Dynamic Update Siamese Networks with Deeper Features

Jianhao Tan\textsuperscript{1,2}, Shanshan Tan\textsuperscript{1,2,*}, Wang Yin\textsuperscript{1,2}, Liming Liu\textsuperscript{1,2}, Siyuan Zhang\textsuperscript{1,2} and Yaonan Wang\textsuperscript{1,2}

\textsuperscript{1}College of Electrical and Information Engineering, Hunan University, Changsha 410082, China
\textsuperscript{2}National Engineering Laboratory for Robot Visual Perception and Control Technology, Changsha 410082, China

*Corresponding author

Abstract. Visual object tracking has been a concern topic these years, and many trackers have achieved good results in various fields. These researches and breakthroughs have made many improvements to solve problems such as drift, lighting, deformation and occlusion. In this paper, we improve the structure of the AlexNet\textsuperscript{[1]} network by designing the three important influencing factors of the receptive field size, total network step size, and feature filling of the twin network. Apart from this, we add a smoothing matrices and a background suppression matrices to effectively learn the features of the first few frames as much as possible. Fuse multilayer feature elements can learn online about target appearance changes and background suppression, and we train them by using continuous video sequences.

Keywords: Target tracking, Siamese network, Residual network, Dynamic update.

1. Introduction

Target tracking is to find the target moving position of interest in each image in the video sequence. The overall evolution of the algorithm is from traditional feature extraction to machine learning, and now it is based on neural network deep learning.

Correlation filtering is a focus of current research and originated in the field of signal processing. One of its advantages is the introduction of fast fourier transform (FFT), which makes the algorithm have a great speed increase. The first algorithm that enables correlation filtering to be used in real-time applications is CSK\textsuperscript{[2]} (the Circulant Structure of Tracking-by-detection with Kernels). It calculates the correlation between two adjacent frames by using a Gaussian kernel, and takes the point with the largest response as the predicted target center.

In the context of big data, using deep learning to train a network model become universal. In object tracking, the initial application method is to apply the features learned by the network directly to the correlation filtering or Struck's tracking framework to get better tracking results, such as the DeepSRDCF\textsuperscript{[3]} method. In essence, the feature expression obtained from the convolution output is better than the HOG or \textsuperscript{CN}\textsuperscript{[4]} feature. This is one of the advantages of deep learning, but it also brings an increase in the amount of calculation. Many current research tracking frameworks and methods often compare two features at the same time to verify the improvement of the tracking method or framework: one is the traditional manual feature, and the other is the feature of deep network learning. The convolution output of different layers of the network can be used as tracking features. D.Martin has also done a lot of work on how to effectively use the features of deep learning, and proposed a series of related methods, such as C-COT\textsuperscript{[5]} and ECO\textsuperscript{[6]}.
In this paper, our proposed network adds transformation matrices about temporal context and background. We uses deeper residual networks to extract features to obtain more feature information of the target. The two matrices described above are quickly calculated by FFT in the frequency domain. The transformation matrices $V$ is obtained from the tracking result of the t-1 frame and the target of the first frame. It acts on the convolution features of the target template, learns the change makes the convolution feature of the template at time t approximately equal to the template convolution feature at time t-1, making the change of the current frame relative to the previous frames smooth. The transformation matrices $W$ is obtained from the tracking result of frame t-1. It is obtained that, by acting on the convolutional features of the candidate region at time t, learning background suppression eliminates the influence caused by unrelated background features in the target region. As shown in Figure 1, our tracker performs better than other trackers.

Figure 1. Tracking results of 3 typical video sequences, using our tracker and 2 real-time trackers. Red represents our tracker, green represents SiamFC, and blue represents Staple.

