Learning Attribute-Specific Representations for Visual Tracking

Yuankai Qi[1] Shengping Zhang[1] Weigang Zhang[1,2] Li Su[2] Qingming Huang[1,2] Ming-Hsuan Yang[3]
[1]Harbin Institute of Technology, Weihai, China
[2]University of Chinese Academy of Sciences, Beijing, China
[3]University of California at Merced, America
{yk.qi, s.zhang, wgzhang}@hit.edu.cn, {suli, qmhuang}@ucas.ac.cn, mhyang@ucmerced.edu

Abstract

In recent years, convolutional neural networks (CNNs) have achieved great success in visual tracking. Most of existing methods train or fine-tune a binary classifier to distinguish the target from its background. However, they may suffer from the performance degradation due to insufficient training data. In this paper, we show that attribute information (e.g., illumination changes, occlusion and motion) in the context facilitates training an effective classifier for visual tracking. In particular, we design an attribute-based CNN with multiple branches, where each branch is responsible for classifying the target under a specific attribute. Such a design reduces the appearance diversity of the target under each attribute and thus requires less data to train the model. We combine all attribute-specific features via ensemble layers to obtain more discriminative representations for the final target/background classification. The proposed method achieves favorable performance on the OTB100 dataset compared to state-of-the-art tracking methods. After being trained on the VOT datasets, the proposed network also shows a good generalization ability on the UAV-Traffic dataset, which has significantly different attributes and target appearances with the VOT datasets.

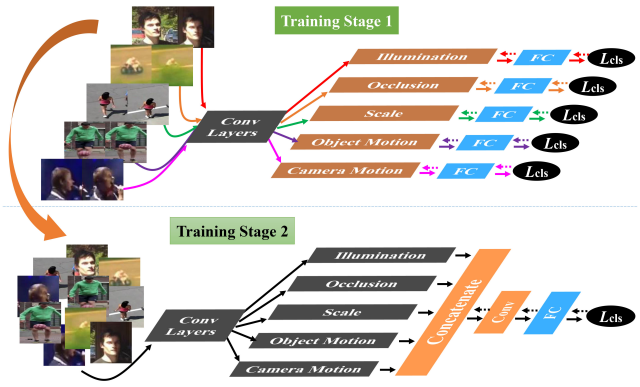

Figure 1: The proposed attribute-based CNN is trained by two stages. In the first stage, five attribute branches are trained individually to learn representations under different attributes. In the second stage, two ensemble layers are trained to learn how to effectively combine attribute-specific representations together.

1 Introduction

Visual tracking has attracted increasing attention in the past decades due to its key roles in numerous applications such as automatic driving, surveillance, and video analysis. While considerable effort has been devoted to developing robust algorithms, tracking a target in a complex environment is still a challenging task due to factors such as heavy occlusion, shape deformations, fast motion, etc.

Recently, features learned from CNNs have shown effectiveness for numerous vision tasks (Ren et al. 2016; Jiang et al. 2018b; Wang et al. 2018; Fan et al. 2018; Chen et al. 2018). Several CNN-based trackers have since been proposed (Nam and Han 2016; Valmadre et al. 2017; Danelljan et al. 2017; Yun et al. 2017; Park and Berg 2018; Luo et al. 2018; Zhang et al. 2018) for visual tracking and achieved state-of-the-art performances. Most of these methods formulate visual tracking as a target/background classification problem and train CNNs to perform the classification. As like other applications, their success relies on the amount of the training data. However, existing tracking datasets, such as VOT (Kristan et al. 2015), OTB100 (Wu, Lim, and Yang 2015) and UAV123 (Mueller, Smith, and Ghanem 2016), are far smaller than image classification datasets such as ImageNet (Russakovsky et al. 2015). The insufficient training data may cause the trained model ineffective when facing all kinds of tracking challenges, such as illumination changes, occlusion, and motion. In this paper, we show that tracking challenges (also known as attributes) are able to facilitate the training of the network with insufficient training data.

Different from existing CNN-based trackers, we propose to first train specific CNN branches under individual attributes as shown in the top panel of Figure 1. Such a design reduces the appearance diversity of the target under each attribute and thus eases the learning of the network parameters. Since the attribute is not known in testing and one frame may have multiple attributes, we propose to feed the input image into all branches to avoid explicitly recognizing the attributes of each frame. We further train ensemble layers to adaptively combine all attribute-specific representations as a

strong representation for the final target/background classification as shown in the bottom panel of Figure 1.

It is worth noticing that the training procedure of the proposed attribute-based CNN is not straightforward because the classification loss of a training sample with any attribute will be backwardly propagated to *all* the five branches. Thus, a simple end-to-end training cannot guarantee each branch to learn to classify data with the corresponding attribute. To address this problem, we propose a two-stage training method as illustrated in Figure 1. In the first stage, we train the attribute branches one-by-one with a temporary fully-connected (FC) layer for binary classification of samples with a specific attribute. In the second stage, we only train the ensemble layers to learn how to combine attribute-specific representations effectively. Experiments show that the two-stage training method helps to achieve better performance than a simple end-to-end training.

