# A Proposal to Estimate Trajectory of Objects Without Using Images or Video

Bharat Bhargava
Intern, Samsung Shishya Program 2020
bharatb010699@gmail.com

12/06/2020

## 1 Introduction

The paper **Towards Real-Time Multi-Object Tracking**[3] proposes a model which simultaneously performs object detection and data association in a single-shot deep network. Though more time-efficient in comparison to the other state-of-the-art methods involving per-frame detection followed by the use of graph-based primitives or Hungarian algorithm for finding out the trajectory of the detected object(s), it does not treat data association as a separate deep learning problem.

As a matter of fact, in case I'm given object detections, why would I use the aforementioned model to re-detect the objects? Moreover, how can I perform object detection when I'm not given any video/image dataset? Obviously, I should think about directly linking these detections across frames to estimate the path, or, the trajectory of the object. Thus, the idea to build a framework so as to perform such "data association" is probably the first thing which comes to our mind. However, it is important to first investigate what this association comprises of, in terms of object tracking.

Data association can be further divided into three broad categories:

1. **Feature extraction on a per-frame basis:**[2] This would reduce our problem to feature comparison, rather than comparison of entire images. Co-ordinates of the bounding boxes can serve our purpose here.

2. **Feature comparison across detections:**[1][2] Association would certainly involve comparisons of features (actually, objects, but with very high computational efficiency), treating the entire problem as a supervised learning problem.

3. **Intra-video feature aggregation:**[1] Video datasets carry along with them rich temporal information. With our model predicting per-frame associations, linking such (single-frame) associations to obtain the trajectory would complete our task of object tracking. A "naive" thinking would be- why not perform a time-series data analysis here? Note that we also have the classes of objects provided; afterwards, we can apply a softmax classification for class prediction and through the use of an appropriate loss function, we can train the model in an end-to-end fashion.

The rest of the report is organized as follows. The following section explains each category separately. These sub-networks are eventually combined to give an end-to-end model for the proposed methodology.

# 2 Methodology

As pointed out earlier, the detections will first pass through a network extracting out the spatial features. These features can be a representative of multiple levels of abstraction, and hence a single output feature vector for a particular object can be a conglomeration of vectors obtained at several intermediate stages. Thereafter, an exhaustive permutation of such feature vectors from two different frames can be the input to the network which would estimate similarity between objects in a supervised manner by predicting the correlation between two feature vectors per estimation. Eventually, data association would simply involve performing a comprehensive time-series data analysis of the feature matrices obtained earlier, and we would store a feature matrix per frame for future reference for trajectory estimation.

## 2.1 Choosing the Appropriate Features

In the absence of a video/image dataset, this question becomes even more relevant. It is because we now have no means to extract features from an image- we rather have to create new features.

Given the coordinates of the bounding boxes, the feature vector for an object can include the coordinates of the top-left and bottom-right corners. Now, neglecting the object scale variation, we can consider size to be an appropriate distinguishing feature (considering the information which we have). Thus, our feature vector for an object can further include the **height** and **width** as two separate features. Additionally, or as a substitute, one can consider the **area** occupied by a bounding box as a parameter. Thus, the parameters which we include in our feature vector may vary greatly, and this, in itself is a problem which needs to be resolved depending upon the preciseness of the objective.

## 2.2 Input to the Network

It is obviously pertinent to feed the feature vector described above as input to the data association model to be described in parts shortly. The question is what else should be fed to the network?

Well, complying with the methodology as followed by the state-of-the-art methods, we can append each feature column vector (corresponding to each detection in a single frame) to obtain the **feature matrix** for a frame. Then, our input to the model would be two such feature matrices, where these matrices would correspond to frames which are 'n' time stamps apart (the value of 'n' can be treated as a hyper-parameter, which can be treated as an indicator of the robustness of the model. Though, in this report, it is just treated as a variable and not a hyper-parameter).

The other input to the network has to be the **ground truth labels**. Let's say, I can have a maximum of twenty objects ($N_m = 20$) in any frame. We first construct an $N_m$ X $N_m$ matrix. Let's call it M. Since we would be supplying

identification numbers to the detected objects as well (let's assume we are, indeed), the matrix would be indexed by these IDs. Say, an object with ID = 3 is detected in both frames. So, M[3][3] would equal 1. In fact, it is quite intuitive to realize that only M[i][i] can be equal to 1, for $1 \leq i \leq N_m$. In case for some j, $1 \leq j \leq N_m$, we have M[j][j] = 0, then the object is not present in at least one of the frames, and can considered as a **dummy object**. We still need to handle the objects entering and exiting the area of surveillance. This can be achieved by appending an extra row for all the objects which have left the scene, and appending an extra column for all the objects which have entered the scene. Here, a 1 at any index in the added row or column would indicate the exit or entry respectively. So, this can serve as a part of our ground truth (which is now an $(N_m+1)$ X $(N_m+1)$ matrix).

Another part of our ground truth would be the information about the **class** of each object. Linking the detections across frames to generate trajectory of the object would involve the use of a temporal framework as a **feature aggregator**, preferably **LSTM** or **RNN**, followed by **softmax classification** for class prediction. The predicted classes will then be compared with the ground truth class for training the model.

