ATLANTIS PRESS

Advances in Intelligent Systems Research, volume 133

2nd International Conference on Artificial Intelligence and Industrial Engineering (AIIE2016)

RoI Pooling Based Fast Multi-Domain Convolutional Neural Networks for Visual Tracking

Yuanyuan Qin, Shiying He, Yong Zhao* and Yuanzhi Gong

School of Electronic and Computer Engineering, Shenzhen Graduate School of Peking University, Shenzhen, China

*Corresponding author

***Abstract*—This paper proposes a fast multi-domain convolutional neural networks method (Fast MDNet) for visual tracking. Fast MDNet builds on fast region-based convolutional neural networks (Fast R-CNN) and MDNet to efficiently track arbitrary objects using deep convolutional networks. We introduce a RoI pooling layer which shares full-image convolutional features, thus significantly speed up MDNet. Compared to previous works, Fast MDNet's online tracking rate is 15x faster than MDNet, and it performs favorably against the state-of-the-art methods on large benchmark datasets.**

Keywords-component; visual tracking; Fast MDNet; CNN; RoI

I. INTRODUCTION

Visual tracking aims to estimate the position of targets under various scenarios. It has many applications, such as intelligent video surveillance, human-computer interaction and vehicle navigation [1, 2]. Human readily find useful features to distinguish object from background. However, Visual tracking is a challenge problem in computer vision due to complicated interfering factors like occlusions, fast motions, drastic change of target, light conditions.

In recently years, the most popular approach for visual tracking is tracking-by-detection which treats the tracking problem as a classification task of finding the decision boundary that best separates the target from the background. This popularity is due to great progress in object detection, with many inspired ideas can learned from. Numerous classification algorithms such as support vector machines [3], boosting [4, 5] and random forests [6, 7] have been used in tracking-by-detection method and remarkable advances have been achieved in visual tracking research. A kernelized Structured SVM (Struck) [8] made use of hand-crafted features such as Haar-like, HOG [9], SIFT [10], and color histogram [11, 12] shown excellent performance in the past few years. However, these hand-crafted features are limited in the ability to deal with complex scenes. Recently, inspired by great success of convolutional neural networks (CNN) in object detection and recognition [13, 14, 15], more and more trackers based on CNN have been proposed. Many experiments on benchmark [16] shows that CNN based trackers [17, 18] achieve far better performance than trackers based on hand-crafted features.

Multi-Domain convolutional neural networks (MDNet) [17] proposed a novel CNN architecture to learn generic feature representations of different sequences for visual tracking. It regards each video as a separate domain, and has separate branches of domain-specific layers for binary classification at the end of the network. By this way, MDNet shows the state-of-the-art performance.

However, MDNet [17] has notable drawbacks. 1. Online tracking is slow. With a small number of layers, three convolutional layers (conv1-3), two fully connected layers (fc4-5) and the input size is only 107x107, MDNet can only run at around 1fps with an NVIDIA Tesla K20m GPU as in [17]. 2. Can't adapt to a larger network. When MDNet is tested on larger networks, the algorithm is less accurate and becomes slower significantly. 3. Fixed input size of 107x107. Regardless of the size of the candidate box, candidate boxes are normalized to 107x107, thus will be distorted.

Motivated by this fact, we propose a novel CNN architecture that fixes the speed disadvantages of MDNet, referred to as Fast Multi-Domain Network (Fast MDNet). MDNet is slow because it performs a ConvNet forward for every target candidate, without sharing convolutional features computation. We compute a convolutional feature map for the entire input image and introduce an ROI pooling layer which extracts features for every candidate into a fixed-size feature map then mapped to a feature vector by fully connected layers (FCs).

The main contributions of this work are summarized as follows:

- We introduce a RoI pooling layer which allows to compute convolutional features at only once, thus the proposed method waives nearly all time cost for convolutional features computation.
- Fast MDNet's online tracking is 15x faster and offline pre-training is 2x faster than MDNet.
- More flexible network structure. We don't need to worry about the increase in the number of network layers leading to seriously drag slow speed.
- Fast MDNet performs favorably against the state-of-the-art methods on large benchmark datasets.

II. RELATED WORK

A brief review of object detection and tracking methods closely related to this work are given as follow. For more comprehensive literature reviews, readers are directed to [1, 2, 19].

