Learning Cascaded Context-aware Framework for Robust Visual Tracking

Ding Ma, Xiangqian Wu
School of Computer Science and Technology, Harbin Institute of Technology
{madingcs, xqwu}@hit.edu.cn

Abstract

Context information on each corner of the whole image is useful for visual tracking. However, some trackers may not be able to model such information, this will result in suboptimal performance. To directly model fully context information is intractable since first the region of the foreground is relatively small, the structure of foreground is lost for some part by straightforwardly aware. Second, the target may share a similar structure of the surrounding distractors. To this end, we propose a cascaded context-aware framework based on two networks that progressively model the foreground and background of the various targets over time. The first network pays attention to the most discriminative information within the whole context and coarser structure of the target, the second network focuses on the self-structure information of the target. Depending on the output of these two networks—the final context-aware map, we can generate the bounding box of the target flexibly. Extensive experiments on 3 popular benchmarks demonstrate the robustness of the proposed CAT tracker.

1. Introduction

With the powerful representation of the *Convolutional Neural Networks* (CNNs), many CNNs-based trackers have been proposed. Among them, most trackers use a rectangle bounding box to label the position of the target. In such cases, the target own model will contain the context information more or less with the rectangle bounding box labeled result. Meanwhile, ignoring the context information may have a drastic impact on tracking performance. First, learning from a limited search area may lead to over-fitting which is not robust to rapid deformation. Second, lacking of real negative training examples can drastically cripple the robustness of such trackers against clustered background, which will increase the risk of tracking drift specifically when the target and context display similar visual cues. Third, the trackers may be ambiguous to distinguish the target from occlusion when the context information is inadequately considered.

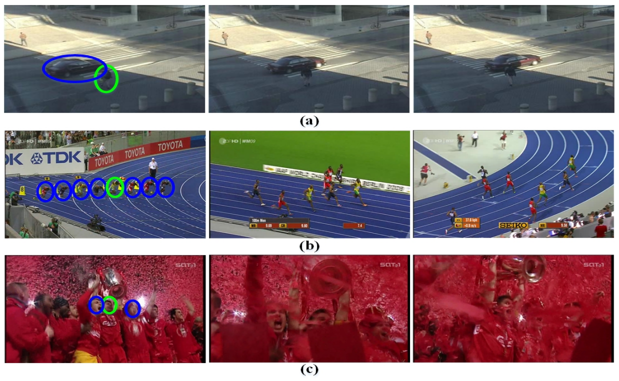

Figure 1: The examples of three categories of the context information. The target and distractors are labeled with green ellipse and blue ellipse, respectively.

To minimize interference from the background, our main idea is to pay attention to the context information on each corner of the whole image. In our observations, the context information can be divided into three categories broadly: low difficulty, medium difficulty, and high difficulty. To illustrate the basis of such classification, we give an example in Figure 1. For the low difficulty level, i.e. smoothly changing context with no/low difficulty distractors. As shown in Figure 1(a), the whole context changes little. Although the distractor (car) has the same texture as the target, the two have very different semantic information. It is relatively easy to model context information in such cases. For the medium difficulty level, i.e. uniformly changing context with medium difficulty distractors. The whole context in (b) is more complex than (a), but distractors (runner) share the same semantic as the target. On this occasion, the information with full context and all distractors are helpful for improving the discriminative of the learned features. For the high difficulty level, i.e. fast-changing context with high difficult distractors. The context in (c) is changing quickly, and the surrounding distractors (faces of other players) share not only the same semantic but also the same attributes (color, pose and so on) as the target. More-

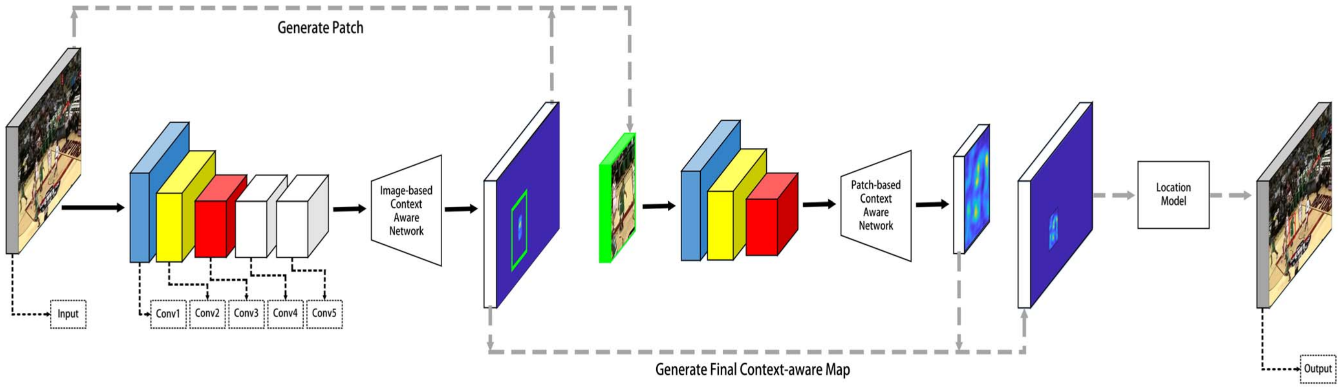
Generate Patch
Image-based Context Aware Network
Patch-based Context Aware Network
Location Model
Input
Conv1
Conv2
Conv3
Conv4
Conv5
Generate Final Context-aware Map
Output

Figure 2: An overview of the proposed CAT. The proposed framework consists of an image-based context-aware network and a patch-based context-aware network. Depending on the output of these two networks, we use different strategies to generate the bounding box of the target in location model. Best viewed in color.

over, all three categories have the same characteristic: the target only accounts for a small region of the whole image, which is intractable to model the full context directly. Motivated by this, we propose a context-aware framework which consists of two networks for modeling the foreground and background in a cascaded way. Given a frame of a video, our goal is to capture the full context of the whole image and produce a clean context map for locating the target.

