Real-Time Generic Object Tracking via Recurrent Regression Network

Rui CHEN†, Ying TONG†(a), Nonmembers, and Ruiyu LIANG†, Member

SUMMARY Deep neural networks have achieved great success in visual tracking by learning a generic representation and leveraging large amounts of training data to improve performance. Most generic object trackers are trained from scratch online and do not benefit from a large number of videos available for offline training. We present a real-time generic object tracker capable of incorporating temporal information into its model, learning from many examples offline and quickly updating online. During the training process, the pre-trained weight of convolution layer is updated lagging behind, and the input video sequence length is gradually increased for fast convergence. Furthermore, only the hidden states in recurrent network are updated to guarantee the real-time tracking speed. The experimental results show that the proposed tracking method is capable of tracking objects at 150 fps with higher predicting overlap rate, and achieves more robustness in multiple benchmarks than state-of-the-art performance.

key words: object tracking, deep learning, convolutional neural network, features comparison, recurrent neural network

1. Introduction

Object tracking plays an important role in computer vision and has a variety of applications such as video surveillance, missile guidance, UAV (Unmanned Aerial Vehicle) tracking, robotics, and so forth. In recent years, the Convolutional Neural Networks (CNNs), which demonstrate the powerfulness in extracting deep-level feature representation, have drawn extensive attention and brought great progress in object tracking. Many CNN-based trackers have been proposed, such as MDNet [1], STCT [2], SINT [3], SiamFC [4], GOTURN [5], TRACA [6], ADNet [7], SANet [8], and etc. The success of these trackers largely depends on the capability of CNN to learn a good feature representation for the object. To predict the object location in a new frame, either MDNet [1] applies a search-and-classify approach, or GOTURN [5] and SiamFC [4] apply crop-and-regress approach. SiamFC [4] converts the tracking problem to a similarity learning problem for the appearance of the target, adopting the siamese fully convolutional neural network to extract the features of the template image and the image to be tracked. Then a two-dimensional scoring map corresponding to the position of the original map is obtained by cross-correlation operation based on the feature maps. The location of the maximum value corresponds to the predicting center position of the original image. GOTURN [5] considers the tracking problem as a regression problem adopting two CNN branches. But the inputs of the two branches are the cutting area of the target in the previous frame and the same area in the current frame, respectively. The depth regression information of the features is extracted by the fully connected layer, and the predicted target location information is output by the last fully connected layer.

Because the tracking object has strong temporal correlation in successive video frames, so the tracking algorithm should be designed for the invariant characteristics of data transmission over time series. Both SiamFC [4] and GOTURN [5] can compare the different appearance features of the object between two sequential frames, lacking of the online update mechanism. Recurrent Neural Network (RNN) can transfer the characteristics of the current samples to the processing procedure of the next or even future samples, so it is suitable for processing time series of different lengths. Furthermore, the internal state transfer mechanism in RNN can solve the problem of online update in object tracking to ensure the real-time tracking. Re3 [9] network combines CNN and RNN successfully, but the redundancy and complexity of CNN output features limits the object resolution of RNN.

In this paper, inspired by the successful works that have applied RNN on computer vision tasks such as video classification and visual attention, we explore and investigate a more general strategy to develop a novel visual tracking approach based on RNN. The major intuition behind our method is that the historical visual semantics and tracking proposals encode pertinent information for future predictions and can be modeled as a recurrent convolutional network. However, unlike video classification or visual attention where only high-level semantic or single-step predictions are needed, visual tracking requires continuous and accurate predictions in both spatial and temporal domain over a long period of time, and thus, requires a novel network architecture design as well as proper training algorithms. So, we use the siamese CNN structure to extract features, add fully connected network for feature comparison, and adopt two-layer RNN to capture complex object transformations and remember longer term relationships. The training method of RNN translates the image embedding into an output bounding box while simultaneously updating the internal appearance and motion model.
2. Related Work

Object tracking is a fundamental problem in computer vision and it has been studied in great depth for decades. Single-object tracking problem can be described as: given some object of interest marked in one frame of a video, the goal of single-object tracking is to locate this object in subsequent video frames, despite object movement, changes in viewpoint, or other incidental environmental variations such as lighting changes and occlusions. Many single-object trackers focus on known object types or specific object instances, such as boxes, hands, people, cars, and etc. On the other hand, in many scenarios it is not feasible to pre-specify the kind of objects need to be tracked. Some trackers focus on generic object tracking which can be described as: given a bounding box around an arbitrary object at time $t$, produce bounding boxes for the object in all future frames [9]. In this paper, we focus on generic object tracking method which predominantly rely on learning a tracker online. Tracking-by-detection is a popular paradigm for tracking algorithms and can be described as: training an object-specific detector, and updating it with the object’s new appearance at every frame. It has a disadvantage that updating the tracker usually requires a significant amount of time and computational resources.

