GATE CONNECTED CONVOLUTIONAL NEURAL NETWORK FOR OBJECT TRACKING

T.Kokul[⋆†] *C.Fookes*[⋆] *S.Sridharan*[⋆] *A.Ramanan*[‡] *U.A.J.Pinidiyaarachchi*[◇]

[⋆]Image and Video Lab, SAIVT Program, Queensland University of Technology, Australia
[‡]Dept. of Computer Science, University of Jaffna, Sri Lanka
{[†]PGIS, [◇]Dept. of Statistics and Computer Science}, University of Peradeniya, Sri Lanka

ABSTRACT

Convolutional neural networks (CNNs) have been employed in visual tracking due to their rich levels of feature representation. While the learning capability of a CNN increases with its depth, unfortunately spatial information is diluted in deeper layers which hinders its important ability to localise targets. To successfully manage this trade-off, we propose a novel residual network based gating CNN architecture for object tracking. Our deep model connects the front and bottom convolutional features with a gate layer. This new network learns discriminative features while reducing the spatial information lost. This architecture is pre-trained to learn generic tracking characteristics. In online tracking, an efficient domain adaptation mechanism is used to accurately learn the target appearance with limited samples. Extensive evaluation performed on a publicly available benchmark dataset demonstrates our proposed tracker outperforms state-of-the-art approaches.

Index Terms— object tracking, CNN, domain adaptation

1. INTRODUCTION

Visual object tracking is a fundamental task in computer vision and many other applications [1]. Object tracking has attracted considerable research in the past and much progress has been made recently. However, it is still far from reaching the accuracy of the tracking ability of humans, due to appearance changes, pose variations, occlusions, illumination variations and background clutter.

Many of the appearance-based tracking frameworks [2, 3] depend on low-level hand-crafted features. These features fail to capture semantic information of targets and are not robust to significant appearance changes. Therefore sophisticated learning methods are needed to improve the feature representation in tracking frameworks.

Through a rapid increase in computational power, Deep Neural Networks (DNNs) can directly learn features from raw data without resorting to hand-crafting. DNNs, especially convolutional neural networks (CNNs), have demonstrated state-of-the-art performance in several vision tasks [4, 5, 6]. However, few visual tracking frameworks that have been proposed make use of DNNs. The main reason for this is as DNNs employ an extensive set of parameters, they require enormous amounts of training data, which is not yet available for visual tracking.

Several early deep learning based tracking approaches [7, 8] manage the training data deficiency by transferring offline learned DNN features to online tracking. These approaches obtain generic image features from object classification models. However, they do not learn similar local structure and inner geometric layout information among the targets, which are more important to discriminate a target from distractors.

Recent deep tracking approaches [9, 10, 11] analyse internal properties of CNN features from the perspective of visual tracking and propose tracking algorithms based on that. Even though these approaches showed state-of-the-art-performance, they suffer from over-fitting as they are fine-tuned online with only limited samples.

A robust visual tracking approach should learn different representations of targets and adapt to their appearance changes. In addition, deep learning based trackers should successfully manage the trade-off between over-fitting and the learning capability of the network with limited samples. Based on these requirements, we propose a novel deep tracking approach (called GNET) to learn generic tracking features. Our main contributions are:

- A residual network based gating CNN architecture, which learns generic tracking characteristics by effectively capturing features from front and end convolutional layers.
- An online domain adaptation mechanism, which is used to train the model with limited samples and reduce over-fitting.

The rest of this paper is organised as follows: Section 2 analyses state-of-the-art tracking trends and deep domain adaptation techniques. Section 3 describes the pros and cons of deep learning based tracking and an overview of our approach. Section 4 demonstrates our proposed approach in detail. The experimental setup and testing results are described in Section 5. Finally the paper is concluded in Section 6.

2. RELATED WORK

Tracking is the process of continuously detecting a target in a video sequence, from its first appearance to last. In this paper, we aim to develop a robust appearance based tracking framework.

