5], extensions include adding a triplet loss [16], region proposal networks [36], unsupervised learning [66], [67], and adversarial attacks [28]. These Siamese trackers compare template features from previous frames with region features from the current frame to locate targets. These region features as proposal representations are from independent frames. The matching mechanisms in Siamese methods do not incorporate temporal coherence into these representations. Our approach uses a classifier to distinguish temporally coherent representations which are from sequence candidates. The temporal modeling in our method enhances representation capability to facilitate visual tracking.

B. Recurrent Neural Networks

The input and output of a recurrent neural network form a loop in a sequential manner. This loop consists of several states passing along the timesteps for a temporal memory simulation. The formulation of sequence data and memory capacity of the RNNs have attracted a lot of attention recently with applications in machine translation [61], video analysis [15], image captioning [64], low-level vision [39], video

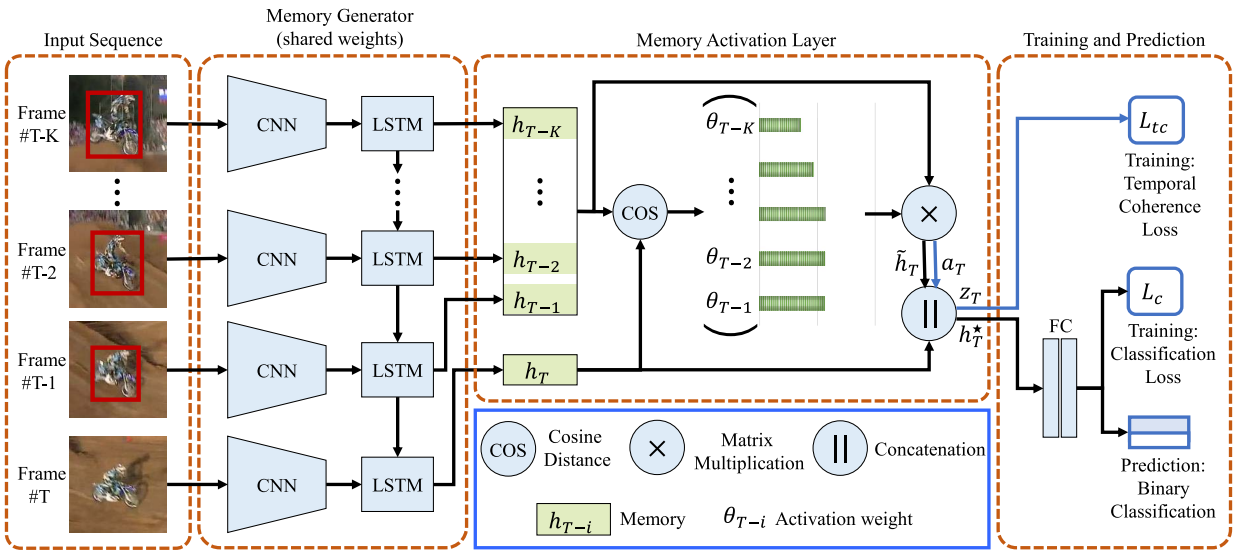

Fig. 2. Overview of the RMAN architecture. We use sequence samples as input to predict target locations of each frame. The RMAN consists of a memory generator, a recurrent memory activation layer, and a binary classifier. The memory generator consists of $K+1$ branches to record the hidden states of the current and historical input patches. These states are selectively activated by the recurrent memory activation layer to enrich the target representation to facilitate the binary classification process. For presentation clarity and convenience, we denote the states $(a_T, h_T, h_T^\star)$ as z_T which is a general description of all the input elements to compute the temporal coherence loss, while in the prediction process, we only use $h_T^\star$ which is $\begin{bmatrix} \widetilde{h}_T \\ h_T \end{bmatrix}$.

detection [41], [62], multi-object tracking [44], [50], and person re-identification [43], [76]. To construct discriminative temporal representations to facilitate visual tracking, a number of attempts [17], [31], [47], [70] have been made by applying RNNs to capture the relevant information in the sequence data. However, inaccurate and occluded detections likely exist in the previous sequence, which leads to noisy updates and makes the temporal representation less effective. We propose a recurrent memory activation layer to reconstruct memory generated by RNNs. The layer enables our network to focus on related memory states and alleviate the effects of noisy memory. Our method demonstrates the feasibility of RNNs in improving the tracking-by-detection framework with state-of-the-art tracking accuracy.

III. Proposed Algorithm

Figure 2 shows an overview of the proposed algorithm. Our recurrent memory activation network contains a memory generator, a recurrent memory activation layer, and a binary classifier. The memory generator consists of a CNN module and an LSTM layer to generate target memory states and a current hidden state from an input sequence. The memory states are activated by the recurrent memory activation layer using the current hidden state as guidance to generate an activated state. The activated state is concatenated with the current hidden state to form a temporal representation for binary classification. In the following, we first introduce the model architecture. We then illustrate the proposed temporal coherence loss and show how it improves the tracking performance.

A. Recurrent Memory Activation Network

Our network takes a sequence of samples as input. We denote a sequence of samples as $X = (x, b_1, \ldots, b_K)$. When classifying a bounding box sample (i.e., x) in any frame T, we pass the bounding box sample x and the predicted target patches (i.e., $(b_1, \ldots, b_K)$) from previous frames into the memory generator of the RMAN. These previous frames are frames $T-1$ to $T-K$. The memory generator contains $K+1$ branches to formulate different states of the memory for the target object. Each branch consists of a fixed CNN feature extractor and an LSTM unit, and all branches share weights. These $K+1$ branches correspond to $K+1$ patches of a sequence of samples. Each branch only considers one patch. That is, we use a CNN with fixed parameters to extract features from different patches of a sample sequence, and the CNN features are forwarded to an LSTM which is presented in the unrolled form [20], [49]. We denote the outputs of the LSTM as the memory states $(h_{T-K}, \ldots, h_{T-1})$ and the current hidden state h_T. That is, the hidden state as the memory state generated from the predicted target patch b_i is h_{T-i} $(i \in [1, \ldots, K])$. The hidden state generated from the current bounding box sample x is h_T. When we obtain the memory states, we use a recurrent memory activation layer to reorganize the importance of each state. The recurrent memory activation layer first generates activation weights $(\theta_{T-K}, \ldots, \theta_{T-1})$, where θ_{T-i} $(i \in [1, \ldots, K])$ is the cosine distance between h_T and h_{T-i} $(i \in [1, \ldots, K])$. The weight θ_{T-i} is computed by

$$\theta_{T-i} = \frac{h_T^\top h_{T-i}}{|h_T^\top||h_{T-i}|}. \tag{1}$$

The recurrent memory activation layer uses activation weights $(\theta_{T-K}, \ldots, \theta_{T-1})$ to activate memory states $(h_{T-K}, \ldots, h_{T-1})$ to generate the activated state $\widetilde{h}_T$ as:

$$\widetilde{h}_T = \sum_{i=1}^{K} \theta_{T-i} \cdot h_{T-i}. \tag{2}$$

Using Eq. 1 and Eq. 2, the memory states are activated to generate the activated state $\widetilde{h}_T$. The hidden state h_T corresponding to the current bounding box sample x guides the activation process. If the current bounding box sample x contains the target object, the activated state $\widetilde{h}_T$ will integrate specific memory states to form an enriched historical target representation. This is because the historical target patches corresponding to these specific memory states are similar to the current bounding box sample x and the corresponding activation weights are large. If the current bounding box sample x only contains the background, the activated state $\widetilde{h}_T$ will not form a historical target representation. We concatenate the activated state $\widetilde{h}_T$ and the current hidden state h_T to form a temporal representation denoted as $h_T^\star = \begin{bmatrix} \widetilde{h}_T \\ h_T \end{bmatrix}$. The representation $h_T^\star$ encodes historical discriminative information into the current representation to facilitate the classification process. $h_T^\star$ is delivered to two fully connected layers which serve as our classifier to predict the probability of the current bounding box sample x being the binary label $y \in \{0, 1\}$. If the IOU (i.e., intersection over union) ratio between the current bounding box sample x and the predicted target exceeds a threshold τ_1, the label y is set to 1. If the above IOU ratio is less than a threshold τ_2, the label y is set to 0. The predicted probability by the classifier is used to compute the cross-entropy loss $\mathcal{L}_c$:

$$\mathcal{L}_c = \frac{1}{2N}\sum_{m=1}^{2N} -[y_m \cdot \log(p(X_m)) + (1 - y_m) \cdot \log(1 - p(X_m))], \quad (3)$$

where $2N$ is the number of sequence samples in a training batch, $p(X_m)$ is the predicted probability of the current bounding box sample x_m in the sequence sample X_m being the target. We use the loss $\mathcal{L}_c$ as the classification loss coupled with the proposed temporal coherence loss to train the RMAN. The temporal coherence loss is illustrated in Section III-B.

