IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON INTEGRATIVE COMPUTER VISION AND MULTIMEDIA ANALYTICS

Received December 24, 2019, accepted January 10, 2020, date of publication January 17, 2020, date of current version January 28, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2967031

Object Tracking Based on Channel Attention

ZHIQUAN HE[1,2,3] AND XUEJUN CHEN[1]

[1]Shenzhen Key Laboratory of Media Security, Shenzhen University, Shenzhen 518060, China
[2]Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen 518060, China
[3]Guangdong Multimedia Information Service Engineering Technology Research Center, Shenzhen 518060, China

Corresponding author: Zhiquan He (zhiquan@szu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61771322, Grant 61375015, and Grant 61971290.

ABSTRACT Object tracking is one of the important research topics in computer vision. Despite the great progress in this area, effectively and efficiently tracking object in videos still remains challenging especially in scenarios of rapid movement of objects, illumination changes, appearance of similar objects in the background. In this work, we propose an object tracking method by improving the state-of-the-art tracker MDNet based on channel attention, which can distinguish among similar objects by suppressing the background and highlighting the object. We integrate the channel attention module and group normalization into MDNet network. To validate the effectiveness of the proposed tracker, we compare it with a number of existing state-of-the-art trackers in terms of success rate and precision on OTB-100, OTB-50 and CVPR 2013 datasets. The test results have demonstrated the effectiveness and improvement of the proposed tracking algorithm.

INDEX TERMS Channel attention, deep convolutional network, object tracking.

I. INTRODUCTION

Object tracking has long been a challenging and hot research problem in computer vision, which has broad applications in areas of visual navigation, traffic monitoring, military guidance, astronomical observation and meteorological analysis. Object tracking estimates the motion state of the object, including the position, speed and acceleration in subsequent video frames given the information of the initial frame. To track the target in videos, the algorithm usually consists of two models, the appearance model and motion model. The appearance model can be further divided into two categories, generative model and discriminative model. The generative model maintains a target template by learning features online and search for the optimal image region that best matches the template. The corresponding region is the predicted position of the target. The discriminative model considers the tracking process as a binary classification problem which extracts the features from the target and background to train a classifier, which is used to separate the target from the image background of video frames. Many tracking algorithms have been developed to attack the problem, for example, classic tracking methods such as Mean shift [1], [2], correlation filtering [3], [4]. In recently years, deep learning technologies have achieved great success in computer vision tasks such as image classification [5], segmentation [6], cross-modal retrieval [7]. In object tracking, deep learning based methods are also widely used, such as DeepSRDCF [8], CCOT [9], ECO [10], and Siames network [11].

The typical object tracking algorithm consists of several modules [12]: motion model, feature extraction, observation model, and model updater, as shown in Fig. 1.

1) Motion model: It generates a set of candidate regions or bounding boxes that locate the target based on the estimation of the previous frame. The motion model mainly establishes the relationship between the object motion states in the previous and subsequent frames in the entire video sequence, and predicts the target directly or indirectly in the candidate frames.
2) Feature extraction: It converts targets into easy-to-handle representations that affect the final performance of the tracker. Feature extraction methods include manual feature extraction [13] and automated feature extraction such as image processing and deep learning based methods [14], [15].
3) Observation model: The observation model makes a confidence judgment on the candidate area in the current frame, and calculates the probability that the candidate region is the target.

The associate editor coordinating the review of this manuscript and approving it for publication was Ye Duan.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/

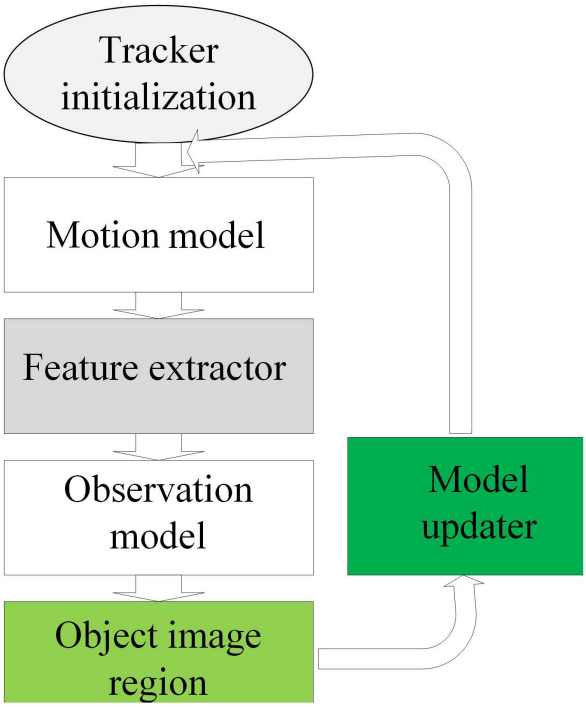

FIGURE 1. Component modules of a typical tracker.

4) Model updater: The model updater determines the strategy and frequency of updating the observation model.

