EasyChair Preprint

№ 223

Real Time Object Detection And Tracking

Dária Baikova, Rui Maia, Pedro Santos, João Ferreira and Joao Oliveira

June 2, 2018

Real Time Object Detection And Tracking

Dária Baikova[1], Rui Maia[1] Pedro Santos[1], João Ferreira[2], and João Oliveira[2]

INOV, Lisbon, Portugal
Instituto Universitário de Lisboa (ISCTE-IUL), Information Sciences, Technologies and Architecture Research Center (ISTAR-IUL), Portugal,
{daria.baikova, rui.maia, pedro.santos}@inov.pt, {jcafa, joao.p.oliveira}@iscte-iul.pt

Abstract. The present work proposes a real-time multi-object detection and tracking system to be implemented in commercial areas. The purpose is to gather and make sense of costumer behavior data extracted from surveillance footage (available from ceiling cameras) in order supply retailers with a set of analytics, management and planning tools to help them perform tasks such as planning demand and supply chains and organizing product placement on shelfs. To achieve this goal, deep learning techniques are used, which have been yielding outstanding results in computer vision problems in recent years.

Keywords: Deep Learning, Computer Vision, Object Detection, Object Tracking

1 Introduction

Today, big grocery retail chains account for the great majority of grocery shopping made globally, generating about $4 trillion annually, which makes this industry one of the biggest worldwide. To keep competitive, retail chains like WalMart (U.S.), Carrefour (France), TESCO (United Kingdom) adopt a wide range of strategies in order to collect information about its customers, to build profiles and to analyze shopping habits, such as targeting clients with personalized vouchers or using loyalty cards. However these strategies don't provide the retailer with sufficient information to have a good understanding about the shopping patterns of its costumers, which leads to inefficiencies felt throughout the current model adopted by most retailers. This includes performing tasks such as planning demand and supply chains or placement on shelfs, monitoring costumer satisfaction and overseeing the payment process in the supermarket checkouts. To oversee and fix these problems in real-time, retailers need to access to information that is currently impossible to obtain, for instance: knowing how many customers are inside the store, determining the distribution of costumers inside the stores, keeping track of the path taken by the costumers throughout the store or learning the costumers shopping patterns under different conditions. The present work is focused on: 1) studying and implementing computer vision algorithms based on Deep Learning, which are the state-of-the art for computer vision tasks; and 2) Apply these techniques to a domain-specific application (retail), which will be focused on obtaining the information retailers need.

2 Related Work

The main purpose of object tracking it to detect multiple objects in a video frame and maintain these identities in the subsequent frames (over time) in order to identify the trajectory of each object. Classical approaches to object tracking include Multiple Hypothesis Tracking (MHT) [20], recursive methods which make use of Kalman filtering to predict locations [5], Joint Probabilistic Data Association (JPDA) [15] or Particle filtering techniques [11]. However, with the recent success of Deep Learning techniques in tasks such as image classification [14,16,22,24,12] and object detection [10,9,19,17,21], and more access to data, deep learning models also started being applied to object tracking. Namely, tracking by detection has become a popular paradigm to solve the tracking problem [2,7]. This type of framework consists of two steps: applying a detector, and subsequently, a post-processing step witch involves a tracker to propagate detection scores over time. The main challenge is grouping the detected targets in contiguous frames in order to represent the targets by their trajectory over all frames. The detection step can be solved with high performing deep learning detection model [10,9,19,17,21], which in may use deep learning classification models as its base - models such as the ones in [24,12,24,23]. The detection association step has many solutions: some approaches include associating different tracks by calculating the Intersection over Union (IOU) of the bounding boxes in consecutive frames [6]; other approaches leverage Kalman filters or Hungarian algorithm such as the SORT method (Simple Online and Realtime Tracking) [4]. Other approaches skip tracking by detection and use fully Deep Learning based methods, for instance, methods based on Recurrent Neural Networks (RNNs) [18,8] or siamese convolutional networks archichectures [3]. The siamese CNN (Convolutional Neural Network) proposed in [13] is the base for the present work.

3 Proposed Architecture

At a high level the project consists of three modules represented in Fig. 1: a frame reader module, responsible for reading frames from a real-time RTSP stream and writing these to a multi-threading queue which also serves as a buffer, a detection module, responsible for outputting object bounding boxes, and the tracking module which uses the previous detection outputs in order to compute the final verdicts about the tracks of each object. Both detection and tracking modules use recent deep learning techniques: the detection module was tested with different CNN detection architectures and the tracking module uses the architecture introduced in [13], codenamed GOTURN (Generic Object Tracking Using Regression Networks). The following sections describe these modules.

