Journal of Computers Vol. 31 No. 5, 2020, pp. 112-126
doi:10.3966/199115992020103105009

Focus on Specific-Video Objects: Learning Various Sample Representations for Visual Tracking

Bo-Yan Zhang[1,2*], Yong Zhong[1,2]

[1] Chengdu Institute of Computer Applications, Chinese Academy of Sciences, No.27 Section 4, South Renmin Road, Chengdu 610041, China

[2] University of Chinese Academy of Sciences, No. 19 (A) Yuquan Road, Shijingshan District, Beijing 100049, China
zhangboyan_0823@163.com, zhongyong@casit.com.cn

Received 15 April 2019; Revised 21 August 2019; Accepted 12 October 2019

Abstract. Visual object tracking is one of the most challenging tasks in the field of computer vision. Many trackers can achieve impressive performance in the field; however, there is still some room for improvement, especially when it comes to tough cases, such as fast motion, blur, and rotation. Deep feature-based trackers have been employed due to their outstanding ability for representation, but their performance suffers from over-fitting due to lack of sufficient labeled training data, as well as similar distractors. Besides, the categories of targets are diverse in the tracking task. In this paper, we introduce a positive data augmentation module (PDAM) during the offline phase to generate various positive samples. The generated samples with the original data are clustered to form a different classes of training data. Each class is used to train one of multiple deep-tracking models which have an identical structure. At the tracking stage, a selection module chooses the most suitable pretrained tracking model according to the target information in the given video sequence. We conducted the experiments to validate the effectiveness of our method with some state-of-the-art trackers on a standard benchmark. The results showed that the proposed method achieved excellent tracking performance and robustness in videos involved in deformation, scale variation, motion blur.

Keywords: clustering, deep learning, generative model, object tracking

1 Introduction

Visual object tracking has come to be widely applied in almost every part of our life due to the rapid development of information and computer technology. Given a video sequence and the initial location of the target, the task of a tracking system is to estimate the target's locations in the subsequent image sequence. Generally, achieved higher overlap rate between the prediction results and the truth bounding box results in more accurate localization of the target. Even though many studies have been carried out in recent decades, visual objects tracking still a critical challenge for many researchers in computer vision. One of the most effective methods in the field of visual objects tracking is discriminative methods, namely tracking-by-detection. These methods have been proposed to achieve visual object tracking by designing a classifier that distinguishes the target region from the background information.

Several works related to discriminative visual object tracking have appeared with the development of deep learning methods. Because of the strong representation power of deep learning for target objects, they are able to achieve the performance of conventional state-of-the-art techniques and they can even outperform some of them [1-3]. They take advantage of the powerful learning ability of Convolutional Neural Networks (CNNs) for extracting appearance features and learn the general feature representation. In addition, some algorithms based on a correlation filter employ the deep features of targets to obtain accurate tracking performance. Hong [3] proposed a CNN-based tracker that combines a pre-trained

* Corresponding Author

CNN with a Support Vector Machine (SVM) to extract the appearance features of targets to distinguish them from the background. These studies show promise for applying CNNs to the field of tracking. However, their effectiveness is limited due to the over-fitting in challenging tracking scenes, such as different levels of lighting or occlusion, motion blur, fast motion, and scale variation.

To deal with the above-stated difficulties, most researchers use a large-scale video dataset to train their tracking network, such as ImageNet [4], COCO [5]. They contain a massive number of various objects. The usage of these datasets improves the performance of trackers under several challenging situations to some degree. Large datasets are required to train deep-learning based trackers with powerful feature extraction. However, in practice, the targets are severely occluded and change their shape and appearance dramatically, which are quite rare to find in most large-scale datasets. A direct solution to solving this problem might be to apply much larger training datasets although this would barely account for some intractable cases. Since each training video contains only one positive sample in the existing tracking task, most tracking algorithms use dense sampling on much larger training datasets to generate massive positive samples. The generated samples lack in diversity, which leads to unsatisfied tracking performance. Another method that is widely applied in the training deep neural networks is data augmentation. This method provides novel training samples by randomly changing the shape and direction of the labelled samples. However, the wide range variations of an object are covered in real tracking scenes, including but not limited to deformation and illumination variation. Thus, the reliability of the mainstream data augmentation approach needs to be proved [6].

Motivated by the above-mentioned facts, this paper proposes a method of providing hard positive samples and effective use for offline training of a CNN-based tracking network. In addition, based on different types of targets, a selection mechanism is employed to enable accurate tracking for sequence-related targets. The main methods and contributions of this study are summarized as follows:

(1) We employ a variational auto-encoder (VAE) to act as a positive data augmentation module (PDAM), which enriches the diversity of positive samples and expands the training data. This module aims at alleviating the over-fitting of deep tracking models that is caused by the absence of effective samples and the existence of redundant training data.

(2) A scene-aware mechanism is built to choose the best-suited pretrained deep tracking model for tracking the specific target of a given video. These pretrained deep tracking models are trained offline with different clusters of training data. It fully considers the video-related information and provides better adaptability for various targets from different class.

(3) The proposed custom tracking system is successfully employed to a visual tracking task. It is compared with some state-of-art trackers on Object Tracking Benchmark (OTB100). The evaluation results demonstrate that our method achieves competitive performance, especially in challenging scenes, including low resolution, motion blur, out-of-view objects, in-plane rotation, and deformed target.

The rest of the paper is arranged as follows. First, we review and analyze recent deep-based trackers. In the next section, the proposed optimization method is described and discussed for the offline training and tracking stage. The tracking details of the implementation and algorithm are described in Section 4. In the final section, the system is applied to tracking benchmark datasets and an analysis of the results is presented.

2 Related Works

The main idea of discriminative tracking algorithms is to teach a classifier to extract target key information from the image. It is quite important for a tracking system to design a method of accurately leaning the target's appearance information from the training samples. Since they have significant potential in representation power for the target object, trackers based on deep networks have become one of the most efficient approaches for tracking tasks. A deep network-based tracker usually contains multiple hidden layers and obtains the deep image information through multi-level abstraction. Every level abstracts the feature of the previous level to a higher feature [7].

Among the existing deep trackers, Han [8] designed BranchOut, which is a regularization technique improving the robustness of the CNN tracking model to distractors in cluttered environments. It achieves a light ensemble tracker but loses targets due to occlusion and other challenges. Another example is an end-to-end CNN tracker, called GOTURN [9]. It learns the motion pattern and appearance of objects from video data during an offline stage. This network achieves high-speed tracking performance.