The contributions of this paper are summarized below:

- We propose an attribute-based CNN framework for visual tracking, which exploits attribute information of video frames to overcome the problem of lacking sufficient training data in visual tracking.
- We develop a two-stage training method to enable each branch to learn a specific representation and effectively combine attribute-specific representations of all branches for the final target/background classification.
- We carry out experiments on the large challenging UAV-Traffic and OTB100 datasets to demonstrate the generalization ability and the effectiveness of the proposed method against state-of-the-art tracking algorithms.

2 Related Work

In this section, we present a brief review of recent CNN-based tracking methods. For algorithms utilizing hand-crafted features, we refer the readers to the surveys (Zhang et al. 2013; Smeulders et al. 2014). As CNNs achieve great success on many computer vision tasks (Ma et al. 2015; Ren et al. 2018; Sun et al. 2017; Jiang et al. 2018a; Yang et al. 2018; Liu et al. 2018; Huang et al. 2018), numerous visual object tracking methods based on CNNs have been proposed, which roughly fall into two categories according to whether the stem of the network has parallel structures: one-stream CNN and multi-stream CNN.

One-stream CNN-based trackers. Reinforcement learning is utilized in ADNet (Yun et al. 2017) to learn an action-decision network so as to track a target by making a series of actions, e.g., translation moves and scale changes, to the target candidate. MDNet (Nam and Han 2016) adopts a network with multiple target/background classification branches at the last layer, where each branch is responsible for only one video, so that it is able to reduce the ambiguity caused by objects that are targets in some training videos but are backgrounds in others. SANet (Fan and Ling 2017) incorporates recurrent neural network into CNNs to model the structure information within an object in addition to the traditional semantic information. To efficiently compute correlation filters from large feature maps obtained by pre-trained image classification CNN models (e.g., VGG-M (Chatfield et al. 2014)), ECO (Danelljan et al. 2017) reformulates learning a large filter as learning a coefficient matrix and a compact set of basis filters. These one-stream CNNs aim to learn a set of parameters that can distinguish the target from its background under all kinds of attributes, which is rather hard due to large appearance difference of the same target under different attributes and the deficiency of training data.

Multi-stream CNN-based trackers. In (Tao, Gavves, and Smeulders 2016), the CNN is designed with two symmetrical streams (also known as the Siamese network) to learn a similarity function of two input images: one target template and one target search region. Later, an asymmetrical Siamese network is proposed (Valmadre et al. 2017) to simultaneously learn CNN features and correlation filters by introducing a differentiable correlation filter layer. To enhance the representation capability, spatial and temporal content differences between the first frame and the current frame are modeled in (Song et al. 2017) by two parallel residual streams after some common convolutional layers. The CNN in (Han, Sim, and Adam 2017) also has multiple streams in later layers, but these streams with different number of layers aim to maintain multi-level target representations. Different from the goals of the above mentioned multi-stream networks, our multi-stream CNN is proposed to address the problem of lacking sufficient training data by exploiting the attribute information in video frames.

3 Proposed Method

In this section, we first present the architecture of the proposed attribute-based CNN, and then describe the two-stage training method. Finally, we apply the attribute-based CNN to visual tracking.

3.1 Attribute-based Neural Network

The architecture of the proposed attribute-based CNN is shown in the bottom panel of Figure 1. It starts with several convolution layers borrowed from VGG-M (Chatfield et al. 2014), which are used to capture common low-level information (e.g., edges and texture) across objects. Then, five attribute branches are followed to learn representations under corresponding attributes. We adopt the five attributes provided by the VOT datasets (Kristan et al. 2013; 2014; 2015), namely target motions, camera motions, illumination variations, occlusions, and scale changes, which are able to cover most of the 11 attributes defined by the OTB100 dataset (Wu, Lim, and Yang 2015). In addition, the attributes of the VOT datasets are annotated for every frame, which allows us to divide training data into different attribute groups to train the corresponding branches.

Following the attribute branches are ensemble layers and an FC layer. The ensemble layers are composed of a concatenation layer and a convolution layer. In the testing stage, the frame attributes are not known and a frame may simultaneously have multiple attributes. Thus, it is unreasonable to process a frame using only one branch. Instead, we propagate an input image region through all attribute branches and

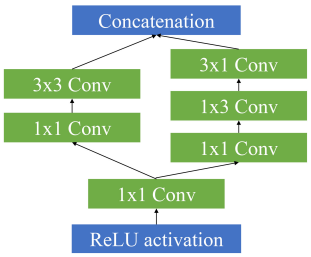

Figure 2: Architecture of each attribute branch. The ReLU layers after each Conv layer are omitted for clarity. Each number in the bracket denotes the number of channels of the output feature map.

train the ensemble layers to adaptively combine all attribute-specific representations together to obtain a comprehensive and discriminative representation. The output of ensemble layers is fed to the FC layer for the final target/background classification.