A. *Object Detection*

Recent successes of object detection are driven by various region proposal methods and region proposal based convulotional neural networks (R-CNNs) [20, 15, 21, 22]. R-CNN [20] is the first to show that a CNN can achieve a dramatically high object detection performance. It shows how to localize objects with a deep network and how to train a high-capacity model with only a small quantity of annotated detection data. But R-CNN is computationally expensive due to repeated calculation for each proposal. Later, Spatial pyramid pooling networks (SPPnets) [15] and Fast R-CNN [21] take advantage of RoI pooling layer to drastically reduce computational cost by sharing convolutions across proposals. The latest incarnation Faster R-CNN [22] achieves near real-time rates using very deep networks. This work is inspired by RoI pooling layer used in object detection, which can also make real-time CNN based online tracking possible.

B. *CNN Based Trackers*

CNN has shown outstanding representation power in a wide range of computer vision applications [13, 14, 15]. There are some tracking algorithms using the representations from CNN has been proposed so far [17, 18, 19]. MDNet [17] take use of multi-domain learning method which is widely used in natural language processing to pre-train deep CNN networks. This method trains data from multiple domains and incorporate domain information in learning procedure. Our pre-training and online-tracking are inspired from MDNet.

III. FAST-MDNET

A. *Network Architecture*

Fig. I illustrates the Fast MDNet architecture. A Fast MDNet takes as input an entire image and a set of RoI (region of interest) boxes. It receives a 600x1000 input, and has eight hidden layers including five convolutional layers (conv1-conv5), a RoI pooling layer and two fully connected layers (fc6-7). Additionally, the network has K branches for the last fully connected layers ($fc8^1$-$fc8^k$) corresponding to K domains (training sequences). At first, the network processes the whole image with five convolutional, two max pooling, several ReLU and dropout layers to produce a conv5 feature map. Then, for each positive or negative sampling box a region of interest (RoI) pooling layer extracts a fixed-length feature vector from the conv5 feature map. Each feature vector is fed into the next two fully connected layers. Fc6 and Fc7 have 4096 output units and are combined with ReLUs and dropouts. The layers are identical to the corresponding parts of ZF network [23] except the RoI pooling layer and the last K branches layer. For the K branch layers, each layer is specified for a training sequence and connected to a binary soft-max loss layer. The two classes are corresponding to target and background.

B. *RoI Pooling Layer*

As in [21], RoI is a rectangle defined by four parameters (x,y,w,h) which specifies its top-left corner (x,y) and its width and height (w,h). The RoI pooling layer's output is conv features map of a fixed dimension H x W (e.g., 6 × 6). RoI max pooling works by dividing the RoI rectangle (h x w) into H x W blocks. Each block is a sub-rectangle of approximate size h/H × w/W. By max-pooling the value of each block, we get the corresponding output grid cell.

C. *Domain-specific Layer*

Domain-specific layer is also named as K branches layer. It is implemented on caffe [24] with the InnerProduct layer. The output unit of this layer is set to K x 2, and each of the two outputs represents a training sequence. In the process of back propagation, the diff input size of Domain-specific layer is K x 2. By transferring the diff from the soft-max layer (1 x 2) and set the other (K - 1) x 2 diff input to 0, the back propagation of Domain-specific layer can be easily implemented.

D. *Offline Pre-training*

Parameters of the pre-trained ImageNet [13] network are used to initialize our network. Fast MDNet is trained by the Stochastic Gradient Descent (SGD) method. Each SGD mini-batch is constructed from N = 4 images which are chosen uniformly at random. We use mini-batches of size R=128, sampling 32 RoIs from each image. As in Fast R-CNN [21], 25% of RoIs which have intersection over union (IoU) overlap with a groundtruth bounding box of at least 0.7, are taken as targets. These RoIs are treated as foreground target class, i.e. label=1. The remaining RoIs which have intersection over union (IoU) overlap with a groundtruth bounding box under 0.3, are treated as background and are labeled with 0.