A few algorithms have been proposed to tackle the complex context information. For person head detection task, two context-aware CNNs were proposed to jointly model the local, global and pairwise information within the whole image [32]. For density estimation task, Wang *et al*. [33] proposed a dual deep neural network (i.e. *DNN-L* and *DNN-G*) to estimate the saliency region by the local and global search. For crowd counting task, Sam *et al*. [28] put forward switching CNNs that leverage different receptive fields to improve the accuracy and localization of the predicted crowd count. For visual tracking task, Galoogahi *et al*. [12] discovered a fundamental drawback of Correlation Filters (CFs), and fused the background information to CFs. Choi *et al*. [4] proposed a context-aware CFs for collecting the context information in a local region. And Li *et al*. [19] used static background information to help track the target instead of only focusing on the target itself.

In contrast to [12], we utilize two deep networks to model the foreground and background simultaneously, rather than the hand-crafted features which are not robust to coping with the complex context. Different from [4, 19], we aim to capture adaptive context for every corner of the whole image. An image-based context-aware map (*ICA map*) derived from image-based context-aware network (ICANet) captures the discriminative features depending on the whole image and the coarser structure of the target region. And this map is produced by a recurrent network consisting of feature extractor ($conv1$-$conv5$ of *VGG-M*) combined with multi convolutional LSTM units [38] and a few convolution (deconvolution) layers. Meanwhile, A patch-based context-aware map (*PCA map*) generated from a patch-based context-aware network (PCANet) is proposed to pay attention to the self-structure of the target to suppress the surrounding distractors. The *PCA map* is generated by a graph-based RNN with several convolution (deconvolution) layers. After that, the final context-aware map (*FCA map*) is constructed by injecting the *PCA map* to the *ICA map*. In the end, different strategies are used for estimating the location of the target depending on the *FCA map* (see Figure 2).

The contributions of this work are summarized as follows:

I, We propose a context-aware framework which consists of an ICANet and a PCANet. The final context-aware map generated from this framework is robust to complex backgrounds throughout the whole scene.

II, We prove that the final context-aware map can be flexibly embedded in two tracking frameworks.

III, Quantitative and qualitative evaluations demonstrate the outstanding performance of our tracking algorithm compared to the state-of-the-art techniques in OTB100 [37], TC128 [21] and VOT2016 [17] benchmark.

The rest of the paper is organized as follows. We first review the related work in Section 2. The detailed configuration of the proposed algorithm is described in Section 3. Section 4 illustrates the experimental results on three large tracking benchmarks. Finally, conclusions are drawn in Section 5.

2. Related Work

2.1. Visual Object Tracking

By considering the sampling strategy, recently published trackers can be divided into two categories: two-stage [14, 16, 30, 10, 24] and one-stage trackers [7, 8, 6, 34, 29, 22].

The two-stage tracking framework consists of a sampling stage and a classification stage. In practice, a large number of candidate samples are drawn by a sampling strategy in the sampling stage. The purpose of the classification stage is to compute the positive probability of each candidate. By the outstanding representation power of *Convolutional Neural Networks* (CNNs), some two-stage trackers have been proposed [14, 16, 30, 10, 24]. CNN-SVM [14] employed an online *support vector machine* (SVM), which discriminates the target object from the background by learning target-specific information in the CNNs features. However, the feature learning and classification are implemented in a separate way which limits the performance of the CNN-SVM. To overcome the separate strategy in CNN-SVM, Hyeonseob Nam *et al*. [24] proposed to learn the feature and classifier jointly, which is referred to as MDNet. The MDNet was composed of shared layers and domain-specific layers, where each domain was formulated as a binary classification to identify the target object from the background. SANet [10] was proposed to use the *Recurrent Neural Network* (RNN) to model object structure, and incorporated the RNN into MDNet to improve the robustness of similar distractors. Meanwhile, to solve the problem of appearance variations and class imbalance in MDNet, Yibing Song *et al*. [30] used adversarial learning to obtain the most robust features of the objects over a long period and proposed a *high-order cost sensitive loss* to decrease the effect of class imbalance. Inspire by Fast R-CNN, Ilchae Jung *et al*. [16] proposed a tracker named RT-MDNet to speed up the MDNet by an improved RoIAlign technology. Nevertheless, RT-MDNet took little advantage of ROI pooling [13] as it cannot encode the difference between highly spatial overlapped candidates.

Different from the two-stage trackers, the one-stage trackers formulate visual tracking as a specific object searching problem and directly calculate a response map through a regression model. One-stage trackers can be broadly classified into two categories: correlation filters (CFs) based trackers [7, 8, 6] and deep regression networks (DRNs) based trackers [34, 29, 22]. CFs trackers achieve fast speed by taking advantage of computing the correlation in the Fourier domain. [7] attempted to use activations from the convolutional layer of CNNs in a discriminative correlation filter, rather than the deep features in fully connected layers. As the features from different CNNs layers characterize different attributes of an object, [8] proposed to utilize different convolutional layers to learn multiple CFs and fuse multiple correlation maps to obtain the location of the object. To reduce the number of parameters and memory, the channel of features was reduced to speed up learning correlation filters in [6]. Although the CFs based trackers achieve the top performance, the features and correlation filters are optimized independently. As opposed to CFs based trackers, DRNs have the potential opportunity to take full advantage of end-to-end learning. Among the DRNs based trackers, the FCNT [34] was proposed to use an *SNet* and a *GNet* to compute a confidence map for predicting the location of the object. CREST [29] fused the outputs of baseline and another two residuals to estimate the location of the target object. Besides, DSLT [22] proposed a shrinkage loss and an ensemble strategy to improve the performance of the DRNs based trackers.