As for the architecture of each branch, we adopt the inception structure as shown in Figure 2 inspired by the recent success of CNNs in image classification (Szegedy et al. 2017). The first 1×1 Conv layer is used to capture common representations under that attribute. Then it is divided into two flows by using another two 1×1 Conv layers with half channels to decrease the dimensionality in the filter space. This is based on the success of embeddings: even low dimensional embeddings might contain a lot of information about a relatively large image patch (Szegedy et al. 2015). We implement 3×3 filters in two ways in the flows to perceive region fields with different sizes: one using the traditional 3×3 filter and the other one using the cascade of 1×3 and 3×1 filters. Their outputs are concatenated together as an attribute-specific representation.

3.2 Two-stage Training

A simple end-to-end training cannot guarantee each branch to learn to classify data of the corresponding attribute because the classification loss of a training sample with any attribute will be backwardly propagated to *all* the five branches. To address this problem, we propose a two-stage training strategy to first train the attribute branches and then learn the ensemble layers.

We use the VOT datasets (Kristan et al. 2013; 2014; 2015) excluding the ones that appear in the OTB100 dataset as training data. In stage 1, all frames are divided into five groups according to their attribute labels. In stage 2, the five groups are merged into one bigger group. In each frame, P positive and Q negative samples are cropped such that the positive and negative samples have $\geqslant 0.7$ and $\leqslant 0.5$ Intersection-over-Union (IoU) overlap ratios with the ground-truth bounding boxes, respectively.

Stage 1: training attribute branches. The five attribute branches are trained one by one. Specifically, we remove both the ensemble layers and the last FC layer from the pipeline, and add a new FC layer at the end of each branch as shown in the top panel of Figure 1. These FC layers are responsible for target/background classification. Given training samples of an attribute as input, low-level feature maps are first extracted and then fed into the corresponding branch for forward-backward propagation with the softmax-loss. The stochastic gradient descent (SGD) method with momentum is utilized for optimization. The momentum and weight decay are set to 0.9 and 0.0005, respectively. Each branch converges after approximate 200 iterations with a fixed learning rate of the value 0.001.

Stage 2: training ensemble layers. Once the attribute branches are ready, we start to train the ensemble layers in order to effectively combine these attribute-specific representations for the final target/background classification. In this stage, we remove the FC layers which are added into each attribute branch in Stage 1, and plug back the ensemble layers and the FC layer as shown in the bottom panel of Figure 1. The parameters of the five branches are fixed. The learning rates of ensemble layers and the last FC layer are set to 0.001. We adopt the softmax-loss for training. The training data and the SGD parameters are the same as in Stage 1. The network converges after approximate 150 iterations.

3.3 Tracking with Attribute-based CNN

As discussed in the design of the attribute-based CNN (see Section 3.1), there is no need to recognize attributes in the testing phase. Once we learn the attribute branches and ensemble layers, the last FC layer is re-trained from scratch to construct a tracker on a test image sequence. For that, positive and negative samples are extracted from the starting frame, where they have $\geqslant 0.7$ and $\leqslant 0.3$ IoU overlap ratios with the ground truth bounding box, respectively. Meanwhile, the ensemble layers are fine-tuned with a relatively small learning rate 0.0001 to adapt to new appearances of the target.

When the t-th frame arrives, we randomly sample M target candidates $\mathbf{x}_1, \cdots, \mathbf{x}_M$ around the target position in the last frame. The candidate with the maximum target classification score is selected as the tracking result

$$\mathbf{x}_\mathbf{t}^* = \underset{i=1,\cdots,\mathbf{M}}{\operatorname{argmax}} \Theta(\mathbf{x}_i), \tag{1}$$

where $\Theta(\cdot)$ outputs the target score using the learned attribute network. During tracking, the ensemble layers and the last FC layer are updated every 20 frames using positive and negative samples collected in previous frames.

4 Experimental Results

In this section, we present extensive experiment evaluations on the proposed **A**ttribute **N**etwork for visual **T**racking (ANT) algorithm. We first introduce the evaluation protocol. Then, we evaluate the effectiveness of each attribute branch. Finally, we evaluate the proposed ANT algorithm against several state-of-the-art trackers. We implement our algorithm in MATLAB, and use the MatConvNet toolbox (Vedaldi and Lenc 2015) to train the proposed network. Our unoptimized implementation runs at about one frame per second on a computer with an Intel I7-4790 CPU, 16G

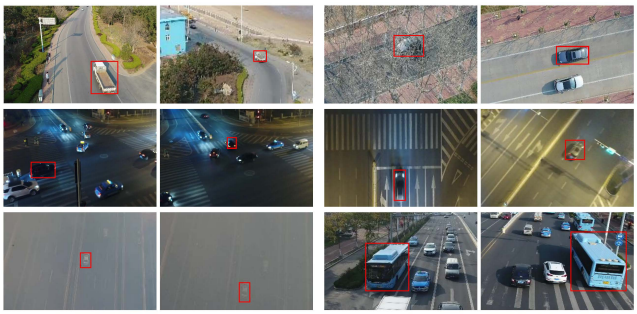

Figure 3: Examples of challenging videos in the UAV-Traffic dataset.