Object tracking can be difficult when the tracking target is occluded by other unrelated objects, moving strenuously or similar objects appear in the view. Tracking algorithms should be robust to these situations. Although there have been various methods trying to solve these problems from different perspectives, a large room still remains to improve. Attention mechanism has been widely used in computer vision tasks such as image classification and segmentation. Several such attempts have been made [16]–[18] to incorporate attention processing to improve the performance of CNNs in large scale image classification tasks [19]. In deep learning based object tracking, considerably less attention is paid to the attention mechanism across different feature channels. In order to decrease the interference from the background and highlight the object, channel-wise attention can be used to help distinguish the object from the background as channel attention focuses on "what" is meaningful given an image [19]. Tracker MDNet [20] is a six-layer model, in which the first five layers are pre-trained using a large number of video sequences to capture the sequence independent features and fine-tune the last a few layers during tracking to capture the sequence dependent information. This design and model update strategy achieves good balance between computational efficiency and tracking accuracy. Therefore, in this work, we propose to improve MDNet based on channel attention mechanism. Our work is motivated by the observation that the channel information of the image mainly conveys object identity information, which can be used to differentiate similar objects. In our work, we embed the channel attention module at the higher layers of MDNet network. We also use group normalization to alleviate the batch size problem caused by batch normalization during training and testing as the tracking network is trained offline using video sequences with ground truth data and fine-tuned online with test videos. We evaluate the proposed tracker on benchmark datasets of OTB-100, OTB-50, and CVPR-13. Extensive experiments with ablation studies demonstrate that the proposed method has achieved better tracking accuracy. In summary,the major contributions of our work are the followings.

1) We propose to use the channel attention mechanism to highlight the foreground target and weaken the background by paying different attentions to different channels. This way, similar objects can be better distinguished.
2) We apply the group normalization to reduce errors caused by the batch size in batch normalization.
3) We implement the proposed method based on MDNet and evaluate it extensively on three benchmark datasets. We show that channel attention mechanism performs well against the state-of-the-art methods in terms of tracking accuracy, especially in scenarios where similar objects appear in the view.

The rest of the paper is organized as follows. Section II presents the related work of attention mechanism and group normalization. In Section III, we describe the channel attention mechanism and group normalization. Section IV presents the experimental results on three benchmark datasets. Section V concludes the paper.

II. RELATED WORK

Visual tracking has been an active research area for a long time. Deep learning technologies, especially deep convolutional neural networks have greatly advanced the accuracy and effectiveness of object tracking due to the strong capabilities of rich feature representation [20]. Recent researches have mainly advanced in three directions: increasing the depth of the network, expanding the width of the network, and the cardinality [19]. Various deep learning trackers have been proposed to attack the core problem in tracking, which is how to detect and locate the object accurately and efficiently in challenging scenarios with occlusions, out-of-view, deformation, background cluttering, existence of similar objects and other variations. In DeepSRDCF [8], the authors propose to use features from the convolutional layers of a CNN for object tracking. In CCOT [9], Martin *et al.* goes beyond the conventional DCF (Discriminative Correlation Filters) and employs an implicit interpolation model to pose the learning in the continuous spatial domain and proposes a continuous convolutional operator.

Siamese networks, which learn similarity functions to match candidate frame against template image [21], [22], have drawn great attention in visual tracking community because of the favorable performance and

end-to-end trainability. Luca Bertinetto from Oxford University put forward a basic tracking framework: SiamFC [21], [23], [24], which trains the model offline to learn the similarity to the initial frame using available training dataset and detects the target online by siamese convolution.

An important characteristic of the human visual system is that it does not attempt to process the entire scene at once. Instead, humans use a series of local glimpses and selectively focus on salient parts to better identify the visual environment. Attention allows humans to sift out valuable part from a large amount of information. Similarly, in computer vision, attention mechanism has been widely used to make the algorithm to focus on the informative and important part of the image. Many work have been carried out on how to design deep neural networks equipped with attention mechanism. According to the attention domain, the attention methods roughly fall into three categories, spatial domain, channel domain, and mixed domain [25]. In the work of [26], a spatial attention deep network combineed with partial PSO (Particle Swarm Optimization) is proposed. In work of squeeze-and-excitation network [17], the squeeze step uses global average pooling to convert the feature map of size $H \times W \times C$ into a vector of $1 \times 1 \times C$ such that the network can see the channels with different weights, where H, W, C are the height, width and number of channels of the feature map, respectively. Wang *et al*. utilize attention mechanism in a residual attention network that consists of bottom-up top-down feedforward structure to perform the image classification task [16]. In Cbam [19], the method combines the spatial and channel attention to take the both advantages.