Unlike MDNet, in which 50 positive small images and 200 negative small images are cropped from each frame, and all these images are resized to 107 x 107 as input data thus leading to repeated calculation 250 times in order to calculate the convolution feature map, Fast MDNet just need to calculate the convolution feature map of the entire image at only once. RoI pooling layer generates the corresponding output grid cell according to RoI rectangles and the entire image's convolutional feature map. Even though the proposed network is much larger than MDNet, the training time is faster than the later.

The iteration method is similar to MDNet [17]. In the k^{th} iteration, the input images are chosen from the $(k \bmod K)^{th}$ sequence, and only a single branch $fc8^{(k \bmod K)}$ is enabled to update weights and bias. Conv1-5 and fc6-7 can learn the common properties that are desirable for target representations in all domains, such as robustness to illumination changes, motion blur, etc. The learning rate is initialized as 0.001, and is reduced to 0.0001 after 10k iterations. A momentum of 0.9 and parameter decay of 0.0005 (on weights and biases) are used

E. *Online Tracking*

After completing pre-training of Fast MDNet, the K branches layer is replaced with a single branch (fc8) for a test sequence. The details of online tracking and online fine-tuning can refer to MDNet [17]. Here, we focus on main procedures and how we can speed up online tracking.

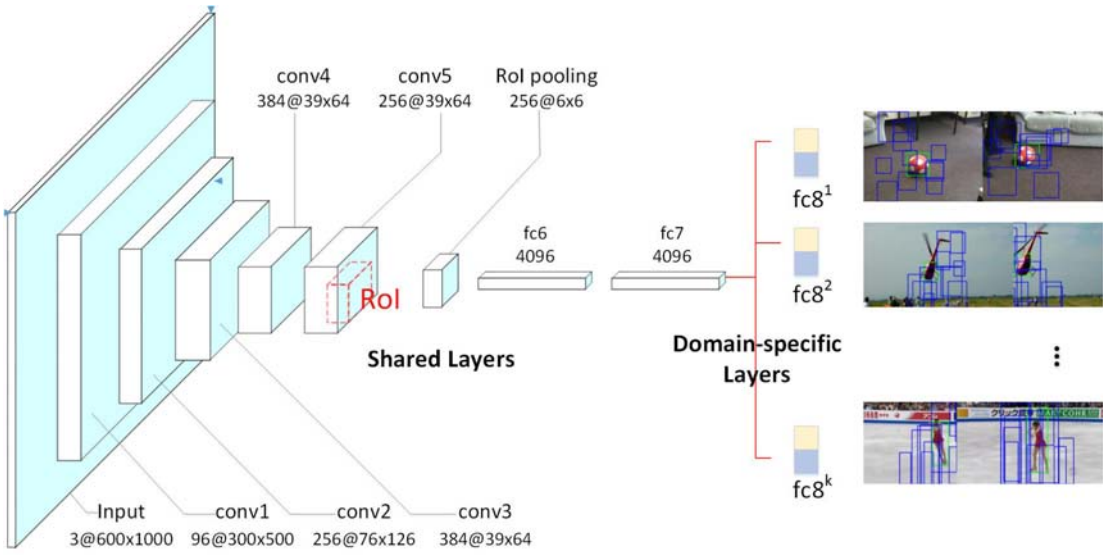

FIGURE I. ARCHITECTURE OF FAST MULTI-DOMAIN NETWORK, WHICH CONSISTS OF SHARED LAYERS, ROI POOLING LAYER AND K BRANCHES OF DOMAIN-SPECIFIC LAYERS.

To estimate the target position, Fast MDNet samples N (256) candidates around the previous target state. The candidates are generated from a Gaussian distribution as in [17]. Then these candidates or RoIs and the entire image are sent into our network. By performing forward process of the entire image at only once, we get the N candidates' convolutional feature map with the help of RoI pooling layer. Then these convolutional features are converted to a fixed size of H x W (e.g., 6 × 6), and are sent to the next fully connected layers. At last, these candidates are evaluated using the network, and their positive scores $f^+(x^i)$ and negative scores $f^-(x^i)$ are given by the output of soft-max layer. Like MDNet, the target of current frame is that with the maximum positive score as:

$$x^* = \arg\max_{x^i} f^+(x^i) \tag{1}$$

Unlike Fast MDNet, MDNet cuts out N (256) small images from the entire image, and resizes them to a fixed size of 107 x 107 as input data. Then MDNet repeatedly compute convolutional feature map N times for each candidates, thus leading to extremely low efficiency. Our algorithm uses RoI pooling to solve this problem.