Despite the success of traditional trackers based on low-level, hand-crafted features, models based on deep convolutional neural network have dominated recent object tracking researches. In 2015, MDNET [1] won the Visual Object Tracking (VOT) challenge. The VOT reports present a succinct overview of many visual trackers. Those most related to ours can be categorized into four sub-groups as follows.

2.1 Tracking Model Based on Transfer Learning and Hierarchical Feature Attributes

For VGGNet, Wang L et al. [10] and Ma Chao et al. [11] find that convolutional layers in different levels characterize the target object from different perspectives. In [10], the feature map of conv4-3 and conv5-4 are analyzed and extracted, then the feature map selection network and heat map prediction network are constructed to prevent tracking drift. Similarly, the feature map of conv3-4, conv4-4 and conv5-4 in VGG-19 are extracted as target representations [11]. By means of transfer learning, Martin et al. activated discriminant correlation filter with CNN convolution layer to improve tracking accuracy [12]. Cui et al. [13] applied the confidence map produced by a RNN model to correlation filter and design a coarse-to-fine tracker. The RNN model is trained from four different directions, which makes the appearance model robust to partial occlusion.

2.2 Tracking Model Combining Traditional and Deep Learning Methods

Considering that the tracking result of a single tracker may be unstable, the discriminant ability and stability of the model can be improved by combining deep learning with traditional ensemble learning method and combining multiple weak classifiers into a strong classifier in the way of adaptive weighted combination. By learning multiple correlation filters on the convolution feature of hierarchical depth, multiple weak trackers were fused into one strong trackers in [14]. Wang et al. [15] presented a RNN-based method to produce a confidence map and applied it as a penalty to solve the occlusion problem. The MDNet [1] is a multi-domain framework based on CNN. By using a large set of tracking videos with ground truth, it pre-trains a CNN model with shared layers and multi-branches of domain-specific layers to obtain a generic object representation. During tracking process, the MDNet constructs a new network to capture the appearance change of the object and uses on-line fine-tuning to solve target deformation, occlusion, illumination change and so on. Ling Haibin et al. [16] used cyclic neural network to model the structure of the object, and achieved good tracking results on OTB100.

2.3 Tracking Model Based on Similarity Contrast Regression

Different from the above CNN-based trackers, Tao R. et al. [3] propose a siamese network model to match the object template and candidates for visual tracking. By extracting a large number of external video sample data offline, they trained a siamese network to learn a priori matching function. The trained siamese network can automatically judge the similarity of the input pairs, the optimal state can be determined based on the highest matching score. After that, the SiamFC proposed by Bertinetto et al. [4] develop a fully connected siamese network to match the object template and current search region in a convolutional manner. Chen K. et al. [17] add the memory of the historical state, utilizing a two-flow CNN to include two inputs to obtain a response map that predicts how likely the object appears in a specific location. GOTURN [5] applies the traditional convolutional layers to extract deep feature representations and fully connected layers to learn a complex feature comparison between the object template and the candidate samples. This method achieves a tracking speed of 100 FPS on GPU GTX680. However, GOTURN method needs to be improved for fast motion and large offset scenarios. He A. et al. [18] add semantic branches on the basis of SiamFC [4], which improve the discrimination of SiamFC. Addressing on non-rigid appearance change and partial occlusion problems, [19] presented a structured Siamese network, which can detect discriminative local patterns automatically and model the contextual information. To seek a trade-off between tracking performance and efficiency, the
authors in [20] decomposed the object tracking into two parallel sub-tasks, tracking and verifying, and presented a tracking framework which consists of two components, a tracker and a verifier. The tracker provided fast tracking and the verifier provided accurate verification. Recently, RPN have been introduced into the Siamese network for tracking. Heng F. et al. [21] proposed a multi-stage Siamese-RPN to refine the object bounding box, which consists of a sequence of RPNs cascaded from high-level to low-level layers in a Siamese network. Zhu Z. et al. [22] unified optical flow extraction and tracking task in siamese network for the first time, fused historical optical flow information with current frame feature map, and then track in siamese network and DCF framework.

2.4 Tracking Model Based on Spatio-Temporal Information

The CNN-based tracking method can only learn the appearance model of the current frame at a time, without considering the correlation with the historical frames. The recurrent neural network (RNN) model can read inputs one at a time, and remember some information through the hidden layer activations that get passed from one time-step to the next. It is suitable for sequence modeling since its neuron’s output can be directly applied to itself in the next time. Cui Z et al. [13] proposed using multi-directional RNN to obtain reliable target parts to avoid tracking drift. G. Zhu et al. [23] study the regression capacity of LSTM (Long Short-Term Memory) in the temporal domain, and present to concatenate high-level visual features generated by convolutional networks with region information. Gordon D et al. propose Re3 [9] using CNN followed by two LSTM blocks structure, in which CNN learns the appearance, the first LSTM block learns the motion characteristics and the second LSTM block outputs the location of the object. The tracking speed of Re3 reaches 150 fps.