2.2. Context Modeling For Tracking

Lacking of context information modeling will lead to poor discrimination against a cluttered background, and thereby, the risk of spurious detection will be increased when the target and its surrounding background shares similar visual cues. There exists a piece of strong discriminative information on the whole scenes of the consecutive frames [40]. Several trackers have been proposed to employ context information for visual tracking. [39] was the first attempt to use segmented regions surrounding the object as auxiliary objects for collaborative tracking. Key points surrounding the object are first extracted to help locate the object location, and hand-crafted features i.e. SIFT or SURF were then used to represent these consistent regions. However, representing and finding consistent regions was computationally expensive. To solve this problem, STC [40] computed the spatio-temporal context model by *Fast Fourier Transform* (FFT). For better generalization, Matthias Mueller *et al*. [23] proposed a generalized framework CACF for *Correlation Filters* (CFs) based trackers. In CACF, more context (i.e. another 4 orientational contexts) was extracted to improve the discriminative capacity of the filters. More context will increase the feature dimensionality, TRACA [4] was proposed to compress deep feature that is achieved by a context-aware scheme and utilize multiple expert auto-encoders for accelerating the tracking speed. However, we discover that such trackers rarely considering the comprehensive discriminative information on the whole scenes of the successive frames. To this end, we learn a cascaded context-aware framework for visual object tracking.

2.3. Cascaded Structure

Cascaded structures have been proposed for improving performance. Yilun Chen *et al*. [3] proposed a cascaded pyramid network (CPN), which integrates the global pyramid network (GlobalNet) and pyramid refined network (Re-

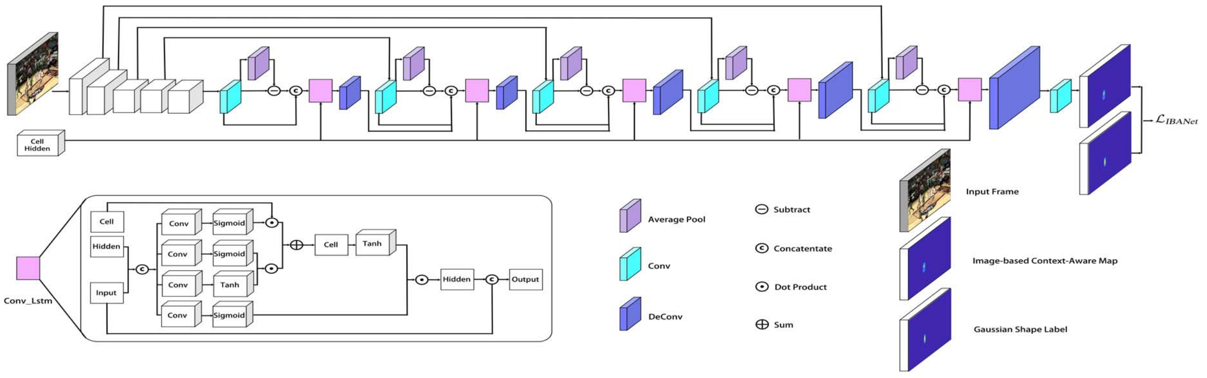

Figure 3: The architecture of our proposed ICANet.

fineNet) for multi-person pose estimation. Yicheng Wang *et al*. [36] presented a cascaded WConv structure to extract the comparison features of two images for person reidentification. Cai *et al*. [1] proposed a multi-stage object detection framework, cascade R-CNN, aiming at high-quality detection by sequentially increasing IoU thresholds. Fan *et al*. [11] utilized a sequence of RPNs cascaded from deep high-level to shallow low-level layers for robust visual tracking. By taking advantage of the cascaded structure, we propose a cascaded context-aware framework for visual tracking. In our framework, an image-based context-aware map (*ICA map*) derived from image-based context-aware network (ICANet) is utilized to capture the discriminative features depending on the whole image and the coarser structure of the target region. Then, a patch-based context-aware map (*PCA map*) generated from the patch-based context-aware network (PCANet) is proposed to pay attention to suppress the surrounding distractors in a local context. After that, the final context-aware map (*FCA map*) is constructed by mapping the *PCA map* to the *ICA map*.

3. Proposed tracking method

The details of the proposed ICANet and PCANet are illustrated in 3.1 and 3.2. The tracking and updating are shown in section 3.3, 3.4, respectively. And then, the training details of the model are illustrated in section 3.5.

3.1. Image-based Context-Aware Network

In our opinion, we argue that recurrent architecture to be important for generating object-free context images, since it allows the network to know where the object is in the sequential frames. As shown in Figure 3, the recurrent architecture is employed to generate image-based context-aware map (*ICA map*) in a recurrent way. The whole network consists of a feature extractor (five convolutional layer in *VGG-M*) and five blocks. Each block is composed of a convolutional layer that encodes features from the corresponding output of feature extractor and an average pooling layer, a convolutional LSTM unit and a deconvolution layer for generating the *ICA map*.

For the ICANet, the target and background are formulated as a binary classification problem. And in most cases, there exists contrast information between the target and its context. In order to capture such contrast information, we propose a contrast layer which is calculated by subtracting the mean value of the features from the features themselves. The mean value is implemented by an average pooling layer with a kernel size of 3×3.