To fine-tune Fast MDNet, 50 positive boxes and 200 negative boxes are sampled around the previous target state for each frame. But we update weights once every 10 frames instead of each frame. Only weights of fc6-8 are updated whereas the ones in convolutional layers conv1-5 are fixed throughout tracking. This is because conv1-5 share the common properties of different domains, and this strategy can prevent over fitting. Other strategies like Hard Minibatch Mining and Bounding Box Regression can refer to [17].

IV. EXPERIMENT

In this section, experimental evaluations on the proposed Fast MDNet are presented. We first discuss the experimental setup, dataset, and evaluation metrics, and then present two sets of experiments: one compares the rate of offline pre-training, online tracking and online fine-tuning to MDNet and the other compares the success plot and precision plot to several state-of-the-art trackers.

A. Dataset, Evaluation Protocol and Metrics

Our network is pre-trained using 58 sequences from Object Tracking Benchmark VOT2013, VOT2014, VOT2015 [25], which do not include the common sequences with the OTB [16] dataset. As suggested in [26], we evaluate the tracking algorithms using one-pass evaluation (OPE). Four recent state-of-the-art trackers including MEEM [27], Struck [8], TLD [29], and SCM [28], are compared with our algorithm. The online tracking is evaluated on the CVPR2013 OOTB [26] benchmark datasets.

B. Online-tracking and Fine-tuning Rate

Table I compares offline pre-training rate (s/iteration), online tracking rate (ms/image) and online fine-tuning rate (ms/iteration) between Fast MDNet and MDNet. Fast MDNet processes images 15.3x faster than MDNet (55.66 ms vs. 853.69 ms). Offline pre-training time is reduced by 2.4x, from 0.60 s/iteration to 0.25 s/iteration. Online fine-tuning time is reduced by 1.6x, from 58.33 ms/iteration to 36.27 ms/iteration. Our algorithm is implemented with ZF (7 layers) model [23], and the input size is 600x1000. MDNet is a VGG-M (3 layers) network which has the input size of 107x107. It runs on a desktop computer with Intel(R) Core(TM) i7-4790K (4.00 GHz), 32 GB memory and an NVIDIA GeForce GTX TITAN X GPU.

C. *OPE Results*

Figure II shows the OPE results on 50 image sequences of CVPR2013 OOTB [26]. The precision and success plots are based on center location and bounding box overlap ratio, respectively. We note that the proposed Fast MDNet performs better than these state-of-the-art methods except MDNet. Fast MDNet performs comparably against MDNet because it uses the similar way to deal with the tracking problem, but it has remarkable advantage in speed.

V. CONCLUSION

In this paper, we propose a novel RoI pooling based multi-domain convolutional neural network, which is referred to as Fast MDNet. With the aid of RoI pooling layer, we can easily speedup the algorithm and achieve nearly real-time tracking. Same with MDNet, our algorithm learns domain-independent representations from offline pre-training, and aquires domain-specific features from online fine-tuning. Experiments shows that Fast MDNet performs favorably against the state-of-the-art methods on large benchmark datasets.

TABLE I. RUNTIME COMPARISON

Terms	**Algorithms**	
	Fast MDNet	***MDNet***
Offline train rate (s/iteration)	0.25	0.60
Offline train speedup	2.4x	1x
Online tracking rate (ms/image)	55.66	853.69
Online tracking speedup	**15.3x**	1x
Online fine-tuning rate (ms/iteration)	36.27	58.33
Online fine-tuning speedup	1.6x	1x

a. Sample of a Table footnote. (Table footnote)