III. METHOD

A. CHANNEL ATTENTION MODULE DESIGN

In this section, we present the design of the proposed channel attention module. This is a simple and effective attention module for feed forward convolutional neural networks. Our module is based on channel attention, which is more suitable for object tracking as spatial attention require much more computation. Channel attention uses channel-to-channel relationships to generate channel attention maps. To efficiently calculate the channel attention, we compress the spatial dimensions of the input feature map. In order to learn spatial information, average pooling has been widely used so far. In addition to previous work, we think that max pooling collects another important clues about object characteristics to infer more elaborate channel attention. Therefore, we use both average pooling and maximum pooling features. Because the number of layers of this network is relatively small, channel attention only applies to high-level semantics at higher layers of the tracking model. The channel block attention module is a lightweight universal module so that it can be seamlessly integrated into any CNN architecture and the calculation and parameter overhead is negligibly small as we only add one channel attention layer to the orignal MDNet model. In addition, the final model can be trained end-to-end. Fig. 2 shows the network structure of the channel attention module. The

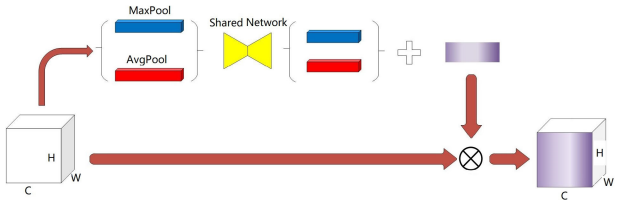

FIGURE 2. Channel attention module. As illustrated, the input feature map is passed through max-pooling and average-pooling in parallel. A following shared network which is a multi-layer perceptron network is used before the information being merged. Shared network is critical as it can reduce the number of parameters and the computation.

TABLE 1. Structure of channel attention network.

Layers	Input	Output
Average pooling	$512 \times 3 \times 3$	512×1
Max pooling	$512 \times 3 \times 3$	512×1
Shared Layers		
Fully connected 1	512×1	32×1
Activation 1	32×1	32×1
Fully connected 2	32×1	512×1
Activation 2	512×1	512×1

input feature map on the left is input to the maximum pooling and the average pooling in parallel. After passing through the shared network, the data is extended to the dimension of the feature map and finally merged. The shared block contains two fully connected layers and corresponding activation layers. Activation layer 1 is a ReLU function and activation layer 2 is a sigmoid function. Table 1 shows the parameters of the module layers. Given an intermediate feature map $F \in R^{C\times H\times W}$ as an input, the channel attention module calculates the sum of the one-dimensional channel attention map $M_c \in R^{C\times 1\times 1}$.

$$F^* = M_c(F) \bigotimes F \tag{1}$$

where $\bigotimes$ represents element-wise multiplication to expand the corresponding attention value, *i.e*. broadcasting channel attention values along the channel dimension. F^* is the final output. We first use the average pooling and maximum pooling operations to aggregate the spatial information of the feature map to generate two different spatial context descriptors; then forward the two descriptors to the shared network to generate channel attention map $M_c \in R^{C\times 1\times 1}$. A shared network consists of a multi-layer perceptron (MLP), whose dimensions are determined by the channels of input features. To reduce parameter overhead, the hidden activation layer size is set to $R^{C/r\times 1\times 1}$, where r is the reduction ratio, which is set to 16 in our implementation. After applying the shared network to each descriptor, we use element-wise summation to merge the output feature vectors. In summary, channel attention is calculated as:

$$\begin{aligned} M_c(F) &= \sigma(MLP(AvgPool(F)) + MLP(MaxPool(F))) \\ &= \sigma(W_1(W_0(F^c_{avg})) + W_1(W_0(F^c_{max}))) \end{aligned} \tag{2}$$

where σ is the sigmoid function, $W_0 \in R^{C/r \times C}$ and $W_1 \in R^{C \times C/r}$. Note that the two inputs share the weight of the MLP, namely $W_0 \in R^{C/r \times C}$ and $W_1 \in R^{C \times C/r}$ and the activation function follows W_0 immediately.

B. GROUP NORMALIZATION

In the process of training and testing, we notice the significant differences between training and testing during tracking. Group-normalized network structure is applied to reduce the bias caused by the inconsistent data volume during training and verification [27]. Unlike batch normalization [28] which performs more global normalization along the batch dimension, group normalization divides the channels into groups and is easy to implement. A series of normalization includes BN(batch normalization), LN (layer normalization) [29], and IN (instance normalization) [30] and GN (group normalization) [27]:

$$x_i = \frac{1}{\sigma_i}(x_i - u_i) \tag{3}$$

Here x represents the feature from a layer and i is the index. For example, for the case of two-dimensional images, $i = (i_N, i_C, i_H, i_W)$ is a 4D vector indexing the features in (N, C, H, W) order, where N is the axis of the batch, C is the dimension of the channel, H and W are the height and width dimensions of the space. The u and σ in the above formula is the mean and standard deviation calculated by:

$$u_i = \frac{1}{m}\sum_{k \in S_i} x_k \tag{4}$$

$$\sigma_i = \sqrt{\frac{1}{m}\sum_{k \in S_i}(x_k - u_i)^2 + \varepsilon} \tag{5}$$

where ε is a small constant, S_i is a set of pixels to compute the mean and standard deviation, and m is the size of this set. The difference between different normalization methods lies in the definition of S_i. The layer normalization set is defined as $S_i = \{k|k_N = i_N\}$, which means that LN computes u and σ along the (C, H, W) axes for each sample. In IN, this set is defined as $S_i = \{k|k_N = i_N, k_C = i_C\}$, which means IN calculates u and σ along (H, W) axes for each sample and each channel.