Compared with the appearance variation of the target, the variation of context is relatively slower in most cases. To this end, LSTM is selected to handle this long-term dependency. As shown in Figure 3, the convolutional LSTM unit (the pink rectangle) consists of an input gate $\mathbf{I}_t$, a forget gate $\mathbf{F}_t$, a cell state $\mathbf{C}_t$ and an output gate $\mathbf{O}_t$. Through the time dimension, the relationships between gates and states are expressed as:

$$\begin{aligned}
\mathbf{I}_t &= \sigma(\mathbf{W}_{xI}*\mathbf{X}_t + \mathbf{W}_{hI}*\mathbf{H}_{t-1} + \mathbf{W}_{cI}\odot\mathbf{C}_{t-1} + \mathbf{b}_I),\\
\mathbf{F}_t &= \sigma(\mathbf{W}_{xF}*\mathbf{X}_t + \mathbf{W}_{hF}*\mathbf{H}_{t-1} + \mathbf{W}_{cF}\odot\mathbf{C}_{t-1} + \mathbf{b}_F),\\
\mathbf{C}_t &= \mathbf{F}_t\odot\mathbf{C}_{t-1} + \mathbf{I}_t\odot\tanh(\mathbf{W}_{xC}*\mathbf{X}_t + \mathbf{W}_{hC}*\mathbf{H}_{t-1} + \mathbf{b}_C),\\
\mathbf{O}_t &= \sigma(\mathbf{W}_{xO}*\mathbf{X}_t + \mathbf{W}_{hO}*\mathbf{H}_{t-1} + \mathbf{W}_{cO}\odot\mathbf{C}_t + \mathbf{b}_O),\\
\mathbf{H}_t &= \mathbf{O}_t\odot\tanh(\mathbf{C}_t).
\end{aligned} \tag{1}$$

Where $\mathbf{X}_t$ is the feature generated by the contrast layer. The cell state $\mathbf{C}_t$ will be fed into the next LSTM. The hidden output features are represented by $\mathbf{H}_t$. And $*$ is the convolution operation. The output features of the LSTM are concatenated with the contrast features, which will be fed into a deconvolution layer. After five blocks, the features maps with different scales are integrated and up-sampled to the input size. At last, a convolution layer with a kernel size of 1×1 is operated on the output of the last deconvolution

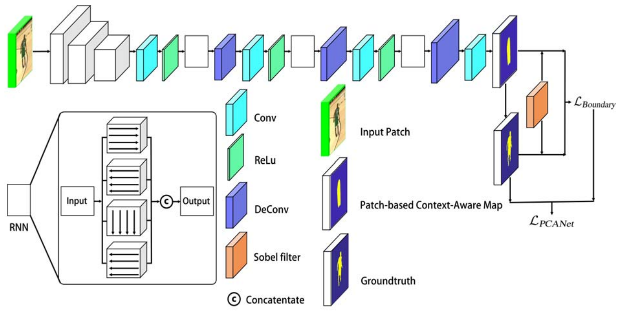

Figure 4: The architecture of our proposed PCANet.

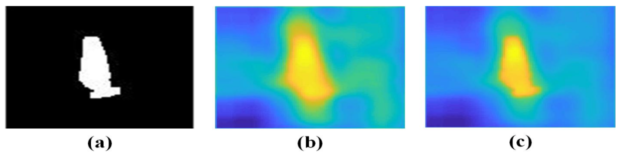

Figure 5: (a) is the groundtruth. (b) is the visualization of the *PCA map* without $\mathcal{L}_{Boundary}$. (c) is the visualization of the *PCA map* with $\mathcal{L}_{Boundary}$.

layer to produce a score layer which contains one channel. For the loss function, we consider the output as the likelihood probability and the distribution of target/background pixels is heavily biased, and hence the class-balancing cross entropy loss function is used for training:

$$\mathcal{L}_{ICANet} = -\frac{1}{K}\sum_{k=1}^{K}[\mathbf{Q}_k \log \mathbf{P}_k + (1-\mathbf{Q}_k)\log(1-\mathbf{P}_k)] \tag{2}$$

where K is the total number of training pixels, $\mathbf{Q}_k$ is the Gaussian shape label of the groundtruth, $\mathbf{P}_k$ is the predicted target probability.

3.2. Patch-based Context-Aware Network

The architecture of ICANet is depending on the 2D CNNs and convolutional LSTM, which is usually focused on capturing coarser and long-term temporal structure. However, such architecture may lack of the capacity of representing finer temporal relation in a local spatiotemporal window. Besides, the output labeled as a Gaussian shape map, we find in some cases, the output cannot describe accurate contour of the target.

Figure 4 illustrates the architecture of PCANet. We crop a patch from the frame, which is centered at the highest response area of the *ICA map*. The proposed PCANet consists of a feature extractor (the first three convolutional layers in ICANet) for constructing feature maps, and three blocks for generating the patch-based context-aware *PCA map*. Each block is composed of a convolutional layer for reducing the dimension of features, an RNN unit for modeling self-structure, and a deconvolution layer for incrementally enlarging the features to the size of the input.

Our PCANet aims to obtain the structure of the target itself from the feature extractor constructed from an image patch. While the resolution of the target feature is low, the target only accounts for a small part of an image. To capture the full structure of the target, we need to construct a feature map with a high resolution. We address this requirement by enlarging the receptive field of each activation. To these ends, the max-pooling layers followed by $conv1$ and $conv2$ layers in *VGG-M* network are removed. Followed by this operation, the output feature map of $conv3$ is four times larger than the original $conv3$ in *VGG-M* network. It allows us to extract high-resolution features and improve the quality of the constructed structure.