RAM, and a GeForce TITAN 1080Ti GPU card. The sample parameters P, Q, M are set to 32, 96, and 256, respectively.

4.1 Evaluation Protocols

To fully assess our method, we carry out extensive experiments on two datasets: (I) The object tracking benchmark dataset OTB100 (Wu, Lim, and Yang 2015), which contains 100 sequences involving 11 tracking challenges, such as illumination changes, in-plane and out-of-plane rotations, scale variation, and occlusion, low resolution, etc. (II) The unmanned aerial vehicle dataset for traffic, UAV-Traffic (Du et al. 2018), collected by ourselves under different weather (daylight, night, and fog), flying altitude (10$\sim$30m, 30$\sim$70m, and $>$70m), and camera view (front-view, side-view, and bird-view). The UAV-Traffic dataset contains 37,084 frames of 50 videos involving 8 challenges: background clutter, camera rotation, object rotation, small object, illumination variation, object blur, scale variation, and large occlusion. 74% of videos contain at least 4 visual challenges, and 27% of frames contribute to long-term tracking videos. Examples of challenging videos in the dataset are shown in Figure 3.

Following the standard paradigm of the OTB100 benchmark, experimental results are reported using success plots and precision plots in the one-pass evaluation (OPE). The success rate of a tracker is the proportion of successful frames with an overlap rate larger than a given threshold. Trackers are ranked based on the area under the curve (AUC) in success plots and are ranked based on the center location error at a threshold of 20 pixels in precision plots.

4.2 Evaluation of Attribute Branches

In Figure 4, we visualize the representations learned by individual attribute branches against those learned by a baseline that has the same architecture as each branch but is trained using data with all attributes. Figure 4(a) shows five groups of positive and hard negative samples, each of which is dominated by one main attribute. Figure 4(b) and (c) present the features obtained by the baseline and by attribute branches after being projected to the 2D space via the t-SNE technique (Laurens and Hinton 2008), respectively. It shows that given an image with a specific attribute, the branch corresponding to that attribute generates more discriminative fea-