3.1 Object Detection Module

The tasks in this module consist of reading frames from the queue, applying a multi-object detector to these frames and finally sending the detections to

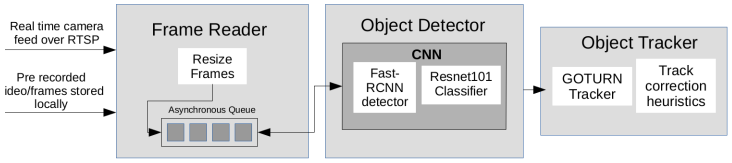


Fig. 1: Architecture overview.

the tracking module. The results sent to the tracker include the found bounding boxes and detection scores (the extent to which the model is certain of that detection). The first problem is to select the detection model, which is a trade-off between accuracy and speed (inference time). In order to find the optimal model for this application two types popular CNN detection architectures were tested: Faster-RCNN and SSD models. The R-CNN family models [10,9,21] have demonstrated excellent performance in recent years. The most advanced of these models, Faster-RCNN [21] tackles object detection as a three step problem: firstly, it uses category-independent region proposal algorithms, e.g. [25], in order to generate possible detection candidates, which are then fed to a CNN that serves as a feature extractor, and lastly, a set of class-specific classifiers are applied on top of the features in order to find the final verdict about the classes of the subjects present in the image. However, due to the region proposal step, the R-CNN models have a slower inference time than single-shot CNN alternatives such as SSD [17] architectures. Speed is a significant feature in the context of the present work due to the real-time requirement of the project. The key idea of an SSD architecture is that each of the last few layers of the network is responsible for the prediction of progressively smaller bounding boxes, and final prediction is the union of all these predictions. In this way, these models are able to simultaneously predict the bounding box and the object class in one iteration, eliminating the region proposal step altogether and are able to achieve an improvement in speed. In contrast 7 frames per second (FPS) speed with 73.2% mAP on VOC2007 test with a Faster-RCNN, SSD operates at 59 FPS with 74.3% mAP on the same dataset. The object detection models tested for this module are: a MobileNet SSD, an Inception-v2 SSD and a ResNet101 Faster-RCNN.

3.2 Tracking Module

The tracking module levreages the single object tracking architecture proposed in [13] codenamed GOTURN (Generic Object Tracking Using Regression Networks). This architecture (Fig. 2) has the structure of a Siamese CNN and is able to perform generic object tracking (which is not trained for a specific class of objects) at very high speeds (100 FPS) making it appropriate for real time applications. The network is trained entirely offline and learns the generic relationship between appearance and motion in order to be able to recognize novel

objects online. In order to learn this relationship, at each training iteration, the network takes two inputs: a crop of a frame at time *t - 1* and a crop of the next frame at time *t*. The crop of the frame at time *t - 1* contains the object the network will be looking for -trying to track-, *i.e.*, the target. The second input is a crop of the frame at time *t*, which represents the area the network will be searching for the target object in (search area), which is centered on the same point that the previous crop was, however, it is scaled by a factor of *k*. If the objects don't move too quickly, this scaling ensures the target object will still be present in the search area. The network directly regresses the bounding box coordinates of the tracked object in the search area of the next frame.

Since the GOTURN architecture is only able to track one object at a time, in order to perform multi-object tracking in real-time the tracker runs multiple times per frame (one time for each object detected when it first enters the field of view). Furthermore, in order to initialize, maintain and end a track additional rules are needed. These rules are described below.

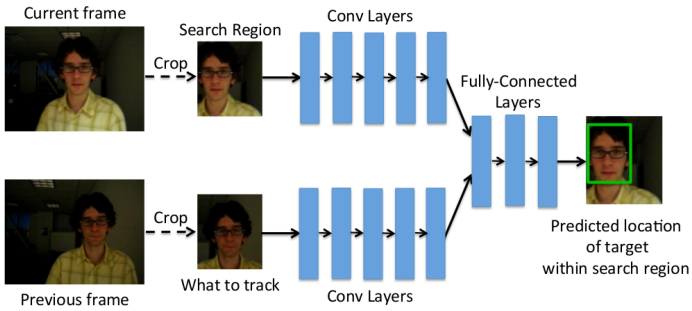


Fig. 2: GOTURN architecture.