However, it does not use an adaptive method to online fine-tune network. This leads to an inaccurate tracking result when dramatic changes happen in the target's shape and environment. In addition, other deep learning models have also been applied to this field. Valmadre et al. [10] proposed an asymmetric Siamese tracker that was constructed to calculate the similarity score between the training sample and the test image, which achieves promising tracker performance on light architecture. However, it fails in dealing with multi-scale targets due to lacking an adaptive technique. Cui et al. [11] employed a multi-directional recurrent neural network (RNN) to capture the context information in the video sequence. It produces a confidence map suppressing the noise and utilizes the reliable part of the target to enable effective tracking. Nevertheless, the network fails to utilize the robust visual feature representation, which degrades its performance in a target's fast movement. Zhu et al. [12] constructed semantic negative data based on a large-scale dataset to improve the discriminative power. It is further expanded to long-term tracking.

In the above deep-based tracking networks, dense sampling strategy and regular data augmentation are applied on a large-scale dataset to obtain massive training samples at offline stage (Fig. 1). They attempt to solve the problem of over-fitting due to the scarcity of training data. However, the performance of trackers with these methods is unsatisfied for several reasons. Firstly, these collected training data are less diverse and redundant. This makes learning effective feature representation arduous. Secondly, targets with some challenging attributes are not covered by most large-scale datasets. This results in trackers lacking enough robustness for complex tracking scenarios. During an online test, it can be found that each tracking video contains different kinds of targets, moving trajectories, and background information. The tracking algorithm needs to be designed and fine-tuned according to the video-related information.

(a) Dense sampling (b) Regular data augmentation

Fig. 1. Dense sampling and regular data augmentation strategy

To solve the above-mentioned problems, this paper utilizes a deep generation network to provide massive diverse samples for training multiple deep trackers that correspond to various class target objects. Some deep generative networks have been employed in the field of computer vision, like variational auto-encoder (VAE) [13] and generative adversarial networks (GANs) [14]. A VAE performs feature learning on the target manifold and has achieved promising performance in reconstruction. A conditional variational auto-encoder (CVAE) for forecasting the dynamics of a target in the near future was proposed by Walker et al. [15]. The latent space of the VAE captures the important information hidden in the image to determine the target trajectory. Yan et al. [16] employed a hierarchical representation CVAE, which was motivated by structure information of images. This method achieves the generation of realistic images of human faces and birds according to the high-level attribution description. Considering the superior performance of reconstructing varied images, a VAE is employed in this paper to act as a deep generation network to generate samples with diversity. All the training samples are then clustered to different classes. Each class of samples is used to train one of multiple deep-tracking models which have an identical structure. During the test stage, a best-suitable pretrained tracking model is selected according to the target's information. This work aims to improve the robustness of models and avoid over-fitting in challenging tracking tasks, such as deformation, scale variation, motion blur, and partial occlusion. At the same time, it has the ability to tackle different category targets adaptively and effectively.

3 The Proposed Method

A scene-aware tracker is proposed in this paper. Specifically, this system selects a pre-trained tracking network suited for specific types of targets from N_v pre-trained CNN tracking networks. The proposed approach consists of two phases: offline training and online tracking. A diagram of the offline training is shown in Fig. 2. N_v CNN-trackers that have identical architecture are trained using clustered positive samples and negative samples generated by a positive data augmentation module and hard negative data mining, respectively. At the beginning of a tracking phase, according to the initial frame of the given video sequence, a trained selection mechanism is triggered to choose the best suitable pre-trained tracker.

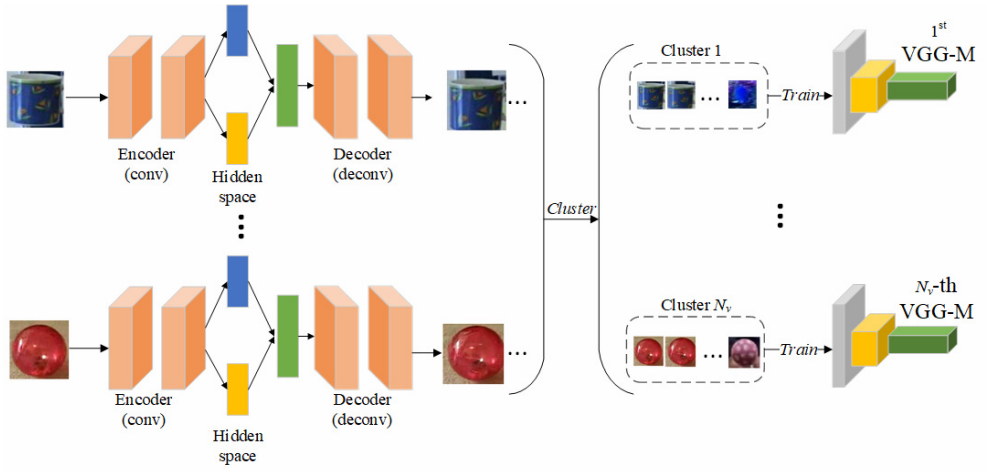


Fig. 2. The offline training diagram of the proposed method

3.1 The Positive Data Augmentation Module (PDAM)

To improve the robustness of the pre-trained tracking model, a positive data generation module is employed to produce varying positive samples with similar features during the offline training stage. To achieve this, offline training is divided into the following steps. A VAE is constructed by combining N_l encoding layers and N_l decoding layers. The VAE is able to discover the target patterns, and thus finds the positive samples that do not occur in the general training data. Convolutional layers are introduced as the encoding layers to improve the robustness of the model. The input of layer $l + 1$ is the output of layer l. The sample mapped to the low-dimensional hidden space facilitates finding the latent pattern of the target. During the decoding process, the decoding part reconstructs the samples along the specific direction and traverses all the patterns.

