Spatially Supervised Recurrent Convolutional Neural Networks for Visual Object Tracking

Guanghan Ning*, Zhi Zhang*, Chen Huang*, Xiaobo Ren†, Haohong Wang†, Canhui Cai‡, Zhihai He *
*University of Missouri, Columbia, MO 65211
{gnxr9, zzbhf, chenhuang, hezhi}@mail.missouri.edu
‡Huaqiao University, Quanzhou, Fujian 362021, China
{chcai}@hqu.edu.cn
†TCL Research America, San Jose, CA 95134
{renxiaobo, haohong.wang}@tcl.com

***Abstract*—In this paper, we develop a new approach of spatially supervised recurrent convolutional neural networks for visual object tracking. Our recurrent convolutional network exploits the history of locations as well as the distinctive visual features learned by the deep neural networks. Inspired by recent bounding box regression methods for object detection, we study the regression capability of Long Short-Term Memory (LSTM) in the temporal domain, and propose to concatenate high-level visual features produced by convolutional networks with region information. In contrast to existing deep learning based trackers that use binary classification for region candidates, we use regression for direct prediction of the tracking locations both at the convolutional layer and at the recurrent unit. Our experimental results on challenging benchmark video tracking datasets show that our tracker is competitive with state-of-the-art approaches while maintaining low computational cost.**

I. Introduction

Visual tracking is a challenging task in computer vision due to target deformations, illumination variations, scale changes, fast and abrupt motion, partial occlusions, motion blur, object deformation, and background clutters. Recent advances in methods for object detection [1] have led to the development of a number of tracking-by-detection [2] approaches. These modern trackers are usually complicated systems made up of several separate components. According to [3], the feature extractor is the most important component of a tracker. Using proper features can dramatically improve the tracking performance. To handle tracking failures caused by the above mentioned factors, existing appearance-based tracking methods [4] adopt either generative or discriminative models to separate the foreground from background and distinct co-occurring objects. One major drawback is that they rely on low-level handcrafted features which are incapable to capture semantic information of targets, not robust to significant appearance changes, and only have limited discriminative power. Therefore, more and more trackers are using image features learned by deep convolutional neural networks [5]. We recognize that existing methods mainly focus on improving the performance and robustness of deep features against hand-crafted features. How to extend the deep neural network analysis into the spatiotemporal domain for visual object tracking has not been adequately studied.

In this work, we propose to develop a new visual tracking approach based on recurrent convolutional neural networks, which extends the neural network learning and analysis into the spatial and temporal domain. The key motivation behind our method is that tracking failures can often be effectively recovered by learning from historical visual semantics and tracking proposals. In contrast to existing tracking methods based on Kalman filters or related temporal prediction methods, which only consider the location history, our recurrent convolutional model is doubly deep in that it examine the history of locations as well as the robust visual features of past frames.

There are two recent papers [6], [7] that are closely related to this work. They address the similar issues of object tracking using recurrent neural networks (RNN), but they focused on artificially generated sequences and synthesized data. The specific challenges of object tracking in real-world videos have not been carefully addressed. They use traditional RNN as an attention scheme to spatially glimpse on different regions and rely on an additional binary classification at local regions. In contrast, we directly regress coordinates or heatmaps instead of using sub-region classifiers. We use the LSTM for an end-to-end spatio-temporal regression with a single evaluation, which proves to be more efficient and effective.

Major contributions of this work include: (1) we introduce a modular neural network that can be trained end-to-end with gradient-based learning methods. Using object tracking as an example application, we explore different settings and provide insights into model design and training, as well as LSTM's interpretation and regression capabilities of high-level visual features. (2) In contrast to existing ConvNet-based trackers, our proposed framework extends the neural network analysis into the spatiotemporal domain for efficient visual object tracking. (3) The proposed network is both accurate and efficient with low complexity.