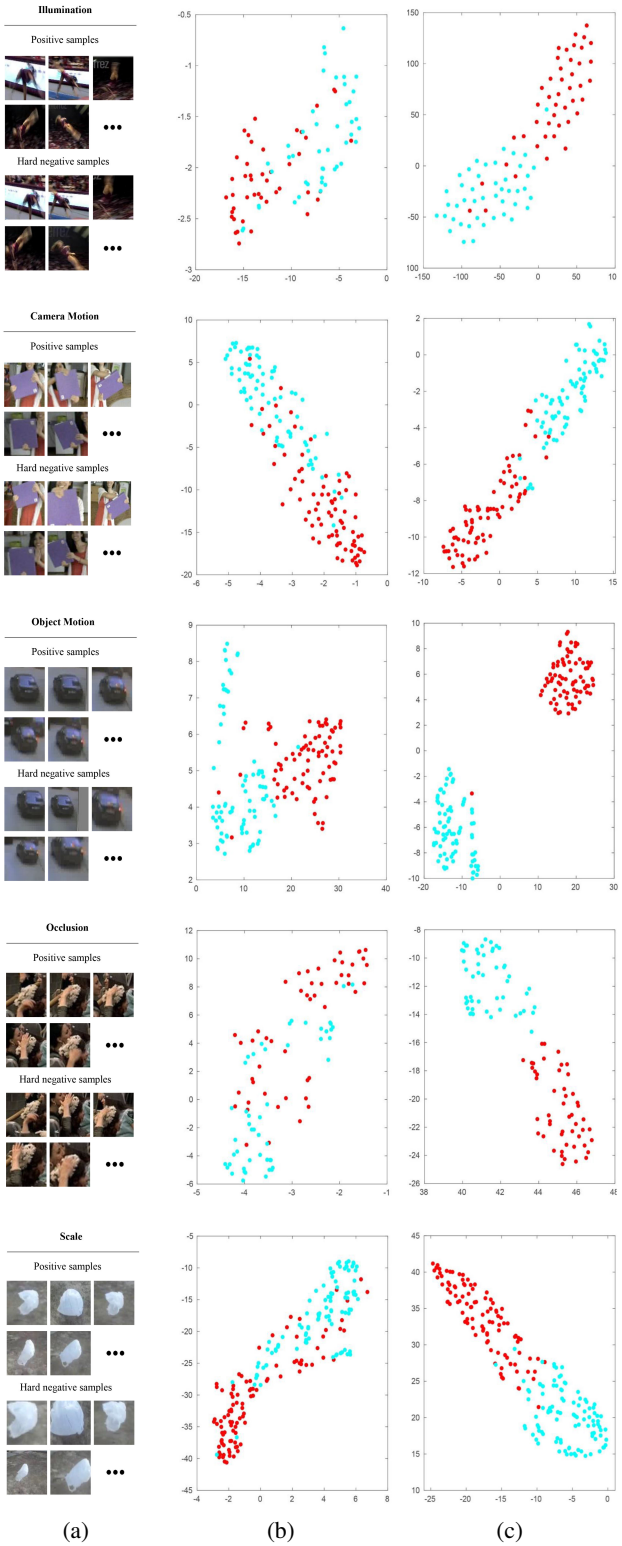

Figure 4: (a) Five groups of samples dominated by different attributes. (b) Projected features of the baseline branch. (c) Projected features of attribute branches.

tures than the network trained using data with all kinds of attributes.

In Table 1, we further quantitatively compare the effectiveness across branches to verify that each branch specializes in the resulting attribute. The evaluation is conducted on five typical videos which are dominantly challenged by different attributes. The results show that each attribute branch achieves better performance on the video dominated by the same attribute than other branches.

Table 1: Tracking results in terms of average success rate on five VOT videos (A: Skating, B: Bolt1, C: Tiger2, D: CarScale, E: Basketball), which are not used for training and denoted with the sequence names in the OTB100 dataset for clarity, each of which is challenged by one dominant attribute as indicated after the video name. The results are obtained by using five branches separately.

	Illu-branch	Mot-branch	Occ-branch	Scale-branch	Cam-branch
A: Illu	**69.7**	64.4	62.6	67.3	62.4
B: Mot	74.3	**76.4**	69.5	74.4	70.8
C: Occ	52.2	52.9	**54.0**	51.2	51.2
D: Scale	41.4	61.2	55.3	**62.2**	60.8
E: Cam	56.8	59.3	62.8	63.8	**70.6**

4.3 Evaluation of Two-stage Training

To evaluate the effectiveness of the proposed two-stage training method, we compare its tracking results on the OTB100 dataset against that obtained by training all branches together. The results are presented in Table 2. The proposed two-stage training method achieves consistent improvement of about 2% in terms of both the precision and AUC metrics.

Table 2: Tracking results on the OTB100 dataset using the proposed two-stage training method and training all branches as a whole, respectively.

	Two-stage Training	Training As a Whole
Prec.	90.5	88.6
AUC	67	65.8

4.4 Comparisons to State-of-the-art Trackers

We compare the proposed ANT algorithm to seven state-of-the-art trackers: ECO (Danelljan et al. 2017), CFNet (Valmadre et al. 2017), ADNet (Yun et al. 2017), CCOT (Danelljan et al. 2016), MDNet (Nam and Han 2016), HDT (Qi et al. 2016), and SINT (Tao, Gavves, and Smeulders 2016). For fair comparison, all these trackers are run on the same platform as ours.

Quantitative evaluation We present the tracking performance in terms of success plots and precision plots on the

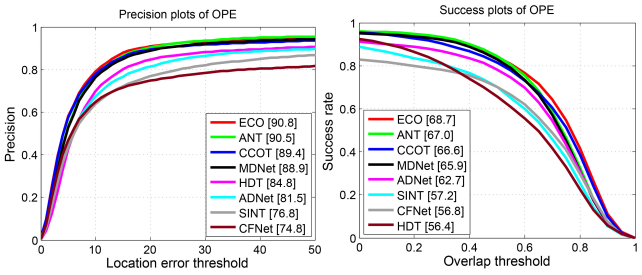

Figure 5: Precision and success plots on the OTB100 dataset.

OTB100 dataset in Figure 5. The plots show that ANT algorithm performs favorably against state-of-the-art trackers such as ECO, CCOT, and MDNet. Specifically, the ANT algorithm ranks top 2 in terms of either the precision plots or the success plots, and the performance gap between ANT and ECO is only 0.3% in terms of the precision plots.

In Figure 6, we present the tracking results on the UAV-Traffic dataset. This dataset is much more challenging than the OTB100 dataset as its best results are nearly 20% lower than that on OTB100 (precision: 77.0 vs. 90.8, success: 46.4 vs 68.7). On such a challenging dataset, the ANT algorithm achieves the best performance in terms of the precision plots, which is 4.5% higher than the runner-up MDNet. At the same time, the performance of ANT in terms of the success plots is very close to the best one. In addition, as the UAV-Traffic dataset has significantly different attributes compared to the training datasets, these tracking results demonstrate the good generalization ability of the proposed network.