2. Related Work

Siamese network based tracking. Siamese network based trackers select target from candidate patches through a matching function offline learned on image pairs. We briefly review SiamFC[7]: SiamFC tracker proposed a basic tracking algorithm, training full convolutional twin networks end-to-end, and training on the ILSVRC15 video object detection dataset. This tracker is faster than real-time, and it can learn to track arbitrary objects. SiamFC has used a fully convolutional strategy to realize this process and formulate is as,

$$ S_t' = corr(f^1(O_t), f^1(Z_t)) $$

where $S_t'$ is a response map denoting the similarity between $O_t$ and candidate patches in $Z_t$, $f^i(\cdot)$ represents the $i$-th layer deep feature of some properly trained CNN model.

Residual network. Scholars discovered during the experiment that the network has degraded with the increase of the number of network layers: when the number of network layers is increased, the training set loss gradually decreases, and then it becomes saturated. When you increase the network depth, the training set instead and losses will increase. But this is not over-fitting phenomenon, because training loss is repeated in over-fitting training.

Residual learning framework[9] can reduce the burden of network training, which is inherently deeper than networks that have been used before. They explicitly use this layer as the learning residual function related to the input layer, rather than learning the unknown function. Deep residual learning for image recognition is a very concise framework that can be used to train deep networks. It can achieve the best performance in the fields of image classification, object detection, and semantic segmentation.
3. Tracking Framework

![Diagram](image)

Figure 2. Main component of tracking framework.

In figure 2, the dashed line in the figure shows the flow of SiamFC. After obtaining the feature map in SiamFC, the correlation filtering operation is directly used, and the extract of the target in the first frame is used as the convolution kernel of the detection frame. The target serves as a response point. In target tracking, a box is generally given in the first frame of the video. This box contains four parameters \((x,y,w,h)\), and the range of the box is the target to be tracked. However, SiamFC is prone to the detection of wrong objects during the tracking process. For example, when a second person suddenly appears when tracking a pedestrian, the tracker may frame the second person and lose the target. This is mainly because SiamFC only uses the first frame as a template frame, which leads to too simple sample data for feature extraction. To solve this problem, we propose a tracker that can learn the tracking results of previous frames.

3.1. Our Final Model

For the transformation matrices \(V\) and transformation matrices \(W\), training is performed using regular linear regression. \(V_{t-1}^i * f^{i}(O)\) and \(W_{t-1}^i * f^{i}(Z)\) are obtained by transforming the matrices \(V\) and \(W\), respectively, where "*" represents a circular convolution operation, and \(V\) represents a change in the appearance of the target. After the target template, \(W\) represents the background suppression transformation to get a more suitable current search template; the final model is as follows:

\[
S_t^i = \text{corr}(V_{t-1}^i * f^{i}(O), W_{t-1}^i * f^{i}(Z))
\]  

The improved twin network structure takes image pairs as input, including an example image \(Z\) and a candidate search image \(X\). Among them, image \(Z\) represents the object of interest (for example, an image block centered on the target object in the first video frame), while \(X\) represents the search area in subsequent video frames, which is usually larger. Both inputs are processed by ConvNet with parameter \(\theta\). This will produce two feature maps, which are related to each other as:

\[
f_{\theta}(z,x) = \phi_{\theta}(z)*\phi_{\theta}(x) + b
\]

Among them, \(b\) represents a bias term, and the entire formula is equivalent to performing an exhaustive search on the image \(X\) in the mode of \(Z\). The purpose is to match the maximum value in the response graph \(f\) with the target position. In order to achieve this goal, the network conducts offline training through random image pairs \((Z, X)\) and corresponding ground labels \(y\) obtained from the training videos. The parameter \(\theta\) in ConvNet is obtained by minimizing the following loss parameters in the training set:

\[
\arg\min_{\theta} E_{(z,x,y)} L(y, f_{\theta}(z,x))
\]

The basic formula of the loss function is:

\[
L(y, \nu) = \log(1 + \exp(-\nu y))
\]
where $y \in (+1, -1)$ represents the true value, and $V$ represents the actual score of the sample search image. According to the sigmoid function, the above formula indicates that the probability of positive samples is $\frac{1}{1+e^{-y}}$, and the probability of negative samples is $1 - \frac{1}{1+e^{-y}}$, then the following formula is easily obtained from the formula of cross entropy:

$$L(y; v) = \frac{1}{D} \sum_{u \in D} l(y[u], v[u])$$

(6)

3.2. Network Diagram
As shown in figure 3, we use a residual network to extract features from the input image and search frame, and add the smoothing matrices $V$ and the background suppression matrices $W$ fusion features.

