Visual Tracking with Fully Convolutional Networks

Lijun Wang[1,2], Wanli Ouyang[2], Xiaogang Wang[2], and Huchuan Lu[1]
[1]Dalian University of Technology, China
[2]The Chinese University of Hong Kong, Hong Kong, China

Abstract

We propose a new approach for general object tracking with fully convolutional neural network. Instead of treating convolutional neural network (CNN) as a black-box feature extractor, we conduct in-depth study on the properties of CNN features offline pre-trained on massive image data and classification task on ImageNet. The discoveries motivate the design of our tracking system. It is found that convolutional layers in different levels characterize the target from different perspectives. A top layer encodes more semantic features and serves as a category detector, while a lower layer carries more discriminative information and can better separate the target from distracters with similar appearance. Both layers are jointly used with a switch mechanism during tracking. It is also found that for a tracking target, only a subset of neurons are relevant. A feature map selection method is developed to remove noisy and irrelevant feature maps, which can reduce computation redundancy and improve tracking accuracy. Extensive evaluation on the widely used tracking benchmark [36] shows that the proposed tacker outperforms the state-of-the-art significantly.

1. Introduction

Visual tracking, as a fundamental problem in computer vision, has found wide applications. Although much progress [7, 29, 38] has been made in the past decade, tremendous challenges still exist in designing a robust tracker that can well handle significant appearance changes, pose variations, severe occlusions, and background clutters.

Existing appearance-based tracking methods adopt either generative or discriminative models to separate the foreground from background and distinct co-occurring objects. One major drawback is that they rely on low-level hand-crafted features which are incapable to capture semantic information of targets, not robust to significant appearance changes, and only have limited discriminative power.

Driven by the emergence of large-scale visual data sets and fast development of computation power, Deep Neural Networks (DNNs), especially convolutional neural networks [17] (CNNs), with their strong capabilities of learning feature representations, have demonstrated record breaking performance in computer vision tasks, *e.g.*, image classification [16, 27], object detection [8, 23, 22], and saliency detection [33, 39]. Different from hand-crafted features, those learned by CNNs from massive annotated visual data and a large number of object classes (such as ImageNet [4]) carry rich high-level semantic information and are strong at distinguishing objects of different categories. These features have good generalization capability across data sets. Recent studies [28, 1] have also shown that such features are robust to data corruption. Their neuron respons-

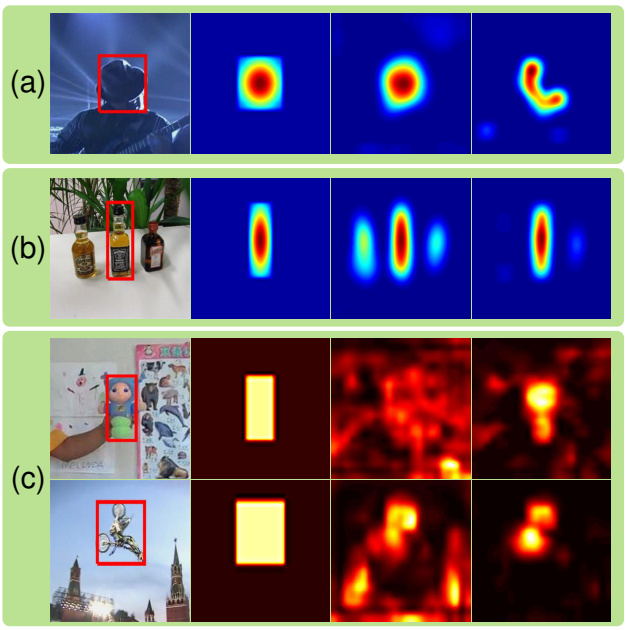


Figure 1. Feature maps for target localization. (a)(b) From left to right: input image, the ground truth target heat map, the predicted heat maps using feature maps of conv5-3 and conv4-3 layers of VGG network [27] (See Section 4.2 for the regression method). (c) From left to right: input image, ground truth foreground mask, average feature maps of conv5-3 (top) and conv4-3 (bottom) layers, average selected feature maps conv5-3 (top) and conv4-3 (bottom) layers (See Section 4.1 for the feature map selection method).

1550-5499/15 $31.00 © 2015 IEEE
DOI 10.1109/ICCV.2015.357

es have strong selectiveness on object identities, *i.e*., for a particular object only a subset of neurons are responded and different objects have different responding neurons.

All these motivate us to apply CNNs to address above challenges faced by tracking. Given the limited number of training samples in online tracking and the complexity of deep models, it is inferior to directly apply CNNs to tracking, since the power of CNNs relies on large-scale training. Prior works [6, 35, 12] attempted to transfer offline learned DNN features (*e.g*. from ImageNet) for online tracking and achieved state-of-the-art performance. However, DNN was treated as a black-box classifier in these works. In contrast, we conduct in-depth study on the properties of CNN features from the perspective of online visual tracking and in order to make better use of them in terms of both efficiency and accuracy. Two such properties are discovered and they motivate the design of our tracking system.

First, CNN features at different levels/depths have different properties that fit the tracking problem. A top convolutional layer captures more abstract and high-level semantic features. They are strong at distinguishing objects of different classes and are very robust to deformation and occlusion as shown in Figure 1 (a). However, they are less discriminative to objects of the same category as shown by the examples in Figure 1 (b). A lower layer provides more detailed local features which help to separate the target from distracters (*e.g*. other objects in the same class) with similar appearance as shown in Figure 1 (b). But they are less robust to dramatic change of appearance, as shown in Figure 1 (a). Based on these observations, we propose to automatically switch the usage of these two layers during tracking depending on the occurrence of distracters.

Second, the CNN features pre-trained on ImageNet are for distinguishing generic objects. However, for a particular target, not all the features are useful for robust tracking. Some feature responses may serve as noise. As shown in Figure 1 (c), it is hard to distinguish the target object from background if all the feature maps are used. In contrast, through proper feature selection, the noisy feature maps not related to the representation of the target are cleared out and the remaining ones can more accurately highlight the target and suppress responses from background. We propose a principled method to select discriminative feature maps and discard noisy or unrelated ones for the tracking target.