B. Temporal Coherence Loss

We propose a temporal coherence loss to help the memory generator better differentiate the positive (i.e., the target object) and negative (i.e., the background) patches. This loss facilitates the activation accuracy and discriminative temporal representations generation in the memory activation process. When taking a frame of a video, we randomly generate multiple positive and negative bounding box samples. The bounding box samples are assigned with binary labels according to their IOU ratios with the predicted bounding box in this frame. These bounding box samples are combined with the predicted target patches from previous K frames to be sequence samples. When training the RMAN, we construct a batch of positive-negative sequence sample pairs to compute the temporal coherence loss. Corresponding to the classification loss $\mathcal{L}_c$, the number of sequence sample pairs in this batch is N. We denote a positive sequence sample in this batch as X_m^p $(m \in [1, \ldots, N])$. Correspondingly, a negative sequence sample in this batch is denoted as X_m^n $(m \in [1, \ldots, N])$. Our temporal coherence loss regularizes the feature distributions of the positive and negative sequence samples.

To construct the temporal coherence loss, the network generates the state a_T which is only used during the training stage. We expect a_T is a coherence reference which represents the predicted target in previous frames, and define a_T as:

$$a_T = \sum_{i=1}^{K} \theta'_{T-i} \cdot h_{T-i}, \quad (4)$$

where θ'_{T-i} is $\frac{\exp(\theta_{T-i})}{\sum_{j=1}^{K}\exp(\theta_{T-j})}$ (i.e., softmax). Similar to $\widetilde{h}_T$, a_T is a linear combination of $(h_{T-K}, \ldots, h_{T-1})$. The difference between a_T and $\widetilde{h}_T$ is θ'_{T-i} and θ_{T-i}. θ_{T-i} is the cosine distance which reflects the discrimination ability of $\widetilde{h}_T$. However, we expect a_T represents the predicted target in previous frames. The magnitude of a_T is required to be comparable to $(h_{T-K}, \ldots, h_{T-1})$. Thus, we use the softmax function to ensure $\sum_{i=1}^{K}\theta'_{T-i} = 1,\ \theta'_{T-i} > 0$.

We use a_T to construct a loss function $\mathcal{L}_a$ as:

$$\mathcal{L}_a = \frac{1}{N}\sum_{m=1}^{N}[||f_{h_T}(X_m^p) - f_{a_T}(X_m^p)||_2^2 - ||f_{h_T}(X_m^n) - f_{a_T}(X_m^n)||_2^2 + \tau]_+, \quad (5)$$

where f_{h_T} and f_{a_T} are the networks from the input layer to the layers which output h_T and a_T, τ is the threshold determining the margin. We note Eq. 5 shares some similarity with the triplet loss [55] where there is a certain margin between the distance $||f_{h_T}(X_m^p) - f_{a_T}(X_m^p)||_2^2$ and $||f_{h_T}(X_m^n) - f_{a_T}(X_m^n)||_2^2$. Since we expect a_T to represent the predicted target in previous frames, a further expectation is that $f_{h_T}(X_m^p)$ is close to $f_{a_T}(X_m^p)$ while $f_{h_T}(X_m^n)$ and $f_{a_T}(X_m^n)$ are far away. That is, we expect the current positive bounding box sample x_m^p corresponding to $f_{h_T}(X_m^p)$ is similar to the predicted target in previous frames represented by $f_{a_T}(X_m^p)$, while we expect the current negative bounding box sample x_m^n corresponding to $f_{h_T}(X_m^n)$ is not similar to the predicted target in previous frames represented by $f_{a_T}(X_m^n)$. The loss $\mathcal{L}_a$ aims to decrease the positive class distance while ensuring the negative class margin. The inequality $||f_{h_T}(X_m^p) - f_{a_T}(X_m^p)||_2^2 + \tau < ||f_{h_T}(X_m^n) - f_{a_T}(X_m^n)||_2^2$ describes the temporal coherence. Furthermore, the loss $\mathcal{L}_a$ aims to ensure θ'_{T-i} and θ_{T-i} reduce the effect of noise for generating reliable a_T and $\widetilde{h}_T$. Overall, the purpose of the loss $\mathcal{L}_a$ is to improve the activation accuracy and facilitate the discriminative representation generation during the memory activation process. The loss function $\mathcal{L}_a$ is proposed to regularize the memory generator from the perspective of temporal coherence. We design another loss $\mathcal{L}_b$ to regularize the memory generator from the perspective of class distributions.

As $h_T^\star$ is the final temporal representation which serves as the input of our classifier, we expect the current and historical discriminative information is encoded into $h_T^\star$. Therefore, in a training batch, we use $\mathcal{L}_b$ to minimize the intra-class variations among all the positive sequence sample features while increasing the distances between the negative sequence sample features and the center of positive sequence sample

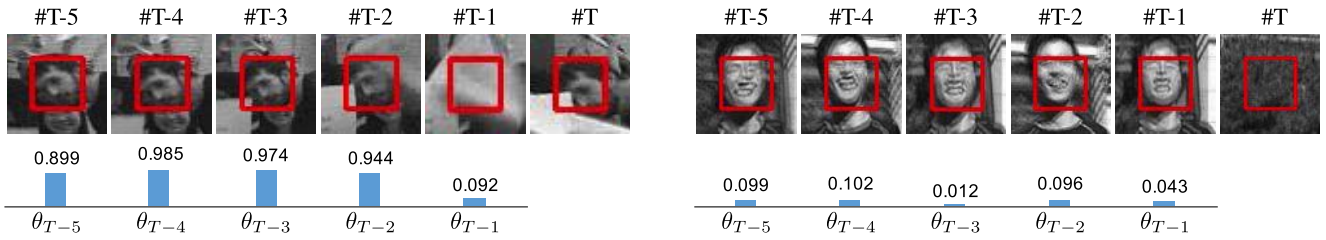

Fig. 3. Visualization of the activation weight θ_{T-i} assigned in the recurrent memory activation layer on the *Freeman4* (i.e., the left example) and *Jumping* (i.e., the right example) sequences [68].

features. The loss $\mathcal{L}_b$ is defined as follows:

$$\mathcal{L}_b = \frac{1}{N}\sum_{m=1}^{N}\frac{||f_{h_T^\star}(X_m^p)-\mu||_2^2}{||f_{h_T^\star}(X_m^n)-\mu)||_2^2}, \tag{6}$$

where $f_{h_T^\star}$ is the network from the input layer to the layer which outputs $h_T^\star$, $\mu = \frac{1}{N}\sum_{m=1}^{N} f_{h_T^\star}(X_m^p)$ is the center of positive sequence sample features in this batch.

Overall, we construct our loss function as follows:

$$\mathcal{L} = \mathcal{L}_c + \lambda_1\mathcal{L}_a + \lambda_2\mathcal{L}_b, \tag{7}$$

where λ_1 and λ_2 are two scalars balancing the three loss functions. The loss $\mathcal{L}_c$ is the cross-entropy loss. The loss $\lambda_1\mathcal{L}_a+\lambda_2\mathcal{L}_b$ is the proposed temporal coherence loss which is denoted as $\mathcal{L}_{tc}$ in Figure 2. We use Eq. 7 to train our RMAN. Note that our temporal coherence loss is proposed to help the memory generator differentiate the target and background. This loss does not affect the classifier during the training process as the classification loss is computed based on the classifier outputs and our temporal coherence loss is computed on the states $(a_T, h_T, h_T^\star)$.

C. Visualization

In this section, we show the effectiveness of our RMAN to record and activate memory states from historical target patches to facilitate the binary classification process. First, we visualize the memory activation weights to see how they are computed based on patch similarity. Second, we visualize the RMAN attention map to see how the proposed network trained by the temporal coherence loss increases the network attention around the target region. Third, we compare the feature distributions of the RMAN and the baseline tracking-by-detection framework by visualizing their features Pearson correlation coefficient (PCC) histograms.

