Recurrent Siamese Network for Object Tracking

Jeet Kanjani

Paper ID 3160

Abstract. The problem of arbitrary object tracking has traditionally been tackled by learning a model of the object's appearance and motion during the online phase. We quip a basic tracking algorithm using a Siamese network composed of fully convolutional network which incorporates a recurrent layer at the end. The convolutional recurrent model learns a motion model and leveraging the power of Siamese network helps in achieving robust tracking. The network is trained end to end on ILVRSC17 for similarity learning in videos. We show competitive results on well known benchmarks. . . .

Keywords: object-tracking, Siamese-network, deep-learning

1 Introduction

We tackle the problem of arbitrary object tracking in videos which receives a single supervision in the initial frame for the object's location. Making the tracking robust to all types of objects makes it impossible to train a specific detector.

The problem of arbitrary Object Tracking has traditionally been tackled by learning a model of the Object's appearance online which compromises on the time complexity of the algorithm. The training is done online with examples extracted from the video in papers like MDNET[1] which used online Stochastic Gradient Decsent. Despite having low frame rates these models can learn only a simple model due to the lack of supervised data available at that time.

With the advent of huge datasets like the ever-growing imagenet, the descriminative power of CNN models is extensively being used to extract useful embeddings. These features were incorporated in a Siamese network to show state of the art result in multiple benchmarks in object tracking by Bertinetto[2]. The simplicity of the architecture enables it to operate at higher frame rates which makes it a strong area for doing further research in the domain.

Despite showing good results the tracker by Bertinetto[1] struggles with handling occlusion and confusion due to the lack of motion model incorporated in the model. The problem of re detection of the object after it is occluded is tackled in this paper by incorporating a recurrent architecture. The network for handling re detection is inspired by Niall[3]. This is the key contribution of the paper.

We train a fully convolutional network incorporating LSTM layers at the end in the offline phase on ILVRSC17 train dataset. Our approach focuses on finding the exemplar in the search region using the similarity parameters learned by the network. The network compromises on the frame rate but still achieves close to real time performance.

2 Deep Similarity for Tracking

The proposed network architecture computes the similarity between the helps in proposing propose to learn a function f(z,x) that computes the similarity between the exemplar and the search region.

2.1 Input

The inputs consists of an exemplar image Z and a search image X of dimension 127x127 and 255x255 respectively.

2.2 Convolutional Network

In general, CNN processes the input using series of layers composed of convolution, pooling, non linear activation function steps. The parameters are identically applied to the exemplar and search images during training and testing. For an input x, we get a feature vector f = C(x). The fully convolutional network is very similar to the architecture proposed by Krizhevsky et al[4]. During inference the temporal framess$= s^1 s^2 .... s^T$ of length T where s^t is the image at time t is passes through the these layers resulting in $f^t = C(s^t)$, where f^t is the vectorized representation of the CNN's final layer activation maps. The vector f^t is passed forward to the recurrent layer, where it is projected into a low-dimensional feature-space and combined with the information from previous time steps. Dropout[5] is used between CNN layers and recurrent layers to reduce over-fitting.

2.3 RNN layers

The RNN layers incorporated are useful for learning a motion model from the temporal information in the video sequence. We can incorporate recurrent connections between the CNN and temporal pooling as follows:

$$o^t = W_i f^t + W_s r^t - 1$$
$$r^t = Tanh(o^t)$$

The output $o^t \in \mathbb{R}^{e-1}$at each time step is a linear combination of the vectors, $f^t \in \mathbb{R}^{Nx1}$ containing information on the current input image, and $r^{t-1} \in \mathbb{R}^{ex1}$, containing information on RNN's state at the previous time step.

2.4 Temporal Pooling

The output from the each of the LSTM unit pooled to produce a single temporal component. We use mean-pooling over the temporal dimension to produce a single feature vector v representing the object's appearance averaged over the whole input sequence as follows:

$$v_s = \frac{1}{T} \sum_{t=1}^{T} o^t$$

2.5 Siamese Network

The network is fully-convolutional with respect to the candidate image x similar to Bertinetto[1]. The advantage of a fully convolutional network is that, instead of a candidate image of the same size, we can provide as input to the network a search of larger size and it will compute the similarity at all translated sub-windows on a dense grid in a single evaluation.Deep Siamese conv-nets have previously been applied to tasks such as face verification [6][7], keypoint descriptor learning [8] and one-shot character recognition[9].

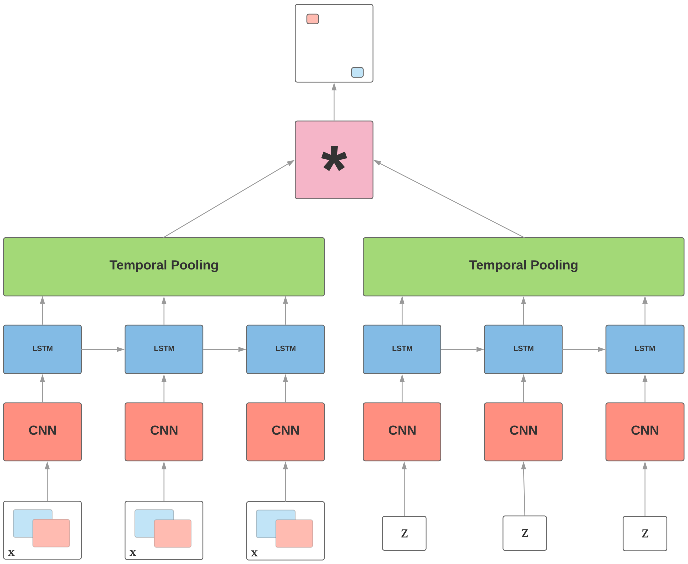

Fig. 1. Fully-convolutional Siamese architecture. The architecture is fully convolutional siamese network with recurrent layers and temporal pooling. The output is a scalar-valued score map whose dimension depends on the size of the search image. This enables the similarity function to be computed for all translated sub-windows within the search image in one evaluation. In this example, the red and blue pixels in the score map contain the similarities for the corresponding sub-windows. Best viewed in colour.