The contributions of this work are three folds:

i) We analyze CNN features learned from the large-scale image classification task and find important properties for online tracking. It facilitates further understanding of CNN features and helps to design effective CNN-based trackers.

ii) We propose a new tracking method which jointly considers two convolutional layers of different levels so that they complement each other in handling drastic appearance change and distinguishing target object from its similar distracters. This design significantly mitigate drifts.

iii) We develop a principled method which automatically selects discriminative feature maps and discards noisy or unrelated ones, further improving tracking accuracy.

Evaluation on the widely used tracking benchmark [36] shows that the proposed method well handles a variety of challenging problems and outperforms state-of-the-art methods.

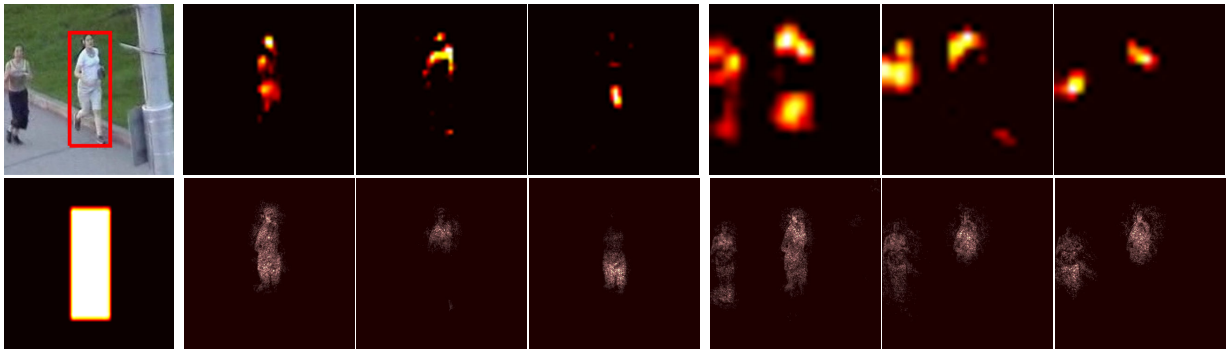
2. Related Work

A tracker contains two components: an appearance model updated online and a search strategy to find the most likely target locations. Most recent works [2, 41, 11, 20] focus on the design of appearance models. In generative models, candidates are searched to minimize reconstruction errors. For example, Ross *et al*. [25] learned subspace online to model target appearance. Recently, sparse coding has been exploited for tracking [21, 3, 32, 31, 30], where the target is reconstructed by a sparse linear combination of target templates. In discriminative models, tracking is cast as a foreground and background separation problem [24, 37, 34]. Online learning algorithms based on CRFs [24], boosting [9], multiple instance learning [2] and structured SVM [10] were applied in tracking and achieved good performance. In [40], the generative and discriminative models were incorporated for more accurate online tracking. All these methods used hand-crafted features.

The application of DNNs in online tracking is under fully explored. In [35], a stacked denoising autoencoder (SDAE) was offline trained on an auxiliary tiny image data set to learn generic features and then used for online tracking. In [19], tracking was performed as foreground-background classification with CNN trained online without offline pre-training. Fan *et al*. [6] used fully convolutional network for human tracking. It took the whole frame as input and predicted the foreground heat map by one-pass forward propagation. Redundant computation was saved. Whereas [35] and [19] operated in a patch-by-by scanning manner. Given N patches cropped from the frame, DNNs had to be evaluated for N times. The overlap between patches leads to a lot of redundant computation. In [12], pre-trained CNN features were used to construct target-specific saliency maps for online tracking. Existing works treated DNNs as black-box feature extractors. Our contributions summarized in Section 1 were not explored in these works.

3. Deep Feature Analysis for Visual Tracking

Analysis on deep representations is important to understand the mechanism of deep learning. However, it is still very rare for the purpose of visual tracking. In this section, we present some important properties of CNN features which can better facilitate visual tracking. Our feature analysis is conducted based on the 16-layer VGG network [27]



(a) (b) (c)

Figure 2. CNNs trained on image classification task carry spatial configuration information. (a) input image (top) and ground truth foreground mask. (b) feature maps (top row) of conv4-3 layer which are activated within the target region and are discriminative to the background distracter. Their associated saliency maps (bottom row) are mainly focused on the target region. (c) feature maps (top row) of conv5-3 layer which are activated within the target region and capture more semantic information of the category (both the target and background distracter). Their saliency maps (bottom row) present spatial information of the category.

pre-trained on the ImageNet image classification task [4], which consists of 13 convolutional layers followed by 3 fully connected layers. We mainly focus on the conv4-3 layer (the 10-th convolutional layer) and the conv5-3 layer (the 13-th convolutional layer), both of which generate 512 feature maps.

Observation 1 *Although the receptive field* [1] *of CNN feature maps is large, the activated feature maps are sparse and localized. The activated regions are highly correlated to the regions of semantic objects* .

Due to pooling and convolutional layers, the receptive fields of the conv4-3 and conv5-3 layers are very large (92×92 and 196×196 pixels, respectively). Figure 2 shows some feature maps with the maximum activation values in the object region. It can be seen that the feature maps have only small regions with nonzero values. These nonzero values are localized and mainly correspond to the image region of foreground objects. We also use the approach in [26] to obtain the saliency maps of CNN features. The saliency maps in Figure 2 (bottom row) show that the change of input that results in the largest increase of the selected feature maps are located within the object regions. Therefore, the feature maps are capturing the visual representation related to the objects. These evidences indicate that DNN features learned from image classification are localized and correlated to the object visual cues. Thus, these CNN features can be used for target localization.