1) Activation Weights: The recurrent memory activation layer reorganizes the historical memory states based on the current bounding box sample. Figure 3 shows two examples. When we predict the target location on the T-th frame of the *Freeman4* sequence [68] (i.e., the left example), we show the activation weights from the K historical patches. In Figure 3, the first row shows the patches and the second row shows the corresponding weights. The left example indicates that our recurrent memory activation layer assigns a low weight value to the memory state from the occluded patch (i.e., $(T-1)$-th), and selects other historical memory states to enrich the target representation. In contrast, when the input bounding box sample from the T-th frame of the *Jumping* sequence (i.e., the right example) only contains the background, our recurrent memory activation layer assigns low weight values to all historical memory states.

2) Network Attentions: We use the visualization method [56] to obtain network attention maps. Specifically, the attention map is obtained by taking the partial derivatives of the positive classification score with respect to the input bounding box patch. That is, an attention map A corresponding to an input patch x_0 can be denoted as:

$$A = \left.\frac{\partial f_p(x)}{\partial x}\right|_{x=x_0},$$

where f_p is the network from the input layer to the layer which outputs the positive classification score. We perform the back-propagation from the positive classification score to the input to obtain the attention map. Therefore, the resolution of the attention map is the same as the input bounding box patch. The pixel intensity of the attention map indicates the amount of contribution to the classification score. We use this visualization method to show the network attention maps of the baseline, the RMAN and the RMAN trained using the Temporal Coherence Loss (TCL). The baseline method is the original tracking-by-detection framework which consists of CNN layers and a fully connected classifier. The RMAN is constructed upon the baseline by adding an LSTM and a recurrent memory activation layer.

Figure 4 shows two example from the *Shaking* and *Singer2* sequences [68]. The input frames are shown in Figure 4(a), (e), (i) and (m) with ground truth annotations. At the beginning of the video sequences (i.e., Frame #2), the attention maps of all the networks are similar as shown from (b)-(d) and (j)-(l). When the target objects undergo severe appearance variations shown in (e) and (m), there are limited highlight regions shown in (f)-(g) and (n)-(o). It indicates that both the baseline and the RMAN without the TCL training are not able to fully exploit the target regions for binary classification. The regions they have utilized are not effective to represent the target objects. In contrast, the highlight regions almost cover the target regions shown in (h) and (p). It indicates that when the TCL is utilized during training, the RMAN fully exploits the target regions to generate hidden states similar as that shown in (d) and (l). Figure 4 illustrates that when the target objects encounter appearance variations, both the baseline framework and RMAN focus on the partial regions. They are not effective in capturing the invariant representations

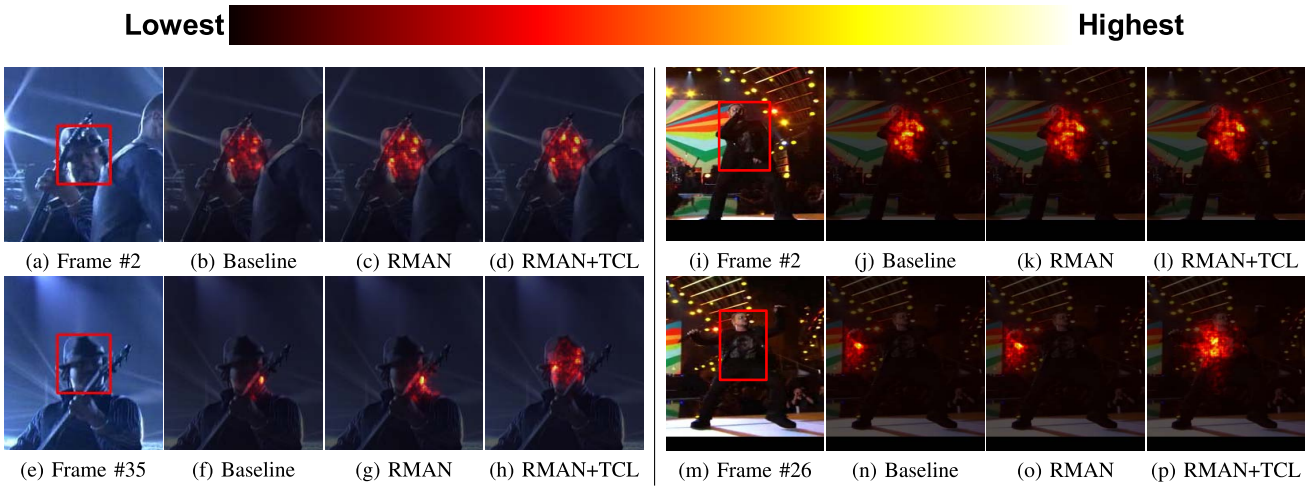

Fig. 4. Visualization of network attention maps on the *Shaking* (i.e., the left example) and *Singer2* (i.e., the right example) sequences [68]. We show where networks focus by highlighting the attentive region when they predict the target location. Our RMAN with TCL training is effective to utilize the whole target region for hidden states generation and activation. It performs favorably against the baseline framework and the RMAN without TCL training when the target object encounters large appearance variations.

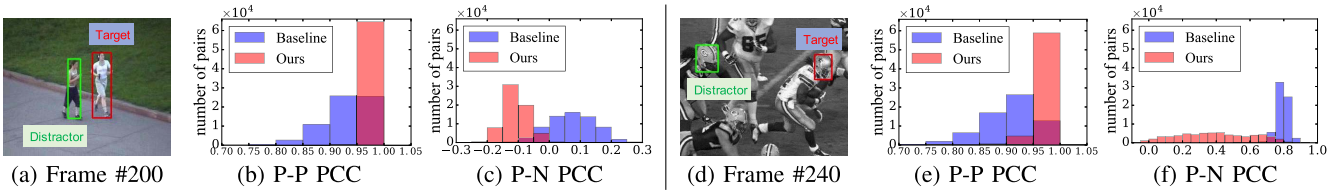

Fig. 5. Visualization of feature distributions through computing the Pearson correlation coefficient (PCC) histograms on the *Jogging* (i.e., the left example) and *Football* (i.e., the right example) sequences [68]. We extract hundreds of positive sample features around the target and negative sample features around the distractor. Then we compute the feature PCC values of each positive-positive (P-P) sample pair and each positive-negative (P-N) sample pair. All PCC values are collected to build the histograms.

of the target objects. After training the RMAN using the TCL, our memory generator and recurrent memory activation layer are able to focus on the whole target regions for hidden state recording and activation. It selectively activates memory states to enrich the representation of the target sample in the current frame, which facilitates the classification process.

3) Feature Distributions: In addition to target appearance variations, another challenging factor is the instance-level classification where there are similar instances around the target object. Figure 5 shows two example from the *Jogging* and *Football* sequences [68]. In Figure 5(a) and (d), there are similar instances moving side-by-side, and the targets are on the right. The classifier may not be able to differentiate these instances as two instances share similar appearances.

We use the Pearson correlation coefficient (PCC) [3] histogram to measure the feature distributions of the baseline and our method. We randomly draw hundreds of bounding box samples around the target as the positive samples and a similar amount of samples around the distractor as the negative samples. These samples are then passed into the two networks. We extract the input features corresponding to their classifiers and compute the feature PCC values of each positive-positive sample pair and each positive-negative sample pair. We collect these PCC values to generate PCC histograms as shown in Figure 5(b)-(c) and (e)-(f). In (b) and (e), we find that compared with the baseline, there is a strong correlation among positive sample features generated by our network. Furthermore, in (c) and (f), we observe that the positive sample features are less correlated to the negative sample features in our method than that in the baseline. Figure 5(b)-(c) and (e)-(f) indicate that our positive sample feature distribution is more compact than that of the baseline, and the distance between our positive and negative sample feature distributions is larger compared with that of the baseline. Our features facilitate the binary classification process by increasing the instance-awareness.

IV. Tracking Process

As our tracker does not require an offline training step, we present the tracking process through model initialization, online detection, and model update.