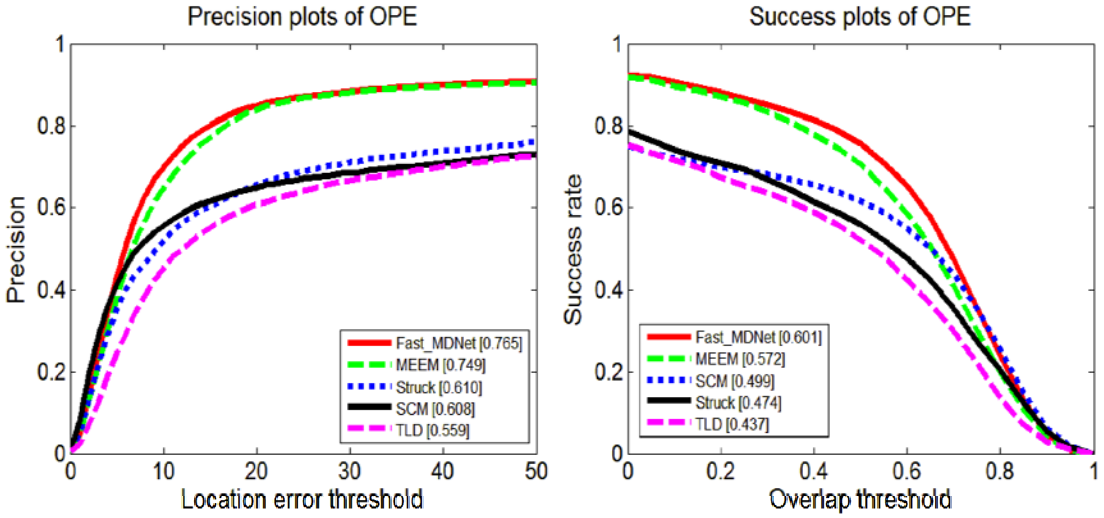

FIGURE II. PRECISION AND SUCCESS PLOTS ON CVPR2013 OOTB [26]

REFERENCES

[1] K. Cannons. A review of visual tracking. Dept. Comput. Sci.Eng., York Univ., Toronto, Canada, Tech. Rep. CSE-2008-07,2008.

[2] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara,A. Dehghan, and M. Shah. Visual tracking: An experimental survey. PAMI, 36(7):1442–1468, 2014.

[3] S. Avidan. Support vector tracking. PAMI, 26(8):1064–1072,2004.

[4] S. Avidan. Ensemble tracking. PAMI, 29(2):61–271, 2007.

[5] L. Wen, Z. Cai, Z. Lei, and S. Li. Online spatio-temporal structural context learning for visual tracking. In ECCV, 2012.

[6] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof. On-line random forests. In ICCV, 2009.

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

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

[9] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In CVPR, 2005.

[10] D. G. Lowe. Object recognition from local scale-invariant features. In ICCV, 1999.

[11] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragmentsbased tracking using the integral histogram. In CVPR, 2006.

[12] G. Tian, R. Hu, Z. Wang, and Y. Fu. Improved object tracking algorithm based on new HSV color probability model. In ISNN, 2009.

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

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

[15] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In ECCV, 2014.

[16] Y. Wu, J. Lim, and M. Yang. Object tracking benchmark. IEEE Transactions on Pattern Analysis and Machine Intelligence, 37(9):1834–1848, Sept 2015.

[17] H. Nam and B. Han. Learning Multi-Domain Convolutional Neural Networks for Visual Tracking. In CVPR, 2016.

[18] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, J. Lim and M. Yang. Hedged Deep Tracking. In CVPR, 2016.

[19] X. Li, W. Hu, C. Shen, Z. Zhang, A. R. Dick, and A. van den Hengel. A survey of appearance models in visual object tracking. ACM TIST, 4(4):58, 2013.

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

[21] R. Girshick. Fast R-CNN. arXiv:1504.08083, 2015.

[22] S. Ren, K. He, R. Girshick and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. Advances in neural information processing systems. 2015.

[23] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional neural networks. In ECCV, 2014.

[24] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. arXiv:1408.5093, 2014.

[25] M. Kristan, J. Matas and A. Leonardis. The visual object tracking vot2015 challenge results. Proceedings of the IEEE International Conference on Computer Vision Workshops. 2015.

[26] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In CVPR, 2013.

[27] J. Zhang, S. Ma, and S. Sclaroff. MEEM: Robust tracking via multiple experts using entropy minimization. In ECCV, 2014.

[28] W. Zhong, H. Lu, and M.-H. Yang. Robust object tracking via sparsity-based collaborative model. In CVPR, 2012.

[29] Z. Kalal, J. Matas, and K. Mikolajczyk. P-N learning: Bootstrapping binary classifiers by structural constraints. In CVPR, 2010.