II. System Overview

The overview of the tracking procedures is illustrated in Fig. 1. The proposed model is a deep neural network that takes as input raw video frames and returns the coordinates of a bounding box of an object being tracked in each frame. Mathematically, the proposed model factorizes the full tracking probability into

$$p(B_1, B_2, , B_T | X_1, X_2, , X_T) = \prod_{t=1}^{T} p(B_t | B_{<t}, X_{\leq t}), \quad (1)$$

978-1-4673-6853-7/17/$31.00 ©2017 IEEE

where B_t and X_t are the location of an object and an input frame, respectively, at time t. $B_{<t}$ is the history of all previous locations before time t, and $X_{\leq t}$ is the history of input frames up to time t. In the following section, we describe the major components of the proposed system in more detail.

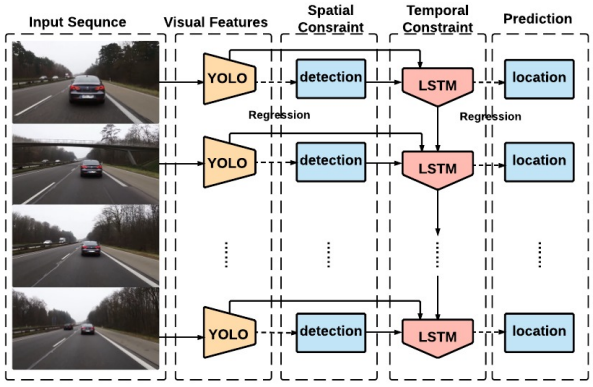


Fig. 1: Simplified overview of our system and the tracking procedure.

A. Long Short Term Memory (LSTM)

Conventional RNNs cannot access long-range context due to the back-propagated error either inflating or decaying over time, which is called the vanishing gradient problem. By contrast, LSTM RNNs overcome this problem and are able to model self-learned context information. The major innovation of LSTM is its memory cell c_t which essentially acts as an accumulator of the state information. The cell is accessed, written and cleared by several self-parameterized controlling gates. Every time a new input comes, its information will be accumulated to the cell if the input gate i_t is activated. Also, the past cell status c_{t-1} could be forgotten in this process if the forget gate f_t is on. Whether the latest cell output c_t will be propagated to the final state h_t is further controlled by the output gate o_t. In our system, we use the LSTM unit as the tracking module. Letting $\sigma = (1+e^{-x})^{-1}$, be the sigmoid nonlinearity which squashes real-valued inputs to a $[0, 1]$ range, and letting $\phi(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$, the LSTM updates for timestamp t given inputs x_t, h_{t-1}, and c_{t-1} are:

$$\begin{aligned} i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i), \\ f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f), \\ o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o), \\ g_t &= \sigma(W_{xc}x_t + W_{hc}h_{t-1} + b_c), \\ h_t &= o_t \odot \phi(c_t). \end{aligned} \tag{2}$$

B. Object Detection Using YOLO

While accuracy is important in visual tracking systems, speed is another significant factor to consider in practice. Existing tracking approaches employing ConvNets are already computationally expensive. Applying it to each frame for visual object tracking will result in prohibitively high computational complexity. Recently, a new approach to object detection is proposed in [8]. They frame object detection as a regression problem to spatially separated bounding boxes and associated class probabilities. The baseline YOLO model processes images in real-time at 45 fps. A smaller version

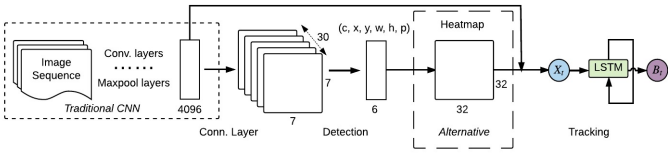


Fig. 2: Our proposed architecture.

of the network, Fast YOLO, processes at 155 fps while still the state-of-the-art object detection performance. In one frame, YOLO may output multiple detections. In assigning the correct detection to the tracking target, we employ an assignment cost matrix that is computed as the intersection-over-union (IOU) distance between the current detection and the mean of its short-term history of validated detections. The detection of the first frame, however, is determined by the IOU distance between the detections and the ground truth. Additionally, a minimum IOU is imposed to reject assignments where the detection to target overlap is less than IOU_{min}.

