How To Train Your Deep Multi-Object Tracker

Team Members

Atharva Joshi

2020111010

Dhruv Hirpara

2020102029

Naimeesh Tiwari

2020101074

Soveet Nayak

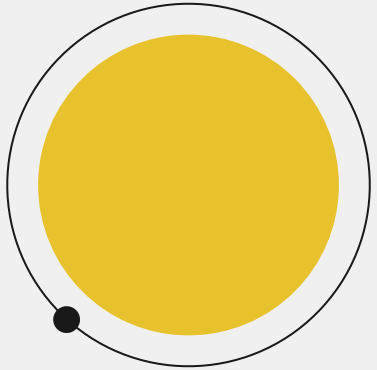
2020101086

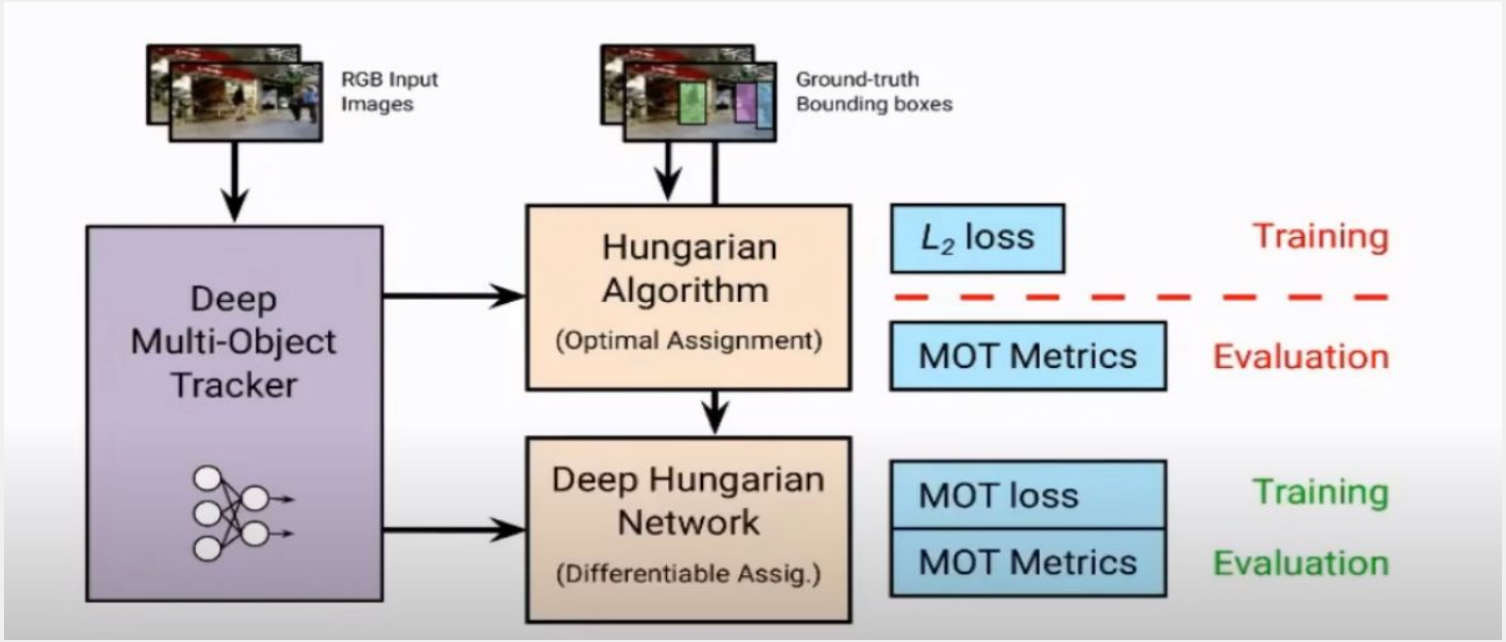
The paper proposes an end-to-end MOT training framework, based on a differentiable approximation of HA and CLEAR-MOT metrics. It experimentally demonstrates that the proposed MOT framework improves the performance of existing deep MOT methods.