The original training sample is denoted by x, $p(x)$ is the probability distribution of the training samples, and $p(h)$ is the probability distribution of the hidden variables. Given a hidden variable, the conditional probability of the sample distribution produced is $p(x|h)$, where φ denotes the generative parameters. The computation of the posterior denoted by $p_\varphi(h|x)$ during inference is a complex task. Therefore, the variational lower bound is employed and optimized to estimate the posterior. For the i-th sample x_i, the lower bound loss is formulated using the following equation:

$$L(\varphi,\tau,x_i)=KL\left(q_\tau(h|x_i)\|p_\varphi(h)\right)+\mathbb{E}_{q_\tau(h|x_i)}\left[\log p_\varphi(x_i|h)\right]. \quad \textbf{(1)}$$

In formula (1), $q_\tau(h|x_i)$ denotes the hidden variable. The first part computes the Kullback-Leibler (KL) divergence between the estimation and the ground-truth distribution of the posterior. Furthermore, the restored loss is presented as the second part of the above equation. The goal is to minimize the loss function of formula (1) using gradient descent. To simplify the KL term, it can be defined through formula (2) as follows:

$$KL\left(q_\tau(h|x_i)\|p_\varphi(h)\right)=\frac{1}{2}\sum_{d=1}^{D}\left(1+\log\left(\sigma_d^2\right)-\mu_d^2-\sigma_d^2\right), \quad \textbf{(2)}$$

where d stands for the dimension of the hidden variable h. Here, the mean μ is the output of encoder x while σ denotes the results of the variational parameter τ. To facilitate solving the second part of the loss function, formula (1) is approximated using a differentiable transformation with a noisy term as follows:

$$L(\varphi,\tau,x_i) \approx \frac{1}{2}\sum_{d=1}^{D}\left(1+\log\left(\sigma_d^2\right)-\mu_d^2-\sigma_d^2\right)+\frac{1}{J}\sum_{j=1}^{J}\log p_\varphi\left(x_i\left|h_{i,j}\right.\right). \quad \textbf{(3)}$$

In equation (3), $h_{i,j}$ is calculated using $h_{i,j}=\mu_i+\sigma_i\odot\in_j$, where $\in_j$ follows the normal distribution of $N(0, \mathbf{I})$. To solve the second component in Formula (3), binary cross-entropy is employed. A positive data augment network is trained for each target in different video sequences. Once the training of the VAE is completed. The original training data are provided to the VAE to generate extensive varying positive data for sample clustering. During the generation of positive data, some channel of the feature map is randomly set to 0, which corresponds to an occlusion in some scene and serves to improve the robustness of the tracking network.

The data augmentation network consists of 2 conv-layers, 2 transpose conv-layers (i.e, $N_l = 2$) and a hidden space. For each video sequence, a VAE is trained individually for 10,000 epochs with RMSprop to optimize the loss function. The learning rate is set to 0.001 and the learning process of VAE is conducted offline. Fig. 3 demonstrates some generated hard positive samples. Some of them are blur and part-occluded, which provides the robustness for motion blur, low resolution and occlusion.

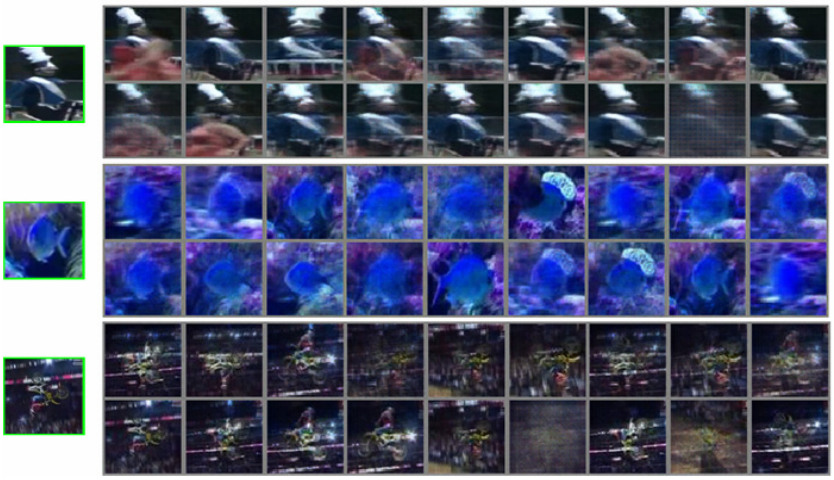


Fig. 3. The original positive (green box) and the generated samples using the proposed method

3.2 Training Samples Clustering

Targets in different scenes present various patterns and features in visual tracking tasks. Let $\boldsymbol{X}_i$ (i = 1,…, N_s) be the set of all training samples, which includes the samples generated using the positive data augmentation network and the original training data.

To cluster the training data, a sample selection was conducted for 2000 iterations and each time included $2N_v$ training samples with different tracking scenes and types of targets. The samples that have the largest Euclidean distance between each other were chosen as the centroids. Then, k-means was employed to cluster all the training samples with k=$2N_v$. Out of the $2N_v$ centroids, the N_v centroids with the fewest samples were discarded. Training data were then clustered with the remaining centroids, which yields N_v clusters and sufficient samples for training the trackers. The cluster index for $\boldsymbol{X}_i$ is denoted as $c_i \in \{1,\ldots, N_v\}$. Each d-th (d =1,…, N_v) tracking net was trained individually with the training samples from the d-th cluster. The number of centroids N_v was 10.

3.3 Selection Mechanism

During the tracking stage, a selection network is employed to choose the best-suited tracking network trained according to the target information of the initial frame. The VGG-M network [17] was pre-trained on ImageNet [4] as the selection network. It included 3 Conv (convolutional) layers (conv1, conv2, and

conv3) and 3 FC (Fully Connected) layers (FC4, FC5, and FC6). The parameters of the three convolutional layers were the same as the corresponding part of VGG-M while the fully connected layers were first initialized using a Gaussian distribution with a zero mean. According to the target in a given video sequence, the final fully connected layer (FC6) is designed to obtain the probability of every tracking network.

The training of this network was conducted by receiving training data $\boldsymbol{X}_i$ and generating the probability of the index d_i that the input sample belonged to. The i-th training sample was formed by the $\boldsymbol{X}_i$, d_i (i=1,…, n) pair, in which n is the size of a mini-batch. The loss function can be presented as follows:

$$L_s = \frac{1}{n}\sum_{i=1}^{n} f\left(d_i, h\left(\boldsymbol{X}_i\right)\right), \tag{4}$$

where $f(\cdot)$ determines the cross-entropy loss between the ground-truth d_i and the estimated cluster index $h(\boldsymbol{X}_i)$. The optimization goal of formula (4) is to minimize the loss function L_s using stochastic gradient descent.

3.4 The CNN Tracking Network

A simplified VGG-M network was adopted for this study as the architecture of the CNNs-tracker demonstrated in the right part of Fig. 1. It began with three convolutional layers (Conv1-Conv3) and was followed by two fully connected layers (FC4-FC5) in a similar manner to VGG-M.