Our approach is a hybrid tracker, using siamese CNN structure to extract the image features. Compared with the network in [9], we add a fully connected network for feature comparison. An improved two-layer LSTM uses information from previous frames to make future predictions. The temporal dependencies between sequential images can be modeled and let the tracker has robustness to occlusion problem. Finally, a fully connected layer is used to regress the bounding box and output the object location. The accuracy or robustness of the tracking algorithm can be improved by adjusting the fully connected feature comparison network in the middle.

3. Method

Our approach is a hybrid tracker adopting CNN to extract features and RNN to recurrent state updates. The proposed network consists three parts: feature extraction network $N_1$, feature comparison network $N_2$ and feature update network $N_3$, which is depicted in Fig. 1. $N_1$ extracts deep features and constructs multiple classifiers with different convolutional layers. $N_2$ learns effective matching functions to match the object template and candidates for visual tracking. $N_3$ network includes a two-layer LSTM blocks, in which the first layer remembers appearance and motion information and the second layer output the location of the object.

For each video sequence, we partition it into a series of frame-pair containing inter-frame correlation and feed the network a pair of frames, denoted as $F_t$ and $F_{t+1}$. The final output of the network is the object location in the current frame $F_{t+1}$.

3.1 Feature Extraction Network

The task of visual tracking starts with an initial bounding box around an object in the first video frame, with the goal of keeping track of that object the remainder video frames. For each video frame, the tracker must locate the object as well as update its internal state so it can continue tracking in future frames. One of the main sub-tasks of the framework is to convert the original pixel into a higher level feature vector representation. Traditional object trackers rely on extracting appearance information from the object pixels by hand-crafted features [24]. Recently, CNN has been used as a good appearance feature extractor. The explanatory work of feature extracted by CNN also proves that CNN can extract details such as edges and colors of objects in low-level layer, and global information such as shape and texture of objects in high-level layer [25]. The decomposition of target tracking algorithm is studied in document [26]. It is also pointed out that the most important step to decide whether a target tracking algorithm is good or not is the feature extraction part of image.

On the basis of Re3 network, in order to preserve the richness of image features so that CNN can process image details, we input the output feature maps of AlexNet’s first and second convolutional layers into two additional convolutional layers (denoted as conv-ex1 and conv-ex2) respectively. Then we obtain two feature maps $f_1$ and $f_2$. According to transfer learning theory [27], removing AlexNet’s last
three fully connected layers for classification can be better applied to visual tracking tasks. Combining the output feature map \( f_3 \) of the last convolutional layer and the previous two feature maps \( f_1 \) and \( f_2 \), then we get three sets of features at different levels: \( f_1 \) and \( f_2 \) can represent the details of the image, and \( f_3 \) tends to represent the deep response of the image to the convolution cores of each layer. The feature extraction network of one video frame is shown in Fig. 2.

To reduce computational complexity, like many other tracking algorithms, based on the reasonable assumption that the same object will not have a large drift within two adjacent video frames, the input can be a region near the known location of the object in the previous frame. If it is the first frame, the input is set to be a region near the known location of the object in it. Thus, it ensures that the input contains both the object and background information. Due to the large number of output features of the first two groups of convolutional layers in AlexNet, the effectiveness of features and whether the number of features will affect the tracking speed of the network should be considered when extracting low-level features. Thus, the kernel sizes of the two additional convolutional layers adopt 1 * 1, which not only improves the non-linear expression ability of our network, but also reduces the feature dimension. In this paper, the sizes of convolutional kernel and output feature dimension of each layer are shown in Table 1.

### Table 1 Size of kernels and feature map at different levels

| Conv1 | Conv2 | Conv3 | Conv4 | Conv5 |
|-------|-------|-------|-------|-------|
| Kernel | 11*11 | 5*5 | 1*1 | 1*1*25 | 3*3*38 |
| Feature map | 27*27 | 96*256 | 96*16 | 6*32 | 4*256 |

![Fig. 2 Feature extraction network structure](image)

**3.2 Effective Feature Comparison Network**

After getting the feature information for each frame from CNN, we need to learn effective matching functions by feature comparison network. Similar to GOTURN [5], we feed the network a pair of video frames as two branches shown in Fig. 3. The function of feature comparison network is to compare the output features from two branches at the same level and obtain the information related to the object location. Generally, the similarity between adjacent frames is the greatest in video, so comparing the CNN output features of adjacent frames can highlight the feature changes of objects to the greatest extent.