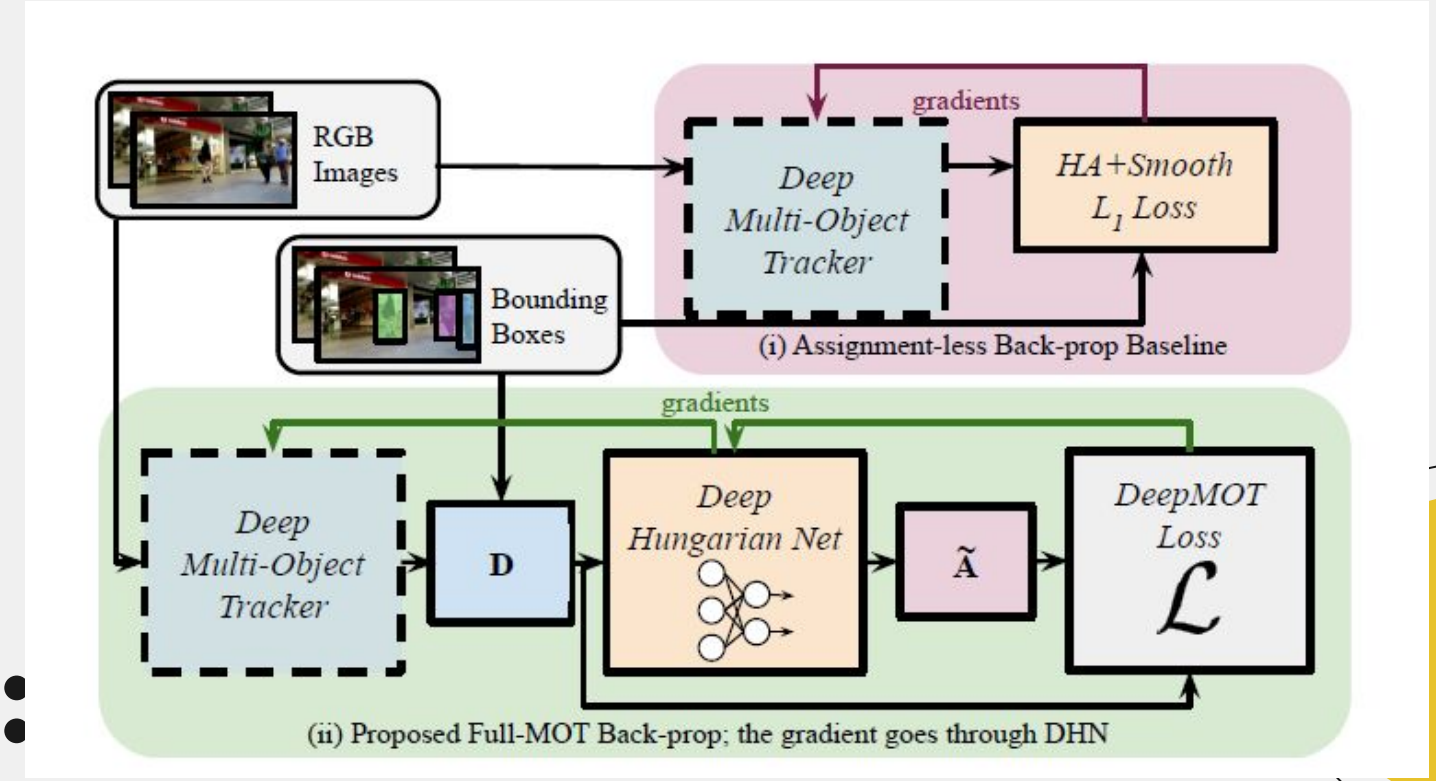
*HA: Hungarian Algorithm
*MOT: Multi Object Tracker

Motivation

- Recent data-driven trends in MOT leverage the representational power of deep networks for learning identity preserving embeddings for data association.
- However, these methods train individual parts of the MOT pipeline using proxy losses (e.g. triplet loss for learning identity embeddings), that are only indirectly related to the MOT evaluation measures. The main difficulty in defining loss functions that resemble standard tracking evaluation measures is due to the need of computing the optimal matching between the predicted object tracks and the ground-truth objects which is usually done by **Hungarian Algorithm (HA).**
- The HA consists of solving an Integer Program (involves counting operations) and hence it is non-differentiable.
- In order to back-propagate losses through the network, we need a differentiable loss function which is related to CLEAR-MOT standards (MOTA/P).
- Hence, we propose a differentiable network module - Deep Hungarian Net (DHN) - that approximates the HA and provides a soft approximation of the optimal prediction-to-ground-truth assignment.




RGB Input Images
Ground-truth Bounding boxes
Deep Multi-Object Tracker
Hungarian Algorithm
(Optimal Assignment)
Deep Hungarian Network
(Differentiable Assig.)
L_2 loss
Training
MOT Metrics
Evaluation
MOT loss
Training
MOT Metrics
Evaluation


RGB Images
Bounding Boxes
gradients
Deep Multi-Object Tracker
HA+Smooth L_1 Loss
(i) Assignment-less Back-prop Baseline
gradients
Deep Multi-Object Tracker
D
Deep Hungarian Net
$\tilde{A}$
DeepMOT Loss
$\mathcal{L}$
(ii) Proposed Full-MOT Back-prop; the gradient goes through DHN

Hungarian Algorithm

$$\mathbf{A}^* = \underset{\mathbf{A}\in\{0,1\}^{N\times M}}{\operatorname{argmin}} \sum_{n,m} d_{nm} a_{nm}, \quad \text{s.t.} \sum_{m} a_{nm} \leq 1, \forall n;$$

$$\sum_{n} a_{nm} \leq 1, \forall m; \quad \sum_{m,n} a_{nm} = \min\{N, M\}.$$

- Upon solving the above Integer program, we get a mutually consistent association between ground-truth objects and track predictions.
- Constraints ensure that all rows and columns of the assignment should sum to 1, thus avoiding multiple assignments between the two sets.
- Using A* and D, we can compute MOTA and MOTP.