In all methods, BN, LN, IN compensate for the lack of representation by learning the linear transformation for each channel: $y_i = \gamma\hat{x}_i + \beta$ where γ and β are learnable scale transforms and offsets. The calculations of set GN can be defined by:

$$S_i = \{k|k_N = i_N, \lfloor\frac{k_C}{C/G}\rfloor = \lfloor\frac{i_C}{C/G}\rfloor\} \tag{6}$$

Here G is the number of groups to be grouped, and is set to 32 as a hyperparameter by default. C/G is the number of channels in each group, GN is the operation between LN and IN, and the mean and variance are calculated along $(C/G, W, H)$.

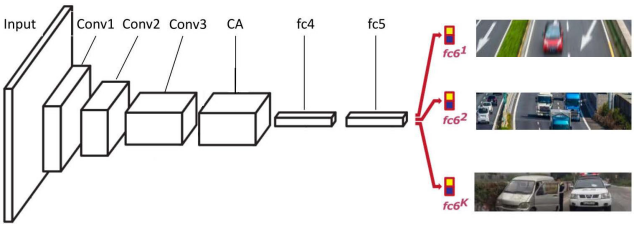

FIGURE 3. Model structure of the proposed tracker. CA is the channel attention module in Fig. 2.

TABLE 2. Network architecture. GN is group normalization, CA is channel attention. ReLU is the activation layer.

Layer1	convolution	ReLU	GN	max pooling
Layer2	convolution	ReLU	GN	max pooling
Layer3	convolution	ReLU	CA	
Layer4	fully connected	ReLU		
Layer5	fully connected	ReLU		
Layer6	softmax			

C. TRACKING PROCESS

1) MAIN MODEL

We modify the network of MDNet [20] by integrating channel attention process and group normalization. Fig. 3 shows the main model structure. Table 2 shows the network architecture. The first three layers are clipped from a pre-trained VGG model [31]. Each layer starts with a convolution layer, followed by the activation layer with the ReLU function. The first and second layers are then followed by group normalization and maximum pooling layer. Considering that the network should be small and easier to train, channel attention layer is appended to the end of the third layer. The fourth and fifth layer are fully connected layer and activation layer. Similar to MDNet, the main model is optimized through multi-domain training and the last fully connected branches ($fc6^1$ - $fc6^K$) are replaced by single one (fc6), which will be fine-tuned during online training.

2) TRACKING ALGORITHM

The overall procedure of our tracking algorithm is presented in Algorithm 1, which is similar to that of [20].

Weight Initialization: The tracker has only one model. Let w_j denote the weight of the jth layer filter in the tracking network. $w_{1:5}$ loads the parameters from the pre-trained model and w_6 is randomly initialized. In the process of online tracking, the parameters of the convolution kernel $w_{1:3}$ and the parameters of the attention layer are fixed, while the fully connected layers $w_{4:6}$ are updated online. Such an update strategy not only makes the tracking network more efficient, but also avoids the over-fitting problems caused by pre-training.

Motion Model: we generate candidate regions around the target in the previous frame according to the Gaussian

Algorithm 1 Online Tracking Algorithm

Input:
Pre-trained CNN filters $w_1, \ldots, w_5$,
Attention layer mlp1, mlp2
Initial target state x_1
Output: Estimated target state x_t

1) Randomly initialize the last layer w_6
2) Train a bounding box regression model
3) Draw positive samples S_1^+ and negative samples S_1^-
4) Update w_4, w_5, w_6 using S_1^+ and S_1^-
5) $T_s \leftarrow \{1\}$ and $T_l \leftarrow \{1\}$
6) Repeat:
7) Draw target candidate samples x_t^i
8) Find the optimal target x_t^* by $x^* = \arg\max_{x^i} f^+(x^i)$
9) if $f^+(x_t^*) > 0$
10) Draw training samples S_t^+ and S_t^-
11) $T_s \leftarrow \{t\}, T_t \leftarrow \{t\}$
12) Adjust x_t^* using bounding box regression
13) if $f^+(x_t^*) < 0$
14) Update w_4, w_5, w_6 using $S^+_{v\in T_s}$ and $S^-_{v\in T_s}$
15) else if $t\ mod(10) == 0$
16) Update w_4, w_5, w_6 using $S^+_{v\in T_l}$ and $S^-_{v\in T_s}$
17) Until the end of sequence

distribution. The mean value of the Gaussian distribution is x_{t-1}^*, which is object state in previous frame and the covariance is a diagonal matrix $diag(0.09r^2, 0.09r^2, 0.25)$, where r is the length of object in previous frame.