## 2.3 Feature Extraction

Once the feature matrix for two images separated by 'n' time stamps is obtained (see previous section(s)), we would run two separate deep networks for feature extraction which may share the parameters. Now, we might prefer our output (the spatial feature vector) to be an indicator of the different levels of abstraction, so we might desire to join the intermediate outputs obtained after each CNN layer. However, we may get a very large feature descriptor. Depending upon computational constraints and/or computational efficiency, we may rather conglomerate only the outputs of a few intermediate CNN layers so as to obtain our spatial feature vector per object per frame. We'd rather see our output as a **spatial feature matrix**, with a feature vector for each object.

We now have two spatial feature matrices, one for each image. What we can now do is, simply combine these feature vectors in every possible way so as to form a **tensor** to be processed by the subsequent sub-network. We'll take a spatial feature vector (a column) from first feature matrix and append it with a feature vector from the other matrix in every possible way.

Let's say, each of the $N_m$ objects detected has a corresponding feature descriptor, which is an L-dimensional vector. This means that each of the feature matrices has dimension L X $N_m$. So, the result of exhaustive permutation would be a 2L X $N_m$ X $N_m$ dimensional tensor. This tensor will serve as the input to the **feature comparison** network (or, in a more sophisticated sense, an "affinity estimator").

## 2.4 Feature Comparison

The first thing which needs to be done is to reduce the tensor obtained into a matrix, so that the ground truth matrix can serve as a direct basis for comparison. Thus, a compression network can be used to achieve this feat, which can take in the aforementioned tensor as input and can output an $N_m$ X $N_m$ matrix, with each index of the matrix indicating the similarity between the appended

L-dimensional feature vectors. An appropriate **threshold** can further refine the index values.

Now, the task to cater to the objects which are present in one frame but absent in the other one can be completed by following a similar procedure of **row and column concatenation** as described earlier. We're now ready to perform our first comparison with the ground truth, and back-propagating this loss would train our network for detecting similarity between detections across different frames.

The only thing which now remains is feature aggregation, which would eventually perform the required association. The following section describes how we can now achieve data association by following this approach.

## 2.5 Temporal Feature Aggregation

As mentioned earlier, the two video frames are 'n' time stamps apart, though this 'n' is not a hyper-parameter but a mere variable. This means, the frames can be consecutive (n = 1), or separated by a single frame (n = 2), or they may be two frames apart (n = 3), and so on.

Let's consider an example to understand how the feature aggregation for data association is expected to work. Let's say, I currently have the detections of the fifth frame. I pass the detections through the feature extractor to obtain comprehensive spatial features. Now, as a matter of fact, I would simultaneously store the feature matrices for the first four frames. Obviously, feature comparator network would estimate the affinities, comparing in this particular step each of the first four feature matrices separately with the fifth feature matrix. At the same time, batch-wise time-series data analysis on these feature matrices through the use of RNN/GRU/LSTM can allow us to possibly get a generic **video-level feature descriptor**. A soft-max classification thereafter can be used to predict the appropriate class for an object, taking the help of the ground-truth classes of the objects.

The idea of using a feature aggregator is to obtain certain video-level features, or, rather, to enhance the **temporal continuity** of the data provided. On the other hand, the feature comparator is more towards an inter-frame re-identification of objects. Presence of both these networks is expected to be quintessential for object tracking- one is **predicting associations** and the other is essentially **linking these associations** by performing a (probable) time-wise ordering.

## 2.6 Linking Sub-Networks

Unless the proposed methodology is put into real practice, it cannot be ascertained what exactly is the proper way to connect these sub-networks, especially the feature comparator and the feature aggregator. One can find out **two separate losses** from these two sub-networks and then, maybe, perform a **weighted average** of the losses, where even these weights could be learnt by the model. One can even follow an end-to-end extraction and aggregation approach, possibly by modifying the approach used for spatial feature extraction and then using the output of feature comparator as input to feature aggregator. Further, an altogether different area of research may be to carry out optimizations in the purported methodology.

# 3 Conclusion

The report is a theoretical attempt to propose data association using a deep network, when no image/video data is provided and the only data available is the bounding boxes of detected objects and their classes. New features formed from the given data can be fed to a spatial feature extraction sub-network. The output of this sub-network can be used to perform a highly exhaustive feature comparison. Simultaneously, a comprehensive, supervised time-series analysis can be used to not only associate features across frames, but also to impart a temporal sequence to the associations. This approach is highly debatable and can be modified/refined depending upon the output we obtain and/or the optimisation constraints imposed.

# References


[1] Arulkumar Subramaniam, Athira Nambiar, and Anurag Mittal. Co-segmentation inspired attention networks for video-based person re-identification. *The IEEE International Conference on Computer Vision (ICCV)*, 2019.

[2] ShiJie Sun, Naveed Akhtar, HuanSheng Song, Ajmal Mian, and Mubarak Shah. Deep affinity network for multiple object tracking. *Journal of LaTeX class files*, 13(9), 2017.

[3] Zhongdao Wang, Liang Zheng, Yixuan Liu, and Shengjin Wang. Towards real-time multi-object tracking. *arXiv preprint arXiv:1909.12605*, 2019.