2.6 Dataset Curation

During training we adopt exemplar images that are 127x127 and search images that are 255x255 pixels. Images are scaled such that the bounding box, plus an added margin for context, has a fixed area. More precisely, if the tight bounding box has size (w,h) and the context margin is p, then the scale factor s is chosen such that the area of the scaled rectangle is equal to a constant

$$s(w + 2p) \text{ x } s(h+2p) = A$$

We use the area of the exemplar images A = 127^2 and set the amount of context to be half of the mean dimension p = $\frac{(w+h)}{4}$. Exemplar and search images for every frame are extracted offline to avoid image resizing during training. In a preliminary version of this work, we adopted a few heuristics to limit the number of frames from which to extract the training data. For the experiments of this paper, instead, we have used all 4417 videos of ImageNet Video, which account for more than 2 million labelled bounding boxes.

Table 1. Architecture of embedding function whose Convolutional part is similar to the Krizhevsky et al. [4].

Layer	Support	Chan. map	Stride	Activation size for exemplar	for search	chans.
				127 x 127	255 x 255	x3
conv1	11 x 11	96 x 3	2	127 x 127	255 x 255	x3
pool1	5 x 3		2	29 x 29	61 x 61	x96
conv2	5 x 5	256 x 48	1	25 x 25	57 x 57	x96
pool2	3 x 3		2	12 x 12	28 x 28	x256
conv3	3 x 3	384 x 256	1	10x10	26 x 26	x192
conv4	3 x 3	384 x 192	1	8 x 8	24 x 24	x192
conv5	3 x 3	256 x 192	1	6 x 6	22 x 22	x128
lstm1	256			6 x 6	22 x 22	x128
lstm2	256			6 x 6	22 x 22	x128

2.7 Tracking algorithm

Since our purpose is to prove the efficacy of our fully- convolutional Siamese network and its generalization capability when trained on ImageNet Video, we use an extremely simplistic algorithm to perform track- ing. Unlike more sophisticated trackers, we do not update a model or maintain a memory of past appearances, we do not incorporate additional cues such as opti- cal flow or colour histograms, and we do not refine our prediction with bounding box regression. Yet, despite its simplicity, the tracking algorithm achieves sur- prisingly

good results when equipped with our offline-learnt similarity metric
Online, we do incorporate some elementary temporal constraints: we only search for the object within a region of approximately four times its previous size, and a cosine window is added to the score map to penalize large displacements. Tracking through scale space is achieved by processing several scaled versions of the search image. Any change in scale is penalized and updates of the current scale are damped.

3 Evaluation

We evaluate our results on OTB-13 benchmark which considers the averageper-frame success rate at different thresholds: a tracker is successful in a given frame if the intersection-over-union (IoU) between its estimate and the ground-truth is above a certain threshold. Trackers are then compared in terms of area under the curve of success rates for different values of this threshold. In addition to the trackers reported by [11], in Figure 3 we also compare against seven more recent state-of-the-art trackers presented in the major computer vision conferences and that can run at frame-rate speed: Staple, LCT, CCT, SCT4, DLSSVM NU, DSST and KCFDP. Given the nature of the sequences, for this bench- mark only we convert 25hyper-parameters (for training and tracking) are fixed.

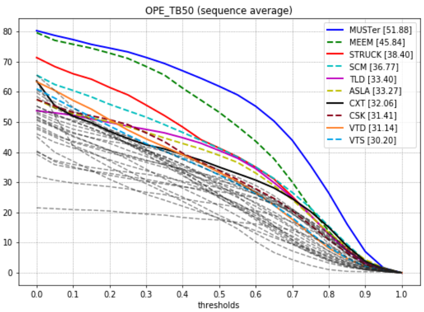

Fig. 2. Success plots on OTB-13

4 Conclusions

We propose a novel algorithm for arbitrary object tracking using a siamese network with recurrent layers and temporal pooling. We show competitive results on well known benchmarks.

References

1. Nam, H., Han, B.: Learning multi-domain convolutional neural networks for visual tracking. CoRR **abs/1510.07945** (2015)
2. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.: Fully-convolutional siamese networks for object tracking. arXiv preprint arXiv:1606.09549 (2016)
3. McLaughlin, N., Martinez del Rincon, J., Miller, P.: Recurrent convolutional network for video-based person re-identification. (2016)
4. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Commun. ACM **60**(6) (2017) 84–90
5. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1) (January 2014) 1929–1958
6. Taigman, Y., Yang, M., Ranzato, M., Wolf, L.: Deepface: Closing the gap to human-level performance in face verification. 2014 IEEE Conference on Computer Vision and Pattern Recognition (2014) 1701–1708
7. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. CoRR **abs/1503.03832** (2015)
8. Zagoruyko, S., Komodakis, N.: Learning to compare image patches via convolutional neural networks. CoRR **abs/1504.03641** (2015)
9. Koch, G., Zemel, R., Salakhutdinov, R.: Siamese neural networks for one-shot image recognition. (2015)