A. Model Initialization

In the first frame, we randomly draw N_1 bounding box samples around the target location and label each of them as either positive or negative according to their IOU ratios with the ground truth annotation. We use these bounding box samples to construct sequence samples to train the recurrent memory activation network with the temporal coherence loss.

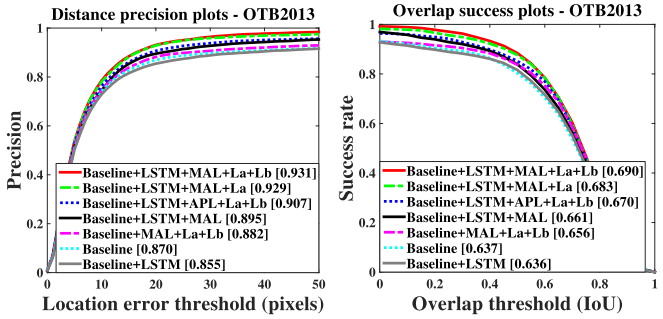

Fig. 6. Ablation studies with different modules and different losses using the one-pass evaluation on the OTB-2013 dataset.

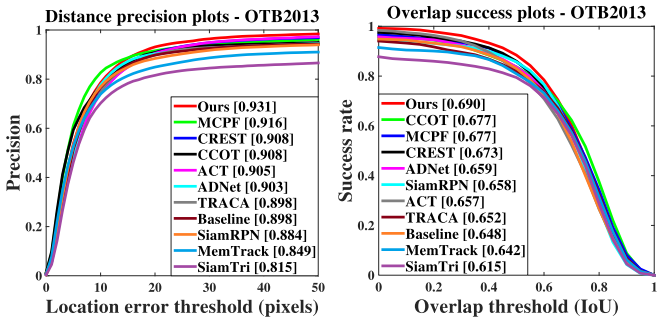

Fig. 7. Overall performance on the OTB-2013 dataset using the one-pass evaluation. Our method ranks first in both the distance precision and overlap success metrics.

Note that in the initialization step, all the input patches for the network are from the first frame.

B. Online Detection

When predicting the target location in any frame T in the tracking sequence, in the current frame we draw N_2 bounding box samples randomly around the target location predicted in the previous frame $T-1$. One of our candidates to the classifier is a sample sequence that consists of a bounding box sample in the current frame and K previously tracked target patches. We feed these candidates to our network which predicts the corresponding probabilities. We select the candidate with the highest positive classification probability and refine the target location using the bounding box regression method as in [19].

C. Model Update

After online detection, we first draw N_2 bounding box samples randomly around the predicted target location and label these bounding box samples with positive or negative labels based on their IOU scores with the predicted bounding box. Then these bounding box samples are combined with the predicted target patches to construct sequence samples. We collect sequence samples every N_3 frames and train the network via the proposed loss function.

V. Experiments

We first introduce the implementation details of our method and analyze the effect of each module in our model on performance improvement. Then we compare our tracker with the state-of-the-art methods on the OTB-2013 [68], OTB-2015 [69], VOT-2016 [35], Temple128 [38], and UAV123 [45] datasets.

A. Experimental Setup

We construct the CNN feature extraction module based on the VGG-M model [57]. At the online training stage, the CNN module is not updated, and the subsequent layers are updated along the tracking process. We set the number of hidden units in the LSTM as 512. That is, the dimensions of h_T are 512. The node connections in the two fully connected layers are set as 512×1024 and 2×512, respectively. The number of the predicted target patches in each sequence sample is set as $K = 5$. The threshold τ in Eq. 5 is set as 1, and the scalars λ_1 and λ_2 in Eq. 7 are set as 2 and 5e-4, respectively. The thresholds τ_1 and τ_2 are set to 0.7 and 0.3, respectively. We use the SGD to train the proposed model using a mini-batch of N=32 positive-negative sequence sample pairs. In the first frame, we use 50 iterations to initialize our network with a learning rate of 2e-4, the number N_1 of samples is set to 5500. For the online model update, we use a learning rate of 3e-4 to update the network every $N_3 = 10$ frames using 15 iterations. During online detection, the number N_2 of proposals is set to 256. Our tracker runs on a PC with an i7 3.4 GHz CPU and a GeForce GTX 1080 GPU. The average computational speed is 1 FPS. The implementation is based on pytorch [51].

B. Evaluation Metrics

On the OTB-2013, OTB-2015, Temple128, and UAV123 datasets, we use one-pass evaluation (OPE) with distance precision (DP) and overlap success (OS) metrics [68]. The threshold of the DP metric is set as 20 pixels (DP_{20}) to determine whether the target object is tracked or not. We report the area-under-the-curve scores of the OS metric plots (OS_{AUC}). In addition, we compute the OS rates at a threshold of 0.5 IOU ($OS_{0.5}$) and the center location error (CLE). On the VOT-2016 dataset, the performance evaluation is presented in terms of Expected Average Overlap (EAO), Accuracy ranks (Ar) and Robustness ranks (Rr).

C. Ablation Studies

We conduct ablation studies on the OTB-2013 dataset to validate the effectiveness of each module. We set an existing tracking-by-detection method as the baseline, which consists of the CNN feature extraction layers and fully connected layers. The baseline is online trained using the cross entropy loss $\mathcal{L}_c$. We add the LSTM layer into this baseline and denote this configuration as Baseline + LSTM. In the Baseline + LSTM configuration, the current hidden state h_T is used for classification and the cross-entropy loss $\mathcal{L}_c$ is also used to train the model. We use this configuration to evaluate whether directly applying the LSTM can capture the relevant information in the sequence data for tracking performance improvement. Next, we add the recurrent memory activation layer (MAL) into the Baseline + LSTM method and denote

TABLE I

COMPARISONS WITH THE STATE-OF-THE-ART TRACKERS ON THE OTB-2013 AND OTB-2015 DATASETS. OUR TRACKER PERFORMS FAVORABLY AGAINST EXISTING TRACKERS IN THE CENTER LOCATION ERROR (CLE), AND THE OVERLAP SUCCESS RATE AT A THRESHOLD OF 0.5 IOU ($OS_{0.5}$). RED: BEST. BLUE: SECOND BEST

	Trackers	Ours	CCOT	MCPF	TRACA	MemTrack	ACT	CREST	SiamRPN	SiamTri	ADNet	Baseline
CLE	OTB-2013	8.06	15.58	11.19	15.44	27.58	9.75	10.22	14.21	29.47	13.79	17.03
	OTB-2015	13.86	13.99	20.86	27.63	27.83	15.78	21.19	19.21	33.13	14.65	15.41
$OS_{0.5}$	OTB-2013	0.884	0.837	0.858	0.818	0.809	0.821	0.860	0.857	0.794	0.836	0.831
	OTB-2015	0.822	0.823	0.780	0.738	0.783	0.765	0.776	0.819	0.747	0.802	0.791