In the offline training stage, the last FC layer with the softmaxloss layer was extended to n branches corresponding to n training video sequences. This gives the network the ability to simultaneously learn multiple training samples in different videos [18], which captures the generic features in different training sequences. Hard negative mining [19] is utilized to avoid the drift problem caused by redundant negative training samples, which enables the tracker to identify the false positive samples effectively and thus improves the tracking accuracy. In addition, the last Fully-connected layer (FC5) calculates the scores of the target and background in the current frame. According to the different clusters generated by the training samples clustering, N_v pre-trained tracking VGG-M networks were pre-trained. Each of these networks had the ability to learn the generic representation for the corresponding category.

The tracker is available for tracking tasks when the learning has been completed. Since objects in the same category may be a distractor in the current video sequence, the pre-trained network needs to be specialized for precision tracking. To achieve this, the FC layers were initialized with the conv-3 feature map of the target in the first frame. This process is described in Section 3.4. In the tracking that follows, online updating for the parameters of the network was carried out in both long and short-term fashion to maintain its robustness and adaptability. The parameter configuration of the online update is presented in Section 4.

3.5 FC Training

As mentioned in the last section, the FC layers need to be initialized and updated at the first frame and following the tracking phase, respectively. That operation provides tracker with better adaptiveness for the various targets and their frequent changes in the given video. The process can be divided into two steps in a similar fashion to training the convolutional layers: forward pass and back-propagation. A mini-batch training dataset consists of x^i and y^i (i = 1, 2, 3,..., b) denoting the video sequences and the corresponding class labels, respectively. j = 1 stands for positive samples and 2 means negative samples. The parameters in the FC layers are expressed using θ_{FC}. At the forward stage, the total loss can be calculated using the following equation:

$$L = -\sum_{i=1}^{b}\sum_{j=1}^{2} y_j^i f_j\left(x^i, \theta_{FC}\right), \tag{5}$$

where $f_j(x_i, \theta_{FC})$ is the output of the softmaxloss layer for the i-th frame. The parameters in the FC layers are determined in the next step by reducing the gradient using equation (6) as follows:

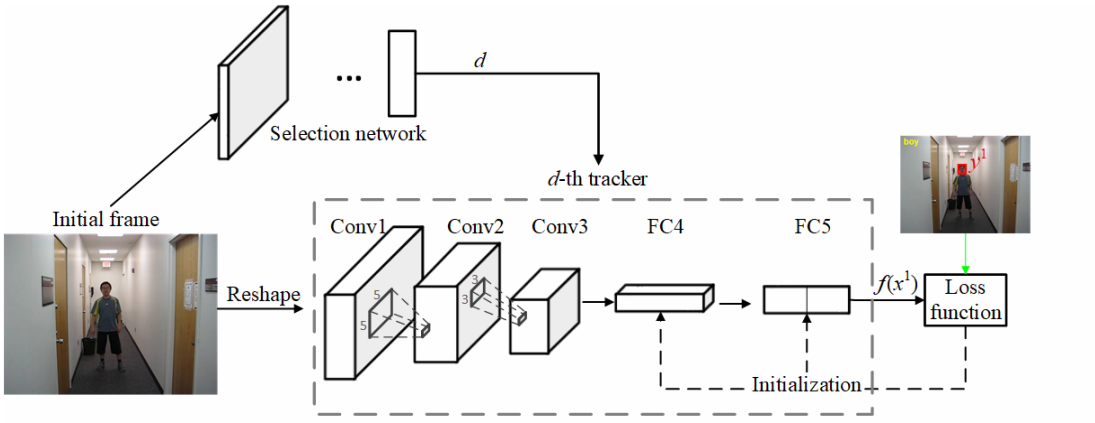
$$\frac{\partial L}{\partial \theta_{FC}} = -\sum_{i=1}^{b} \frac{\partial}{\partial \theta_{FC}} \left[\sum_{j=1}^{2} y_j^i f_j \left(x^i, \theta_{FC} \right) \right]. \tag{6}$$

4 The Tracking Process

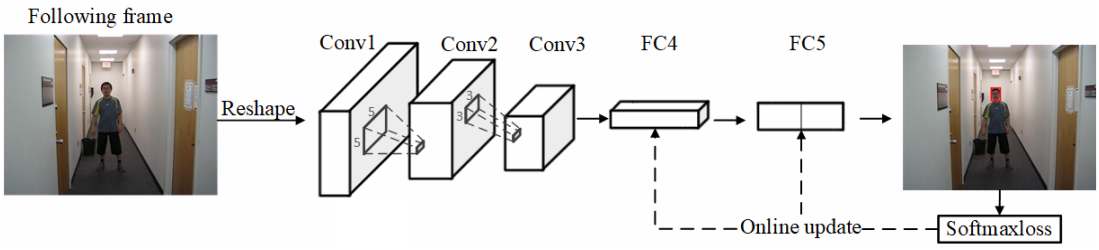
This section presents the tracking network with the proposed method, followed by the configuration of the network and its online update strategy.

4.1 The Tracker Architecture

As shown in Fig. 4, the convolutional layers in each tracking network extract deep feature maps from the cropped 3-dimensional RGB video frame. Between the convolutional layers, a Rectified Linear Unit (ReLU) [20] acts as the activation function to model the neuron's output. Then, a batch normalization layer and a max-pooling down-sampling were added to the feature map to reduce the size of the feature vector and prevent over-fitting. The initial parameters of the convolutional block were obtained using transfer learning. The second component was composed of two FC layers. Between these layers, a ReLU layer and a dropout were inserted to act as the activation function and avoiding over-fitting, respectively. Softmaxloss acts as a loss function during initialization and the online update.



(a) initialization



(b) tracking

Fig. 4. The pipeline of the proposed network during tracking. It consists of two phases

4.2 Implementation Details

During offline training, N_v (N_v = 10) VGG-M networks were trained using N_v clusters training data. For every network, the multi-domain learning method was applied with a learning rate of α_{offline} = 0.0005 and this process included 100 epochs. For every epoch, the training samples in n video sequences were used to train CNN-tracking network with n branches. In each video sequence, a mini training batch included 64 positive and 96 negative samples, respectively, from the proposed positive augmentation network and

hard negative mining technique. This effectively reduces the redundant negative training samples and mitigates the drift that results from a lack of effective negative samples. Once the set training epochs are reached or the loss function converges, the training for the tracking network is completed. As a result, the tracker achieves generic object representations.