The feature maps extracted by CNN are all 3-D, and the fully connected network can handle 3-D features well. It retains the spatial information of the image, and can receive input of any size [26]. In this paper, we pre-process the video frame to a fixed size and input them into the network by interpolating the region near the known location of the object. The target object may deform during interpolation, which can improve the tracking robustness and help generic object tracking. Assuming the dimension of 3-D feature map is \( h * w * c \). The convolutional layer plus the fully connected layer can be taken as the convolutional operation of the feature map with \( h * w * c \) kernel. The number of kernels depends on the number of nodes in the fully connected layer. Re3 [9] simply spliced the features of different levels extracted by CNN and input them into a fully connected layer to reduce the dimension of features while extracting the depth features. This concatenating method will force the network to correlate the features of different layers without comparing the features of two adjacent frames at the same level. In order to highlight the feature differences of the same layer, we design a feature comparison network as shown in Fig. 3.

We concatenate the feature maps at peer levels of the two adjacent frames into a 1-D vector, and obtain three 1-D vectors. The three 1-D vectors are input into three fully connected layers for feature comparison, respectively. The comparing results are concatenated again (denoted as \( f_{c,t,t+1} \)) and input into the first LSTM.

### 3.3 Feature Updating Network

Traditional neural networks can only deal with a single sample in a forward propagation process (without considering the training process), ignoring the temporal relevance be-
the bias vector, and $\sigma$ among frames in visual tracking. Similar to Re3 [9], we opt
attempts have been done to exploit temporal association
can be directly applied to itself in the next time. Some
is suitable for sequence modeling since its neuron’s output
between samples. The recurrent neural network (RNN) model
represents the frame index, $t$, is the current input vector, $h_{t-1}$ is the previous output vector, $W$ is weight matrix, $b$ is
the bias vector, and $\sigma$ is the sigmoid function.

$$
\begin{align*}
    f_t &= \sigma(W_f[c_{t-1}, h_{t-1}, x_t] + b_f), \\
    i_t &= \sigma(W_i[c_{t-1}, h_{t-1}, x_t] + b_i), \\
    o_t &= \sigma(W_o[c_t, h_{t-1}, x_t] + b_o),
\end{align*}
$$

The exact formulation of LSTM is shown in (1) where $t$ represents the frame index, $x_t$ is the current input vector, $h_{t-1}$ is the previous output vector, $W$ is weight matrix, $b$ is
the bias vector, and $\sigma$ is the sigmoid function.

$$
\begin{align*}
    f_t &= \sigma(W_f[c_{t-1}, h_{t-1}, x_t] + b_f), \\
    i_t &= \sigma(W_i[c_{t-1}, h_{t-1}, x_t] + b_i), \\
    o_t &= \sigma(W_o[c_t, h_{t-1}, x_t] + b_o),
\end{align*}
$$

This structure shows better performance on updating
the object appearance changes. We use this unit structure
to build two-layer LSTM for feature updating. The input of
the first LSTM is the compared feature vector. The input of
the second LSTM is the merge of the compared feature
vector and the output of the first layer. The second LSTM’s
outputs are fed into a fully connected layer with four output
values representing the top-left and bottom-right corners of
the predicted target rectangle box $p_t$. The structure of the
feature update network is shown in Fig. 5.

4. Training Procedure

To allow our deep network available to multiple datasets, we use a combination of real and synthetic data to train our
deep network. The training data set and training procedure
is described below. In both cases, we trained the network
with an $L_1$ loss between the predicted bounding box and the
ground-truth (GT) bounding box.

4.1 Training from Datasets

The real training set is from ILSVRC 2016 Object Detection from Video dataset (ImageNet Video), which provides
3862 training videos with 1,122,397 images, 1,731,913 ob-
ject bounding boxes, and 7911 unique object trackers. How-
ever, it only contains videos for 30 object categories. Ref-
ences [5] and [6] adopted single picture data to train net-
work, and [7], [15] adopted synthetic data. Their experi-
mental results showed that these two extending training set
methods can improve the tracking ability of the network.
Similar to [7], the synthetic video sequences are generated
from a single image and combined with ImageNet Video to
train our network.

The final output of the network is a regression process
for the object location calculation, and the $L_1$ loss between
the predicted bounding box and the GT bounding box is:

$$
loss = \frac{\sum_{i=1}^{4} |p_i - g_i|}{4}
$$

where $p_i$ and $g_i$ are the predicted and the GT of the object
location respectively. When training LSTM, the GT of each
time step is usually used as input. GOTURN [5] proposed
a more robust training method, which changes the input of
LSTM on some time steps to the predicted output of the pre-
vious frame. In our training process, we adopt this method
to train the network to make the network has a higher fault-
tolerant rate.