![Network Diagram](image)

**Figure 3.** Detailed network diagram of our tracker.

About transformation matrices $V$, first perform a crop operation on the input image to obtain the target $O$, then use the residual network to extract the features of $O$, and finally learn the target appearance change parameter $V$ through RLR.

About transformation matrices $W$, first perform a crop operation on the input picture to obtain $G$ (the area centered on the target, including part of the background, the same size as the search area), and obtain the target $G'$ by highlights the foreground through the Element wise operation. The features of $G$ and $G'$ are extracted through the residual network, and finally the background suppression parameter $W$ is learned through the RLR unit.

Perform a crop operation on the input image (get the cropped area from the response image of the previous frame) to get the search area $Z$, and extract the feature of input frame (the target to be tracked in the first frame) and the current frame (the search area of the current frame) through the residual network, denoted as $F_1$ and $F_z$. The transformation matrices $V$ and $F_1$ perform a circular convolution operation to obtain $F_1'$, simultaneously $F_z$ and the transformation matrices $W$ perform a circular convolution to obtain $F_z'$. Among them, $F_1'$ and $F_z'$ perform a cross-correlation operation to obtain a response graph.

4. Frame Supplement

4.1. Appearance Change Matrices
According to Regularized linear regression (RLR), learn the changes of the previous frame tracking result and the first frame template frame:

$$R = \arg \min_T \| T \ast X - Y \|^2 + \lambda \| T \|^2$$  \hspace{1cm} (7)

A quick calculation in the frequency domain yields:

$$R = F^{-1} \left( \frac{F^*(X) \odot F(Y)}{F^*(X) \odot F(X) + \lambda} \right)$$  \hspace{1cm} (8)

The resulting change is expressed as follows:

$$V^l_{t+1} = F^{-1} \left( \frac{F^*(f^t_i) \odot F(f^l_{i+1})}{F^*(f^t_i) \odot F(f^l_i) + \lambda_w} \right)$$  \hspace{1cm} (9)

where $f^t_i = f^t(O)$, $f^l_i = f^l(O_{i+1})$, $O$ represents the targets, $f$ are matrices, the upper right index indicates the $l$-th channel, and the right lower index indicates the number of frames, which is obtained from the tracking result of the previous frame and the first frame target.

4.2. Background Suppression Matrices

Obtain the suppression amount of the current frame background according to the RLR calculation formula in the frequency domain $W^l_{t-1}$:

$$W^l_{t-1} = F^{-1} \left( \frac{F^*(G_{t+1}) \odot F(G^l_t)}{F^*(G^l_t) \odot F(G^l_t) + \lambda_w} \right)$$  \hspace{1cm} (10)

Among them, $G_{t+1}$ is a picture of the same size as the search area in the previous frame, and $G^l_t$ is a Gaussian smoothing of the center point of the $G_{t+1}$ picture. The purpose is to highlight the center and suppress the edges. By learning target changes $V^l_{t+1}$ and background suppression transformations $W^l_{t-1}$ online, the improved model enables static twin network adaptation capabilities online, thereby improving tracking accuracy and real-time speed.

4.3. Residual Network

We build up deeper and wider networks. The constructions (as shown in Tab.1) follow our design guidelines, and the goal is to ensure that the receptive field size of neurons in the final layer lies within the derived range. In the first stage of the network, the image is subjected to a cropping operation (size 2) and then enters a 7 × 7 convolution to remove the features affected by the filling.