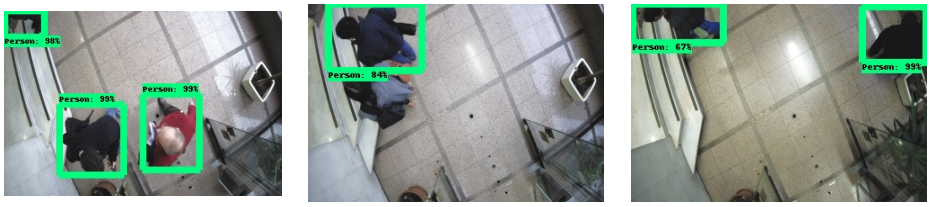
Tracker initialization and maintenance and ending The tracker is initialized when it receives the first detection from the detection module. This detection constitutes the first input fed to the tracker (target) and defines what object the tracker will be looking for during it's runtime. At each iteration, the tracker uses the previously predicted frame crop (at time *t - 1*) and the current frame search area (scaled area of the frame at time *t*, centered on the previous frame target's center) as an input. In order to make sure the tracker does not lose it's target, from *N* to *N* frames the detection re-runs on the current frame and the result is compared with the tracker's prediction via IOU (Intersection Over Union) measure; The tracker bounding box prediction is then switched by the bounding box predicted by the detector since the detection bounding boxes tend to be more accurate. When an object leaves the camera field of view it's track should be removed. In order to achieve this, a parameter is used: the track of an object is considered finished when no detections of that object occurred in *M* frames.

4 Dataset and Network Training

In order to create a new training dataset and to simulate a commercial area, a controlled test environment was created at the INOV-INESC building entrance area where hundreds of different people pass through daily. The setup consists of one top camera which captures one frame per second in the period from 07h00am to 20h00pm. These frames are afterwards stored in an SMB storage system remotely accessible for future use. The live video feed can be accessed in real time via RTSP protocol. The training for the detection models consists of labeled images annotated with object bounding boxes. The tracking dataset has a similar structure with the addition of the object IDs in each frame. Since the tracking module takes a pair of frames as inputs, an additional pre-processing step was to create pairs of target/search area images. The initial labels were created by hand, and afterwards, the ensable learning technique was used to generate new data. Three different detectors were trained on the initial dataset, and when shown a new frame, each of the models "voted" on where it though a bounding box should exist and the final annotation was a conjugation of all the predictions. In this way, the final training dataset had about 10000 labeled images. All the models (detection and tracking) were trained offline with the previously described datasets using the Tensorflow [1] framework. In order to reduce training time and data overfitting, transfer learning was used in the detection models, which were all pre-trained on the COCO dataset.

5 Results

In order to determine which of the object detection models is more suitable for this project, mean average precision (mAP) and inference time of the predictions for each model is measured and shown in Table 1. Between the three tested detection models, the Resnet101/Faster R-CNN model yields the best results Fig. 3*(a)*, however, inference time is the highest, which could cause a delay in reading and processing frames from the queue and consequently cause the queue to overflow and discard frames. This would be a problem since the tracking module relies on the assumption that the objects move slowly through time, and can lose track of an object if there are gaps in the frame sequences. The Mobilenet/SSD model is more appropriate for a real time applications since it is very fast (inference takes 5 ms), however, the detections are faulty, and in some cases, it even fails to detect completely for numerous frames (Figure 3*(b)*). The Inception-v2/SSD model seems a good compromise between inference time and precision, which is able to have a good mAP value while maintaining a fast inference time. In conclusion, if the stream reading rate is set between 5 FPS and 10 FPS, Resnet101-Faster R-CNN is still a better choice since no frames are lost in the queue (the queue writing rate is lower or equal to the frame processing rate). Since this range of frame rates does not damage tracking results, Faster-RCNN/Resnet101 was the chosen architecture for the detection module.



(a) Resnet classifier with Faster R-CNN detector

(b) Mobilenet base network with SSD detector

(c) Inception-v2 base network with SSD detector

Fig. 3: Samples of the test set inference outputs for the different detection models.

Table 1: Detection results.