Observation 2 *Many CNN feature maps are noisy or unrelated for the task of discriminating a particular target from its background.*

The CNN features pre-trained on ImageNet can describe a large variety of generic objects and therefore they are supposed to detect abundant visual patterns with a large number of neurons. However, when tracking a particular target object, it should focuses on a much smaller subset of visual patterns which well separate the target from background. As illustrated in Figure 1 (c), the average of all the feature maps is cluttered with background noise. And we should discard feature maps that have high response in both target region and background region so that the tracker does not drift to the background regions. Figure 3 shows the histograms of the activation values for all the feature maps within the object region. The activation value of a feature map is defined as the sum of its responses in the object region. As demonstrated in Figure 3, most of the feature maps have small or zero values within the object region. Therefore, there are lots of feature maps that are not related to the target object. This property provides us the possibility in selecting only a small number of feature maps with small degradation in tracking performance.

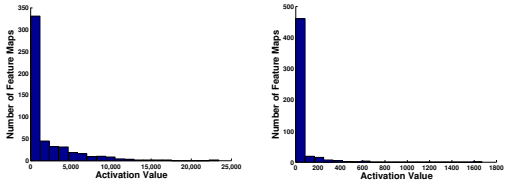


Figure 3. Activation value histograms of feature maps in conv4-3 (left) and conv5-3 (right).

Observation 3 *Different layers encode different types of features. Higher layers capture semantic concepts on object categories, whereas lower layers encode more discriminative features to capture intra class variations.*

Because of the redundancy of feature maps, we employ a sparse representation scheme to facilitate better visualization. By feeding forward an image of an object through the VGG network, we obtain the feature maps $\mathbf{F} \in \mathbb{R}^{d\times n}$ of a convolutional layer (either the conv4-3 or conv5-3 layer),

[1] We use the term *receptive field* to denote the input image region that are connected to a particular neuron in the feature maps

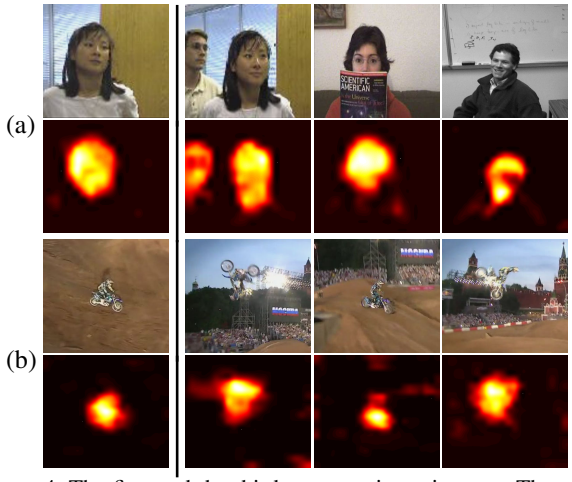


Figure 4. The first and the third rows are input images. The second and the fourth rows are reconstructed foreground masks using conv5-3 feature maps. The sparse coefficients are computed using the images in the first column and directly applied to the other columns without change.

where each feature map is reshaped into a d-dimensional vector and n denotes the number of feature maps. We further associate the image with a foreground mask $\boldsymbol{\pi} \in \mathbb{R}^{d\times 1}$, where the i-th element $\pi_i = 1$ if the i-th neuron of each feature map is located within the foreground object, and $\pi_i = 0$, otherwise. We reconstruct the foreground mask using a subset of the feature maps by solving

$$\begin{aligned} \min_{\mathbf{c}} \|\boldsymbol{\pi} - \mathbf{F}\mathbf{c}\|_2^2 + \lambda\|\mathbf{c}\|_1, \\ \text{s.t. } \mathbf{c} \succeq 0, \end{aligned} \tag{1}$$

where $\mathbf{c} \in \mathbb{R}^{n\times 1}$ is the sparse coefficient vector, and λ the parameter to balance the reconstruction error and sparsity.

Figure 4 shows some of the reconstructed foreground masks using the feature maps of conv5-3. For the two examples (face and motorcycle) in Figure 4, we only compute the sparse coefficients for images in the first column and use the coefficients to reconstruct foreground masks for the rest of the columns. The selected feature maps in Figure 4 (a) capture the semantic concepts of human faces and are robust to faces with appearance variation and even identity change. The selected feature maps in Figure 4 (b) accurately separate the target from cluttered background and are invariant to pose variation and rotation. Although trained on the image classification task, the high-level semantic representation of object categories encoded by the conv5-3 layer enables object localization. However, these features are not discriminative enough to different objects of the same category, thus they can not be directly applied to visual tracking.

Compared with the conv5-3 feature maps, the features captured by conv4-3 are more sensitive to intra-class appearance variation. In Figure 2, the selected feature maps of conv4-3 can well separate the target person from the other non-target person. Besides, different feature maps focus on different object parts.

Table 1. Face classification accuracy using different feature maps. Experiment 1 is to classify face and non-face. Experiment 2 is classify face identities.

Feature map	Experiment 1	Experiment 2
conv4-3	76.42%	83.83%
conv5-3	89.56%	57.28%

To further verify this, we conduct two quantitative experiments. 1800 human face images belonging to six identities and 2000 images containing non-face objects are collected from the benchmark sequences [36]. Each image is associated with a foreground mask to indicate the region of the foreground object. In the first experiment, we evaluate the accuracy in classifying the images into face and non-face using the conv4-3 and conv5-3 layers separately. Three face images belonging to three identities are selected as positive training samples to compute a set of sparse coefficients $\{\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_2\}$ via (1). At the test stage, given the feature maps $\mathbf{F}$ and the foreground mask $\boldsymbol{\pi}$ of an input image, the reconstruction error e for the foreground map is computed by

$$e = \min_i \|\boldsymbol{\pi} - \mathbf{F}\mathbf{c}_i\|_2^2. \tag{2}$$

The image is classified as a face image if its reconstruction error e is less than a predefined threshold. Otherwise, it is classified as a non-face image.