III. Our Proposed System

Inspired by the recent success of regression-based object detectors, we propose a new system of neural networks in order to effectively (1) process spatiotemporal information and (2) infer region locations. Our methods extends the YOLO deep convolutional neural network into the spatiotemporal domain using recurrent neural networks. So, we refer to our method by ROLO (recurrent YOLO). The architecture of our proposed ROLO is shown in Fig. 2. Specifically, (1) we use YOLO to collect rich and robust visual features, as well as preliminary location inferences; and we use LSTM in the next stage as it is spatially deep and appropriate for sequence processing. (2) Inspired by YOLOs location inference by regression, we study in this paper the regression capability of LSTM, and propose to concatenate high-level visual features produced by convolutional networks with region information. There are three phases for the end-to-end training of the ROLO model: the pre-training phase of convolutional layers for feature learning, the traditional YOLO training phase for object proposal, and the LSTM training phase for object tracking.

A. Network Training of the Detection Module

We first pre-train weights with a traditional CNN for general feature learning. The convolutional neural network takes a video frame as its input and produce a feature map of the whole image. The convolutional weights are learned with ImageNet data of 1000 classes such that the network has a generalized understanding of almost arbitrary visual objects. During pre-training, the output of the first fully connected layer is a feature vector of size 4096, a dense representation of the mid-level visual features. In theory, the feature vector can be fed into any classification tool (such as an SVM or CNN) to achieve good classification results with proper training.

Once we have the pre-trained weights able to generate visual features, we adopt the YOLO architecture as the detection module. On top of the convolutional layers, YOLO adopts fully connected layers to regress feature representation into region predictions. These predictions are encoded as an

$S \times S \times (B \times 5 + C)$ tensor. It denotes that the image is divided into $S \times S$ splits. Each split has B bounding boxes predicted, represented by its 5 location parameters including x, y, w, h, and its confidence c. A one-hot feature vector of length C is also predicted, indicating the class label of each bounding box. In our framework, we follow the YOLO architecture and set $S = 7$, $B = 2$, $C = 20$. Each bounding box originally consists of 6 predictions: x, y, w, h, *class label* and *confidence*, but we nullify class label and confidence for visual tracking, as the evaluation consists of locations only.

$$B_t = (0, x, y, w, h, 0), \tag{3}$$

where (x, y) represent the coordinates of the bounding box center relative to the width and the height of the image, respectively. The width and height of the bounding box, are also relative to those of the image. Consequently, $(x, y, w, h) \in [0, 1]$, and it is easier for regression when they are concatenated with the 4096-dimensional visual features, which will be fed into the tracking module.

B. Network Training of the Tracking Module

At last, we add the LSTM RNNs for the training of the tracking module. There are two streams of data flowing into the LSTMs, namely, the feature representations from the convolutional layers and the detection information $B_{t,i}$ from the fully connected layers. Thus, at each time-step t, we extract a feature vector of length 4096. We refer to these vectors as X_t. In addition to X_t and $B_{t,i}$, another input to the LSTM is the output of states from the last time-step S_{t-1}. In our objective module we use the Mean Squared Error (MSE) for training:

$$L_{MSE} = \frac{1}{n} \sum_{i=1}^{n} ||B_{target} - B_{pred}||_2^2, \tag{4}$$

where n is the number of training samples in a batch, y_{pred} is the models prediction, y_{target} is the target ground truth value and $|| \cdot ||$ is the squared Euclidean norm. We use the Adam method for stochastic optimization.

C. Alternative Heatmap

Regressing coordinates directly is highly non-linear and it is difficult for us to interpret the mapping. In order to know what really happens in LSTM during tracking, especially under occlusion conditions, we alternatively convert the ROLO prediction location into a feature vector of length 1024, which can be translated into a 32-by-32 heatmap. The advantage of the heatmap is that it allows to have confidence at multiple spatial locations and we can visualize the intermediate results. The heatmap not only acts as an input feature but can also warp predicted positions in the image. During training, we transfer the region information from the detection box into the heatmap by assigning value 1 to the corresponding regions while 0 elsewhere. Specifically, the detection box is converted to be relative to the 32-by-32 heatmap, which is then flattened to concatenate with the 4096 visual features as LSTM input. Let H_{target} denote the heatmap vector of the groundtruth and H_{pred} denote the heatmap predicted in LSTM output. The objective function is defined as:

$$L_{MSE} = \frac{1}{n} \sum_{i=1}^{n} ||H_{target} - H_{pred}||_2^2, \tag{5}$$

D. Spatio-temporal Regression and Spatial Supervision by Region Proposals

In our findings, LSTM is not only capable of sequence processing but also competent in effective spatio-temporal regression. This regression is two-folds: (1) The regression within one unit, i.e., between the visual features and the concatenated region representations. LSTM is capable of inferring region locations from the visual features when they are concatenated to be one unit. (2) The regression over the units of a sequence, i.e., between concatenated features over a sequence of frames. LSTM is capable of regressing the sequence of features into a predicted feature vector in the next frame. During the regression, LSTM automatically exploits the spatiotemporal information represented by visual features and region locations/heatmaps. In the YOLOs structure, regression in the fully connected layer results in object proposals. They act as soft spatial supervision for the tracking module. The supervision is helpful in two aspects: (1) When LSTM interpret the high-level visual features, the preliminary location inference helps to regress the features into the location of a certain visual elements/cues. The spatially supervised regression acts as an online appearance model. (2) Temporally, the LSTM learns over the sequence units to restrict the location prediction to a spatial range.

IV. Experimental Analysis

Our system is implemented in Python using Tensorflow, and runs at 20fps/60fps for YOLO/LSTM respectively, with eight cores of 3.4GHz Intel Core i7-3770 and an NVIDIA TITAN X GPU. The source code of our tracker, the pre-trained models, and results available on our project page. Extensive empirical evaluation has been conducted, comparing the performance of ROLO with 10 distinct trackers on a suite of 30 challenging and publicly available video sequences. Specifically, we compare our results with the top 9 trackers that achieved the best performance evaluated by the benchmark [9].We also use a modified version of SORT [10] to evaluate the tracking performance of YOLO with kalman filter. We choose the default YOLO model for fair comparison. The model is capable of detecting objects of 20 classes. We pick a subset of 30 videos from the benchmark, where the targets belong to these classes. According to experimental results of benchmark methods, the average difficulty of OTB-30 is harder than that of the full benchmark.

A. Handling Occlusions

Fig. 3: Spatio-temporal robustness against occlusion in unseen frames.

Qualitative result in Fig.3 shows that ROLO successfully tracks the object under occlusion challenges in unseen frames. Note that during frames 776-783, ROLO continues tracking the vehicle even though the detection module fails. We also train an alternative ROLO model with heatmap instead of location coordinates, in order to analyze LSTM under occlusion

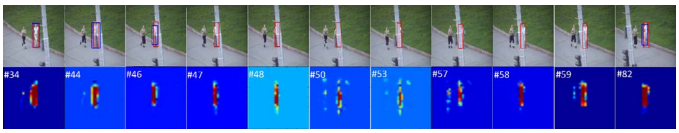


Fig. 4: Robustness against occlusion in unseen video clip. Results are shown in heatmap. Blue and Red bounding boxes indicate YOLO detection and the ground truth, respectively.

conditions. It is shown in Fig. 4 that ROLO tracks the object in near-complete occlusions. Even though two similar targets simultaneously occur in this video, ROLO tracks the correct target as the detection module inherently feeds the LSTM unit with spatial constraint. Note that between frame 47-60, YOLO fails in detection but ROLO does not lose the track. The heatmap is involved with minor noise when no detection is presented as the similar target is still in sight. Nevertheless, ROLO has more confidence on the real target even when it is fully occluded, as ROLO exploits its history of locations as well as its visual features.

B. Quantitative Results

The LSTM model is pre-trained on 3862 videos from the VID dataset of ILSVRC 2015, in order to learn the general spatial supervisions from the detection module. Then it is finetuned on a subset of videos from the VOT-2015 dataset to learn the location inference from similar dynamics of the benchmark. The OPE result is shown in Fig. 5. YOLO with kalman filter takes into account the temporal evolution of locations, while ignorant of actual environments. Due to fast motions, occlusions, and therefore occasionally poor detections, YOLO with the kalman filter perform inferiorly lacking knowledge of the visual context. In contrast, with LSTM ROLO synthesizes over sequences the robust image features as well as their soft spatial supervision.