4.2 Implementation Details

We use TensorFlow to train and test our networks. Unless
otherwise noted, we use the CaffeNet convolutional pipeline
initialized with the CaffeNet pre-trained weights for our
convolutional layers. The embedding fully connected layer has
2048 units and the LSTM layers have 1024 units each.
The new layers are initialized with the MSRA [29] initializa-
tion method. We adopt the ADAM gradient optimizer [30]
with the default momentum and weight decay and an ini-
tial learning rate of $10^{-5}$, which decreased to $10^{-6}$ after
10,000 iterations and continue for approximately 200,000
iterations. All tests were carried out using an Intel Core i7-
7800X CPU @ 3.5GHz and an Nvidia GeForce GTX 1080
Ti. For timing purposes, we ignore disk read speeds as they
are independent of the tracking algorithm used.

In order to make the network converge faster, we use
the corresponding layer weight of AlexNet network as the
initial weight for training. Due to the network has temporal-spatial complexity, the basic convolution layer parameters may degenerate with the increase of training iterations with the visualization of kernels, as shown in Fig. 6. As we can see from Fig. 6 that the changes of 96 convolutional kernels in the first layer when the number of iterations is 20,000, 25,000 and 30,000, respectively. The degeneration of convolution layer parameters leads to inaccurate feature extraction, especially when the number of iterations reaches 30,000, most of the convolutional kernels lose the ability to extract details such as edges and colors.

Addressing on this problem, during training we first fix the parameters of AlexNet’s five basic convolutional layers and update the parameters of other layers with a larger learning rate, and decrease the learning rate when the loss is stable. Then, we allow the weight updating of the first five convolutional layers when loss is stable again to avoid the degeneration.

For recurrent structure, the length of sequence is an important factor to be considered. Getting rid of the limitation that the Siamese network can only handle sequence has length 2, the proposed network can handle video sequences has longer length. But during training, it is necessary to specify a uniform sequence length. In [7], the authors proposed a solution which gradually increased the sequence length from 2 to 32. But in actual experiment, because the length of the test video sequence is unknown, the network trained with 32 frame samples has poor tracking effect on targets beyond 32 frames. By analyzing the intermediate state of LSTM, we find that the values in unit c representing long-term memory no longer concentrate near 0, but spread to the range of −1,000 to +1,000. The absolute value of unit c is too large, which impairs the performance of the neural network. The diffusion of the LSTM state is shown in Fig. 7 (a). In our experiments, every 32 frames are treated as a new sequence. So, the state transmitted in two-layer LSTM can be initialized with the information between the previous frames at the end of each 32-frame, and the values of unit c can be concentrated near 0, which is shown in Fig. 7 (b).

5. Experimental Results

We compare our network to other tracking methods on VOT2016 datasets in terms of both overall performance and robustness.

5.1 The Evaluation Indexes

Many of the videos contain difficulties such as heavy occlusion, object size change, and camera motion. The evaluation indexes are accuracy and robustness, where accuracy represents how well the predicted box matches with the GT and robustness represents how infrequently a tracker fails and is reset. The accuracy at time-step t measures how well the predicted bounding box $A^T_t$ overlaps with the GT bounding box $A^G_t$, and it is defined as intersection-over-union $\phi_t$:

$$\phi_t = \frac{A^G_t \cap A^T_t}{A^G_t \cup A^T_t}$$

The robustness is the number of times the tracker drifted from the object, i.e. the tracker failed. The definition of tracking failure in VOT is that the intersection-over-union $\phi_t$ is lower than a threshold $T$. When tracking fails at a particular frame, VOT performs a unique re-initialization strategy: the tracker is re-initialized $N_{skip} = 5$ frames after the failure. The average accuracy $\rho_A$ is obtained by taking the average over $\pi_t$, as

$$\rho_A = \frac{1}{N_{valid}} \sum_{t} \pi_t$$

where $N_{valid}$ represents the number of the valid tracking frames. Let $F_k$ be the number of times the tracker failed in the experiment repetition $k$ over a set of frames. The average robustness is then

$$\rho_R = \frac{1}{N_{rep}} \sum_{k=1}^{N_{rep}} F(k)$$

Where $N_{rep}$ is the number of times running on each sequence. In this paper, the robustness index is converted to the probability of tracking failure after $S$ (in our experiments, $S = 30$) frames.