In the second experiment, our task is to classify all the face images into different identities. For each identity, 20 images are used as the training samples to learn the sparse coefficients $\mathbf{c}_i,\ i = 1, 2, \ldots, 6$ using (1). At the test stage, the foreground mask reconstruction error for each identity is calculated, and the test image is classified as the identity that has the minimum error as follows:

$$id = \arg\min_i \|\boldsymbol{\pi} - \mathbf{F}\mathbf{c}_i\|_2^2. \tag{3}$$

The classification accuracy using the feature maps of conv4-3 and conv5-3 for the two experiments are demonstrated in Table 1. The feature maps of conv5-3 encode high level semantic information and can better separate face from non-face objects. But they achieve lower accuracy than the features maps of conv4-3 in discriminating one identity from another. The feature maps of conv4-3 preserve more middle-level information and enables more accurate classification of different images belonging to the same category (human faces). But they are worse than the feature maps of conv5-3 in discriminating face from non-face. These results motivate us to consider these two layers jointly for more robust tracking.

4. Proposed Algorithm

An overview (Figure 5) of the proposed fully convolutional network based tracker (FCNT) is as follows:

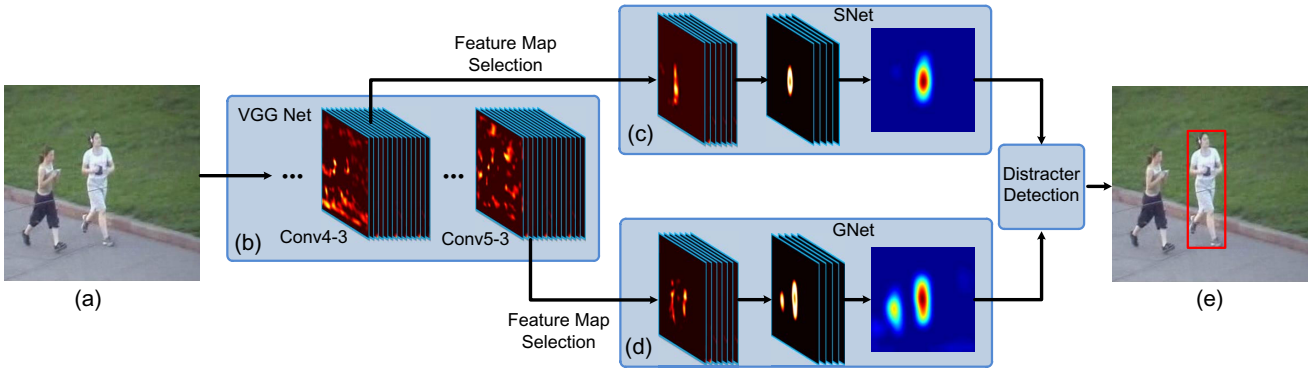


Figure 5. Pipeline of our algorithm. (a) Input ROI region. (b) VGG network. (c) SNet. (d) GNet. (e) Tracking results.

1. For a given target, a feature map selection process is performed on the conv4-3 and conv5-3 layers of the VGG network to select the most relevant feature maps and avoid overfitting on noisy ones.

2. A general network (GNet) that captures the category information of the target is built on top of the selected feature maps of the conv5-3 layer.

3. A specific network (SNet) that discriminates the target from background with similar appearance is built on top of the selected feature maps of the conv4-3 layer.

4. Both GNet and SNet are initialized in the first frame to perform foreground heat map regression for the target and adopt different online update strategies.

5. For a new frame, a region of interest (ROI) centered at the last target location containing both target and background context is cropped and propagated through the fully convolutional network.

6. Two foreground heat maps are generated by GNet and SNet, respectively. Target localization is performed independently based on the two heat maps.

7. The final target is determined by a distracter detection scheme that decides which heat map in step 6 to be used.

4.1. Feature Map Selection

The proposed feature map selection method is based on a target heat map regression model, named as sel-CNN, and is conducted independently on the conv4-3 and conv5-3 layers of VGG. The sel-CNN model consists of a dropout layer followed by a convolutional layer without any nonlinear transformation. It takes the feature maps (conv4-3 or con5-3) to be selected as input to predict the target heat map $\mathbf{M}$, which is a 2-dimensional Gaussian centered at the ground truth target location with variance proportional to the target size (See Figure 1 (a) and (b) for an example). The model is trained by minimizing the square loss between the predicted foreground heat map $\hat{\mathbf{M}}$ and the target heat map $\mathbf{M}$:

$$L_{sel} = \|\hat{\mathbf{M}} - \mathbf{M}\|^2. \tag{4}$$

After parameter learning using back-propagation converges, we fix the model parameters and select the feature maps according to their impacts on the loss function. The input feature maps $\mathbf{F}$ are vectorized into a vector denoted by $vec(\mathbf{F})$. Denote f_i as the i-th element of $vec(\mathbf{F})$. The change of the loss function caused by the perturbation of the feature map $\delta\mathbf{F}$ can be computed by a two-order Taylor expansion as follows:

$$\delta L_{sel} = \sum_i g_i \delta f_i + \frac{1}{2}\sum_i h_{ii}(\delta f_i)^2 + \frac{1}{2}\sum_{i\neq j} h_{ij}\delta f_i \delta f_j, \tag{5}$$

where $g_i = \frac{\partial L_{sel}}{\partial f_i}$ and $h_{ij} = \frac{\partial^2 L_{sel}}{\partial f_i \partial f_j}$ are, respectively, the first and second order derivatives of the objective function with respect to the input feature maps. The number of elements in the feature maps is very large ($> 270,000$). The complexity for computing all the second order derivatives h_{ij} is $O(270,000^2)$, which is too time consuming. We approximate the Hessian matrix with a diagonal matrix, in which the third term of the right hand side of (5) is neglected. Both the first derivatives g_i and the second derivatives h_{ii} can be efficiently computed via back-propagation.

We define the significance of the element f_i as the change of the objective function after setting f_i to zero, *i.e.*, $\delta f_i = 0 - f_i$. According to (5), the significance of f_i can then be computed as

$$s_i = -g_i f_i + \frac{1}{2} h_{ii} f_i^2. \tag{6}$$

The significance of the k-th feature map are further defined as the summation of significance of all its elements $S_k = \sum_{x,y} s(x,y,k)$, where $s(x,y,k)$ is the significance of the element indexed by location (x,y) on the k-th feature map. All the feature maps are sorted in the descending order by their significance, and the top K feature maps are selected. These selected feature maps have significant impact on the objective function and thus are most relevant to the tracking task. Our feature map selection method can be conducted in an online fashion. In our experiments, we only conduct feature selection at the first frame and have achieved good performance. This should be partially attributed to the robustness of CNN features.