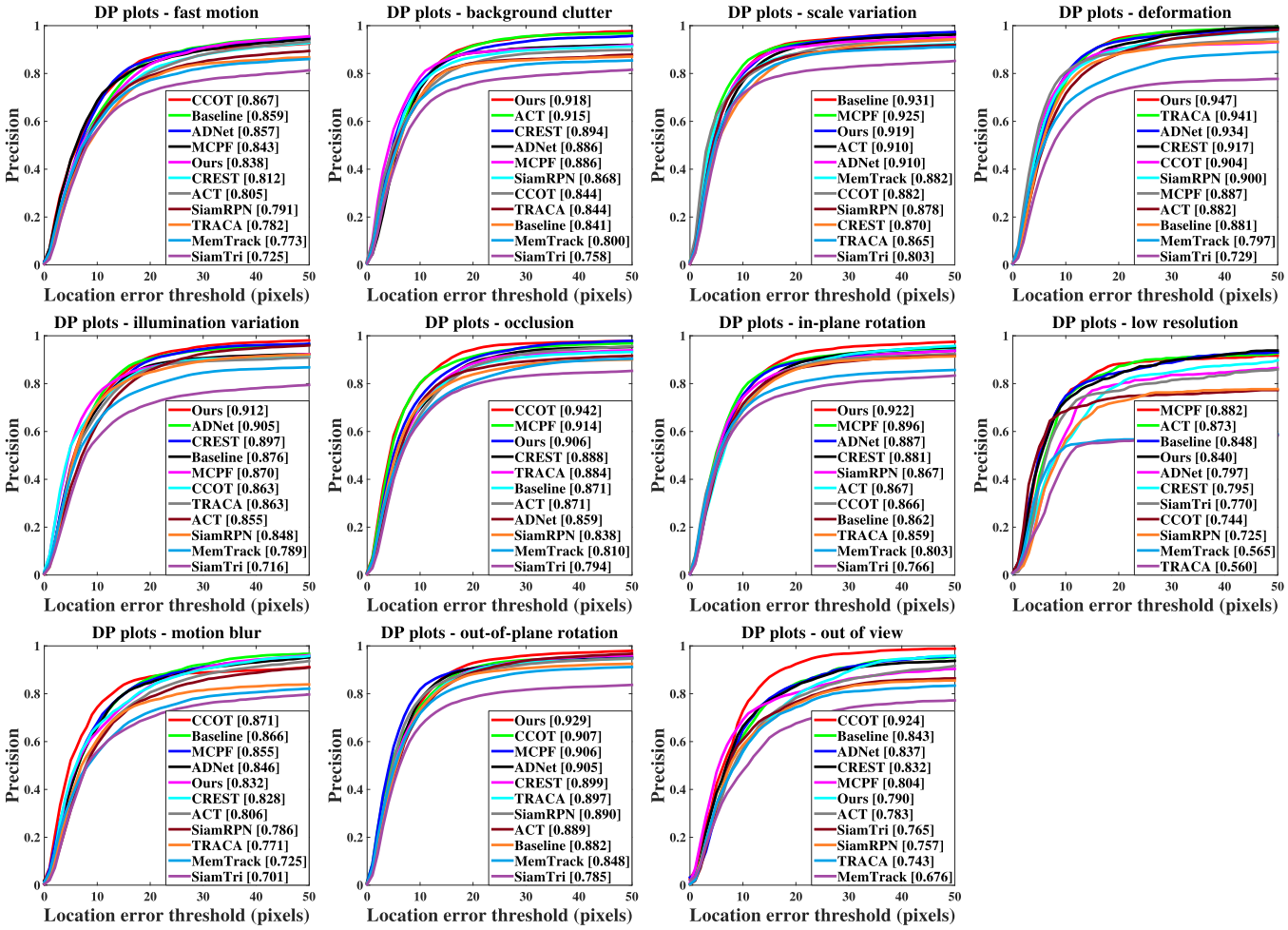

Fig. 8. Distance precision plots over 11 tracking video attributes, including fast motion, background clutter, scale variation, deformation, illumination variation, occlusion, in-plane rotation, low resolution, motion blur, out-of-plane rotation, and out-of-view.

this configuration as Baseline + LSTM + MAL. The configuration is used to evaluate the ability of the MAL to selectively activate historical memory states and generate enriched temporal representations to improve tracking performance. Finally, we train the Baseline + LSTM + MAL model using different loss functions $\mathcal{L}_c + \mathcal{L}_a$ and $\mathcal{L}_c + \mathcal{L}_a + \mathcal{L}_b$, respectively. It aims to demonstrate that the temporal coherence loss gradually helps the memory generator differentiate the target and background to facilitate classification. In addition, we compare the Baseline + LSTM + MAL + $\mathcal{L}_a + \mathcal{L}_b$ method with the Baseline + MAL + $\mathcal{L}_a + \mathcal{L}_b$ model to evaluate the ability of the LSTM layer to model temporal relationships in our method. We use an average pooling layer (APL) instead of the MAL to reorganize the memory states. This configuration aims to present that the MAL reduce the effect of noise is beneficial to tracking performance improvement.

Figure 6 shows the performance of our method with different configurations. The method by directly applying the LSTM layer to deep models for tracking does not perform well (Baseline + LSTM). It is because of inaccurate and occluded detections cause that noise exists in target sequences, which negatively affect the LSTM layer to model temporal relationships. The effect of the noise is reduced when we add the MAL into the Baseline + LSTM configuration. Meanwhile,

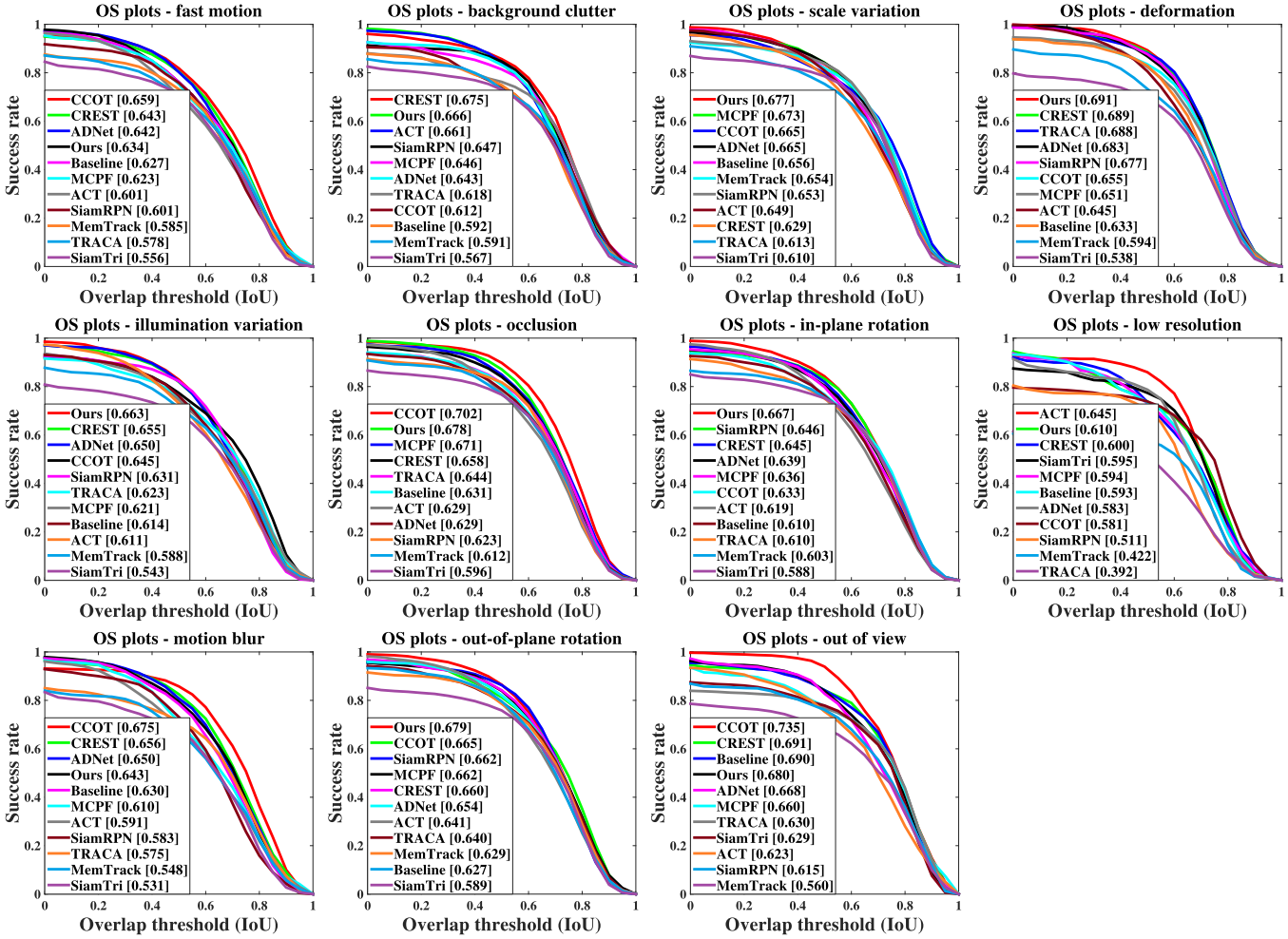

Fig. 9. Overlap success plots over 11 tracking video attributes, including fast motion, background clutter, scale variation, deformation, illumination variation, occlusion, in-plane rotation, low resolution, motion blur, out-of-plane rotation, and out-of-view.

the MAL encodes historical discriminative information into the current representation to facilitate the classification process. Figure 6 shows that the Baseline + LSTM + MAL configuration improves the tracking performance significantly. Our temporal coherence loss gradually improves the Baseline + LSTM + MAL through incorporating $\mathcal{L}_a$ and $\mathcal{L}_b$. It indicates that the proposed loss function is effective to facilitate the activation accuracy and discriminative temporal representations generation in the memory activation process. By integrating the recurrent memory activation layer and temporal coherence loss into the Baseline + LSTM scheme, we activate the ability of the LSTM layer to capture the relevant information. Compared with the method (Baseline + MAL + $\mathcal{L}_a$ + $\mathcal{L}_b$) in Figure 6, our algorithm with the LSTM layer improves the tracking performance significantly. In addition, Figure 6 shows that our scheme (Baseline + LSTM + MAL + $\mathcal{L}_a$ + $\mathcal{L}_b$) significantly outperforms the Baseline + LSTM + APL + $\mathcal{L}_a$ + $\mathcal{L}_b$ method which fails to reduce the effect of noise. Overall, the results in Figure 6 show that the proposed recurrent memory activation network and temporal coherence loss are effective for improving the original tracking-by-detection framework.