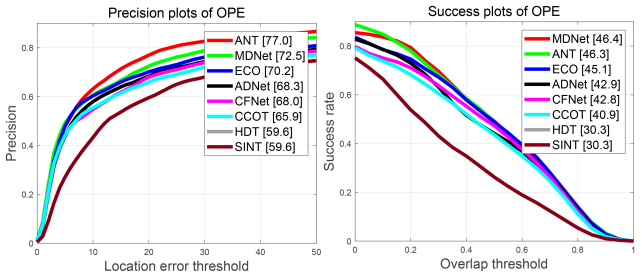

Figure 6: Precision and success plots on the UAV-Traffic dataset.

Qualitative evaluation. We present sample tracking results of the evaluated methods in Figure 7. For presentation clarity, only results of the top seven performing trackers are shown.

Camera motion. In the UAV-Traffic video *S0602*, the camera hovers at the crossroads, which leads to great appearance changes of the target (the blue bus). The bounding boxes show that the proposed ANT method locates the target more accurately than others during the hover, while ADNet falsely locates the road and MDNet fails to identify the target from background as shown in frame 291. In the *BlurBody* sequence, the target appearance is blurred due to cam-

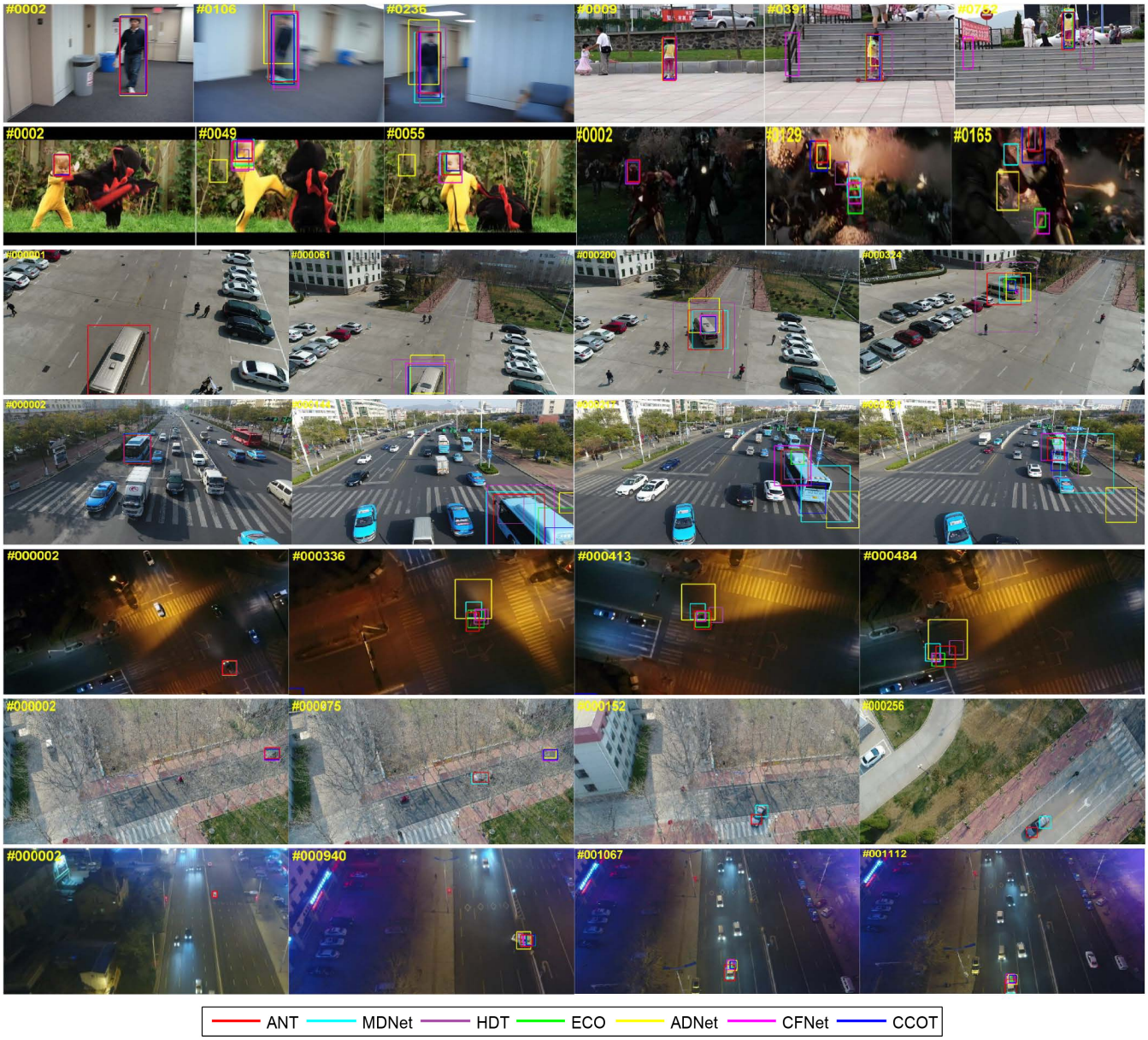

Figure 7: Bounding box comparison on several challenging videos (from top to bottom, left to right: *BlurBody, Girl2, DragonBaby, Ironman, S1701, S0602, S1301, S0103,* and *S1303*). The first four sequences are taken from OTB100 and the last five from the UAV-Traffic dataset.

era shaking. In this video, both ANT and CCOT methods are still able to precisely locate the target, while other trackers such as CFNet and MDNet locate the target with much background as shown in frame 236. The effectiveness of the ANT algorithm benefits from the camera motion branch as evaluated in Figure 4 and Table 1.

