Fast multi-object tracking by detection

Suraj M S
Georgia Institute of Technology
mssuraj@gatech.edu

Takuma Nakamura
Georgia Institute of Technology
takuma.nakamura@gatech.edu

Abstract

Object detection and tracking is a crucial part of computer vision systems with wide applications in autonomous navigation, surveillance, augmented reality, and video editing. Most of these applications require systems that are fast and accurate enough to be used in real-time settings.
Recently, many convolutional neural network (CNN) models for fast object detection have emerged. We explore these models and ways to extend them for multi-object tracking whilst retaining their accuracy and speed.

1. Introduction

Multi-object tracking is a fundamental problem in computer vision. However, the current state-of-the-art is far from human level performance and there is considerable room for improvement. Tracking-by-detection is one of the many methods used for this purpose. With the recent progress in real-time object detection models using CNNs, it is tempting to explore how we can exploit the temporal dependencies within the object detections in frames. We aim to build a system that retains the high accuracy and speed of these detection systems.

2. Baseline tests

As a starting point, we decided to choose amongst the current state-of-the-art fast object detectors. Note that our criteria for these detectors are as follows:

- have a reported implementation that runs on a Titan X GPU at a frame-rate of at least 5 frames per second
- have a reported mAP on the VOC2007[1] dataset above 70%

We found the following detectors that satisfy this criteria:

Detector	FPS(Titan X)	mAP(%)
Faster R-CNN[6]	7	73.2
YOLOv2[5]	**67**	76.8
YOLOv2 544x544[5]	40	78.6
SSD300[3]	46	77.2
SSD512[3]	19	**79.8**

Given the relatively slow frame-rate of Faster R-CNN without any substantial mAP improvement over the other detectors, we discard it.
Both YOLO and SSD have comparable frame-rate and mAP. Hence, we compare their performances to a new dataset.

2.1. Dataset

We choose the MOT16[4] dataset for this purpose since it has a variety of challenging sequences and also provides us a standardized benchmark for both detection and tracking tasks.
The MOT16 dataset contains 14 video sequences (7 training, 7 test) in unconstrained environments filmed with both static and moving cameras. We use only the training set for performing our tests since the ground truth annotations for them are openly available. The annotations are also accompanied by detections with the Deformable part models (DPM)[2].
The training set consists of the following sequences:

Name	Frame count	Description
MOT16-02	600	People walking around a large square
MOT16-04	1050	Pedestrian street at night, elevated viewpoint
MOT16-05	837	Street scene from a moving platform
MOT16-09	525	A pedestrian street scene filmed from a low angle
MOT16-10	654	A pedestrian scene filmed at night by a moving camera
MOT16-11	900	Forward moving camera in a busy shopping mall
MOT16-13	750	Filmed from a bus on a busy intersection

2.2. YOLOv2

We decided to use YOLOv2 416416 which is available off-the-shelf at `https://pjreddie.com/darknet/yolo/`. We tested the CPU implementation of this detector only with the MOT16-02 sequence due to hardware and time constraints - this let us compare it against our measurements on SSD which were performed only with a CPU implementation.
We decided to switch to the GPU implementation for measuring the accuracy performance.

2.3. SSD300

We chose SSD300 since it is faster than its 512 counterpart and also available off-the-shelf at `https://github.com/weiliu89/caffe/tree/ssd`. We tested the CPU implementation with the entire MOT16 training dataset. We could not run the GPU implementation yet. We plan to do so next week.

2.4. Testing hardware

We used an Intel(R) Core(TM) i7-6500U CPU @ 2.50GHz with 12 GB RAM running (Arch)Linux kernel 4.10.2 for testing SSD and YOLO (CPU mode). For the YOLO GPU mode tests, we used an Intel(R) Core(TM) i7-6820 CPU @ 2.7GHz with an nVidia GTX980 GPU.

2.5. YOLO vs SSD

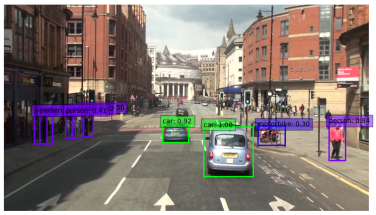
Quantitative analysis

Dataset	Time/frame (sec)		Average Precision (%)		
	YOLO	SSD	YOLO	SSD	DPM
MOT16-02	7.57145	1.36	25	35	56
MOT16-04	-	1.27	11	36	68
MOT16-05	-	1.43	59	61	56
MOT16-09	-	1.37	58	75	76
MOT16-10	-	1.26	07	35	52
MOT16-11	-	1.26	59	62	77
MOT16-13	-	1.25	00	05	33

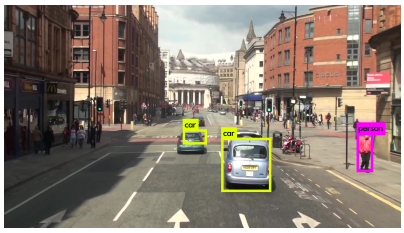
In terms of speed, SSD outperforms YOLO by a significant margin (approximately 5.5 times faster). Perhaps, this margin will not be so drastic when both are tested on a GPU. We plan to run the SSD tests on a GPU system next week.
In terms of precision, again SSD performs better than YOLO on all the sequences.

Qualitative analysis

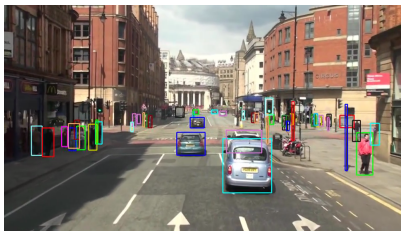
We find that both SSD and YOLO fail badly with distant objects and also when the camera viewpoint is at a high angle. In addition, on many occasions they fail when there are dense multiple instances of an object (eg. pedestrians in a crowded scene). However, in general, SSD seems to generate more accurate detections than YOLO.
Since most of the object instances are pedestrians, DPM performs better than SSD and YOLO. This can be rectified by fine-tuning our detector with pedestrian heavy dataset such as MOT16.



(a) SSD



(b) YOLO



(c) Ground truth

Figure 1: Detections on frame 100 in MOT16-13

3. Next steps

Given the results we have, we will most likely proceed with SSD as our base detector model. We plan to investigate the following tracking models:

- Detector model → [detections] → simple off-the-shelf post-process tracker → [tracks] (our baseline model)
- Detector model → [detections] → siamese/triplet loss network on consecutive frames → [tracks]
- Detector model → [detections] → RNN based post process tracker → simple post-processing → [tracks]
- We use multiple layers from the detector model and augment it with other layers to create an end-to-end

trainable network. Essentially, we use the trained detector as a good initialization for the sub-segment corresponding to the original detector layers in our new model.

References

[1] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[2] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010.

[3] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *ECCV*, 2016.

[4] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. MOT16: A benchmark for multi-object tracking. *arXiv:1603.00831 [cs]*, Mar. 2016. arXiv: 1603.00831.

[5] J. Redmon and A. Farhadi. YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242, 2016.

[6] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.