DeepSORT Overview

Advance Technology Team

Exported on 10/31/2020

Table of Contents

1 Introduction

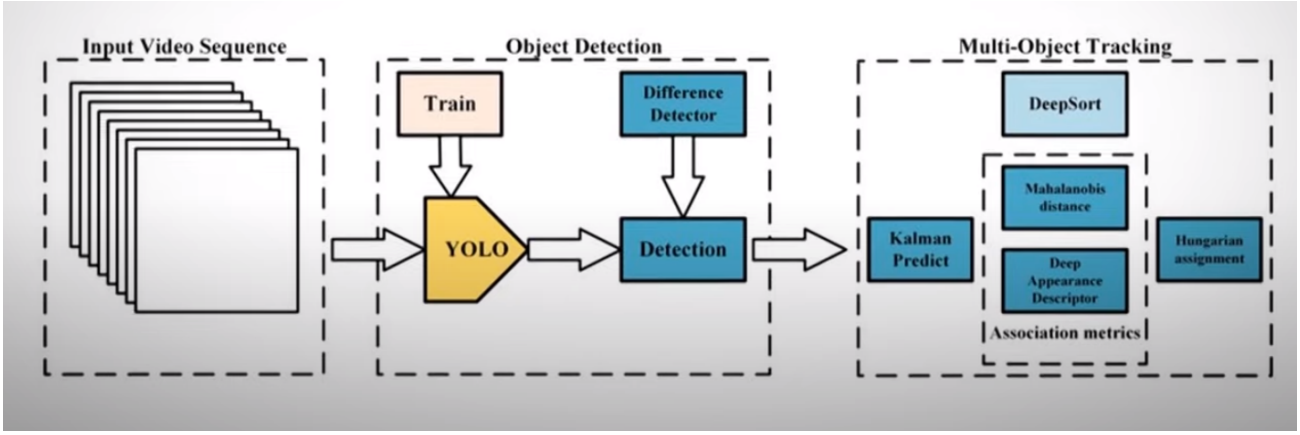
DeepSORT is an algorithm that tracks custom objects in video frames. It is by far the most popular and widely-used object tracking method used. Simple Online and Realtime Tracking with a Deep Association Metric[1] explains the approach in-depth.

Although other approaches could be used such for this problem such as Mean Shift and Optical Flow, they are computationally expensive, and tracking can get lost during occlusion. The deep learning aspect of this algorithm solves this.

1 https://arxiv.org/pdf/1703.07402.pdf

2 Object Detection/Tracking Architecture with DeepSORT and YOLO

Referring to this image may help while reading the steps explained below.



3 How it works

DeepSORT is essentially an enhancement of the algorithm SORT with a few key differences.

3.1 Simple Online Real-time Tracking (SORT)

1. Detection (YOLOv4) - Already implemented
2. Estimation (Predicting bounding boxes and creating tracks)
3. Target Association (Associating tracks with new detections)

3.2 Detection

In order to track objects, we need to detect the objects in the frame first. In our case, we are using **YOLOv4** for object detection.

Side note: As object detection improves, tracking also improves.

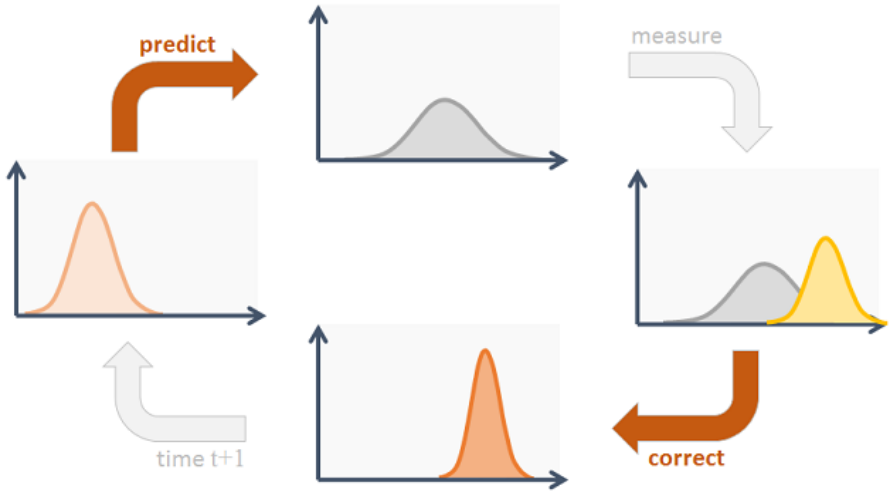
3.3 Estimation

Now, we need to estimate the state of each object in the next frame.