The technique of constructing the structure of the target itself is referred to as the RNN unit [10]. In each RNN unit, we use several directed RNNs to model self-structure of the target, i.e., we approximate the topology of an undirected graph by the combination of some directed graphs. In our RNN unit, the undirected graph is decomposed into four directed graphs, i.e., right ($\mathbf{G}_1$), left ($\mathbf{G}_2$), up ($\mathbf{G}_3$) and down ($\mathbf{G}_4$). By performing RNN, the hidden state $\mathbf{h}_n (n = 1, 2, 3, 4)$ is obtained by the corresponding $\mathbf{G}_n$. And the summation of all hidden layers is fed into the output layer. The process can be expressed as:

$$\begin{aligned} \mathbf{h}_n^{(v_i)} &= \phi(\mathbf{U}_n x^{(v_i)} + \sum_{v_j \in P_{\mathbf{G}_n}(v_i)} \mathbf{W}_n \mathbf{h}_n^{(v_j)} + b_n), \\ y^{(v_i)} &= \sigma(\sum_{\mathbf{G}_n \in \mathbf{G}^u} \mathbf{V}_n \mathbf{h}_n^{(v_i)} + c). \end{aligned} \tag{3}$$

Where $\mathbf{U}_n$, $\mathbf{W}_n$ and $\mathbf{V}_n$ are the matrix parameters for corresponding $\mathbf{G}_n$. b_n and c are the bias terms. $P_{\mathbf{G}_n}(v_i)$ is the predecessor of v_i. After that, the output RNN unit is fed into a deconvolution layer to enlarge the features. In the end, the final output is a one-channel score map with the same size as the input patch.

To emphasize the boundaries of objects, we utilized an auxiliary loss called Boundary Loss $\mathcal{L}_{Boundary}$. The total loss consists of the class-balancing cross entropy loss and the Boundary Loss $\mathcal{L}_{Boundary}$ which are summed. To compute the Boundary Loss $\mathcal{L}_{Boundary}$, we first need to extract the boundaries of the predict and groundtruth. In details, the Sobel filters are selected as a convolution with a 3×3 kernel to detect boundaries. Mathematically, the Sobel filters can be expressed as:

$$S_h = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, S_v = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{4}$$

which encodes the horizontal and the vertical gradient respectively. Then, the Sobel filters are constructed by concatenating the above filters. The $\mathcal{L}_{Boundary}$ is calculated by the mean square error between the groundtruth $\mathbf{q}_k$ and the prediction $\mathbf{p}_k$. Afterward, the whole loss function for training the proposed PCANet is calculated by:

$$\begin{aligned}\mathcal{L}_{PCANet} =& -\frac{1}{K}\sum_{k=1}^{K}[\mathbf{q}_k \log \mathbf{p}_k + (1-\mathbf{q}_k)\log(1-\mathbf{p}_k)] \\ &+ \mathcal{L}_{Boundary}\end{aligned} \tag{5}$$

Figure 5 shows the visualization of the *PCA map*. According to Figure 5, our PCANet focuses more on the finer structure of the target.

3.3. Target location determination

To estimate the location of a target, the final context-aware map (*FCA map*) is first constructed by the projection of two results, e.g. the result of PCANet is mapped to the result of the ICANet by the way of pixel value mapping. Then, we consider two different strategies to generate a bounding box from the *FCA map*:

(1) Given the *FCA map*, we apply a per-pixel sigmoid on the *FCA map*. Then, a binary mask is obtained by binarization of the *FCA map* with a threshold of 0.5. Depending on the binary mask, we generate the bounding box by axis-aligned bounding rectangle. (denote as *Seg*)

(2) We embed the *FCA map* into the Bayesian framework in which the maximum posterior estimate is computed based on the likelihood of the candidate belonging to the target. To capture more clear and robust description of the target after obtaining the *FCA map*, the $independent$ $component$ $analysis$ (ICA) is utilized to describe the detailed information of a target. (denote as *ICA*)

The ICA is developed to extract the desired signal among source components guided by references. To get the reference, we first convolve the input frame with a Laplacian of Gaussian filter and output a boundary map. Then, the reference $\mathbf{m}_r$ generated by the boundary map takes element-wise multiplication with the *FCA map*. Given the reference $\mathbf{m}_r$ and $\mathbf{m}_s$ (e.g. the *FCA map*) as the signal, the desired signal is separated by a projection space $s = w^T\mathbf{m}_s$. And the goal is to maximize the negentropy $J(s)$:

$$J(s) \approx \rho \|\mathbb{E}[\mathcal{Q}(s)] - \mathbb{E}[\mathcal{Q}(\epsilon)]\|^2 \tag{6}$$

$$\varepsilon(s, \mathbf{m}_r) \leq \xi \tag{7}$$

where $\mathcal{Q}(s)$ is the non-quadratic function, ρ is a constant, ϵ is g Uniform variable, $\varepsilon(\cdot)$ is a norm function and $\mathbb{E}[\cdot]$ is the expectation. And the result of ICA is fed into the observation model in the Bayesian framework for visual tracking. In this framework, the location of the target is denoted as $l_t = (x, y, \sigma)$, where x, y and σ represent the center coordinates and scale of the bounding box, respectively. And the candidate samples are normalized to the canonical size maps $\{CC_t^{(r)}\}$ with $v_t^{(r)}(i,j)$ being the value derived from Eq.7 at location (i,j) of the r-th candidate at time t.