5.2 A-R Plot

Table 2 lists the results of our tracking method compared with five real-time tracking method officially mentioned by VOT: ASMS [31], KCF [32], FoT [33], BDF [35] and Re3 [9]. In Table 2, “Baseline mode” means running a tracker on all sequences in VOT datasets by initializing it on the GT bounding boxes; “Unsupervised” mode means
Table 2: Results of tracking algorithm

| Tracker   | baseline overlap | failure | unsupervised overlap | EAO   | AUC   |
|-----------|------------------|---------|----------------------|-------|-------|
| ASMS      | 0.50             | 30.14   |                      | 0.21  | 0.33  |
| sKCF      | 0.47             | 52.03   |                      | 0.15  | 0.30  |
| FoT       | 0.387            | 49.20   |                      | 0.14  | 0.17  |
| BDF       | 0.37             | 51.45   |                      | 0.14  | 0.18  |
| Re3       | 0.52             | 32.01   |                      | 0.23  | 0.34  |
| Ours      | 0.51             | 30.62   |                      | 0.24  | 0.36  |

Fig. 8: A-R plot. Tracking results of our method and other 5 methods. Our tracking method’s performance is indicated with a blue cross, outperforming the other methods on robustness.

Table 3: Accuracy performance on VOT for 6 attributes

| Visual attribute | Tracker | ASMS | sKCF | FoT | BDF | Re3 | Ours |
|------------------|---------|------|------|-----|-----|-----|------|
| camera motion    | 0.52    | 0.51 | 0.37 | 0.36 | 0.53 | 0.51 |
| empty            | 0.53    | 0.55 | 0.43 | 0.43 | 0.53 | 0.53 |
| illumination change | 0.49 | 0.43 | 0.47 | 0.39 | 0.63 | 0.64 |
| motion change    | 0.49    | 0.42 | 0.35 | 0.35 | 0.49 | 0.47 |
| occlusion        | 0.45    | 0.43 | 0.28 | 0.36 | 0.43 | 0.38 |
| size change      | 0.45    | 0.36 | 0.40 | 0.30 | 0.32 | 0.55 |
| average          | 0.49    | 0.45 | 0.38 | 0.37 | 0.52 | 0.51 |

Table 4: Robustness performance on VOT for 6 attributes

| Visual attribute | Tracker | ASMS | sKCF | FoT | BDF | Re3 | Ours |
|------------------|---------|------|------|-----|-----|-----|------|
| camera motion    | 40.00   | 78.00 | 67.00 | 72.00 | 51.00 | 48.001 |
| empty            | 21.00   | 37.00 | 35.00 | 35.00 | 22.00 | 23.00 |
| illumination change | 11.00 | 7.00 | 17.00 | 10.00 | 3.00 | 2.00 |
| motion change    | 34.00   | 67.00 | 69.00 | 78.00 | 40.00 | 38.00 |
| occlusion        | 34.00   | 26.00 | 34.00 | 26.00 | 27.00 | 16.00 |
| size change      | 25.00   | 37.00 | 34.00 | 35.00 | 15.00 | 18.00 |
| average          | 27.50   | 42.00 | 42.67 | 42.67 | 26.33 | 24.17 |

5.3 Evaluation per Attribute

VOT labels the test video with attributes, such as camera motion, empty (the target object disappears from the screen), illumination changes, object motion change, occlusion, and object size change, which excels in Table 3 and Table 4. These tables analyze different trackers in terms of accuracy and robustness per attribute, respectively. As we can see that the average accuracy of our method is 0.51 ranking second and the average robustness is 24.17 ranking first. The significant accuracy improvement is achieved in the cases of illumination change and size change. In these cases, considering the network can’t segment the object at the pixel level, there may still be more background information interference in the prediction frame in some cases. In our method, we adopt feature classification and obtain the details and the global features of the object to better distinguish the object from the similar background. The significant robustness improvement is achieved in the cases of illumination change and occlusion. This improvement is due to the structure of the LSTM can ignore information via the input and forget gates. So, the LSTMs can implicitly learn to hand occlusions and predict the target’s trajectory well (better than other algorithms in the test set of occlusion attributes, with an optimal value of 16.00 in the occlusion row of Table 4). For example, the qualitative results on “road” and “marching” video sequences in VOT 2016 are shown in Fig. 9 and Fig. 10, respectively. As we can see, when the moving vehicle in Fig. 9 is completely occluded by trees or the moving target in Fig. 10 is completely occluded by a girl, the algorithm can still predict the target position (red box is the GT, and the green box is the predicted location). For illumination changes, the qualitative results on “racing” and “shaking” video sequences are shown in Fig. 11 and Fig. 12, respectively. As we can see, our tracker can still track the target well when the illumination changes.
Additionally, the ASMS [31], sKCF [32], FcT [33] and BDF [35] trackers run under 100 fps. Re3 [9] can run at 100 fps. Compared with Re3 [9], our proposed network adds feature comparison network before LSTM to reduce the number of parameters, allowing our tracker to run at 150 fps. Using AlexNet’s weight as the initial value, the iteration number of Re3 [9] network training reaches more than 200,000 times, but the iteration number of our network training is 80,000 to achieve the same result.