At the beginning of the tracking phase (Fig. 2 (a)), the target feature information in the first frame was employed by the selection network to choose the best suitable tracking VGG-M network for the tracking scene. The multi-branch layers in the VGG-M network were replaced with a new single branch FC layer. Positive and negative samples, denoted by $|S_1^+|$ and $|S_1^-|$, were extracted based on the target information in the first frame together with softmaxloss to initialize the FC4 and FC5 layers. Specifically, $|S_1^+|$=500 and $|S_1^-|$=5000. The network starts its tracking loop once the initialization of the FC part has finished. Following each frame, N (N=256) object samples were captured around the target state of the previous frame, and then the network evaluates these samples to acquire their scores of positive $f^+(x_i)$. The desired object state x^* is obtained by finding the sample with the highest positive score. This is expressed using equation (7). The bounding box containing the target state x^* was precisely placed using the bounding box regression model [21], which has been widely applied in object detection.

$$x^* = \underset{x^i}{\operatorname{argmax}} f^+\left(x^i\right). \tag{7}$$

4.3 Online Update

To improve the robustness and adaptiveness to object variants, it is necessary for the tracker to update the network during tracking. As shown in Fig. 2(b), the online update was activated at long and short-term periods. When the target state score x^* in formula(7) was below 0.5, a short-term update was triggered using the conv3 feature of positives samples collected in the latest 20 frames in which successful tracking was achieved. In the long-term case, that is, every 20 frames, the FC part was fine-tuned using the conv3 feature of positive samples from the most recent successfully tracked 100 frames. In the above-stated situations, negative samples consisted of samples collected from the short-term period. Unlike the initialization process, we collected 50 positives and 200 negatives per frame as the training data to update our model. For each online update, the FC layers were trained 30 times with a learning rate α_{online} = 0.0003, momentum $\in_{\text{online}}$ = 0.9 for accelerating learning, and a weight decay rate of 0.0005.

5 Experiment

In this section, we evaluated the proposed method and compared it with other state-of-the-art visual trackers on the visual tracking benchmark dataset, OTB100 [22].

5.1 Experimental Setup

For the generation of training samples, 20 videos from the ImageNet video [4] and VOT 2016 [23] datasets were selected as the original training video sequences. It should be noted that the training datasets were different from the test videos for a fair comparison. Whole training data were clustered to N_v=10 classes. Therefore, 10 VGG-M tracking models were trained corresponding to the different training data category. The parameter settings of the proposed work during the online test were the same as in Sections 4.2 and 4.3. The proposed framework was implemented using MATLAB with MatConvNet [24]. The computational platform consisted of an Intel Xeon E5-2643 CPU @ 3.40G Hz, 32GB RAM, and an Nvidia GTX1080Ti GPU.

5.2 Test Dataset

OTB100 is employed as the test dataset in this paper. It contains 100 individual video clips with diverse target objects and various durations. It includes common challenging factors in visual tracking, such as low resolution, deformation, in-plane rotation, out-of-view target, etc. Some examples from this benchmark dataset are presented in Fig. 5.

Fig. 5. Some examples from OTB100

To evaluate the tracking performance, some metrics need to be introduced, that is, the mean center error, mean overlap rate, and success rate. The mean center error is based on the measurement of the Euclidean distance between the estimated location center and the ground truth location center of all the video sequences on a frame-by-frame basis. The overlap rate is obtained through o_s, as defined in equation (8).

$$o_s = \frac{\left| r_t \cap r_g \right|}{\left| r_t \cup r_g \right|}, \tag{8}$$

where $\cup$ and $\cap$ mean the union and intersection between two parts, respectively; r_t stands for the target bounding box; and r_g denotes the ground truth box. $|\cdot|$ is used to calculate the number of pixels. The mean overlap rate is obtained by calculating the mean value of o_s on all involved frames in a video. In addition, the frame on which o_s is greater than a set threshold is considered to be a successful frame. The success rate is defined by the percentage of successful frames on a video sequence.