**Table 1.** Architectures of designed backbone networks for our trackers.

| Stage | CIResNet |
|-------|----------|
| Conv1 | 7×7, 64, stride 2 |
| Conv2 | 2×2 max pool, stride 2 |
|       | | 1×1 64 |
|       | | 3×3 64 ×1 |
|       | | 1×1 256 |
| Conv3 | 1×1 128 |
|       | | 3×3 128 ×4 |
|       | | 1×1 512 |
The second stage network has 3 layers, the first layer is $1 \times 1$ convolution, the number of channels is 64; the second layer is $3 \times 3$ convolution, the number of channels is 64; the third layer is $1 \times 1$ convolution, the number of channels is 256. The feature map after the convolution layer is subjected to the addition operation and then enters the crop operation. The crop operation is a $3 \times 3$ convolution to offset the features affected by padding of 1.

The third stage network has 12 layers in total, and the first, second, and third layers are used as the unit block for four times. The first layer is a $1 \times 1$ convolution with 128 channels; the second layer is a $3 \times 3$ convolution with 128 channels; the third layer is a $1 \times 1$ convolution with 512 channels.

4.4. Eltwise and Join Training

Shallow features have high center weights and deep features have high peripheral weights and low centers. If the target is in the center of the search area, shallow features can better locate the target. If the target is outside the search area, deep features can also effectively determine the target position.

Multi-layer feature fusion:

$$S_t = \sum_{i \in \Gamma} Y^i \odot S_t^i$$

where $\odot$ denotes the elementwise multiplication. Specifically, we can use Eq. (2) to produce $|\Gamma|$ response maps $\{S_t^i | i \in \Gamma\}$ with multi-level features of some deep feature network.

That is, when the target is near the center of the search area, the darker layer features help to eliminate background interference, and the lighter layer features help to obtain the precise positioning of the target; if the target is located outside the search area, only deeper layer features can effectively determine the target location.

For joint training, first through forward propagation, after tracking a given N-frame video sequence $\{I_t | t = 1, ..., N\}$, N response graphs are obtained, denoted by $\{S_t | t = 1, ..., N\}$, and N target frames are denoted by $\{J_t | t = 1, ..., N\}$:

$$L_t = \frac{1}{|S_t|} \log(1 + \exp(-S_t \odot J_t)) \quad (12)$$

The single-layer network structure diagram is shown in figure 3. “Eltwise” (elementwise multi-layer fusion) is training a matrices, and the values in the matrices represent the weights of different feature maps at different positions. Backpropagation through time (BPTT) and Stochastic Gradient Descent (SGD) are used for gradient propagation and parameter update. In order to effectively use the network trained by BPTT and Stochastic Gradient (SGD), all the parameters must be obtained, as shown in figure 3, which calculates $\nabla_f L_t$, $\nabla_{f_i} L_t$, and $\nabla_{\lambda_i} L_t$ from $\hat{\nabla}_{\alpha} L_t$, and then passes through the "CirConv" and "RLR" layers on the left, to ensure that the loss gradient can be effectively transmitted to $f^i$:

$$\nabla_f L_t = F^{-1}(\hat{f}_i \odot \hat{\nabla}_{\alpha} L_t) \quad (13)$$

$$\nabla_{f_i} L_t = F^{-1}(U \odot \hat{f}_i^* \odot \hat{\nabla}_{\alpha} L_t) \quad (14)$$

$$\nabla_{\lambda_i} L_t = F^{-1}(-U^2 \odot \hat{f}_i^* \odot \hat{f}_i^T) \hat{\nabla}_{\alpha} L_t \quad (15)$$

$$U = (\hat{f}_i^* \odot \hat{f}_i + \lambda_i)^{-1} \quad (16)$$

$$\nabla_{\lambda_i} L_t = E\{ -2U^2 \odot (\hat{f}_i^*)^T \odot \hat{f}_i^T \} E^T \nabla_f L_t + F^{-1}(\hat{\nabla}_{\alpha} L_t) \quad (17)$$
Among them, \( \hat{f} \) represents \( f \) after the Fourier transform, and \( E \) is a discrete Fourier transform matrices. For the cell-based multi-layer fusion formula, the above process can also be used for calculation. For the multi-feature fusion formula, it can be transformed into \( \nabla_y L_i = S'_i \odot \nabla s_i \), and then we can learn \( Y^t \) from this.