Hungarian Algorithm

$$\mathrm{MOTA} = 1 - \frac{\sum_t (\mathrm{FP}_t + \mathrm{FN}_t + \mathrm{IDS}_t)}{\sum_t M_t}, \quad (1)$$

$$\mathrm{MOTP} = \frac{\sum_t \sum_{n,m} d_{tnm} a^*_{tnm}}{\sum_t |\mathrm{TP}_t|}, \quad (2)$$

FP : False Positives Count at t-th frame (the number of non-matched predicted tracks)
FN : False Negatives Count at t-th frame (the number of ground-truth objects without a match)
IDS : ID Switches Count at t-th frame (to keep track of past-frame assignments)
TP: True Positives Count at t-th frame (correspond to the number of matched predicted tracks)

Since, these evaluation measures are not differentiable, existing strategies only optimize the trackers' hyper-parameters that maximize MOTA and MOTP or their combination.
In the above version, we can not use them for tracker optimization with gradient descent techniques.

Two Step Strategy

STEP 1: Compute CLEAR-MOT tracking evaluation measures is to perform bi-partite matching between the sets of ground-truth objects and of predicted tracks.

STEP 2: Count TP, FN, FP, IDS required to get MOTA and MOTP.

STEP 1: Soft matching done using a differentiable function parameterized as a Deep Neural Network.
STEP 2: Once we establish the matching, we design a loss as a combination of differentiable functions of the (soft) assignment matrix ($\tilde{A}$) and the distance matrix (D) approximating the CLEAR-MOT measures.

Deep Hungarian Net

Replacing A* with a soft approximation, DHN produces a proxy $\tilde{A}$ that is differentiable w.r.t. D.
We model DHN by a neural network,

$$\tilde{A} = g(D, \omega d), \text{ with parameters } \omega d$$

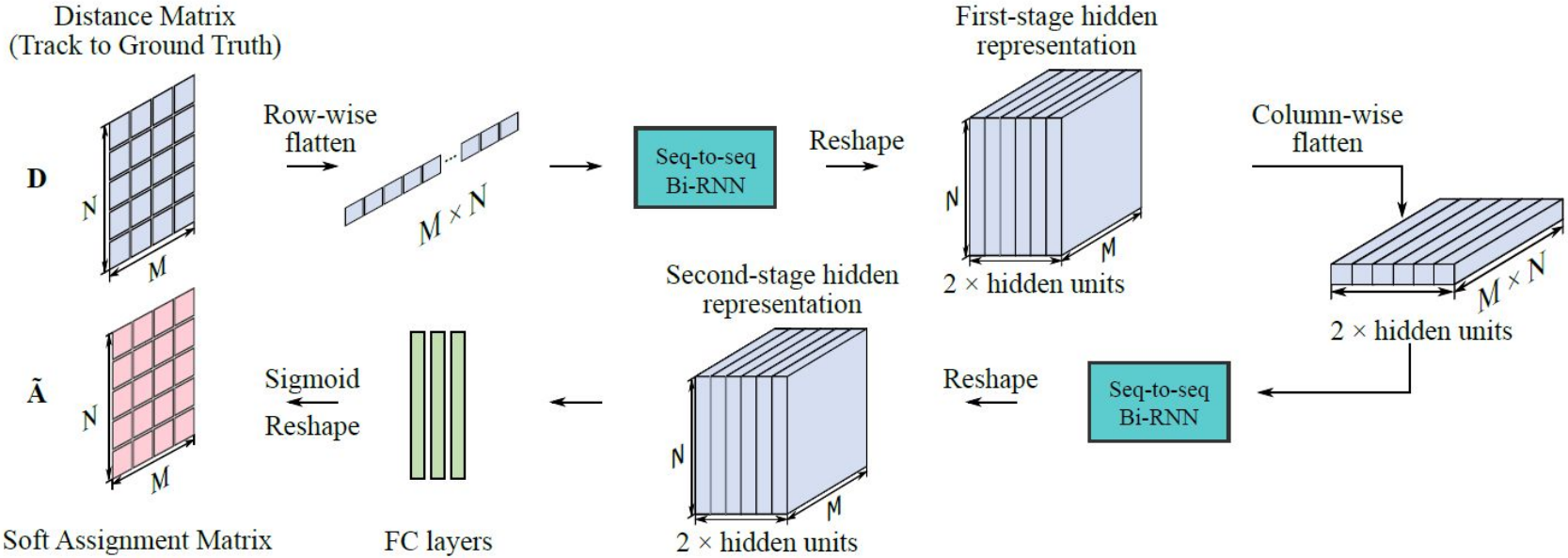
The DHN mapping must satisfy:

1. The output $\tilde{A}$ must be a good approximation to the optimal assignment matrix A*.
2. This approximation must be differentiable w.r.t. D.
3. Both input and output matrix are of equal, but varying size.
4. g must take global decisions as the HA does.

We will achieve these by:

1. Setting appropriate loss function when training DHN.
2. Designing DHN as a composite of differentiable functions
3. Making every neutron have a receptive field equivalent to the entire input.
4. We have Bi-RNNs that can guarantee the above even in large assignment problems.

Deep Hungarian Net



Distance Matrix Computation

$$d_{nm} = \frac{f(\mathbf{x}^n, \mathbf{y}^m) + \mathcal{J}(\mathbf{x}^n, \mathbf{y}^m)}{2}.$$

- For measuring similarity between two bounding boxes, we use IoU.
- Since we want D to be any differentiable distance function, we define D as an average of Euclidean center-point distance and Jaccard distance J (1 - IoU).

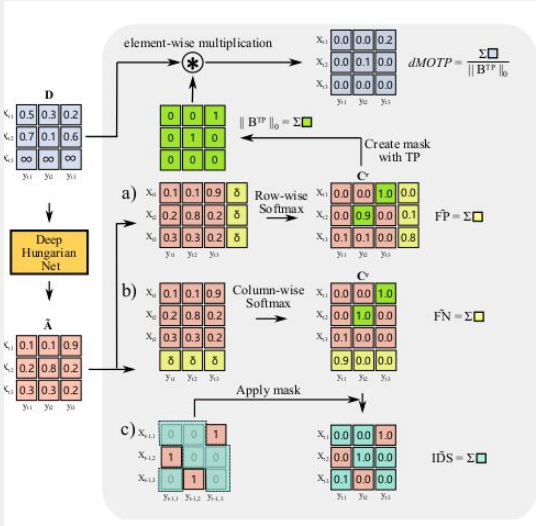
$$f(\mathbf{x}^n, \mathbf{y}^m) = \frac{\|c(\mathbf{x}^n) - c(\mathbf{y}^m)\|_2}{\sqrt{H^2 + W^2}},$$

New MOTA and MOTP

$$\tilde{\mathrm{FP}} = \sum_{n} \mathbf{C}^{r}_{n,M+1}, \quad \tilde{\mathrm{FN}} = \sum_{m} \mathbf{C}^{c}_{N+1,m} \cdot$$

$$\tilde{\mathrm{IDS}} = \| \mathbf{C}^{c}_{1:N,1:M} \odot \overline{\mathbf{B}}^{\mathrm{TP}}_{-1} \|_1,$$

$$dMOTA = 1 - \frac{\tilde{\mathrm{FP}} + \tilde{\mathrm{FN}} + \gamma\, \tilde{\mathrm{IDS}}}{M}.$$



$$dMOTP = 1 - \frac{\| \mathbf{D} \odot \mathbf{B}^{\mathrm{TP}} \|_1}{\| \mathbf{B}^{\mathrm{TP}} \|_0}.$$

Training DHN

We pose the DHN training as a 2D binary classification task using the focal loss.

We evaluate the performance of DHN by computing the weighted accuracy (WA):

$$\mathrm{WA} = \frac{w_1 n_1^* + w_0 n_0^*}{w_1 n_1 + w_0 n_0},$$

where n∗1 and n∗0 are the number of true and false positives
Class-imbalance handled by weighting the classes

Once the DHN is trained, its weights are fixed: they are not updated in any way during the training of the deep trackers.

Training DeepMOT

We randomly sample a pair of consecutive frames from the training video sequences. These two images together with their ground-truth bounding boxes constitute one training instance.

For each such instance, we first initialize the tracks with ground-truth bounding boxes (at time t) and run the forward pass to obtain the track's bounding-box predictions in the following video frame (time t+1).

We then compute D and use our proposed DHN to compute Ã. Finally, we compute our proxy loss based on D and Ã. This provides us with a gradient that accounts for the assignment, and that is used to update the weights of the tracker.

$$\mathcal{L}_{\text{DeepMOT}} = (1 - dMOTA) + \lambda(1 - dMOTP),$$

Thanks!