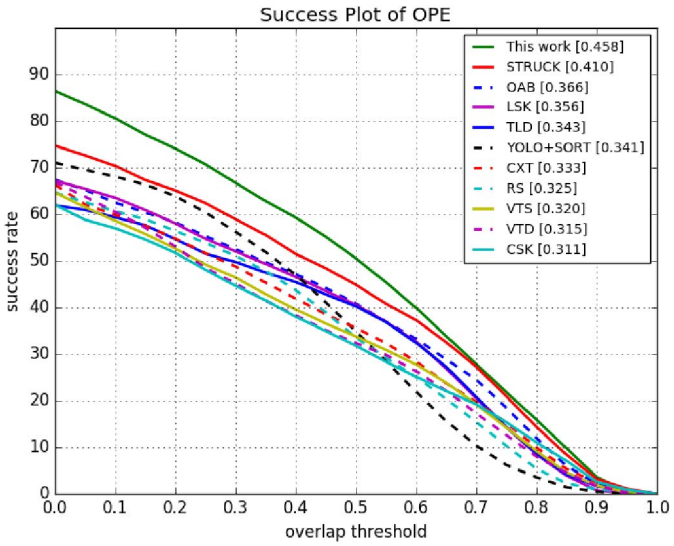
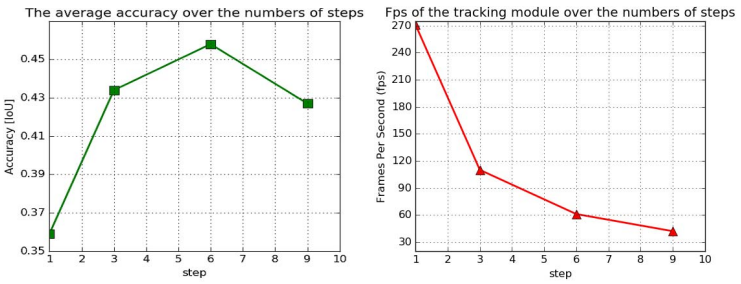


Fig. 5: Success Plots of OPE (one pass evaluation)



(a) (b)

Fig. 6: Average IOU scores and fps under various step sizes.

C. Parameter Sensitivity and Tracker Analysis

Step size denotes the number of previous frames considered each time for a prediction by LSTM. In previous experiments, we used 6 as the step number. In order to shed light upon how sequence step of LSTM affects the overall performance and running time, we repeat the experiment with various step sizes, and illustrate the results in Fig. 6. In our experiments, we also tried dropouts on visual features, random offset of detection boxes during training intended for more robust tracking, and auxiliary cost to the objective function to emphasize detection over visual features, but these results are inferior to what is shown.

V. Conclusion and Future Work

In this paper, we have successfully developed a new method of spatially supervised recurrent convolutional neural networks for visual object tracking. Our proposed ROLO method extends the deep neural network learning and analysis into the spatiotemporal domain. We have also studied LSTM's interpretation and regression capabilities of high-level visual features. Our proposed tracker is both spatially and temporally deep, and can effectively tackle problems of major occlusion and severe motion blur.

References

[1] R. Girshick, "Fast r-cnn," in *ICCV*, 2015, pp. 1440–1448.

[2] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Stct: Sequentially training convolutional networks for visual tracking." CVPR, 2016.

[3] N. Wang, J. Shi, D.-Y. Yeung, and J. Jia, "Understanding and diagnosing visual tracking systems," in *ICCV*, 2015, pp. 3101–3109.

[4] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *ECCV*. Springer, 2012, pp. 702–715.

[5] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Visual tracking with fully convolutional networks," in *ICCV*, December 2015.

[6] S. E. Kahou, V. Michalski, and R. Memisevic, "Ratm: Recurrent attentive tracking model," *arXiv preprint arXiv:1510.08660*, 2015.

[7] Q. Gan, Q. Guo, Z. Zhang, and K. Cho, "First step toward model-free, anonymous object tracking with recurrent neural networks," *arXiv preprint arXiv:1511.06425*, 2015.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," CVPR, 2016.

[9] Y. Wu, J. Lim, and M.-H. Yang, "Object tracking benchmark," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 37, no. 9, pp. 1834–1848, 2015.

[10] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," *arXiv:1602.00763*, 2016.