D. OTB-2013 Dataset

We evlaute the proposed tracker against the state-of-the-art methods on the OTB-2013 benchmark dataset. These methods include CCOT [14], MCPF [75], CREST [59], ADNet [73], ACT [7], TRACA [8], SiamRPN [36], MemTrack [71], and SiamTri [16]. Figure 7 shows that our tracker performs well against the state-of-the-art approaches on the dataset with the distance precision and overlap success metrics. The legends of the figure contain the DP_{20} scores and the OS_{AUC} scores. The baseline method as a representative tracking-by-detection method is also shown in Figure 7. It can be treated as the MDNet [46] method, but for fair comparisons, it does not use tracking videos to conduct offline training. Our tracker improves the representative tracking-by-detection method by a large margin. Table I shows the tracking results in terms of CLE and $\text{OS}_{0.5}$ on the OTB-2013 dataset. These results demonstrate our method is effective to reduce the average center distance error and increase the tracking success rates. We attribute the effectiveness of our method to that the temporal coherent modeling within our method is beneficial to feature representation against various appearance changes. In Figure 8 and 9, we show the tracking performance under

TABLE II
COMPARISONS OF COMPUTATIONAL SPEED BETWEEN OUR TRACKER WITH THE STATE-OF-THE-ART METHODS ON THE OTB-2013 DATASET

	Ours	CCOT	MCPF	CREST	ADNet	ACT	MemTrack	TRACA	SiamRPN	SiamTri
FPS	1	0.3	0.58	2.4	2.9	30	50	65	71	86.3

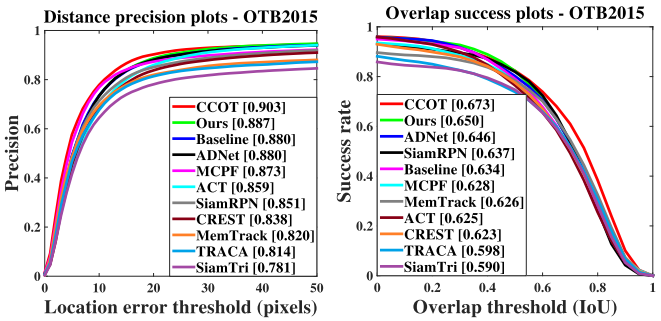

Fig. 10. Overall performance on the OTB-2015 dataset using the one-pass evaluation. Our method ranks second in both the distance precision and overlap success metrics.

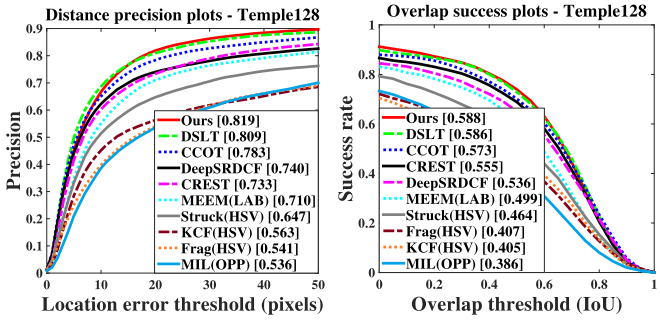

Fig. 11. Overall performance on the Temple128 dataset using the one-pass evaluation. Our method ranks first in both the distance precision and overlap success metrics.

different video attributes which include various appearance changes. The distance precision and overlap success plots with the one-pass evaluation are used in these experiments. When there are severe appearance variances for the target object (i.e., deformation, illumination variation, in-plane rotation, out-of-plane rotation, and occlusion), our tracker performs better than the original tracking-by-detection tracker [46]. Existing trackers mainly focus on appearance modeling in single frames and ignore exploit temporal coherence as an additional clue to adapt to appearance variations. These experimental results demonstrate the effectiveness of our coherent modeling on handling appearance variations and the favorable performance of our method against the state-of-the-art trackers. Table II shows the comparisons of computational speeds between our tracker with the state-of-the-art methods. The average speed of our proposed method is 1FPS which is comparable to the non-real-time trackers (e.g., CCOT, MCPF, CREST, and ADNet). The speed bottleneck of our method is the online sampling strategy of the base framework. The base framework draws samples from the input video frame and takes a lot of time to compute CNN features of samples. Our proposed modules with few parameters bring little additional time cost for the base framework. An alternative sampling strategy to improve the average speed of the base framework is the ROI align method [32]. As the ROI align method directly extract CNN features of samples from the CNN features of the input video frame, it reduces time cost in feature extraction. We will use the ROI align method and engineering acceleration in the future work.

E. OTB-2015 Dataset

We evaluate our tracker on the OTB-2015 benchmark against the aforementioned trackers. We show the results using one-pass evaluation with distance precision and overlap success metrics in Figure 10. Our tracker achieves 88.7% and 65.0% on the DP_{20} and the OS_{AUC}, which ranks second in the two metrics among these recent state-of-the-art trackers. While the CCOT tracker performs better than our method when using the DP_{20} metric, Table I shows that our approach has a smaller center location error on all the benchmark sequences than the CCOT method. These results can be explained by the fact that we incorporate temporal coherence to facilitate visual tracking. The evaluated methods do not explicitly exploit temporal coherence to strengthen representation. Our tracker does not perform as well as the CCOT method in the overlap success rate on this dataset. The reason can be attributed to that the CCOT method crops the samples in a continuous space for scale estimation. In the tracking-by-detection framework, scale estimation is mainly affected by proposal sampling and bounding box regression. In the proposed method, we randomly draw a sparse set of samples and use the bounding box regression method as in [19] for scale estimation. We do not use the dense sampling method with numerous scales as adopted in the CCOT method. The reason is that adopting dense sampling in the two-stage tracking-by-detection framework exacerbates the class imbalance between positive and negative samples. In addition, dense sampling heavily downgrades the computational speed of our model. An alternative strategy to improve scale estimation is to use the IOU scores generated by the IOU-Net [30] for fine-tuning image patches based on bounding boxes. The IOU-Net improves significantly in the detection task. We will discuss the temporal coherence modeling in tracking, and use the IOU-Net to further improve the scale estimation module in the future work.

F. VOT-2016 Dataset

We compare our tracker with the state-of-the-art trackers on the VOT-2016 benchmark. These trackers include Staple [4], MDNet [46], CCOT [14], EBT [77], DeepSRDCF (DSR) [12],

TABLE III

COMPARISONS WITH THE STATE-OF-THE-ART TRACKERS ON THE VOT-2016 DATASET. THE RESULTS ARE PRESENTED IN TERMS OF EXPECTED AVERAGE OVERLAP (EAO), ACCURACY RANK (AR) AND ROBUSTNESS RANK (RR). RED: BEST. BLUE: SECOND BEST

	Ours	CCOT	Staple	MDNet	EBT	DSR	SiamFC
EAO	0.2965	0.3310	0.2952	0.2572	0.2913	0.2763	0.2766
Ar	1.82	1.95	1.88	1.68	3.65	2.02	1.30
Rr	2.37	1.98	3.25	2.83	2.15	2.77	3.18

TABLE IV

COMPARISONS WITH THE STATE-OF-THE-ART TRACKERS ON THE TEMPLE128 DATASET. OUR TRACKER PERFORMS FAVORABLY AGAINST EXISTING TRACKERS IN THE CENTER LOCATION ERROR (CLE), AND THE OVERLAP SUCCESS RATE AT A THRESHOLD OF 0.5 IOU ($OS_{0.5}$). RED: BEST. BLUE: SECOND BEST