Model Updater: In order to increase the robustness and adaptability of the tracker, we adopt long-term and short-term update strategy during the tracking process. When the model score $f^+(x_t^*) > 0$ on the target, the data is updated, but the model is not updated. This is the long-period update. When a tracking failure is detected, *i.e.* $f^+(x_t^*) < 0$, a short-period update is performed, in which the model is updated using the samples collected during successful tracking. In both update schemes, the negative samples are collected from short-period updates as the negative samples of the previous frames are usually irrelevant to the current frame target. The execution of long-period updates and short-period updates depends on how fast the appearance of the target changes.

IV. EXPERIMENT

A. IMPLEMENTATION DETAILS

1) OFFLINE TRAINING

We choose ImageNet 2015-VID to train the offline model as it contains large number of videos and are quite different from the test videos. 3683 video sequences are selected with high integrity and the target occupance ratio is less than 0.5. We randomly select 8 frames from each of these video sequences. In each frame, 50 positive and 200 negative patches are generated. A patch is labeled positive if the IOU

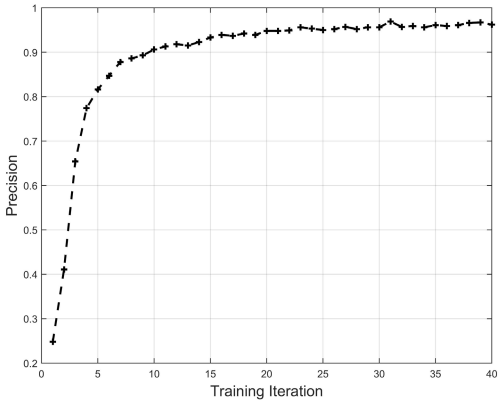

FIGURE 4. Precision of the model during offline training. X axis is the iteration, Y axis is the precision score.

to the ground truth bounding box is over 0.7, negative is less than 0.5. The model is updated with a mini-batch which contains only 96 negatives and 32 positives produced by the hard negative example mining procedure. Fig. 4 shows the precision curve during the offline training. We can see that the model precision converges to around 0.96.

2) MODEL UPDATING AND ONLINE TRAINING

The same as MDNet, the time period for short term and long term update is set to $|T_s| = 20$, $|T_l| = 100$ and the model generates 256 candidate regions around the target in the previous frame according to the Gaussian distribution. During online updating, the learning rates for ordinary convolution layers, convolution layer in the attention module and the fully connected layers are 0.0001, 0.004 and 0.001 respectively. Similar to offline training, we collect 200 positive samples and 50 negative samples according to the IOU value with the calibration bounding box using thresholds 0.7 and 0.3. When the first frame is initialized, there are 5000 negative samples (S_1^-) and 500 positive samples (S_1^+). For the bounding box regression model, we train it using 1000 training samples with the same parameter settings as in [32].

B. EXPERIMENTAL RESULTS

OTB50 or OTB2013 [33] contains 50 fully annotated sequences that are collected from commonly used tracking sequences. OTB100 or OTB2015 [34] is the extension of OTB2013 and contains 100 video sequences. Some new sequences are more difficult to track. CVPR-13 [33] has 51 fully annotated sequences. The evaluation is based precision plot and success plot. The precision plot shows the percentage of frames with respect to the center distance in pixels between the tracking result and ground truth. The success plot shows the percentage of successful frames with respect to the overlap ratio between the tracking result and ground truth. The area under curve (AUC) of each success plot is used to rank the tracking algorithm.

We compare our methods to the state-of-the-art trackers, in which MDNet [20] is built on ubuntu16.04's

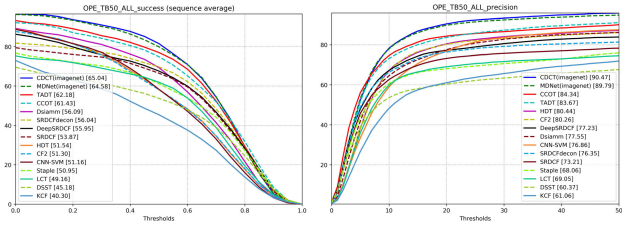

FIGURE 5. Performance comparison of trackers on OTB-50. The left one is success plot, right one is precision plot.

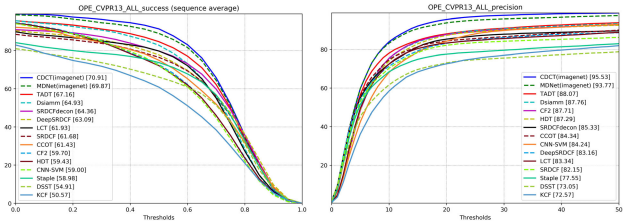

FIGURE 6. Performance comparison of trackers on CVPR-13. The left one is success plot, right one is precision plot.