Occlusion. The target in the UAV-Traffic sequence *S0103* undergoes occlusions by trees. Only our ANT method and MDNet are able to track the target, while other trackers such as ECO and CCOT falsely locate on background out of the view as shown in frames 75 and 152. Similar performance can be observed in the OTB100 sequence *Girl2*, where the target girl suffers occlusions by a man walking with a bicycle. This can be attributed to the particular branch which learn discriminative representations in occlusion scenarios as evaluated in Figure 4 and Table 1.

Object motion. The target in the OTB100 sequence *DragonBaby* is hitting his opponent using the turn-around kick. As shown by the bounding boxes, both ANT and ECO methods are able to locate the target accurately in such a procedure, while other trackers lose the target, such as ADNet. With reference to the branch evaluations in Figure 4 and Ta-

ble 1, the object-motion branch in our attribute network is able to capture discriminative information in such a scene.

Scale. In the UAV-Traffic sequence *S1701*, the size of the target bus varies largely and the observation view changes from bird-view to side-view, which cause huge appearance variations. In such a challenging scene, the proposed ANT method locates the target more accurately than others such as MDNet and ECO, as shown in frames 200 and 324. Considering that MDNet and ECO use the same basal network as ours and referring to the branch evaluations in Table 1, the performance gain of the ANT algorithm can be mainly attributed to the attribute network, which learns specific representations under various challenges.

Illumination. The target in *Ironman* has drastic movements in a dark night with large illumination changes in the backgrounds. In such a poor lighting condition, the ANT algorithm accurately locates the target in most frames while other trackers drift far away as shown in frames 129 and 165. Similar performance can be observed in the UAV-Traffic sequences *S1301* and *S1303*. As evaluated in Table 1, the illumination branch contributes most in such situations.

5 Conclusion

To address the problem of lacking sufficient training data in CNN-based tracking methods, this paper proposes an attribute-based CNN framework with multiple branches. Each branch is separately trained to learn an attribute-specific representation. The representations of all branches are combined together to obtain a more discriminative representation to handle complicated tracking challenges. To the best of our knowledge, the proposed algorithm is the first to learn attribute-specific representations for visual tracking. Extensive experimental results on a large number of sequences demonstrate the effectiveness and the generalization ability of the proposed method.

6 Acknowledgments

This work was supported in part by National Natural Science Foundation of China: 61620106009, 61332016, U1636214, 61872112, 61472389 and 61672497, Key Research Program of Frontier Sciences, CAS: QYZDJ-SSW-SYS013, Shandong Provincial Natural Science Foundation, China: ZR2017MF001, the NSF CAREER Grant (No. 1149783), and gifts from Adobe and Nvidia.

References

Chatfield, K.; Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2014. Return of the devil in the details: Delving deep into convolutional nets. In *British Machine Vision Conference*, 1–8.

Chen, J.; Chen, X.; Ma, L.; Jie, Z.; and Chua, T.-S. 2018. Temporally grounding natural sentence in video. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 162–171.

Danelljan, M.; Robinson, A.; Khan, F. S.; and Felsberg, M. 2016. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *European Conference on Computer Vision*, 472–488.

Danelljan, M.; Bhat, G.; Khan, F. S.; and Felsberg, M. 2017. ECO: efficient convolution operators for tracking. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 6931–6939.

Du, D.; Qi, Y.; Yu, H.; Yang, Y.; Duan, K.; Li, G.; Zhang, W.; Huang, Q.; and Tian, Q. 2018. The unmanned aerial vehicle benchmark: Object detection and tracking. In *European Conference on Computer Vision*, 375–391.

Fan, H., and Ling, H. 2017. Sanet: Structure-aware network for visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2217–2224.

Fan, L.; Huang, W.; Gan, C.; Gong, B.; Ermon, S.; and Huang, J. 2018. End-to-end learning of motion representation for video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6016–6025.

Han, B.; Sim, J.; and Adam, H. 2017. Branchout: Regularization for online ensemble tracking with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 521–530.

Huang, W.; Fan, L.; Harandi, M.; Ma, L.; Liu, H.; Liu, W.; and Gan, C. 2018. Towards efficient action recognition: Principal backpropagation for training two-stream networks. *IEEE Transactions on Image Processing*.

Jiang, W.; Ma, L.; Chen, X.; Zhang, H.; and Liu, W. 2018a. Learning to guide decoding for image captioning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 6959–6966.