Researchers have started to use DNNs in visual tracking because of their strong feature learning capabilities. Initial approaches of deep learning based trackers treat the DNN as a black box feature extractor and use its generic feature representation for tracking. Wang and Yeung [7] used a stacked denoising auto-encoder, which is trained offline on small image datasets and was then used for online tracking. Hong *et al.*, [8] used the features of Region-based CNN (R-CNN) to construct a discriminative model with an online learning SVM. Hanxi *et al.*, [12] used a two-layer CNN for tracking without any pre-training procedure. These approaches treat the CNN as a black box and rely on features obtained from the last layer.

Recently, researchers [9, 10, 11] use internal properties of deep features to construct a strong appearance model for tracking. These models rely on the concept that different layers of a CNN extract a different perspective of information, which can be used to better discriminate a target from background and other objects. Lijun *et al.*, [9] used features from the top and bottom layers of a CNN with a switch mechanism to accurately learn class specific and target specific features. Chao *et al.* [11] used hierarchies of CNN features to learn correlation filters and use their response map to locate a target. Yuankai *et al.* [10] proposed a similar approach, in which they train a number of week trackers, each learning features from corresponding layers of a CNN with the response of correlation filters. Finally, all trackers form a stronger one with a hedging algorithm. Even though these approaches revealed state-of-the-art performance, they learn their model from a generic image representation and they do not feed any prior knowledge to the network about generic tracking characteristics such as how a target's appearance changes through illumination changes, scale variations and background clutter.

Some recent approaches effectively fine-tune or transfer pre-trained features online and as a result showed better performance. Lijun *et al.*, [13] transfer offline trained CNN features to online tracking by treating each channel of CNN features as an individual base learner. Nam and Han [14] trained a small CNN architecture with a set of annotated video sequences. They rely on the concept that each sequence represents a separate domain and their network learns the tracking features with a domain specific network architecture.

Recently, deep domain adaptation techniques [15, 16] have become popular since data deficiency is a common issue in many applications. These approaches transfer the learned knowledge from one domain (source) to another (target) with few samples. They minimize the distance between source distribution and target distribution and learn the target network accordingly. These techniques can be applied to deep learning based object tracking since they are affected by over-fitting.

3. OVERVIEW

Our aim is to track a single target in a video sequence while only an initial frame information is provided. It is truly challenging to propose a deep learning based tracker since there are several factors influencing the network architecture design in the perspective of tracking.

A deeper network requires a large amount of training data to avoid over-fitting. Data deficiency is the biggest issue in tracking. Deeper networks learn much richer level discriminative features. On the other hand, spatial information is diluted with the depth of the network due to pooling layers of CNNs, which is important for target localisation. In addition, recent studies [9, 11] show that early convolutional layer features capture fine grain details and therefore are useful for accurate localisation of the target while the last layer features are more effective for inter-class classification.

By considering all these factors, we propose a new network architecture for visual tracking, which effectively learns tracking features by fusing the front and bottom layer features of a network to accurately learn target appearance and provide its localisation. In addition, we use an efficient online domain adaptation technique to learn the target domain with limited samples.

4. PROPOSED APPROACH

4.1. Network Architecture

As demonstrated in Fig. 1, our proposed network has five convolutional layers and three fully connected layers. In addition to these sequential layers, it has an additional convolutional layer, called a gate layer, which bridges the front and bottom convolutional layers. Sequential convolutional layers are obtained from VGG-M [17] architecture. Different to VGG-M, convolutional layers are followed by fully connected layers without any pooling operation, which is removed to reduce further spatial dilution. Fully connected layers are interleaved with an activation layer and dropout. We keep the same parameters of VGG-M for convolutional layers (including padding size) and modify output neurons of $fc6$ and $fc7$ to 512. Our network takes input with size of 96×96 and produces output as binary classes (target and background).

The major aim of the gate layer is to reduce the spatial information loss with network depth. As shown in Fig. 1, the gate layer receives input as $conv2$ features before max pooling (shown as in brown colour) operation and it's output is concatenated with $conv5$ features before ReLU activation through an element-wise addition. It has kernel size of 3×3 with stride of 2.