The idea of using quadratic approximation of the cost function to remove connections in networks can be traced back to 1989 [18]. The aim was to reduce the number of parameters and improve speed, while we target on removing noisy feature maps and improving tracking accuracy.

4.2. Target Localization

Figure 5 (c) and (d) show the design of the CNNs for target localization. After feature map selection in the first frame, we build the SNet and the GNet on top of the selected conv4-3 and conv5-3 feature maps, respectively. Both networks share the same architecture that consists of two additional convolutional layers. The first additional convolutional layer has convolutional kernels of size 9×9 and outputs 36 feature maps as the input to the next layer. The second additional convolutional layer has kernels of size 5×5 and outputs the foreground heat map of the input image. ReLU is chosen as the nonlinearity for these two layers.

SNet and GNet are initialized in the first frame by minimizing the following square loss function:

$$\begin{aligned} L &= L_S + L_G, \\ L_U &= \|\hat{\mathbf{M}}_U - \mathbf{M}\|_F^2 + \beta\|\mathbf{W}_U\|_F^2, \end{aligned} \tag{7}$$

where the subscript $U \in \{S, G\}$ indicates SNet and GNet, respectively; $\hat{\mathbf{M}}_U$ represents the foreground heat map predicted by the network; $\mathbf{M}$ is the target heat map, $\mathbf{W}_U$ is the weight parameter of the convolutional layers; β is a trade off parameter for weight decay.

Note that the sel-CNN for selecting features and the SNet and GNet for localization are different in CNN structures. The sel-CNN architecture is very simple to avoid using noisy feature maps to overfit the objective function, whereas the SNet and GNet are more complex. Since the noisy feature maps have been discarded by the feature map selection, more complex models facilitate more accurate tracking. Detailed experimental results and analysis on the selection of different model architectures for localization and feature map selection are provided in the supplementary materials.

In a new frame, we crop a rectangle ROI region centered at the last target location. By forward propagating the ROI region through the networks, the foreground heat maps are predicted by both GNet and SNet. Target localization is first performed on the heat map produced by GNet. Denote the target location as $\hat{\mathbf{X}} = (x, y, \sigma)$, where x, y and σ represent the center coordinates and scale of the target bounding box, respectively. Given the target location $\hat{\mathbf{X}}^{t-1}$ in the last frame, we assume the locations of target candidates in the current frame are subject to a Gaussian distribution

$$p(\mathbf{X}^t|\hat{\mathbf{X}}^{t-1}) = \mathcal{N}(\mathbf{X}^t; \hat{\mathbf{X}}^{t-1}, \mathbf{\Sigma}), \tag{8}$$

where $\mathbf{\Sigma}$ is a diagonal covariance matrix that indicates the variances of the location parameters. The confidence of the i-th candidate is computed as the summation of all the heat map values within the candidate region $conf_i = \sum_{j\in\mathbf{R}_i} \hat{\mathbf{M}}_G(j)$, where $\hat{\mathbf{M}}_G$ denotes the heat map generated by GNet; $\mathbf{R}_i$ is the region of the i-th target candidate according to its location parameter $\mathbf{X}_i^t$ (8); j denotes the coordinate index. The candidate with the highest confidence is predicted as the target by GNet.

According to the analysis in Section 3, GNet based on the conv5-3 layer captures semantic features and is highly invariant to intra class variation. Hence, the foreground heat map generated by GNet highlights both the target and background distracters with similar appearances.

To prevent the tracker from drifting to background, we further promote a distracter detection scheme to determine the final target location. Denote the target location predicted by GNet as $\hat{\mathbf{X}}_G$, the corresponding target region in the heat map as $\mathbf{R}_G$. The probability of distracter occurring in background is evaluated by the proportion between the confidence values outside and inside the target region

$$P_d = \frac{\sum_{j\in\hat{\mathbf{M}}_G-\mathbf{R}_G} \hat{\mathbf{M}}_G(j)}{\sum_{k\in\mathbf{R}_G} \hat{\mathbf{M}}_G(k)}, \tag{9}$$

where $\hat{\mathbf{M}}_G - \mathbf{R}_G$ represents the background region on heat map $\hat{\mathbf{M}}_G$. When the proportion P_d is less than a threshold (0.2 in all the experiments), we assume no co-occurring distracter and use the target location predicted by GNet as the final result. Otherwise, the same target localization procedure described above is performed on the heat map $\hat{\mathbf{M}}_S$ predicted by SNet to determine the final target location.

4.3. Online Update

To avoid the background noise introduced by online update, we fix GNet and only update SNet after the initialization in the first frame. SNet is updated following two different rules: the adaptation rule and the discrimination rule, which aim to adapt SNet to target appearance variation and improve the discriminative power for foreground and background, respectively. According to the adaptation rule, we finetune SNet every 20 frames using the most confident tracking result within the intervening frames. Based on the discrimination rule, when distracters are detected using (9), SNet is further updated using the tracking results in the first frame and the current frame by minimizing

$$\begin{aligned} \min \beta\|\mathbf{W}_S\|_F^2 + \sum_{x,y} \Big\{ &\left[\hat{\mathbf{M}}_S^1(x,y) - \mathbf{M}^1(x,y)\right]^2 \\ &+ \left[1 - \mathbf{\Phi}^t(x,y)\right] \left[\hat{\mathbf{M}}_S^t(x,y) - \mathbf{M}^t(x,y)\right]^2 \Big\}, \end{aligned} \tag{10}$$

where $\mathbf{W}_S$ denotes the convolutional weight of SNet; (x, y) are spatial coordinates; $\hat{\mathbf{M}}_S^t$ and $\mathbf{M}^t$ represent the heat map for the t-th frame predicted by SNet and the heat map