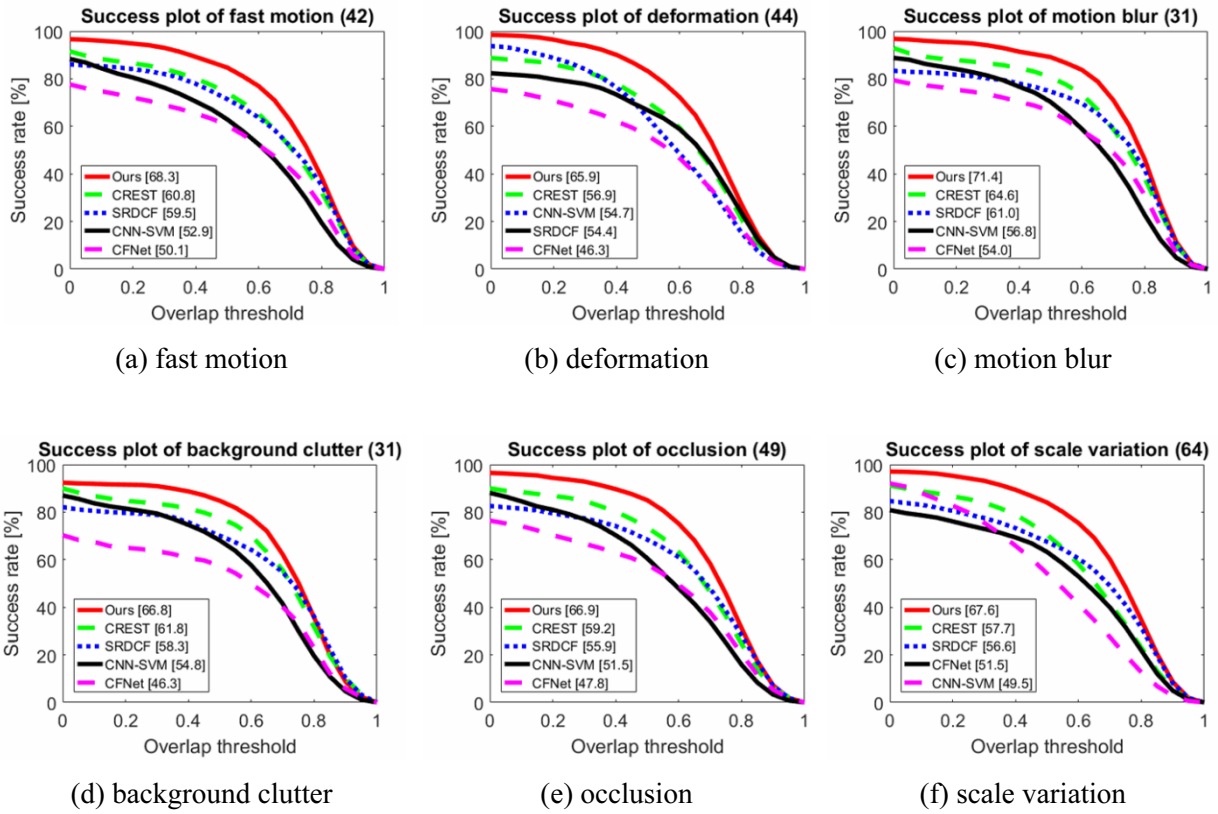
5.3 State-of-the-Art Comparison

The proposed network was compared with some state-of-the-art approaches:
CFNet. This approach obtains an end to end lightweight tracking network by achieving the deep integration of the correlation filter and CNNs [10]. It shows fast and promising tracking accuracy.
CNN-SVM. This method employs a CNN to generate the feature representation of the object [3]. These features are used for the SVM to obtain the saliency maps based on positive samples. The position of the target is determined with the saliency maps.
SRDCF. This method designs a spatial regularization function to deal with the boundary effects and uses various scale searches to solve the problem of scale change [25]. The iterative Gauss-Seidel method is introduced to simplify calculation during the learning process.
CREST. This method uses DCF as a layer of a CNN-based tracker to integrate feature extraction, response map generation, and model update into a neural network for end-to-end training [26]. Residual learning is utilized to update the tracking model during the online stage.

Among these trackers, CFNet, CNN-SVM, and CREST are built based on a deep-learning architecture that employs the deep convolutional feature for target description. Both CFNet and CREST introduce the correlation filter into their network to form the end-to-end training tracker. As for SRDCF, it constructs a tracking system based on a discriminative correlation filter and uses manually selected features. The testing results of these four trackers are provided by their authors.

The comparison result is presented in Fig. 6. It shows success plots of all the methods in some challenging scenes that have difficulty due to the target's movement. The plots were obtained based on the overlap rate o_s mentioned in formula (8) and given a threshold from 0 to 1. The Area-Under-Curve (AUC) score for each tracker is reported in the legend and was computed in the range from 0 to 1. As shown in Fig. 6, the proposed method achieved the best performance in all six scenes. Its curve of

performance is higher than other methods with an obvious margin.



(a) fast motion (b) deformation (c) motion blur

(d) background clutter (e) occlusion (f) scale variation

Fig. 6. The success plots on motion-related challenging scenes

Compared with the second-best method CREST, the proposed work provided a better AUC score in around 66% of the cases of deformation and scale variation. It outperformed CREST by 10% in AUC score. Regarding CFNet, it uses a CNN to extract the features of objects, and a lack of robustness can be observed from the above challenging scenes. The proposed method achieved a significant gain of above 10% in all the presented scenes. In addition, CNN-SVM applies a large-scale dataset for training due to the inherent drawback of a deep-learning network. However, it did not take any further process for training samples. Its performance is far behind the proposed method in the above six tracking scenes. SRDCF showed promising results in most scenes due to overcoming the boundary effect and scale change; however, it degrades in the deformation cases since hand-crafted features fail to handle dramatic changes in the target. The inferior performance of above trackers can be attributed to a lack in any further effective process for training samples. In contrast, our tracker employs the hard samples generation module to provide various representations of objects to ensure the robustness. Besides, the test videos contain different classes of target objects. The selection mechanism enables adaptive tracking according to the feature information of various targets. On another hand, the proposed method achieved the highest AUC on motion blur among all the scenes. Considering that the samples generated by the VAE were blurry, the trained model was more robust to blurred targets caused by fast movement.

5.4 Qualitative Analysis

Due to space limitations, the visualized tracking results of some crucial frames for all the compared methods on two representative hard video sequences are shown below.

Bird1. Bird1 is one of the most difficult tracking video clips. It focuses on some challenging attributes: deformation; fast motion; and scale variation, which is closely related to the target's motion. The tracking results of some key frames are visualized in Fig. 7. In the 120th frame, SRDCF lost the target bird due to the continuous change of the bird's shape before it flew into the cloud. At the 140th frame, the target is in the cloud and invisible. The proposed method and SRDCF attempted to search for the target due to an

awareness of the target's disappearance while the other three trackers did not conduct the operation. This state lasted for a while until the 185th frame when the target passed through the cloud and appeared with dramatic motion. The proposed method enabled re-identification and tracking of the target. We consider that this can be attributed to the power of various samples learning and long-term online update. The problem of target drift arises in other methods since they fail to handle long-term occlusion. Deformation happens in the 290th frame during the sudden movement of the target to the right. The proposed method kept tracking the target as opposed to other trackers losing the target until the end of the video. This illustrates that the proposed method is robust to the targets that undergo fast motion and deformation. We believe that this can be attributed to the powerful ability of representation obtained through the generated hard positive data. It should also be pointed out that deep feature-based methods outperformed manual feature-based method SRDCF. This result verifies the success plot of deformation and fast motion shown in Fig. 6.