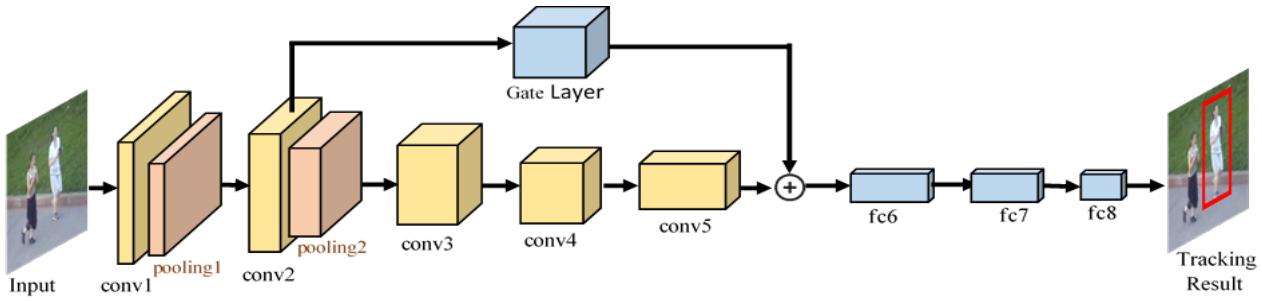

Fig. 1. Proposed network architecture (GNET). Convolutional layers (yellow) are obtained from VGG-M network and remaining layers (blue) are randomly initialized.

We have tested several gating options between front and bottom convolutional layers. While the gate layer maps features from a very high dimension to a lower one, it looses richness in features and the performance becomes lower. In addition, we have tested all possible gating options between convolutional layers (such as $conv2$ to $conv4$), but they don't give much improvement. We also experimented with different feature concatenation options, such as before and after activation of convolution. Similar to the ResNet architecture [18], when we concatenate features after activation of convolution, the performance becomes less accurate. The proposed architecture balances the trade-off between network depth and spatial information loss.

4.2. Pretraining

We trained the proposed network architecture with a set of annotated video sequences. The objective of pretraining is twofold: to train the gate layer to learn the feature mapping between convolutional layers; and to train the whole architecture to learn generic tracking characteristics.

We use Stochastic Gradient Descent (SGD) to train our model. Since our aim is to learn generic tracking characteristics instead of predicting class labels, we assign the same class label to different target objects of each sequence. During the training iteration, the whole network is trained with samples collected from targets and corresponding backgrounds. The pre-training process is stopped when the model converged to a fixed lower training error.

4.3. Online Tracking

Since only the initial frame is given in online tracking, fine-tuning the pre-trained network with very few limited samples makes the network very prone to over-fitting. We have used an online deep domain adaptation technique to learn the target appearance with limited samples.

We re-initialized the last two fully connected layers in the pre-trained model. Since the first three convolutional layers ($conv1$ to $conv3$) are more generic, they are frozen in online learning. The remaining layers are fine-tuned with a lower learning rate, except $fc7$ and $fc8$. They are learnt with the Maximum Mean Discrepancy (MMD) [19] domain adaptation technique.

MMD is a technique to measure the distance metric between two samples. Let F^p and F^t be the feature representation at layer l of the pre-trained model and target model, respectively, then the MMD at layer l can be measured as,

$$MMD_l(F^p, F^t) = \left\| \sum_{\substack{i=1 \\ f_i^p \in F^p}}^{N^p} \frac{\phi(f_i^p)}{N^p} - \sum_{\substack{j=1 \\ f_j^t \in F^t}}^{N^t} \frac{\phi(f_j^t)}{N^t} \right\|, \quad (1)$$

where ϕ denotes reproducing features to a kernel space. N^p and N^t are the number of samples in pre-trained model and target model, respectively. We mapped features to RBF kernel and measure the MMD. We used MMD to learn the target domain by minimizing the distance between pre-trained samples and target samples. Our network learns the target model by minimizing the loss function,

$$L = L_t + \lambda(MMD^2_{fc7} + MMD^2_{fc8}), \quad (2)$$

where L_t is logistic classification loss on limited target data, which can be obtained from the first frame of the sequence. λ is a parameter, which is set experimentally by using a validation sequence. We have computed the classification loss and MMD for each mini batch and trained the target model by minimizing Eq. 2. In addition, inspired by [14], we have used the hard mini-batch mining approach to improve the network performance by learning distracting negative samples.