5. Experiments
In visual object tracking 2016 benchmark (VOT2016) challenge, the sequences are the same as visual object tracking 2015 benchmark (VOT2015), while the bounding boxes are re-annotated. We compare our tracker with some excellent trackers in VOT2016. The performance is evaluated in terms of accuracy (average overlap while tracking successfully) and robustness (failure times). The overall performance is evaluated using Expected Average Overlap (EAO) which takes account of both accuracy and robustness.

![Figure 4. Comparison of overlap of different trackers.](image)

![Figure 5. Comparison of accuracy and robustness of different trackers.](image)

We compared our tracker with other excellent trackers. Although our tracker has lower overlap score than SiamRPN as shown in figure 4, our tracker gets much higher accuracy than it. And we also can know from figure 5 that our trackers are superior in the comparison of accuracy. Tab. 2 lists the details about trackers. As shown in Tab. 2, our method is able to rank first in accuracy, and rank second in
overlap. It can be seen that our method is mainly to improve the accuracy. Because our tracker uses a deeper network for feature extraction, it is much slower than other trackers.

Table 2. Expected overlap of our tracker, Siamese-RPN and SiamFC in VOT2016 challenge. Details about the trackers in VOT2016. Red, blue and green, represent 1st, 2nd and 3rd respectively.

| Tracker | Accuracy | overlap | Speed(fps) |
|---------|----------|---------|------------|
| SiamFC  | 0.4973   | 0.2189  | 65         |
| SiamRPN | 0.5673   | 0.3804  | 85         |
| ours    | 0.5712   | 0.3205  | 16         |

6. Conclusion
In experiment, our method can achieve good performance in accuracy. Our tracker uses a dynamic twin network to ensure the balance between accuracy and real-time tracking, uses a dynamic update network to quickly learn target appearance changes, makes full use of target space-time information, and effectively solves problems such as drift and target occlusion. Our framework selects a deeper network to obtain target features, uses appearance learning and background suppression to perform dynamic tracking, and effectively increases robustness.

In the future, we plan to explore the possibility of online regressing more detailed parameters of a moving object, e.g. its aspect ratio, tight silhouette and scale. At the same time, we will explore how to improve the tracking speed so that the tracker can be better applied in practice.

Acknowledgements
Fund Project: National Natural Science Foundation of China (61433016)

References
[1] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks.
[2] Henriques J.F., Caseiro R., Martins P., Batista J. (2012) Exploiting the Circulant Structure of Tracking-by-Detection with Kernels. In: Fitzgibbon A., Lazebnik S., Perona P., Sato Y., Schmid C. (eds) Computer Vision – ECCV 2012. ECCV 2012.
[3] Martin Danelljan, Gustav Hager, Fahad Shahbaz Khan, Michael Felsberg; The IEEE International Conference on Computer Vision (ICCV), 2015, pp. 4310-4318
[4] Danelljan, M.,Khan, F.S.,Felsberg, M.,van de Weijer, J.. Adaptive Color Attributes for Real-Time Visual Tracking[P]. Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on,2014.
[5] Martin Danelljan et al. Beyond Correlation Filters: Learning Continuous Convolution Operators for Visual Tracking. ECCV 2016, Part V, LNCS 9909, pp. 472-488. Springer (2016)
[6] Martin Danelljan et al. ECO: Efficient Convolution Operators for Tracking. Computer Vision and Pattern Recognition (cs.CV)
[7] Bertinetto L., Valmadre J., Henriques J.F., Vedaldi A., Torr P.H.S. (2016) Fully-Convolutional Siamese Networks for Object Tracking. In: Hua G., Jégou H. (eds) Computer Vision – ECCV 2016 Workshops. ECCV 2016. Lecture Notes in Computer Science, vol 9914. Springer, Cham.
[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778