To this end, the confidence of the r-th candidate is computed as the sum of all the heat map values within the canonical size maps: $c^{(r)} = \sum_{(i,j)\in CC^{(r)}} v^{(r)}(i,j)$. The final localization is computed as:

$$c_t^* = \arg\max_r c_t^{(r)} \tag{8}$$

where $*$ corresponds to the best candidate state l_t^* in the current frame.

3.4. Online Updating

The online updating strategy plays an important role in the process of tracking. For the ICANet, we feed the network with an entire frame. Since the ICANet is trained on sequences with a maximum length of 16 frames, we reset the LSTM state after every 16 frames. The state of the LSTM is set to the output from the first forward pass, which maintains an encoding of the tracked object. For the PCANet, we incrementally update our network frame-by-frame with the estimated binary mask.

3.5. Training Details

For the ICANet, the layers in feature extractor from *VGG-M* are initialized with the pre-trained weights, the other parameters are initialized randomly with a truncated normal distribution. The AdamOptimizer method is used for updating with a learning rate of 10^{-4}. The ICANet is trained with two stages. During the first stage, the ICANet is trained for 300 epochs with a batch size of 16 frames from the CDnet2014 dataset. Then, the ICANet is fine-tuned on the DAVIS2016 dataset for 200 epoch with a batch size of 16 frames. For the PCANet, the feature extractor is initialized with the first three convolutional layers of the ICANet. We use an initial learning rate of 10^{-5}, which continues for approximately 300 iterations. We set all the parameters fixed throughout the experiments and datasets.

4. Experimental Results and Analysis

To evaluate the performance of our CAT, we follow the standard metrics. In the OTB100 [37] dataset, we utilize the popular one-pass evaluation (OPE) with precision and success plots metrics. For the precision metric, the estimated locations are measured within a certain threshold distance from groundtruth locations. In general, the threshold distance is set as 20 pixels. The success plot metric focuses

Table 1: Ablation study on the contribution of different components.

	PCANet	*Seg*	*ICA*	$\mathcal{L}_{Boundary}$	Precision	Success
ICANet		✓			0.774	0.525
ICANet			✓		0.851	0.627
ICANet	✓	✓			0.814	0.602
ICANet	✓		✓		0.903	0.695
ICANet	✓	✓		✓	0.818	0.608
ICANet	✓		✓	✓	0.909	0.697

on the overlap ratio between the predicted bounding box and the groundtruth bounding box. The precision and success plots are also utilized in TC128 dataset [21]. In the VOT2016 dataset [17], each tracker is measured by the metrics of Accuracy Ranks (A), Robustness Ranks (R) and Expected Average Overlap (EAO).

4.1. Implementation Details

The proposed algorithm is implemented in MATLAB with Matconvnet toolbox, runs on a PC with an Intel(R) Core(TM) i7-4790k CPU and an NVIDIA Tesla K40c GPU. The input size of ICANet and PCANet is 300×300 and 100×100, respectively. The LSTM layers have 1024 units each. We initialize all new layers with the MSRA initialization method. The label of the ICANet is generated using a two-dimensional Gaussian function with a peak value of 1.0. The state of the target in the first frame is initialized by the GrabCut [27] for the PCANet. The dimension of hidden layers of RNNs is set to 512, 256 and 128. For the tracking strategy of the Bayesian framework, a Gaussian distribution model is used to generate 600 candidates for each frame. The variance of candidate location parameters are set to $\{10, 10, 0.01\}$ for translation and scale, respectively.

4.2. Ablation study

To investigate the effectiveness of the components of CAT, we conducted six variants of CAT and evaluated them using OTB100. The gray lines represent the variants of the ICANet with the *Seg* strategy. The White lines exhibit the variants of the ICANet with the *ICA* strategy. The precision and success scores of the ablation study are illustrated in the last two columns of Table 1. For the *Seg* strategy, "ICANet + PCANet" and "ICANet + PCANet + $\mathcal{L}_{Boundary}$" achieved 4% and 4.4% improvement in precision performance, respectively. For the *ICA* strategy, "ICANet + PCANet" and "ICANet + PCANet + $\mathcal{L}_{Boundary}$" achieved 5.2% and 5.8% improvement in precision performance, respectively. The results show that the proposed framework can improve performance with considering fully context and boundary information. The architecture of "ICANet + PCANet + *ICA* + $\mathcal{L}_{Boundary}$" is selected to compare with other state-of-the-art trackers on the following 3 benchmarks.

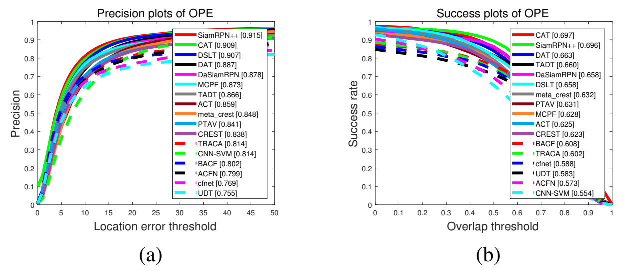

Figure 6: (a) and (b) are the precision and success plots on OTB100, respectively.

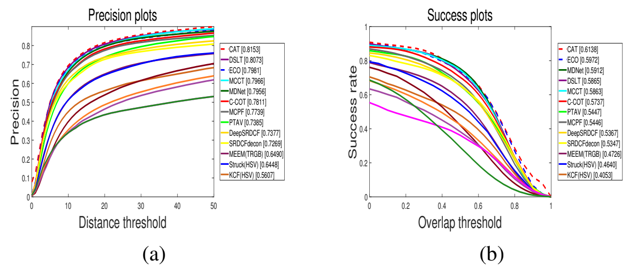

Figure 7: (a) and (b) are the precision and success plots on TC128, respectively.