To locate the target in a new frame, we collect a fixed number of samples with different scales and translations around the last known position of target and obtain their corresponding scores as the network response. The target location is obtained as the mean of the first five maximum scoring samples. We update the network at every fixed interval in between frames. Only fully connected layers are

Table 1. Average precision scores for the top 10 trackers on different attributes: Illumination variation (IV), out-of-plane rotation (OPR), scale variation (SV), occlusion (OCC), deformation (DEF), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-view (OV), background cluttered (BC) and low resolution (LR). The best and the second scores are shown in red and blue colours, respectively.

	VTD	TLD	SCM	Struck	CNN-SVM	STCT	FCNT	HDT	CF2	GNET
IV	0.557	0.537	0.594	0.558	0.780	0.786	0.83	0.845	0.844	0.885
OPR	0.620	0.596	0.618	0.597	0.832	0.828	0.828	0.871	0.869	0.920
SV	0.597	0.606	0.672	0.639	0.827	0.807	0.830	0.866	0.880	0.948
OCC	0.545	0.563	0.640	0.564	0.770	0.816	0.797	0.874	0.877	0.924
DEF	0.505	0.512	0.586	0.521	0.858	0.838	0.917	0.884	0.881	0.896
MB	0.375	0.518	0.339	0.551	0.745	0.683	0.789	0.840	0.844	0.873
FM	0.352	0.551	0.333	0.604	0.723	0.702	0.767	0.782	0.790	0.876
IPR	0.599	0.584	0.597	0.617	0.836	0.782	0.811	0.869	0.868	0.898
OV	0.462	0.576	0.429	0.539	0.687	0.691	0.741	0.679	0.695	0.874
BC	0.571	0.428	0.578	0.585	0.789	0.837	0.799	0.871	0.885	0.897
LR	0.168	0.349	0.305	0.545	0.705	0.682	0.765	0.847	0.897	0.899
Overall	0.576	0.608	0.649	0.656	0.852	0.852	0.856	0.889	0.891	0.922

updated with MMD procedure and remaining layers are fixed throughout the tracking.

5. EXPERIMENTS

5.1. Implementation details

The proposed approach is implemented in MATLAB with MatConvNat [20]. It runs 1 fps on four cores of 2.66 Intel Xeon with NVIDIA Tesla K40 GPU. We have used sequences from VOT challenge [21] (excluding common sequences with test dataset) to train the network. Since our pre-trained model and online learnt model have mostly common features, the parameter λ in Eq. 2 is set to 0.5.

5.2. Evaluation on OTB dataset

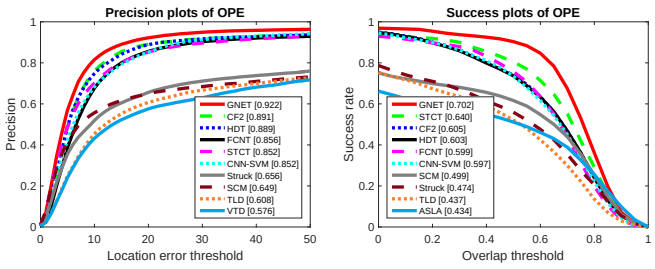

Fig. 2. The precision and success plots for the top 10 trackers. The numbers in legend show the representative precision at error threshold of 20 pixels, and the area-under-curve scores for success plots.

Object Tracking Benchmark (OTB) [22] is a popular benchmark dataset, which has 50 different sequences, that reflect different challenges of tracking. OTB uses center location error and overlap ratio to measure the tracking performance. We use one-pass evaluation (OPE) scheme to compare results with other trackers. We evaluate our proposed tracker with 29 state-of-the-art trackers[1]. In addition, we compare our approach with five recently proposed deep learning based trackers STCT [13], HDT [10], FCNT [9], CF2 [11] and CNN-SVM [8].