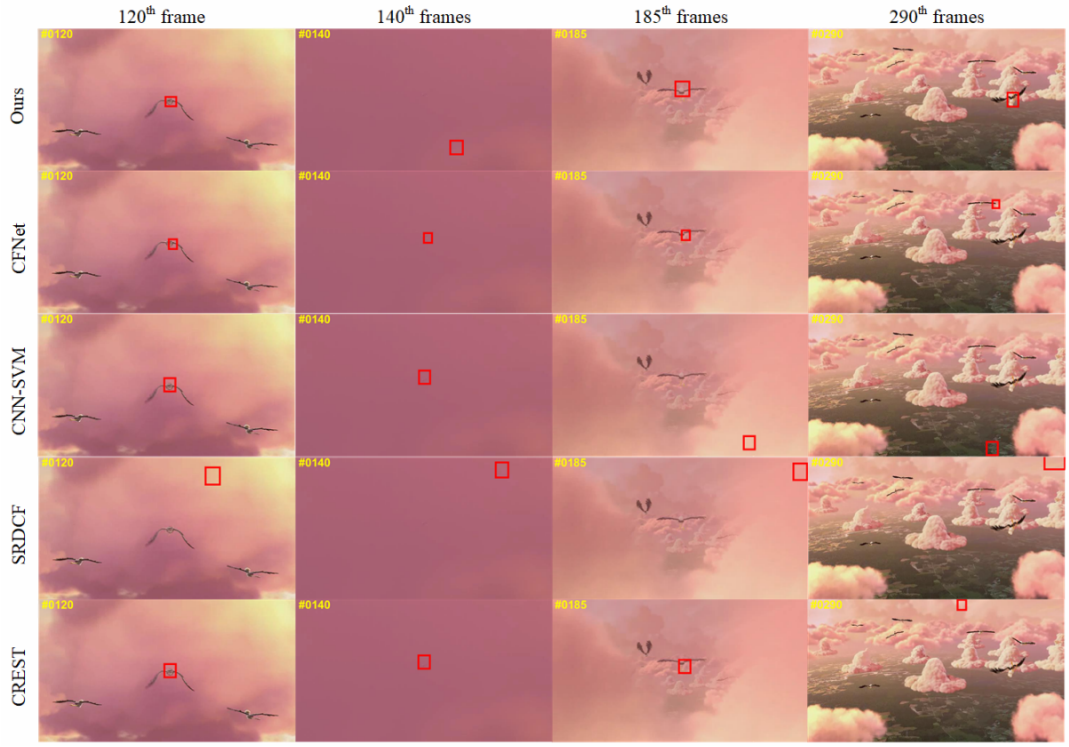
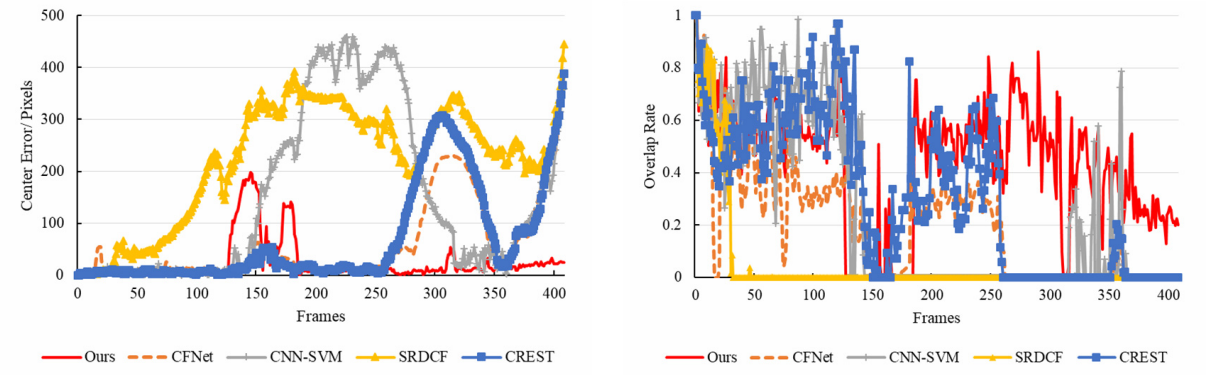


Fig. 7. A visualization of the tracking result for Bird1

Fig. 8 presents the overall performance of all the involved trackers frame by frame on Bird1. Compared with the other four methods, the proposed method achieved a more stable and accurate tracking performance. In most parts of this video clip, the center error of the proposed method was the smallest among all the trackers (Fig. 8(a)). Although CREST and CFNet achieved satisfactory error values at an early stage, they gradually lost the target due to dramatic deformation and fast movement. For the overlap rate, the proposed method stably tracked the target before the 150th frame. After that, the disappearance of the target led to a decrease in the overlap rate. When the target appeared again, the proposed method rapidly found it and tracked it to the end of the video. However, the other methods consecutively failed to capture the target's trajectory because of its fast movement and dramatic deformation. This situation is reflected with great fluctuation in performance plots. Table 1 employs the evaluation metrics mentioned in Section 5.1 to show the overall performance on Bird1. The success rate is defined as the percentage of frames over the whole video sequence in which the overlap rate is greater than 0.5. The proposed method achieved superior results in all three metrics. This means that the proposed method is more robust than the existing trackers in videos involving fast motion, motion blur, and deformation.



(a) Center error plot (b) Overlap rate plot

Fig. 8. Overall performance plot on Bird1

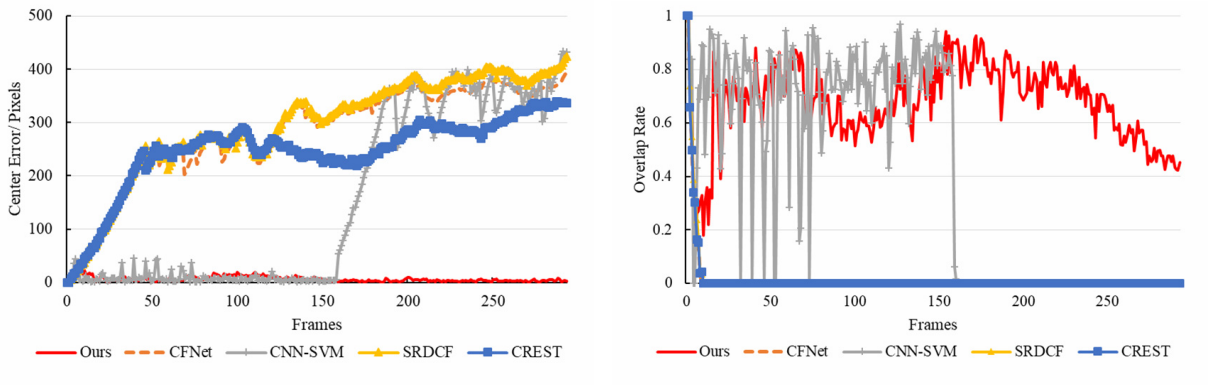
Table 1. A summary of the overall performance on Bird1

Trackers	Mean Center Error/Pixels	Mean Overlap Rate (%)	Success Rate (%)
Ours	23.8765	43.3896	50.9804
CFNet	58.4603	19.3849	7.8431
CNN-SVM	149.2762	24.0043	29.4118
SRDCF	223.0042	4.8483	6.3725
CREST	68.0479	30.9579	32.1078

Bolt 2. Another video sequence is Bolt 2. It contains fast and dramatic deformation of the target, rapid movement, and similar distractors. The target is the fourth athlete from the right side. As shown in the 16th frame, trackers employing DCF, CFNet, SRDCF, and CREST wrongly recognized the neighboring people with similar appearances as the target. At the 160th frame, CFNet and SRDCF completely lost the target and fall into the local region. CREST fails to accurately capture the target. It should be pointed out that drift appears in CNN-SVM when people appeared near the target wearing similar white clothing. In the 216th frame, all the other trackers failed to locate the target object and lost it. The proposed method stably tracked the target and shows robustness to a dramatic change of target's shape and similar distractors.