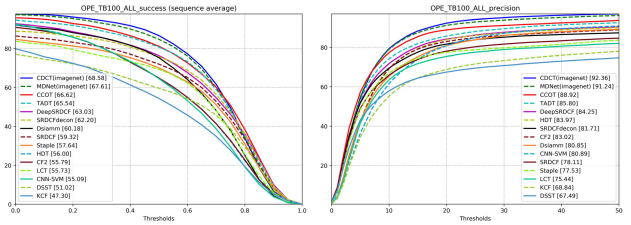

FIGURE 7. Performance comparison of trackers on OTB-100. The left one is success plot, right one is precision plot.

pytorch1.1 deep learning framework, opencv3.3.0 and python3.6. The trackers tested are TADT [35](2019CVPR), Dsiamm [11] (2018ECCV), C-COT [9] (2016ECCV), HDT [36] (2016CVPR), CF2 [37] (2015ICCV), SRD-CFdecon [38] (2016CVPR), SRDCF [39] (2016CVPR), SAMF [40] (2014ECCV), Staple [41](2016CVPR), DSST [42](2014BMVC), KCF [43](2015TPAMI), LCT [44] (2015CVPR), CNN-SVM [45] (2015ICML), DeepSRDCF [8] (2015ICCV).

1) RESULTS ON FULL DATASET

Fig. 5, 6 and 7 show the test results of the proposed tracker (CDCT) on datasets of OTB-50, CVPR-13, and OTB-100. From the curves in the figure, we can see that our tracker is the best among these trackers in terms of success rate and precision. Table 3 shows the concrete numbers. From the table, our tracker is slightly better than that of MDNet and significantly better than that of TADT and DSiamm, which were published in 2019 and 2018 respectively.

2) RESULTS ON DATASET WITH BACKGROUND CLUTTERS

We also run the comparison tests on the subset of 33, 20 and 23 video sequences with background clutters selected from OTB-100, OTB-50, CVPR-13 respectively. Fig. 8, 9 and 10

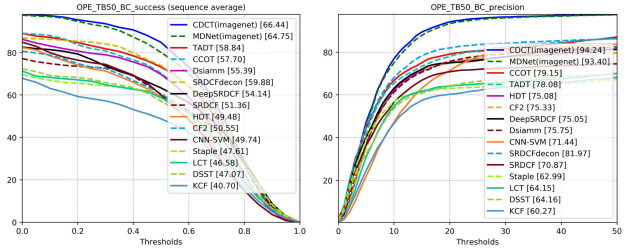

FIGURE 8. Performance comparison of trackers on OTB-50 with BC (Background Clutters). The left one is success plot, right one is precision plot.

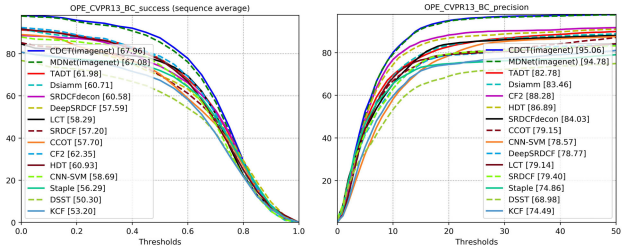

FIGURE 9. Performance comparison of trackers on CVPR-13 with BC (Background Clutters). The left one is success plot, right one is precision plot.

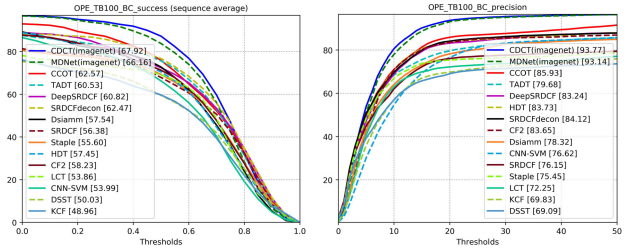

FIGURE 10. Performance comparison of trackers on OTB-100 with BC (Background Clutters). The left one is success plot, right one is precision plot.

show the test results of the proposed tracker (CDCT) on datasets of OTB-50, CVPR-13, and OTB-100. Table 4 shows the numbers for the performance of MDNet and CDCT. For the performance of other trackers, please refer to the curves in the figures. We can see that on this subset of test videos with background clutters, our proposed tracker still performs best. Specifically, on the 33 sequences from OTB-100, CDCT achieves success rate of 67.92 and precision score of 93.77, which are both higher than those of MDNet, 66.16 and 93.14 for success rate and precision respectively.

3) TEST EXAMPLES

Fig. 11, 12 and 13 show an example in which the left figure is the 660th frame and the right one is 663rd frame. Fig. 11 is the ground truth with the target person annotated. Fig. 12 shows the prediction given by MDNet and Fig. 13 is the prediction of our proposed tracker. We can see that our tracker makes a correct prediction even with similar objects around the target.

TABLE 3. Tracker performance comparison on three full datasets.