	Ours	CCOT	DSR	CREST	DSLT	MEEM	Struck	KCF	Frag	MIL
CLE	20.23	33.33	41.07	32.26	25.62	42.60	50.70	66.26	59.71	62.81
$OS_{0.5}$	0.740	0.702	0.652	0.689	0.729	0.615	0.545	0.480	0.469	0.428

TABLE V

COMPARISONS WITH THE STATE-OF-THE-ART TRACKERS ON THE UAV123 DATASET. OUR TRACKER PERFORMS FAVORABLY AGAINST EXISTING TRACKERS IN THE AVERAGE DISTANCE PRECISION SCORES AT A THRESHOLD OF 20 PIXELS (DP_{20}), AND THE AREA-UNDER-THE-CURVE SCORES OF THE OVERLAP SUCCESS METRIC PLOTS (OS_{AUC}).: BEST. : SECOND BEST

	Ours	CCOT	DSLT	SRDCF	MEEM	SAMF	MUST	DSST	Struck	ASLA
DP_{20}	0.757	-	0.746	0.676	0.627	0.592	0.591	0.586	0.578	0.571
OS_{AUC}	0.525	0.517	0.530	0.464	0.392	0.396	0.391	0.356	0.381	0.407

and SiamFC [5]. The state-of-the-art EAO result in the VOT-2016 [35] report is 0.251. As shown in Table III, the CCOT achieves the best results under the EAO metric, followed by our method which is better than the Staple, EBT, DeepSRDCF, and SiamFC. In addition, the proposed tracker performs better than the MDNet method in a large margin under the EAO metric. To further analyze the performances of our tracker and the CCOT method, we add more experiments of our method and the CCOT method on the UAV123 and Temple128 datasets.

G. Temple128 Dataset

Figure 11 and Table IV shows the comparison with the state-of-the-art trackers on the Temple128 dataset. These trackers include CCOT [14], DSLT [40], CREST [59], DeepSRDCF (DSR) [12], MEEM [74], Struck [23], KCF [25], Frag [1], and MIL [2]. Our method achieves the best DP_{20}, OS_{AUC}, CLE and $OS_{0.5}$ scores among these trackers. Compared to the CCOT method on this dataset, our tracker performs well in the distance precision metric (81.9% vs. 78.3%), and has a smaller center location error (20.23 vs. 33.33). In addition, our tracker performs better than the CCOT approach whether in OS_{AUC} (58.8% vs. 57.3%) or in $OS_{0.5}$ (74.0% vs. 70.2%). This demonstrates that our method performs favorably against the CCOT method.

H. UAV123 Dataset

We evaluate our method on the UAV123 dataset with the comparison to representative trackers including the CCOT [14], DSLT [40], SRDCF [13], MEEM [74], SAMF [37], MUST [27], DSST [11], Struck [23], and ASLA [29] methods. As some recent trackers do not release the raw outputs on the UAV123 dataset, we use the reported results for fair comparisons, and we only present the evaluations in terms of DP_{20} and OS_{AUC}. Table V shows that our method performs well against the state-of-the-art trackers. The experimental results with respect to the CCOT method show the improvements and generalization by modeling temporal coherence.

I. Qualitative Evaluation

Figure 12 shows the qualitative evaluation of our tracker and the representative trackers, including CCOT [14], MemTrack [71], SiamRPN [36], and the baseline method [46] on 12 challenging sequences. The baseline method fails to locate the target objects undergoing severe appearance variations. The limitations include deformation (*Basketball*), background clutter (*Football*), and partial occlusion (*Skating2-1*), which cause drifting upon localizing the target object. The CCOT tracker is effective to estimate scale but shows limited performance under the precision metric. When target objects undergo deformation (*Diving*), partial occlusion (*Freeman4*), and rotation (*MotorRolling*), the CCOT method does not perform well. The MemTrack and SiamRPN methods are based on the Siamese framework. The proposal representations are from independent frames and do not incorporate temporal coherence as additional tracking clues. When distractors appear in the background as shown in the Girl2, Human4, Human6, Soccer, CarDark, and SUV sequences, the Siamese based methods fail to locate targets. Despite the MemTrack also employed LSTM, this method use an LSTM layer to control reading and writing memory. We propose an LSTM layer to capture the

Fig. 12. Qualitative evaluation of our tracker, CCOT [14], MemTrack [71], SiamRPN [36], and Baseline [46] on 12 challenging sequences (from left to right and top to down: *Basketball, Football, Skating2-1, Diving, Freeman4, MotorRolling, Girl2, Human4, Human6, Soccer, CarDark,* and *SUV*, respectively). Our method performs favorably against the state-of-the-art trackers.

relevant information in the sequence data for discriminative representations generation. Our temporal modeling using the memory activation layer and the temporal coherence loss facilities the classification process in the temporal span. Overall, our algorithm performs favorably against the state-of-the-art trackers.

VI. Conclusion

In this paper, we propose a recurrent memory activation network to exploit temporal coherence within the tracking-by-detection framework. The network generates memory states of targets objects from input sequences. The memory states are then activated to generate a historical discriminative representation which is encoded into the current representation to facilitate classification. The proposed temporal coherence loss helps the memory generator of our network differentiate the target and background. It coupled with the classification loss ensures the coherent modeling benefits visual tracking. Extensive experimental results on the benchmarks demonstrate the effectiveness and robustness of our tracker against the state-of-the-art tracking methods.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 1, Jun. 2006, pp. 798–805.

[2] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1619–1632, Aug. 2011.

[3] J. Benesty, J. Chen, Y. Huang, and I. Cohen, "Pearson correlation coefficient," in *Noise Reduction in Speech Processing*. Berlin, Germany: Springer, 2009.

[4] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. Torr, "Staple: Complementary learners for real-time tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1401–1409.

[5] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 850–865.

[6] D. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010.

[7] B. Chen, D. Wang, P. Li, S. Wang, and H. Lu, "Real-time 'actor-critic' tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 318–334.

[8] J. Choi *et al.*, "Context-aware deep feature compression for high-speed visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 479–488.

[9] I. Cohen and G. Medioni, "Detecting and tracking moving objects for video surveillance," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 1999, pp. 319–325.

[10] A. I. Comport, E. Marchand, M. Pressigout, and F. Chaumette, "Real-time markerless tracking for augmented reality: The virtual visual servoing framework," *IEEE Trans. Vis. Comput. Graphics*, vol. 12, no. 4, pp. 615–628, Jul./Aug. 2006.

[11] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *Proc. Brit. Mach. Vis. Conf.*, 2014, pp. 1–11.

[12] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. Workshop (ICCVW)*, Dec. 2015, pp. 58–66.

[13] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4310–4318.

[14] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 472–488.

[15] J. Donahue *et al.*, "Long-term recurrent convolutional networks for visual recognition and description," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 2625–2634.

[16] X. Dong and J. Shen, "Triplet loss in siamese network for object tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 459–474.

[17] Q. Gan, Q. Guo, Z. Zhang, and K. Cho, "First step toward model-free, anonymous object tracking with recurrent neural networks," 2015, *arXiv:1511.06425*. [Online]. Available: http://arxiv.org/abs/1511.06425

[18] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361.

[19] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[20] Y. Goldberg, "A primer on neural network models for natural language processing," *J. Artif. Intell. Res.*, vol. 57, pp. 345–420, Nov. 2016.

[21] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," in *Proc. Brit. Mach. Vis. Conf.*, 2006, p. 6.

[22] B. Han, J. Sim, and H. Adam, "Branchout: Regularization for online ensemble tracking with convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 521–530.

[23] S. Hare *et al.*, "Struck: Structured output tracking with kernels," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 10, pp. 2096–2109, Oct. 2016.

[24] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2012, pp. 702–715.

[25] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, Mar. 2015.

[26] S. Hong, T. You, S. Kwak, and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2015, pp. 597–606.

[27] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao, "MUlti-store tracker (MUSTer): A cognitive psychology inspired approach to object tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 749–758.

[28] S. Jia, C. Ma, Y. Song, and X. Yang, "Robust tracking against adversarial attacks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2020, pp. 69–84.

[29] X. Jia, H. Lu, and M.-H. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1822–1829.