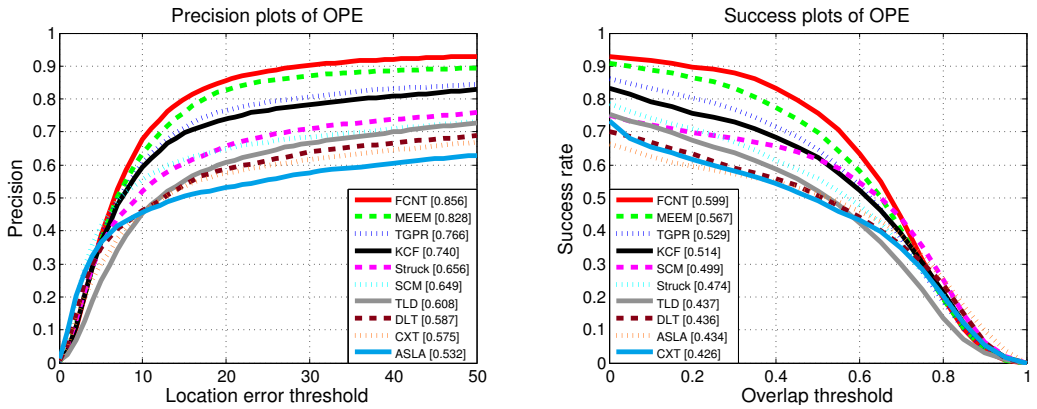


Figure 6. The precision plots and success plots of OPE for the top 10 trackers. The performance score for each tracker is shown in the legend. The performance score of precession plot is at error threshold of 20 pixels while the performance score of success plot is the AUC value.

Table 2. Average precision scores on different attributes: illumination variation (IV), out-of-plane rotation (OPR), scale variation (SV), occlusion (OCC), deformation (DEF), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-view (OV), background cluttered (BC) and low resolution (LR). The best and the second best results are in **red** and **green** colors, respectively.

	SCM	Struck	TLD	DLT	ASLA	CXT	MEEM	KCF	TGPR	FCNT	FCNT_G	FCNT_S	FCNT_{512}	FCNT_{256}	FCNT_{128}	FCNT_{64}
IV	0.594	0.558	0.537	0.534	0.517	0.501	0.778	0.728	0.687	**0.830**	0.776	0.731	**0.791**	0.758	0.760	0.787
OPR	0.618	0.597	0.596	0.561	0.518	0.574	**0.838**	0.729	0.741	**0.831**	0.820	0.775	0.789	0.779	0.758	0.751
SV	0.672	0.639	0.606	0.590	0.552	0.550	0.787	0.679	0.703	**0.830**	**0.806**	0.762	0.799	0.787	0.755	0.785
OCC	0.640	0.564	0.563	0.574	0.460	0.491	**0.801**	0.749	0.708	**0.797**	0.745	0.764	0.740	0.730	0.740	0.707
DEF	0.586	0.521	0.512	0.563	0.445	0.422	0.859	0.740	0.768	**0.917**	**0.917**	0.884	**0.919**	0.857	0.896	0.850
MB	0.339	0.551	0.518	0.453	0.278	0.509	0.740	0.650	0.578	**0.789**	0.666	0.737	0.734	0.712	**0.743**	0.729
FM	0.333	0.604	0.551	0.446	0.253	0.515	**0.757**	0.602	0.575	**0.767**	0.686	0.727	0.725	0.671	0.731	0.713
IPR	0.597	0.617	0.584	0.548	0.511	0.610	0.790	0.725	0.705	**0.811**	0.797	0.749	0.778	**0.824**	0.750	0.751
OV	0.429	0.539	0.576	0.444	0.333	0.510	**0.730**	0.650	0.576	**0.741**	0.636	0.641	0.648	0.520	0.650	0.554
BC	0.578	0.585	0.428	0.495	0.496	0.443	**0.807**	0.753	0.761	**0.799**	0.722	0.679	0.722	0.741	0.674	0.727
LR	0.305	0.545	0.349	0.396	0.156	0.371	0.494	0.381	0.539	**0.765**	0.577	0.633	0.747	0.740	**0.761**	0.750
Overall	0.649	0.656	0.608	0.587	0.532	0.575	**0.828**	0.740	0.766	**0.856**	0.794	0.801	0.824	0.817	0.800	0.798

generated according to the predicted target location (a 2-dimensional Gaussian centered at the target location), respectively; the foreground mask $\mathbf{\Phi}^t$ indicates the predicted target bounding box, *i.e.*, $\mathbf{\Phi}^t(x, y) = 1$ if the location (x, y) belongs to the target region and $\mathbf{\Phi}^t(x, y) = 0$, otherwise.

The second term in (10) corresponds to loss for locating the target in the first frame. When distracters appear or the target undergoes severe occlusion in the current frame, the estimated target region is not reliable for learning target appearance. Therefore, we choose a conservative scheme by adding the first frame to supervise update so that the learned model still captures the appearance in the first frame. Meanwhile, the third term in (10) removes the loss in the unreliable target region and only considers those within the background region in the current frame. It enforces the model to put more efforts on assigning the co-occurring distracters as background. The combination of the second term and the third term in (10) can help SNet to better separate the target from background and alleviate the model degradation caused by occlusion or distracters.

5. Experiments

Setup. The proposed FCNT tracker is implemented in MATLAB based on the wrapper of Caffe framework [14], and runs at 3 fps on a PC with a 3.4GHz CPU and a TITAN GPU. The source code is publicly available[2]. Both the sel-CNN for feature map selection and the GNet and SNet for target localization are trained in the first frame using back-propagation for 50 iterations. Afterwards, SNet are finetuned for 3 iterations at each update step. The learning rates are set to $1e-9$ for the sel-CNN and $1e-7$ for the GNet and SNet. The number of feature maps selected by the proposed feature selection method is set to $K = 384$ for both the conv4-3 and conv5-3 layers. The size of the input ROI region centered at target location is 386×386 pixels. The weight decay parameter β in (7) and (10) is set to 0.005. At each frame, 600 target candidates are randomly sampled. The variance of location parameters in (8) are set to $\{10, 10, 0.004\}$ for x, y translation and scale, respectively. All the parameters are fixed through the experiment.