Jiang, Y.; Yang, Z.; Xu, Q.; Cao, X.; and Huang, Q. 2018b. When to learn what: Deep cognitive subspace clustering. In *ACM Multimedia Conference on Multimedia Conference*, 718–726.

Kristan, M.; Pflugfelder, R. P.; Leonardis, A.; Matas, J.; and et al. 2013. The visual object tracking vot2013 challenge results. In *International Conference on Computer Vision Workshops*, 98–111.

Kristan, M.; Pflugfelder, R. P.; Leonardis, A.; Matas, J.; and et al. 2014. The visual object tracking vot2014 challenge results. In *European Conference on Computer Vision Workshops*, 191–217.

Kristan, M.; Matas, J.; Leonardis, A.; Felsberg, M.; and et al. 2015. The visual object tracking vot2015 challenge results. In *International Conference on Computer Vision Workshops*, 564–586.

Laurens, v. d. M., and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9(Nov):2579–2605.

Liu, H.; Ji, R.; Wang, J.; and Shen, C. 2018. Ordinal constraint binary coding for approximate nearest neighbor search. *IEEE Transactions on Pattern Analysis Machine Intelligence*.

Luo, W.; Sun, P.; Zhong, F.; Liu, W.; Zhang, T.; and Wang, Y. 2018. End-to-end active object tracking via reinforcement

learning. In *Proceedings of the International Conference on Machine Learning*, 3292–3301.

Ma, L.; Lu, Z.; Shang, L.; and Li, H. 2015. Multimodal convolutional neural networks for matching image and sentence. In *IEEE International Conference on Computer Vision*, 2623–2631.

Mueller, M.; Smith, N.; and Ghanem, B. 2016. A benchmark and simulator for UAV tracking. In *European Conference on Computer Vision*, 445–461.

Nam, H., and Han, B. 2016. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4293–4302.

Park, E., and Berg, A. C. 2018. Meta-tracker: Fast and robust online adaptation for visual object trackers. In *European Conference on Computer Vision*, 587–604.

Qi, Y.; Zhang, S.; Qin, L.; Yao, H.; Huang, Q.; Lim, J.; and Yang, M. 2016. Hedged deep tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4303–4311.

Ren, W.; Liu, S.; Zhang, H.; Pan, J.; Cao, X.; and Yang, M.-H. 2016. Single image dehazing via multi-scale convolutional neural networks. In *European Conference on Computer Vision*, 154–169.

Ren, W.; Ma, L.; Zhang, J.; Pan, J.; Cao, X.; Liu, W.; and Yang, M.-H. 2018. Gated fusion network for single image dehazing. In *IEEE Conference on Computer Vision and Pattern Recognition*.

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

Smeulders, A. W. M.; Chu, D. M.; Cucchiara, R.; Calderara, S.; Dehghan, A.; and Shah, M. 2014. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36(7):1442–1468.

Song, Y.; Ma, C.; Gong, L.; Zhang, J.; Lau, R. W. H.; and Yang, M. 2017. CREST: convolutional residual learning for visual tracking. In *International Conference on Computer Vision*, 2574–2583.

Sun, X.; Huang, Z.; Yin, H.; and Shen, H. T. 2017. An integrated model for effective saliency prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 274–281.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S. E.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1–9.

Szegedy, C.; Ioffe, S.; Vanhoucke, V.; and Alemi, A. A. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 4278–4284.

Tao, R.; Gavves, E.; and Smeulders, A. W. M. 2016. Siamese instance search for tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1420–1429.

Valmadre, J.; Bertinetto, L.; Henriques, J. F.; Vedaldi, A.; and Torr, P. H. S. 2017. End-to-end representation learning for correlation filter based tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5000–5008.

Vedaldi, A., and Lenc, K. 2015. Matconvnet: Convolutional neural networks for MATLAB. In *ACM Multimedia*, 689–692.

Wang, B.; Ma, L.; Zhang, L.; and Liu, W. 2018. Reconstruction network for video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7622–7631.

Wu, Y.; Lim, J.; and Yang, M.-H. 2015. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37(9):1834–1848.

Yang, Z.; Xu, Q.; Cao, X.; and Huang, Q. 2018. From common to special: When multi-attribute learning meets personalized opinions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 515–522.

Yun, S.; Choi, J.; Yoo, Y.; Yun, K.; and Choi, J. Y. 2017. Action-decision networks for visual tracking with deep reinforcement learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1349–1358.

Zhang, S.; Yao, H.; Sun, X.; and Lu, X. 2013. Sparse coding based visual tracking: Review and experimental comparison. *Pattern Recognition* 46(7):1772–1788.

Zhang, S.; Qi, Y.; Jiang, F.; Lan, X.; Yuen, P. C.; and Zhou, H. 2018. Point-to-set distance metric learning on deep representations for visual tracking. *IEEE Transactions on Intelligent Transportation Systems* 19(1):187–198.