Fig. 2 shows the overall precision and success plots on 50 sequences of OTB dataset. Our proposed tracker whose performance is shown in red colour outperforms other trackers. Table 1 shows that proposed GNET tracker outperforms state-of-the-art trackers in almost all the challenges. Since the gate connection combines generic features with discriminative features, the proposed GNET tracker showed a huge improvement in scale variation and occlusion sequences.

6. CONCLUSION

In this paper, we presented a novel CNN architecture for visual tracking. Our framework bridges the front and bottom convolutional layer features by a gate connection. Different from other deep learning based trackers, our approach learns generic tracking characteristics by utilising features obtained from different convolutional layers. Our tracker combines a generic feature representation and a discriminative representation of targets while preventing spatial information loss, which is effective for separating the target from distractors especially in occlusion and scale variation scenarios. We also use an online domain adaptation technique for learning the target model while reducing over-fitting. Our approach is evaluated on a publicly available dataset and demonstrated improved performance than other state-of-the-art trackers.

[1] http://cvlab.hanyang.ac.kr/tracker_benchmark/benchmark_v10.html

7. REFERENCES

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM computing surveys (CSUR)*, vol. 38, no. 4, pp. 13–22, 2006.

[2] X. Jia, H. Lu, and MH. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 1822–1829.

[3] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, "Online Multiperson Tacking-by-Detection from a Single, Uncalibrated Camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 9, pp. 1820–1833, 2011.

[4] A. Krizhevsky, I. Sutskever, and GE. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proceedings of the Advances in neural information processing systems*, 2012, pp. 1097–1105.

[5] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "DeCAF: A deep convolutional activation feature for generic visual recognition," in *Proceedings of International conference on Machine Learning (ICML)*, 2014, vol. 2, pp. 988–996.

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 580–587.

[7] N. Wang and D. Y. Yeung, "Learning a deep compact image representation for visual tracking," in *Proceedings of the Advances in Neural Information Processing Systems*, 2013, pp. 809–817.

[8] S. Hong, T. You, S. Kwak, and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network," *arXiv preprint arXiv:1502.06796*, 2015.

[9] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Visual Tracking with Fully Convolutional Networks," in *Proceedings of International Conference on Computer Vision (ICCV)*, 2015, pp. 3119–3127.

[10] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, and J. L. MH. Yang, "Hedged deep tracking," in *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4303–4311.

[11] C. Ma, J. A. Huang, X. Yang, and MH. Yang, "Hierarchical Convolutional Features for Visual Tracking," in *Proceedings of International Conference on Computer Vision (ICCV)*, 2015, pp. 3074–3082.

[12] H. Li, Y. Li, and F. Porikli, "DeepTrack: Learning Discriminative Feature Representations by Convolutional Neural Networks for Visual Tracking," in *Proceedings of British Machine Vision Conference (BMVC)*, 2014, vol. 1, pp. 3–11.

[13] L. Wang, W. Ouyang, X. Wang, and H. Lu, "STCT: Sequentially Training Convolutional Networks for Visual Tracking," in *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1373–1381.

[14] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4293–4302.

[15] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," *arXiv preprint arXiv:1412.3474*, 2014.

[16] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *Proceedings of International conference on Machine Learning (ICML)*, 2015, pp. 97–105.

[17] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," *arXiv preprint arXiv:1405.3531*, 2014.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[19] A. Gretton, K. M. Borgwardt, M. Rasch, B. Schlkopf, and A. J. Smola, "A kernel method for the two-sample-problem," *Advances in neural information processing systems*, vol. 19, pp. 513, 2007.

[20] A. Vedaldi and K. Lenc, "MatConvNet: Convolutional Neural Networks for MATLAB," in *Proceedings of the 23rd ACM international conference on Multimedia*, 2015, pp. 689–692.

[21] "Visual object tracking (VOT) challenge," `http://www.votchallenge.net`.

[22] Y. Wu, J. Lim, and MH. Yang, "Online object tracking: A benchmark," in *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 2411–2418.