Evaluation Methodology. We evaluate the proposed FCNT tracker on the benchmark data set [36] which includes 50 sequences and the results of 29 trackers. In addition, we also compare our method with three recent state-of-the-art methods MEEM [38], TGPR [7] and KCF [11] for a more thorough comparison. The sequences are further tagged with 11 attributes according to different challenging factors. We use the precision plot and the success plot to evaluate all the trackers. The precision plot demonstrates the per-

[2] http://ice.dlut.edu.cn/lu/index.html

Table 3. Average success scores on different attributes: illumination variation (IV), out-of-plane rotation (OPR), scale variation (SV), occlusion (OCC), deformation (DEF), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-view (OV), background cluttered (BC) and low resolution (LR).The best and the second best results are in **red** and **green** colors, respectively.

	SCM	Struck	TLD	DLT	ASLA	CXT	MEEM	KCF	TGPR	FCNT	FCNT_G	FCNT_S	FCNT_{512}	FCNT_{256}	FCNT_{128}	FCNT_{64}
IV	0.473	0.428	0.399	0.405	0.429	0.368	0.548	0.493	0.486	**0.598**	0.519	0.546	**0.565**	0.546	0.558	**0.565**
OPR	0.470	0.432	0.420	0.412	0.422	0.418	**0.562**	0.495	0.507	**0.581**	0.532	0.548	0.550	0.544	0.539	0.529
SV	0.518	0.425	0.421	0.455	0.452	0.389	0.503	0.427	0.443	**0.558**	0.507	0.527	**0.531**	0.525	0.515	0.519
OCC	0.487	0.413	0.402	0.423	0.376	0.372	**0.559**	0.514	0.494	**0.571**	0.491	0.557	0.528	0.519	0.540	0.515
DEF	0.448	0.393	0.378	0.394	0.372	0.324	0.582	0.534	0.556	**0.644**	0.605	0.638	**0.649**	0.618	0.643	0.614
MB	0.293	0.433	0.404	0.363	0.258	0.369	**0.565**	0.497	0.440	**0.580**	0.452	0.563	0.524	0.518	0.539	0.532
FM	0.296	0.462	0.417	0.360	0.247	0.388	**0.568**	0.459	0.441	**0.565**	0.496	0.545	0.523	0.493	0.534	0.513
IPR	0.458	0.444	0.416	0.411	0.425	0.452	0.526	0.497	0.487	**0.555**	0.510	0.519	0.532	**0.561**	0.520	0.520
OV	0.361	0.459	0.457	0.367	0.312	0.427	**0.597**	0.550	0.431	**0.592**	0.478	0.498	0.502	0.419	0.514	0.430
BC	0.450	0.458	0.345	0.339	0.408	0.338	**0.574**	0.535	0.543	**0.564**	0.494	0.510	0.514	0.527	0.498	0.517
LR	0.279	0.372	0.309	0.346	0.157	0.312	0.367	0.312	0.351	**0.514**	0.416	0.452	0.495	0.497	**0.520**	0.483
Overall	0.499	0.474	0.437	0.436	0.434	0.426	0.567	0.514	0.529	**0.599**	0.524	0.574	**0.575**	0.570	0.568	0.559

centage of frames where the distance between the predicted target location and the ground truth location is within a given threshold. All the trackers are ranked according to the precision scores at the threshold of 20 pixels. Whereas the success plot illustrates the percentage of frames where the overlap ratio between the predicted bounding box and the ground truth bounding box is higher than a threshold $\tau \in [0, 1]$. The area under curve (AUC) of each success plot are used to rank the tracking algorithms. We report the results of one pass evaluation (OPE) [36] for the proposed FCNT tracker and the top 10 algorithms in each plot, including MEEM [38], TGPR [7], KCF [11], SCM [40], Struck [10], TLD [15], DLT [35], ASLA [13] and CXT [5].

Results. The precision plot and the success plot of the compared trackers on 50 sequences are demonstrated in Figure 6. The proposed FCNT tracker outperforms all the other trackers in terms of both average precision score and success score. To facilitate better analysis on the tracking performance, we further evaluate all the trackers on sequences with 11 attributes. Table 2 and Table 3 demonstrate that the proposed FCNT can well handle a variety of challenging factors and consistently outperform state-of-the-art methods in almost all the challenges. To demonstrate the robustness of the features learned by DNNs from large scale image classification, we also compare with [19] which use a convolutional network for tracking without pre-training. On the 16 adopted test sequences, the proposed FCNT achieves a precision score of 0.88 and a success score of 0.85, whereas the reported results in [19] are 0.83 and 0.83, respectively. Our pre-trained network outperforms the method in [19] with a large margin. We also note that the proposed FCNT tracker has some failure cases in handling the low resolution (LR) challenge and achieves a relatively lower success score on the LR attribute (Table 3). One reason is that the VGG network is pre-trained on the Imagenet data set with training images of high resolutions.

Ablation Study. To further investigate the effectiveness of tracking by considering multiple layers and the proposed feature map selection method, we report the performance of different variants of the proposed algorithm in Table 2 and 3, where FCNT_G and FCNT_S denote the proposed method using only GNet and SNet for tracking; FCNT_K represents the proposed method using K selected feature maps from both the cnov4-3 and conv5-3 layers of VGG network, and no feature map selection is conducted for $K = 512$. FCNT_S exploits more discriminative feature and with proper online update it has higher performance than FCNT_G. By considering both the higher layer and the middle layer, the proposed FCNT can better deal with different challenging factors and outperform both FCNT_G and FCNT_S which only use the feature maps of a single layer. The proposed FCNT tracker uses less feature maps (386) achieves higher performance than FCNT_{512}. The FCNT_{64} tracker uses only 1/8 of all the feature maps and can still compare favorably against state-of-the-art methods. This further demonstrate that the proposed feature map selection method can effectively remove unreliable and noisy feature maps and retain relevant ones, which effectively avoids overfitting and improves training convergence rate.