Fig. 9. A visualization of tracking result on Bolt 2

The overall performance on this video was plotted frame by frame in the same manner as before. In both plots, the proposed tracker outperformed the other methods and achieved a stable tracking performance. SRDCF, CREST, and CFNet failed to estimate the location of the target at the beginning of this video. CREST's failure is due to merely employing a feature from a single convolutional layer. Meanwhile, it does not apply a further adaptive strategy to tackle targets in a different category, which is same as CFNet. Although CNN-SVM started with the satisfactory tracking results, it lost the target in the 160^{th} frame due to a man with similar clothing near to the target. This can be attributed to the insensitivity of CNN-SVM for intra-class variation. In contrast, the proposed method focused on the variations of the target and kept tracking to the end of the video sequence. This is a benefit of learning from hard positive samples with diversity. Table 2 summarizes the performance results in Fig. 10 and gives the mean center error, mean overlap rate, and success rate for the compared trackers. This validates the effectiveness of our method to tackle tracking with deformation, fast motion, and a similar distractor. Besides, even though the targets in the above videos belong to different categories, the proposed framework achieves superior adaptive tracking performance. This indicates that our online selection mechanism assists the model in conducting adaptive tracking.



(a) Center error plot

(b) Overlap rate plot

Fig. 10. The overall performance plot on Bolt 2

Table 2. A summary of the overall performance on Bolt 2

Trackers	Mean Center Error /Pixels	Mean Overlap Rate (%)	Success Rate (%)
Ours	6.1476	67.7688	88.3959
CFNet	283.2965	1.1208	1.0239
CNN-SVM	151.7323	39.1783	47.7816
SRDCF	295.9071	1.1852	0.6826
CREST	246.2414	1.0911	0.3413

6 Conclusion

This paper proposes a combined and efficient method to mitigate over-fitting and improve the robustness of deep learning-based tracking network in challenging cases of visual tracking. In the offline training stage, a positive training data augmentation module (PDAM) is designed and employed to provide abundant and varying samples. Then, the training data are clustered according to their patterns. For different sample clusters, multiple CNNs are trained using specific sample clusters. During the tracking phase, a selection mechanism is used to select the best suited trained CNNs for a given video sequence. To verify the effectiveness of our method, the proposed method was compared with state-of-the-art trackers on the OTB100 visual tracking benchmark dataset. The experimental results showed that the proposed method achieved outstanding tracking accuracy and robustness, especially in cases of deformation, fast motion, scale variation, and motion blur. In future work, this scheme can be integrated with an effective baseline to be applied in traffic scenes and with a focus on runtime speed.

References

[1] H. Nam, B. Han, Learning multi-domain convolutional neural networks for visual tracking, in: Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[2] L. Wang, W. Ouyang, X. Wang, H. Lu, Visual tracking with fully convolutional networks, in: Proc. 2015 IEEE International Conference on Computer Vision (ICCV), 2015.

[3] S. Hong, T. You, S. Kwak, B. Han. Online tracking by learning discriminative saliency map with convolutional neural network. in: Proc. 2015 International Conference on Machine Learning, 2015.

[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, F.-F. Li, Imagenet large scale visual recognition challenge, International Journal of Computer Vision (IJCV) 115(3)(2015) 211-252.

[5] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C.L Zitnick, P. Dollár, Microsoft COCO: common objects in context, in: Proc. 2014 European Conference on Computer Vision (ECCV), 2014.

[6] T. Tran, T. Pham, G. Carneiro, L. Palmer, I. Reid, A Bayesian data augmentation approach for learning deep models, in: Proc. 2017 Advances in Neural Information Processing Systems, 2017.

[7] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521(2015) 436-444.

[8] B. Han, J. Sim, H. Adam, BranchOut: regularization for online ensemble tracking with convolutional neural networks, in: Proc. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2017.

[9] D. Held, S. Thrun, S. Savarese, Learning to track at 100 FPS with deep regression networks, in: Proc. 2016 European Conference on Computer Vision (ECCV), 2016.

[10] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, P.-H. Torr, End-to-End representation learning for correlation filter based tracking, in: Proc. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[11] Z. Cui, S. Xiao, J. Feng, S. Yan. Recurrently target-attending tracking, in: Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[12] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, W. Hu, Distractor-aware Siamese networks for visual object tracking, In: Proc. 2018 European Conference on Computer Vision (ECCV), 2018.

[13] D. P. Kingma, M. Welling, Auto-encoding variational bayes. <https://arxiv.org/pdf/1312.6114.pdf>, 2014 (accessed 18.10.13).

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Proc. 2014 International Conference on Neural Information Processing Systems (NIPS), 2014.

[15] J. Walker, C. Doersch, A. Gupta, M. Hebert, An uncertain future: forecasting from static images using variational autoencoders, in: Proc. 2016 European Conference on Computer Vision (ECCV), 2016.

[16] X. Yan, J. Yang, K. Sohn, H Lee, Attribute2Image: conditional image generation from visual attributes, in: Proc. 2016 European Conference on Computer Vision (ECCV), 2016.

[17] K. Chatfield, K. Simonyan, A. Vedaldi, A. Zisserman, Return of the devil in the details: delving deep into convolutional nets, in: Proc. 2014 British Machine Vision Conference (BMVC), 2014.

[18] M. Dredze, A. Kulesza, K. Crammer, Multi-domain learning by confidence-weighted parameter combination, Machine Learning 79(1-2)(2010) 123-149.

[19] K.-K. Sung, T. Poggio, Example based learning for view-based human face detection, Transactions on Pattern Analysis and Machine Intelligence 20(1)(1998) 39-51.

[20] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: Proc. 2012 International Conference on Neural Information Processing Systems (NIPS), 2012.

[21] R. Girshicket, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proc. 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2014.

[22] Y. Wu, J. Lim, M.-H. Yang, Online object tracking, IEEE Transactions on Pattern Analysis and Machine Intelligence 37(9)(2015) 1834-1848.

[23] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. ehovin, T. Vojr, G. Hger, A. Lukei, G. Fernndez, The visual object tracking VOT2016 challenge results, in: Proc. 2016 European Conference on Computer Vision (ECCV), 2016.

[24] A. Vedaldi, K. Lenc, MatConvNet: convolutional neural networks for MATLAB, in: Proc. the 23rd ACM International Conf. on Multimedia, 2015.

[25] M. Danelljan, G. Hager, F.S. Khan, M. Felsberg, Learning spatially regularized correlation filters for visual tracking, in: Proc. 2015 IEEE International Conference on Computer Vision (ICCV), 2015.

[26] Y. Song, C. Ma, L. Gong, J. Zhang, R. W.H. Lau, M.-H. Yang, CREST: convolutional residual learning for visual tracking, in: Proc. 2017 IEEE International Conference on Computer Vision (ICCV), 2017.