[30] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang, "Acquisition of localization confidence for accurate object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 784–799.

[31] S. E. Kahou, V. Michalski, and R. Memisevic, "RATM: Recurrent attentive tracking model," 2015, *arXiv:1510.08660*. [Online]. Available: http://arxiv.org/abs/1510.08660

[32] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 386–397, Feb. 2020.

[33] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1409–1422, Jul. 2012.

[34] H. K. Galoogahi, A. Fagg, and S. Lucey, "Learning background-aware correlation filters for visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Oct. 2017, pp. 1135–1143.

[35] M. Kristan, R. Bowden, and K. Lebeda, "The visual object tracking VOT2016 challenge results," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 777–823.

[36] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 8971–8980.

[37] Y. Li and J. Zhu, "A scale adaptive kernel correlation filter tracker with feature integration," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 254–265.

[38] P. Liang, E. Blasch, and H. Ling, "Encoding color information for visual tracking: Algorithms and benchmark," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5630–5644, Dec. 2015.

[39] S. Liu, J. Pan, and M.-H. Yang, "Learning recursive filters for low-level vision via a hybrid neural network," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 560–576.

[40] X. Lu, C. Ma, B. Ni, X. Yang, I. Reid, and M.-H. Yang, "Deep regression tracking with shrinkage loss," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 353–369.

[41] Y. Lu, C. Lu, and C.-K. Tang, "Online video object detection using association LSTM," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2344–2352.

[42] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3074–3082.

[43] N. McLaughlin, J. M. del Rincon, and P. Miller, "Recurrent convolutional network for video-based person re-identification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1325–1334.

[44] A. Milan, S. H. Rezatofighi, A. R. Dick, I. D. Reid, and K. Schindler, "Online multi-target tracking using recurrent neural networks," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 4225–4232.

[45] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for uav tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 445–461.

[46] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4293–43202.

[47] G. Ning *et al.*, "Spatially supervised recurrent convolutional neural networks for visual object tracking," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4.

[48] J. Ning, J. Yang, S. Jiang, L. Zhang, and M.-H. Yang, "Object tracking via dual linear structured SVM and explicit feature map," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4266–4274.

[49] C. Olah. (2015). *Understanding LSTM Networks*. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[50] P. Ondruska and I. Posner, "Deep tracking: Seeing beyond seeing using recurrent neural networks," 2016, *arXiv:1602.00991*. [Online]. Available: http://arxiv.org/abs/1602.00991

[51] A. Paszke *et al.*, "Automatic differentiation in pytorch," in *Proc. NIPS Autodiff Workshop*, Long Beach, CA, USA, 2017.

[52] S. Pu, Y. Song, C. Ma, H. Zhang, and M.-H. Yang, "Deep attentive tracking via reciprocative learning," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2018, pp. 1931–1941.

[53] Y. Qi *et al.*, "Hedged deep tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4303–4311.

[54] S. Salti, A. Cavallaro, and L. Di Stefano, "Adaptive appearance modeling for video tracking: Survey and evaluation," *IEEE Trans. Image Process.*, vol. 21, no. 10, pp. 4334–4348, Oct. 2012.

[55] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 815–823.

[56] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2013, *arXiv:1312.6034*. [Online]. Available: http://arxiv.org/abs/1312.6034

[57] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556

[58] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1442–1468, Jul. 2014.

[59] Y. Song, C. Ma, L. Gong, J. Zhang, R. W. H. Lau, and M.-H. Yang, "CREST: Convolutional residual learning for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2555–2564.

[60] Y. Song *et al.*, "VITAL: VIsual tracking via adversarial learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8990–8999.

[61] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2014, pp. 3104–3112.

[62] S. Tripathi, Z. C. Lipton, S. Belongie, and T. Nguyen, "Context matters: Refining object detection in video with recurrent neural networks," in *Proc. BMVC*, 2016, pp. 1–12.

[63] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. S. Torr, "End-to-end representation learning for correlation filter based tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2805–2813.
[64] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3156–3164.
[65] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Visual tracking with fully convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3119–3127.
[66] N. Wang, Y. Song, C. Ma, W. Zhou, W. Liu, and H. Li, "Unsupervised deep tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1308–1317.
[67] N. Wang *et al.*, "Unsupervised deep representation learning for real-time tracking," *Int. J. Comput. Vis.*, 2020, doi: 10.1007/s11263-020-01357-4.
[68] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2411–2418.
[69] Y. Wu, J. Lim, and M. H. Yang, "Object tracking benchmark," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1834–1848, Sep. 2015, doi: 10.1109/TPAMI.2014.2388226.
[70] T. Yang and A. B. Chan, "Recurrent filter learning for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2017, pp. 2010–2019.
[71] T. Yang and A. B. Chan, "Learning dynamic memory networks for object tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 152–167.
[72] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Comput. Surv.*, vol. 38, no. 4, p. 13, 2006.
[73] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Y. Choi, "Action-decision networks for visual tracking with deep reinforcement learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2711–2720.
[74] J. Zhang, S. Ma, and S. Sclaroff, "MEEM: Robust tracking via multiple experts using entropy minimization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 188–203.
[75] T. Zhang, C. Xu, and M.-H. Yang, "Multi-task correlation particle filter for robust object tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4335–4343.
[76] Z. Zhou, Y. Huang, W. Wang, L. Wang, and T. Tan, "See the forest for the trees: Joint spatial and temporal recurrent neural networks for video-based person re-identification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4747–4756.
[77] G. Zhu, F. Porikli, and H. Li, "Beyond local search: Tracking objects everywhere with instance-specific proposals," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 943–951.

Shi Pu (Member, IEEE) received the Ph.D. degree from the Beijing University of Posts and Telecommunications in 2020. He was sponsored by China Scholarship Council as a Visiting Ph.D. Student with the University of California Merced from the fall of 2017 to the fall of 2018. He is currently working with Tencent. His research interests include computer vision, particularly focuses on visual tracking and video classification.

Yibing Song (Member, IEEE) received the bachelor's degree from the University of Science and Technology of China in 2011 and the M.Phil. and Ph.D. degrees from the City University of Hong Kong in 2014 and 2018, respectively. During graduate study, he has interned in Adobe Research, San Jose, CA, USA, and visited UC Merced. He is currently a Research with Tencent AI Lab. His research interests include computer vision, particularly focuses on visual recognition and visual generation.

Chao Ma (Member, IEEE) received the Ph.D. degree from Shanghai Jiao Tong University in 2016. He was sponsored by China Scholarship Council as a Visiting Ph.D. Student with the University of California Merced from 2013 to 2015. He was a Senior Research Associate with the Australian Center of Robotic Vision, The University of Adelaide, from 2016 to 2018. He has been an Assistant Professor with Shanghai Jiao Tong University since 2019. His research interests include computer vision and machine learning.

Honggang Zhang (Senior Member, IEEE) received the B.S. degree from the Department of Electrical Engineering, Shandong University, in 1996, and the master's and Ph.D. degrees from the School of Information Engineering, Beijing University of Posts and Telecommunications (BUPT), in 1999 and 2003, respectively. He worked as a Visiting Scholar with the School of Computer Science, Carnegie Mellon University (CMU), from 2007 to 2008. He is currently an Associate Professor and the Director of the Web Search Center, BUPT. He has published more than 130 articles on IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (TPAMI), IEEE TRANSACTIONS ON IMAGE PROCESSING (TIP), *Science*, CVPR, ICCV, and NeurIPS. His research interests include image retrieval, computer vision, and pattern recognition.

Ming-Hsuan Yang (Fellow, IEEE) received the Ph.D. degree in computer science from the University of Illinois at Urbana-Champaign in 2000. He is currently a Professor of electrical engineering and computer science with the University of California Merced. He received the Google Faculty Award in 2009 and the Distinguished Early Career Research Award from the UC Merced Senate in 2011. He was a recipient of the Faculty Early Career Development (CAREER) Award from the National Science Foundation in 2012. In 2015, he received the Distinguished Research Award from the UC Merced Senate. He has served as an Associate Editor for the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE from 2007 to 2011. He is also an Associate Editor of the *International Journal of Computer Vision*, *Computer Vision and Image Understanding*, *Image and Vision Computing*, and *Journal of Artificial Intelligence Research*.