6. Conclusion

In this paper, we empirically present some important properties of CNN features under the viewpoint of visual tracking. Based on these properties, we propose a tracking algorithm using fully convolutional networks pre-trained on image classification task. We observe that convolutional layers at different levels have different properties. And we jointly consider these properties in order to capture the semantic information of the target and discriminate the target from background distracters. We further develop a principled feature map selection method to select discriminative features and discard noisy or unrelated ones. Our approach is shown to effectively improve the tracking performance on challenging scenarios.

Acknowledgements. This work is supported by the Natural Science Foundation of China (NSFC) #61472060, the Fundamental Research Funds for the Central Universities under Grant DUT14YQ101, Hong Kong Innovation and Technology Support Programme (ITS/221/13FP), and the Research Grants Council of Hong Kong (CUHK 417011, CUHK 419412, CUHK 14207814).

References

[1] P. Agrawal, R. B. Girshick, and J. Malik. Analyzing the performance of multilayer neural networks for object recognition. In *ECCV*, 2014. 1

[2] B. Babenko, M.-H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *TPAMI*, 33(8):1619–1632, 2011. 2

[3] C. Bao, Y. Wu, H. Ling, and H. Ji. Real time robust l1 tracker using accelerated proximal gradient approach. In *CVPR*, 2012. 2

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 1, 3

[5] T. B. Dinh, N. Vo, and G. Medioni. Context tracker: Exploring supporters and distracters in unconstrained environments. In *CVPR*, 2011. 8

[6] J. Fan, W. Xu, Y. Wu, and Y. Gong. Human tracking using convolutional neural networks. *TNN*, 21(10):1610–1623, 2010. 2

[7] J. Gao, H. Ling, W. Hu, and J. Xing. Transfer learning based visual tracking with gaussian processes regression. In *ECCV*. 2014. 1, 7, 8

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1

[9] H. Grabner, C. Leistner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. In *ECCV*, 2008. 2

[10] S. Hare, A. Saffari, and P. H. Torr. Struck: Structured output tracking with kernels. In *ICCV*, 2011. 2, 8

[11] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *TPAMI*, 37(3):583–596, 2015. 2, 7, 8

[12] S. Hong, T. You, S. Kwak, and B. Han. Online tracking by learning discriminative saliency map with convolutional neural network. In *ICML*, 2015. 2

[13] X. Jia, H. Lu, and M.-H. Yang. Visual tracking via adaptive structural local sparse appearance model. In *CVPR*, 2012. 8

[14] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM Multimedia*, pages 675–678, 2014. 7

[15] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *TPAMI*, 34(7):1409–1422, 2012. 8

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[18] Y. LeCun, J. S. Denker, S. A. Solla, R. E. Howard, and L. D. Jackel. Optimal brain damage. In *NIPS*, 1989. 6

[19] H. Li, Y. Li, and F. M. Porikli. Robust online visual tracking with a single convolutional neural network. In *ACCV*, 2014. 2, 8

[20] X. Li, Z. Han, L. Wang, and H. Lu. Visual tracking via random walks on graph model. *IEEE Transactions on Cybernetics*, PP(99):1–1, 2015. 2

[21] X. Mei and H. Ling. Robust visual tracking using l1 minimization. In *CVPR*, 2009. 2

[22] W. Ouyang and X. Wang. Joint deep learning for pedestrian detection. In *ICCV*, 2013. 1

[23] W. Ouyang, X. Wang, X. Zeng, S. Qiu, P. Luo, Y. Tian, H. Li, S. Yang, Z. Wang, C.-C. Loy, et al. Deepid-net: Deformable deep convolutional neural networks for object detection. In *CVPR*, 2015. 1

[24] X. Ren and J. Malik. Tracking as repeated figure/ground segmentation. In *CVPR*, 2007. 2

[25] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *IJCV*, 77(1-3):125–141, 2008. 2

[26] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013. 3

[27] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 1, 2

[28] Y. Sun, X. Wang, and X. Tang. Deeply learned face representations are sparse, selective, and robust. *CoRR*, abs/1412.1265, 2014. 1

[29] D. Wang and H. Lu. Visual tracking via probability continuous outlier model. In *CVPR*, 2014. 1

[30] D. Wang, H. Lu, Z. Xiao, and M.-H. Yang. Inverse sparse tracker with a locally weighted distance metric. *TIP*, 24(9):2646–2657, 2015. 2

[31] D. Wang, H. Lu, and M. Yang. Robust visual tracking via least soft-threshold squares. *TCSVT*, PP(99):1–1, 2015. 2

[32] D. Wang, H. Lu, and M.-H. Yang. Online object tracking with sparse prototypes. *TIP*, 22(1):314–325, 2013. 2

[33] L. Wang, H. Lu, X. Ruan, and M.-H. Yang. Deep networks for saliency detection via local estimation and global search. In *CVPR*, 2015. 1

[34] L. Wang, H. Lu, and D. Wang. Visual tracking via structure constrained grouping. *Signal Processing Letters*, 22(7):794–798, 2015. 2

[35] N. Wang and D. Yeung. Learning a deep compact image representation for visual tracking. In *NIPS*, 2013. 2, 8

[36] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, 2013. 1, 2, 4, 7, 8

[37] F. Yang, H. Lu, and M.-H. Yang. Robust superpixel tracking. *TIP*, 23(4):1639–1651, 2014. 2

[38] J. Zhang, S. Ma, and S. Sclaroff. Meem: Robust tracking via multiple experts using entropy minimization. In *ECCV*. 2014. 1, 7, 8

[39] R. Zhao, W. Ouyang, H. Li, and X. Wang. Saliency detection by multi-context deep learning. In *CVPR*, 2015. 1

[40] W. Zhong, H. Lu, and M. Yang. Robust object tracking via sparse collaborative appearance model. *TIP*, 23(5):2356–2368, 2014. 2, 8

[41] B. Zhuang, H. Lu, Z. Xiao, and D. Wang. Visual tracking via discriminative sparse similarity map. *TIP*, 23(4):1872–1881, 2014. 2