5.4 Experiments on OTB-2015

We conduct experiments on OTB-2015, which consists of 100 fully annotated videos. The qualitative results on “bird1” and “DragonBaby” video sequences are shown in Fig. 13 and Fig. 14, respectively. In Fig. 15, we also show tracking failure of “car1” sequence due to latching onto a foreground object.

5.5 Experiments on LaSOT

LaSOT [36] is a large-scale dataset which consists of 1,400 videos. We compare the proposed tracking algorithm with SiamFC [4], StructSiam [19], TRACA [6] and fDSST [12]. Following [36], the evaluation results on LaSOT under protocol I using precision, normalized precision and success (SUC) for different trackers as shown in Fig. 16. It can be seen that our method achieves SUC scores of 0.336 under protocol II, and outperforms TRACA [6] and fDSST [12] but approaches to SiamFC [4] and StructSiam [19]. In the future work, we will continue do experiments on LaSOT for long large-scale object tracking. The qualitative results on LaSOT are shown in Fig. 17.

6. Conclusion

In this paper, we present a “CNN+RNN” hybrid network for real-time generic object tracking. Inspired by
siamese network structure, we introduce feature comparison to strengthen the inter-frame connection, and the compared results are input into our two-layer LSTM for quickly updating online and predicting the object location. Because the recurrent network can directly learn robustness to complex visual phenomena such as occlusion and illumination change, our method demonstrates good A-R rank and speed over comparable trackers. We also showed some details about efficient and effective training method to reduce the training time. The test results show that, compared with some high real-time tracking algorithms, the proposed method has better robustness and predictive overlap rate under unsupervised conditions while ensuring speed and accuracy.

Acknowledgements

This work is financially supported by the Natural Science Foundation of China (61703201), Natural Science Foundation of Jiangsu Province (BK20170765), Key Laboratory of meteorological detection and information processing in Jiangsu Province (KDXS1503), and Nanjing Institute of Technology fund for Young Scholar (CKJB201602).