	Success rate			Precision		
Tracker	TB100	TB50	CVPR13	TB100	TB50	CVPR13
MDNet (2016CVPR)	67.61	64.58	69.87	91.24	89.79	93.77
CCOT (2016ECCV)	66.62	61.43	61.93	88.92	84.43	84.34
TADT (2019CVPR)	65.54	83.67	67.61	85.8	62.18	88.07
DeepSRDCF (2015 ICCV)	63.03	55.95	63.09	84.25	77.23	83.16
SRDCFdecon (2016CVPR)	62.20	56.04	64.36	81.71	76.35	85.33
Dsiamm (2018ECCV)	60.18	56.09	64.93	80.85	77.55	87.76
SRDCF (2016CVPR)	59.32	53.87	61.68	78.11	73.21	82.15
Staple (2016CVPR)	57.64	50.95	58.98	77.53	68.06	77.55
HDT (2016CVPR)	56.00	51.54	59.43	83.97	80.44	87.29
CF2 (ICCV2015)	55.79	51.30	59.70	83.02	80.26	87.71
LCT (CVPR2015)	55.73	49.16	61.93	75.44	69.05	83.34
CNN-SVM (2015ICML)	55.09	51.16	59.00	80.89	76.86	84.24
SAMF (2014ECCV)	54.92	46.92	58.98	74.46	65.01	77.17
MEEM (2014ECCV)	52.54	47.33	57.17	77.40	71.22	81.58
DSST (2014BMVC)	51.02	45.18	54.91	67.49	60.37	73.05
KCF (2015TPAMI)	47.30	40.30	50.57	68.84	61.06	72.57
CDCT (imagenet)	**68.58**	**65.04**	**70.09**	**92.36**	**90.47**	**95.53**

TABLE 4. Tracker performance comparison on subset with background clutters.

	Success rate			Precision		
Tracker	TB100	TB50	CVPR13	TB100	TB50	CVPR13
MDNet (2016CVPR)	66.16	64.75	67.08	93.14	93.40	94.78
CDCT (ImageNet)	67.92	66.43	67.96	93.77	94.24	95.06

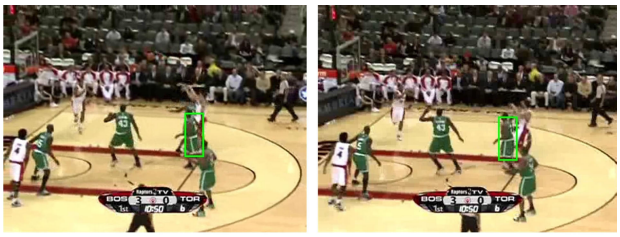

FIGURE 11. Test example: ground truth. The left is frame 660, the right is frame 663. The target person is annotated in the boxes.

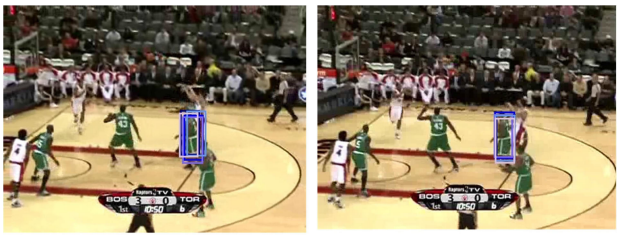

FIGURE 13. Test example: result of the proposed method. The left is frame 660, the right is frame 663. The bounding boxes are the prediction from the proposed tracker.

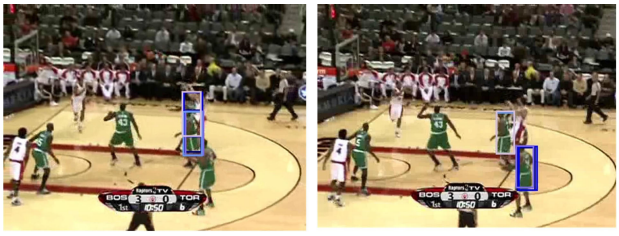

FIGURE 12. Test example: result of MDNet. The left is frame 660, the right is frame 663. The bounding boxes are the prediction from MDNet tracker.

C. ABLATION EXPERIMENT

In this section, we perform an ablation experiment to see performance of the tracker. Our tracker integrates channel attention module and group normalization based on MDNet, which enhances the foreground by weakening the background, making the tracker more robust against similar backgrounds. The network parameters increase by 0.2MB, which is relatively small, and will not significantly affect the tracking performance. We compare our tracker with MDNet with or without channel attention and group normalization. The experiment is done using the video sequences from ImageNet 2015-VID object detection. In Table 5, the first row is the final result. We can conclude that the channel attention deformation module contributes the most to the final effect, and the group normalization has less contribution. Group normalization is effective only when the number of training batches and test batches reaches a certain order of

TABLE 5. Ablation experiment results.

	Success			Precision		
	TB100	TB50	CVPR13	TB100	TB50	CVPR13
MDNet	67.61	64.58	69.87	91.24	89.79	93.77
MDNet+Channel	68.23	64.94	70.38	91.68	89.97	94.04
MDNet+GN	67.82	64.78	69.96	90.35	88.01	94.33
CDCT	**68.58**	**65.04**	**70.91**	**92.36**	**90.04**	**95.53**

magnitude. In this experiment, during training, each batch has 128 patches and in testing, the batch size is 256. The difference between training and testing batch size is not very big. In summary, the channel attention module has made the major contribution to the improvement of the proposed tracker over the original MDNet.