4.3. Experiments on OTB100 Dataset

We compare the proposed CAT tracker on OTB100 dataset with the following recent published 16 trackers: SiamRPN++ [18], DSLT [22], DAT [26], DaSiamRPN [42], MCPF [41], TADT [20], ACT [2], meta_crest [25], PTAV [9], CREST [29], TRACA [4], CNN-SVM [15], BACF [12], ACFN [5], cfnet [31] and UDT [35]. The tracking performance was measured by conducting a one-pass evaluation (OPE) based on two metrics: center location error and overlap ratio. The results are shown in Figure 6. According to Figure 6, the CAT tracker achieves competitive performance among the state-of-the-arts on this dataset. The values of the precision plot and the success plot are 0.909 and 0.697 on OTB100, respectively.

4.4. Experiments on TC128 Dataset

For experiments on the TC128 [21] dataset containing 128 videos, a comparison with 12 state-of-the-art trackers is shown in Figure 7. Among the compared methods, our approach improves the precision score from 0.8073 of the state-of-the-art tracker to 0.8153. Figure 7 (b) shows the success plot overall 128 videos in TC128 dataset. The

Table 2: Comparisons with the state-of-the-art trackers on the VOT2016 dataset. The results are presented in terms of expected average overlap (EAO), accuracy and robustness. (The first and second best results are shown in color.)

Trackers	CAT	ECO	C-COT	Staple	MDNet	CREST	SiamFC	ECO-hc
EAO	0.332	0.367	0.331	0.295	0.257	0.283	0.235	0.322
A	0.57	0.55	0.54	0.54	0.54	0.51	0.53	0.54
R	0.23	0.20	0.24	0.38	0.34	0.25	0.46	0.30

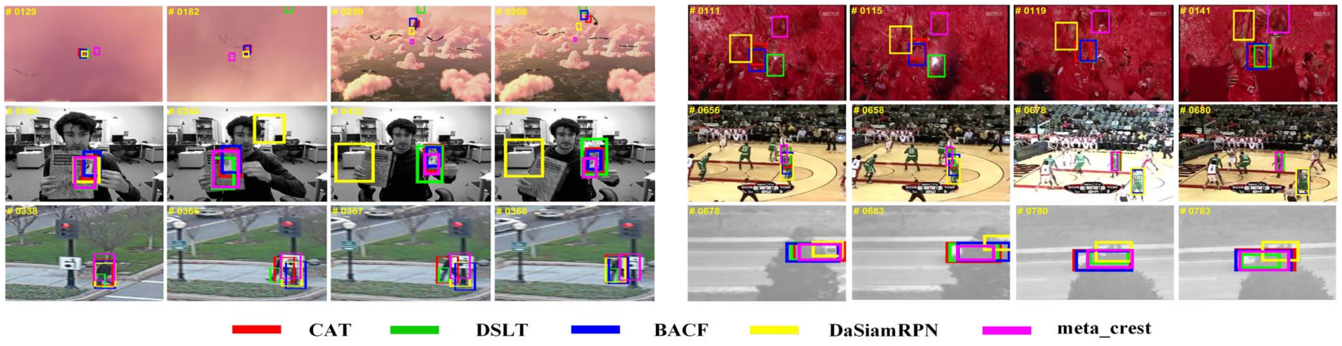

Figure 8: Some results of the proposed CAT tracker on a subset of challenging sequences.

CAT tracker outperforms state-of-the-art approaches with an AUC score of 0.6138. The top rank verifies the robustness of the proposed CAT.

4.5. Experiments on VOT2016 Dataset

Finally, we evaluate our CAT on Visual Object Tracking (VOT2016) benchmark [17]. VOT2016 report shows that the strict state-of-the-art bound is 0.251 under the EAO metric. Trackers whose EAO value exceeds this bound is defined as state-of-the-art. We compare the CAT tracker with 7 state-of-the-art trackers including ECO, C-COT, Staple, MDNet, CREST, SiamFC and ECO-hc. As illustrated in Table 2, the CAT tracker achieves competitive results with higher ranking within all the compared trackers.

4.6. Analysis and Discussion

Qualitative results of the proposed CAT tracker on a subset of challenging sequences are shown in Figure 8. The proposed CAT handles large appearance variations well caused by deformation, in-plane and out-of-plane rotations. The *ICA map* generated via ICANet captures more discriminative features for separating the foreground and background, i.e. it maintains the most robust features over a long temporal span. The advantage of exploiting the temporally robust features by recurrent units in PCANet is proved when the target deals with occlusion. Moreover, our PCANet effectively captures a variety of self-structure variations. Compared to [12, 42], our CAT achieves leading performance in the presence of illumination variation and background clutter. This is because of our extracted context information for every corner within the whole image. Meanwhile, the *FCA map* is capturing the coarse and fine-grained information of the target, our tracker performs better than [12] even the target size of these sequences is small.

5. Conclusions

In this paper, we propose an effective context-aware framework for visual tracking. This framework consists of an ICANet and a PCANet. The *ICA map* derived from ICANet can delineate a coarse map of the target object. To separate the target from surrounding distractors, the *PCA map* captures the self-structure of target by PCANet. After that, both maps are fused to form the final context-aware map (*FCA map*). Depending on the *FCA map*, we use different strategies to generate the bounding box of the target flexibly. Quantitative and qualitative experiments demonstrate the robustness of the proposed method.