References

[1] H. Nam and B. Han, “Learning multi-domain convolutional neural networks for visual tracking,” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.4293–4302, June 2016.
[2] L. Wang, W. Ouyang, X. Wang, and H. Lu, “Stc: Sequentially training convolutional networks for visual tracking,” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1373–1381, June 2016.
[3] R. Tao, E. Gavves, and A.W.M. Smeulders, “Siamese instance search for tracking,” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1420–1429, June 2016.
[4] L. Bertinetto, J. Valmadre, J.F. Henriques, A. Vedaldi, and P.H.S. Torr, “Fully-convolutional siamese networks for object tracking.” In The European Conference on Computer Vision (ECCV), vol.9914, pp.850–865, June 2016.
[5] D. Held, S. Thrun, and S. Savarese, “Learning to track at 100 fps with deep regression networks.” In The European Conference on Computer Vision (ECCV), vol.9905, pp.749–765, June 2016.
[6] J. Choi, H.J. Chang, T. Fischer, S. Yun, K. Lee, J. Jeong, Y. Demiris, and J.Y. Choi, “Context-aware deep feature compression for high-speed visual tracking.” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.479–488, June 2018.
[7] S. Yun, J. Choi, Y. Yoo, K. Yun, and J.Y. Choi, “Action-decision networks for visual tracking with deep reinforcement learning.” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1349–1358, July 2017.
[8] H. Fan and H. Ling, “Sanet: Structure-aware network for visual tracking,” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pp.2217–2224, July 2017.
[9] G. Daniel, F. Ali, and D. Fox, “Re3: Real-time recurrent regression networks for object tracking,” IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2017.
[10] L. Wang, W. Ouyang, X. Wang, and H. Lu, “Visual tracking with fully convolutional networks,” In The IEEE International Conference on Computer Vision (ICCV), pp.3119–3127, Dec. 2015.
[11] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, “Hierarchical convolutional features for visual tracking,” In The IEEE International Conference on Computer Vision (ICCV), pp.3074–3082, Dec. 2015.
[12] M. Danelljan, G. Häger, F.S. Khan, and M. Felsberg, “Discriminative scale space tracking,” IEEE Transactions on Pattern Analysis & Machine Intelligence, vol.39, no.8, pp.1561–1575, 2017.
[13] Z. Cui, S. Xiao, J. Feng, and S. Yan, “Recurrently target-attending tracking,” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1449–1458, June 2016.
[14] Y. Qi, S. Zhang, L. Qin, Q. Huang, H. Yao, J. Lim, and M.-H. Yang, “Hedging deep features for visual tracking,” IEEE Transactions on Pattern Analysis & Machine Intelligence, 99:1–6, 2018.
[15] B. Ma, J. Shen, Y. Liu, H. Hu, L. Shao, and X. Li, “Visual tracking using strong classifier and structural local sparse descriptors,” IEEE Trans. Multimedia, vol.17, no.10, pp.1818–1828, 2015.
[16] Y. Qi, L. Qin, J. Zhang, S. Zhang, Q. Huang, and M.-H. Yang, “Structure-Aware Local Sparse Coding for Visual Tracking,” IEEE Trans. Image Process., vol.27, no.8, pp.3857–3869, 2018.
[17] K. Chen and W. Tao, “Once for all: a two-flow convolutional neural network for visual tracking,” IEEE Transactions on Circuits & Systems for Video Technology, vol.28, no.12, pp.3377–3386, 2018.
[18] A. He, C. Luo, X. Tian, and W. Zeng, “A twofold siamese network for real-time object tracking.” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.4834–4843, June 2018.
[19] Y. Zhang, L. Wang, J. Qi, D. Wang, M. Feng, and H. Lu, “Structured siamese network for real-time visual tracking,” In The European Conference on Computer Vision (ECCV), vol.11213, pp.355–370, Sept. 2018.
[20] H. Fan and H. Ling, “Parallel tracking and verifying: A framework for real-time and high accuracy visual tracking,” In The IEEE International Conference on Computer Vision (ICCV), pp.5487–5495, Oct. 2017.
[21] H. Fan and H. Ling, “Siamese cascaded region proposal networks for real-time visual tracking,” IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.
[22] Z. Zhu, W. Wu, W. Zou, and J. Yan, “End-to-end flow correlation tracking with spatial-temporal attention,” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.548–557, June 2018.
[23] G. Zhu, F. Porikli, and H. Li, “Robust visual tracking with deep convolutional neural network based object proposals on pets,” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pp.1265–1272, June 2016.
[24] M.E. Maresca and A. Petrosino, “The matrioska tracking algorithm for ltdt2014 dataset,” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pp.720–725, June 2016.
[25] H. Fan and H. Ling, “Parallel tracking and verifying: A framework for real-time and high accuracy visual tracking,” In The IEEE International Conference on Computer Vision (ICCV), pp.5487–5495, Oct. 2017.
[26] N. Wang, J. Shi, D.-Y. Yeung, and J. Jia, “Understanding and diagnosing visual tracking systems,” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1449–1458, June 2016.
[27] K. Chen and W. Tao, “Once for all: a two-flow convolutional neural network for visual tracking,” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.3101–3109, Dec. 2015.
[28] S.J. Pan and Q. Yang, “A survey on transfer learning,” IEEE Trans. Knowl. Data Eng., vol.22, no.10, pp.1345–1359, 2010.
[29] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.3431–3440, June 2015.
[30] Y. Qi, S. Zhang, L. Qin, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” In The IEEE International Conference on Computer Vision (ICCV), pp.1026–1034, Dec. 2015.
[31] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 3rd International Conference for Learning Representations (ICLR), July 2015.
[32] T. Vojir, J. Nosková, and J. Matas, “Robust scale-adaptive mean-shift for tracking,” Pattern Recognition Letters, vol.49, pp.250–258, 2014.
[33] T. Vojíř and J. Matas, “The enhanced flock of trackers,” Registration and Recognition in Images and Videos, vol.532, pp.113–136, 2014.

[34] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Cehovin, “A novel performance evaluation methodology for single-target trackers,” IEEE Transactions on Pattern Analysis & Machine Intelligence, vol.38, no.11, pp.2137–2155, 2016.

[35] M.E. Maresca and A. Petrosino, “Clustering local motion estimates for robust and efficient object tracking,” In The European Conference on Computer Vision (ECCV), vol.8926, pp.244–253, 2015.

[36] H. Fan, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, H. Bai, Y. Xu, C. Liao, and H. Ling, “Lasot: A high-quality benchmark for large-scale single object tracking,” IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.

Rui Chen received the B.E degree, the M.E. degree from Southeast University, Nanjing, China, in 1991 and 1996, respectively. She received PhD degree from Nanjing University of Post and Telecommunications in 2013. Her major research interests include deep learning and image processing.

Ying Tong received the B.E degree from PLA University of Science and Technology in 2001 and the M.E. degree from Nanjing University of Post and Telecommunications in 2006. Her major research interests include image processing and face recognition.

Ruiyu Liang received the B.E degree, the M.E. degree from Hehai University, Nanjing, China. He received PhD degree from Southeast University in 2015. His major research interests include audio signal processing and intelligent electrics and system.