We define the **state** of a detected object using the variables (u, v, a, h, u', v', a', h') where (u, v) are centres of the bounding boxes, *a* is the aspect ratio and *h* is the height of the frame. The rest of the variables are the respective velocities of each variable. This state is determined using Kalman filters.

3.3.1 Kalman Filters

Kalman filters use available detections and previous predictions to arrive at the best guess of where the object should be. It does this by assuming a "Constant velocity model", which assumes constant velocity motion and constant acceleration motion.



An object's motion and state can be described using a Gaussian distribution. Kalman filters infer a new Gaussian distribution from the previous state's Gaussian distribution. This new Gaussian distribution is the predicted state/ position of the object. Kalman filters also factor in noise in detection to create better bounding box predictions.

More information on Kalman filters can be found here[2].

After determining the state of an object (so we have now defined (u, v, a, h, u', v', a', h')), we can create **tracks** for each state. A **track** has all the information from the state, and an additional variable that determines if a track should be deleted if it leaves the frame.

3.4 Target Association Problem

Now that we have predicted the bounding boxes, we want to use YOLOv4 again to find new detections. Then, we want to find a way to associate these new detections with the bounding box tracks we just predicted. I.e. How is *track_i* related to *detection_k?* For DeepSORT, we use the **Hungarian algorithm** with 2 metrics: **Mahalanobis distance** and **appearance descriptor** to solve this problem.

Side note: For SORT, we use the Hungarian algorithm with the IOU metric instead.

3.4.1 Hungarian Algorithm and Mahalanobis Distance

This Hungarian algorithm can associate an obstacle from one frame to another, based on a score/metric. The metric used in the paper is the Mahalanobis distance and an appearance descriptor.

The Mahalanobis distance is calculated using an equation 2.2.1 in the paper. However, this distance is generally suitable when motion uncertainty is low. Kalman filters are a way of *roughly* predicting the location and motion of an object, so we face possible issues with this metric. Furthermore, unpredictable camera motions may result in sharp motions which the metric cannot account for when there are occlusions.

These reasons are why DeepSORT uses an additional metric to solve the association problem.

3.4.2 Appearance Descriptor

The appearance descriptor metric is where the deep learning in DeepSORT comes into play.

Name	Patch Size/Stride	Output Size
Conv 1	$3 \times 3/1$	$32 \times 128 \times 64$
Conv 2	$3 \times 3/1$	$32 \times 128 \times 64$
Max Pool 3	$3 \times 3/2$	$32 \times 64 \times 32$
Residual 4	$3 \times 3/1$	$32 \times 64 \times 32$
Residual 5	$3 \times 3/1$	$32 \times 64 \times 32$
Residual 6	$3 \times 3/2$	$64 \times 32 \times 16$
Residual 7	$3 \times 3/1$	$64 \times 32 \times 16$
Residual 8	$3 \times 3/2$	$128 \times 16 \times 8$
Residual 9	$3 \times 3/1$	$128 \times 16 \times 8$
Dense 10		128
Batch and ℓ_2 normalization		128

The CNN produces a single feature vector (classification layer has been stripped), AKA the appearance descriptor. When this is combined with the Mahalanobis distance metric, the tracking model becomes much more effective. In

2 https://www.codeproject.com/articles/865935/object-tracking-kalman-filter-with-ease.

other words, DeepSORT uses an additional distance metric based on "appearance" in order to overcome challenges such as occlusions and varying camera angles.

You can think of this similarly to how humans differentiate between two occluded objects. When two objects overlap each other in an image, we don't just compare the distance between the two objects in order to detect that there are two separate objects in the frame, we also have an understanding of how the objects look individually (features). The appearance descriptor does exactly that for the algorithm. It gives the model an understanding of the defining characteristics of an object.