6. Acknowledgments

This work was supported in part by the National Key R&D Program of China under Grant 2018YFC0832304, by the Natural Science Foundation of China under Grant 61672194, by the Distinguished Youth Science Foundation of Heilongjiang Province of China under Grant JC2018021, by the State Key Laboratory of Robotics and System (HIT) under Grant SKLRS-2019-KF-14, and by the ZTE Industry-Academia-Research Cooperation Funds.

References

[1] Z. Cai and N. Vasconcelos. Cascade r-cnn: Delving into high quality object detection. *arXiv*, 2017.

[2] B. Chen, D. Wang, P. Li, S. Wang, and H. Lu. Real-time actor-critictracking. In *ECCV*, 2018.

[3] Y. Chen, Z. Wang, Y. Peng, Z. Zhang, G. Yu, and J. Sun. Cascaded pyramid network for multi-person pose estimation. 2017.

[4] J. Choi, H. J. Chang, T. Fischer, S. Yun, K. Lee, J. Jeong, Y. Demiris, and J. Y. Choi. Context-aware deep feature compression for high-speed visual tracking. In *CVPR*, 2018.

[5] J. Choi, H. J. Chang, S. Yun, T. Fischer, Y. Demiris, J. Y. Choi, et al. Attentional correlation filter network for adaptive visual tracking. In *CVPR*, 2017.

[6] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg. Eco: Efficient convolution operators for tracking. In *CVPR*, 2017.

[7] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg. Convolutional features for correlation filter based visual tracking. In *ICCVW*, 2016.

[8] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *ECCV*, 2016.

[9] H. Fan and H. Ling. Parallel tracking and verifying: A framework for real-time and high accuracy visual tracking. 2017.

[10] H. Fan and H. Ling. Sanet: Structure-aware network for visual tracking. In *CVPRW*, 2017.

[11] H. Fan and H. Ling. Siamese cascaded region proposal networks for real-time visual tracking. In *CVPR*, 2019.

[12] H. K. Galoogahi, A. Fagg, and S. Lucey. Learning background-aware correlation filters for visual tracking. pages 1144–1152, 2017.

[13] R. Girshick. Fast r-cnn. In *ICCV*, 2015.

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

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

[16] I. Jung, J. Son, M. Baek, and B. Han. Real-time mdnet. In *ECCV*, 2018.

[17] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. ehovin, T. Vojr, G. Hger, A. Lukei, and G. Fernndez. *The Visual Object Tracking VOT2016 Challenge Results*. Springer International Publishing, 2016.

[18] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. *arXiv*, 2018.

[19] H. Li, D. Yang, and Z. Chen. Adaptive background for real-time visual tracking. In *ICME*, 2016.

[20] X. Li, C. Ma, B. Wu, Z. He, and M.-H. Yang. Target-aware deep tracking. In *CVPR*, 2019.

[21] P. Liang, E. Blasch, and H. Ling. Encoding color information for visual tracking: Algorithms and benchmark. *TIP*, 2015.

[22] X. Lu, C. Ma, B. Ni, X. Yang, I. Reid, and M.-H. Yang. Deep regression tracking with shrinkage loss. In *ECCV*, 2018.

[23] M. Mueller, N. Smith, and B. Ghanem. Context-aware correlation filter tracking. In *CVPR*, 2017.

[24] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, 2016.

[25] E. Park and A. C. Berg. Meta-tracker: Fast and robust online adaptation for visual object trackers. *arXiv*, 2018.

[26] S. Pu, Y. Song, C. Ma, H. Zhang, and M.-H. Yang. Deep attentive tracking via reciprocative learning. In *NIPS*, 2018.

[27] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM SIGGRAPH*, 2004.

[28] D. B. Sam, S. Surya, and R. V. Babu. Switching convolutional neural network for crowd counting. In *CVPR*, 2017.

[29] Y. Song, C. Ma, L. Gong, J. Zhang, R. W. H. Lau, and M. H. Yang. Crest: Convolutional residual learning for visual tracking. In *ICCV*, 2017.

[30] Y. Song, C. Ma, X. Wu, L. Gong, L. Bao, W. Zuo, C. Shen, R. W. Lau, and M.-H. Yang. Vital: Visual tracking via adversarial learning. In *CVPR*, 2018.

[31] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. S. Torr. End-to-end representation learning for correlation filter based tracking. 2017.

[32] T. H. Vu, A. Osokin, and I. Laptev. Context-aware cnns for person head detection. In *ICCV*, 2015.

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

[34] L. Wang, W. Ouyang, X. Wang, and H. Lu. Visual tracking with fully convolutional networks. In *ICCV*, 2016.

[35] N. Wang, Y. Song, C. Ma, W. Zhou, W. Liu, and H. Li. Unsupervised deep tracking. In *CVPR*, 2019.

[36] Y. Wang, Z. Chen, F. Wu, and G. Wang. Person re-identification with cascaded pairwise convolutions. In *CVPR*, 2018.

[37] Y. Wu, J. Lim, and M. H. Yang. Object tracking benchmark. *TPAMI*, 2015.

[38] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NIPS*, 2015.

[39] M. Yang, Y. Wu, and G. Hua. Context-aware visual tracking. *PAMI*, 2009.

[40] K. Zhang, L. Zhang, M. Yang, and D. Zhang. Fast tracking via spatio-temporal context learning. arxiv, 2013. *arXiv*.

[41] T. Zhang, C. Xu, and M. H. Yang. Multi-task correlation particle filter for robust object tracking. In *CVPR*, 2017.

[42] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu. Distractor-aware siamese networks for visual object tracking. In *ECCV*.