Model	mAP	Inference Time (s)
Resnet101/Faster R-CNN	0,9198	0.1
MobileNet/SSD	0,8015	0,005
Inception-v2/SSD	0,9	0.04

The tracking module is able to use detection bounding boxes and produce and maintain the ID of each object through the object's path (Fig. 4). It works well even in situations where multiple people go through the camera field of view. The results are the best when the M parameter (number of frames where no detection exists before the track is considered finished) is set to two frames, *i.e.*, when no detection of an object is found in two consecutive frames, the track is considered finished. If this number is greater than two, there is the risk of a new object being associated with another object's track. Despite the results being seemingly good, at this time there is still no way of measuring the exact tracking accuracy however the tracker seems to perform well. This approach is suitable for real time tracking due to the fast nature of the performed computations in contrast to more elaborate trackers that use image information (features) to predict the next position of the tracked objects.

6 Conclusion and Future Work

The present work proposes a real time detection and tracking system aimed at implementation in commercial areas, in which the tracking by detection framework is used. Based on the results, the models (both tracking and detection) are already yielding good results and with more training data should achieve higher levels of precision. The next steps for the tracking module are to annotate new training and test data with paths of each individual object in order to properly measure tracking accuracy. This task can now be easily achieved since the information returned by the tracking module can be recycled (the tracker does most of the work and human input is only needed for minor adjustments).

Fig. 4: Tracking sequence. The tracker attributes IDs and draws colored bounding boxes around the objects in every frame which it considers belong to the same object- each object has it's own ID and color. The tracker is able to maintain these IDs even with multiple people going through the field of view. Here, four different IDs (from 2 to 5) are represented in different colors.

When these tasks are finished, the next steps will be to train deep learning models (for instance R-NNs) to predict object positions since they are the current state-of-the-art and a really promising research field.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mane, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viegas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems (2016)
2. Andriyenko, A., Schindler, K., Group, R.S.: Multi-target tracking by continuous energy minimization (2014)
3. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.: Fully-convolutional siamese networks for object tracking. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9914 LNCS, 850–865 (2016)
4. Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B.: Simple online and realtime tracking. Proceedings - International Conference on Image Processing, ICIP 2016-August, 3464–3468 (2016)
5. Black, J., Ellis, T., Rosin, P.: Multi view image surveillance and tracking. Proceedings - Workshop on Motion and Video Computing, MOTION 2002 pp. 169–174 (2002)
6. Bochinski, E., Eiselein, V., Sikora, T.: High-Speed Tracking-by-Detection Without Using Image Information. In Proceedings of the IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS) (August) (2017)
7. Choi, W.: Near-online multi-target tracking with aggregated local flow descriptor. Proceedings of the IEEE International Conference on Computer Vision 2015 Inter, 3029–3037 (2015)

8. Gan, Q., Guo, Q., Zhang, Z., Cho, K.: First Step toward Model-Free, Anonymous Object Tracking with Recurrent Neural Networks pp. 1–13 (2015)
9. Girshick, R.: Fast R-CNN. Proceedings of the IEEE International Conference on Computer Vision 2015 Inter, 1440–1448 (2015)
10. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition pp. 580–587 (2014)
11. Green, P.: Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. Biometrika 82(4), 711–732 (1995), http://biomet.oxfordjournals.org/content/82/4/711.short
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition (dec 2015)
13. Held, D., Thrun, S., Savarese, S.: Learning to track at 100 FPS with deep regression networks. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9905 LNCS, 749–765 (2016)
14. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet Classification with Deep Convolutional Neural Networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1. pp. 1097–1105. NIPS'12, Curran Associates Inc., USA (2012)
15. Kuhn, H.: The Hungarian Method For The Assignment Problem. Naval Research Logistics 52(1), 7–21 (2005)
16. LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation Applied to Handwritten Zip Code Recognition (1989)
17. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.y., Berg, A.C.: SSD: Single Shot MultiBox Detector (2015)
18. Milan, A., Leal-Taixe, L., Reid, I., Roth, S., Schindler, K.: MOT16: A Benchmark for Multi-Object Tracking pp. 1–12 (2016), http://arxiv.org/abs/1603.00831
19. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You Only Look Once: Unified, Real-Time Object Detection. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 779–788 (2016)
20. Reid, D.: An algorithm for tracking multiple targets. IEEE Transactions on Automatic Control 24(6), 843–854 (1979), http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4046312{\%}5Cnhttp://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1102177
21. Ren, S., He, K., Sun, J., Girshick, R.: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks 39(6), 1137–1149 (2017)
22. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition pp. 1–14 (2014)
23. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning (2016)
24. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 07-12-June, 1–9 (2015)
25. Uijlings, J.R.R., Van De Sande, K.E.A., Gevers, T., Smeulders, A.W.M.: Selective Search for Object Recognition (2012)