V. CONCLUSION

In this work, we propose a tracking method by improving MDNet in two ways. The first one is the integration of channel attention module, which is designed to make the tracker more robust when there are similar objects in the background. The second one is the group normalization technique, which can effectively solve the inconsistency between offline, online training and testing if batch normalization is used. We have tested the proposed tracker on OTB-100, OTB-50 and CVPR 2013 datasets, compared its performance with a number of existing state-of-the-art trackers in terms of success rate and precision. The test results have demonstrated the effectiveness and improvement of the proposed tracking algorithm.

REFERENCES

[1] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 8, pp. 790–799, Aug. 1995.

[2] F. Yang, H. Lu, W. Zhang, and G. Yang, "Visual tracking via bag of features," *IET Image Process.*, vol. 6, no. 2, pp. 115–128, 2012.

[3] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, Mar. 1960.

[4] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 425–437, Aug. 2002.

[5] D. Meng, G. Cao, Y. Duan, M. Zhu, L. Tu, D. Xu, and J. Xu, "Tongue images classification based on constrained high dispersal network," *Evidence-Based Complementary Alternative Med.*, vol. 2017, pp. 1–12, Mar. 2017.

[6] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

[7] W. Cao, Q. Lin, Z. He, and Z. He, "Hybrid representation learning for cross-modal retrieval," *Neurocomputing*, vol. 345, pp. 45–57, Jun. 2019.

[8] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. Workshop (ICCVW)*, Dec. 2015, pp. 58–66.

[9] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 472–488.

[10] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "ECO: Efficient convolution operators for tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6638–6646.

[11] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu, "Distractor-aware siamese networks for visual object tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 101–117.

[12] N. Wang, J. Shi, D.-Y. Yeung, and J. Jia, "Understanding and diagnosing visual tracking systems," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3101–3109.

[13] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. ICCV*, 1999, vol. 99, no. 2, pp. 1150–1157.

[14] W. Cao, Y. Li, Z. He, G. Cao, and Z. He, "Supplementary virtual keypoints of weight-based correspondences for occluded object tracking," *IEEE Access*, vol. 6, pp. 9140–9146, 2018.

[15] R. Wang, Y. Shi, and W. Cao, "GA-SURF: A new speeded-up robust feature extraction algorithm for multispectral images based on geometric algebra," *Pattern Recognit. Lett.*, vol. 127, pp. 11–17, Nov. 2019.

[16] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3156–3164.

[17] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

[18] W. Cao, Y. Li, and Z. He, "Weighted optical flow prediction and attention model for object tracking," *IEEE Access*, vol. 7, pp. 144885–144894, 2019.

[19] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon, "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 3–19.

[20] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4293–4302.

[21] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 850–865.

[22] J. Yan, C. Li, Y. Li, and G. Cao, "Adaptive discrete hypergraph matching," *IEEE Trans. Cybern.*, vol. 48, no. 2, pp. 765–779, Feb. 2018.

[23] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a 1'siamese' time delay neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, 1994, pp. 737–744.

[24] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. S. Torr, "End-to-end representation learning for correlation filter based tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2805–2813.

[25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[26] Q. Ye, S. Yuan, and T.-K. Kim, "Spatial attention deep net with partial PSO for hierarchical hybrid hand pose estimation," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 346–361.

[27] Y. Wu and K. He, "Group normalization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 3–19.

[28] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: https://arxiv.org/abs/1502.03167

[29] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*. [Online]. Available: https://arxiv.org/abs/1607.06450

[30] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," 2016, *arXiv:1607.08022*. [Online]. Available: https://arxiv.org/abs/1607.08022

[31] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Sep. 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556

[32] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[33] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2411–2418.
[34] Y. Wu, J. Lim, and M. H. Yang, "Object tracking benchmark," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1834–1848, Sep. 2015.
[35] X. Li, C. Ma, B. Wu, Z. He, and M.-H. Yang, "Target-aware deep tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1369–1378.
[36] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, J. Lim, and M.-H. Yang, "Hedged deep tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4303–4311.
[37] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3074–3082.
[38] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg, "Adaptive decontamination of the training set: A unified formulation for discriminative visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1430–1438.
[39] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4310–4318.
[40] Y. Li and J. Zhu, "A scale adaptive kernel correlation filter tracker with feature integration," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 254–265.
[41] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr, "Staple: Complementary learners for real-time tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1401–1409.
[42] M. Danelljan, G. Häger, F. Shahbaz Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *Proc. Brit. Mach. Vis. Conf.*, 2014.
[43] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, Mar. 2015.
[44] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 5388–5396.
[45] S. Hong, T. You, S. Kwak, and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 597–606.