Learning a Neural Association Network for Self-supervised Multi-Object Tracking

Shuai Li[1]
lishuai@iai.uni-bonn.de
Michael Burke[2,3]
michael.g.burke@monash.edu
Subramanian Ramamoorthy[3]
s.ramamoorthy@ed.ac.uk
Juergen Gall[1,4]
gall@iai.uni-bonn.de

[1] University of Bonn
Bonn, Germany
[2] Monash University
Melbourne, Australia
[3] University of Edinburgh
Edinburgh, UK
[4] Lamarr Institute for Machine Learning and Artificial Intelligence
Germany

Abstract

This paper introduces a novel framework to learn data association for multi-object tracking in a self-supervised manner. Fully-supervised learning methods are known to achieve excellent tracking performances, but acquiring identity-level annotations is tedious and time-consuming. Motivated by the fact that in real-world scenarios object motion can be usually represented by a Markov process, we present a novel expectation maximization (EM) algorithm that trains a neural network to associate detections for tracking, without requiring prior knowledge of their temporal correspondences. At the core of our method lies a neural Kalman filter, with an observation model conditioned on associations of detections parameterized by a neural network. Given a batch of frames as input, data associations between detections from adjacent frames are predicted by a neural network followed by a Sinkhorn normalization that determines the assignment probabilities of detections to states. Kalman smoothing is then used to obtain the marginal probability of observations given the inferred states, producing a training objective to maximize this marginal probability using gradient descent. The proposed framework is fully differentiable, allowing the underlying neural model to be trained end-to-end. We evaluate our approach on the challenging MOT17, MOT20, and BDD100K datasets and achieve state-of-the-art results in comparison to self-supervised trackers using public detections.

1 Introduction

Multi-object tracking (MOT) is highly relevant for many applications ranging from autonomous driving to understanding the behavior of animals. Thanks to the rapid development of object detection algorithms [8, 29, 33], tracking-by-detection [11, 14, 19] has become the dominant paradigm for multi-object tracking. Given an input video, a set of detection hypotheses is first produced for each frame and the goal of tracking is to link these detection hypotheses across time, in order to generate plausible motion trajectories.

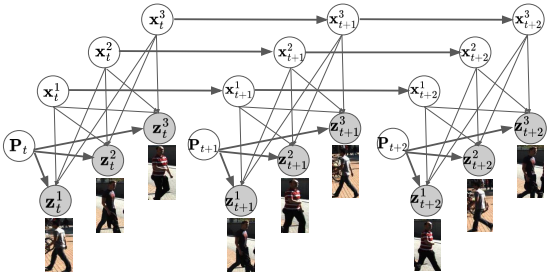


Figure 1: Given observations $\mathbf{z}_{1:T}$, a neural network is trained to predict the permutations $\mathbf{P}_{1:T}$ that assign observations to states $\mathbf{x}_{1:T}$, which define tracks. The network is trained self-supervised, *i.e.*, without given associations between $\mathbf{z}_{1:T}$ and $\mathbf{x}_{1:T}$. After training, the network is used to associate detections in MOT.

A large number of learning-based methods have focused on the fully-supervised MOT setting [6, 11, 19]. These approaches assume that a training set with detections together with the associations between these detections are provided, and the goal is to train a model that can predict data association between detections during inference. While these approaches achieve strong performance on standard tracking benchmarks [7, 26], they demand costly labeling and annotation burdens that can be expensive to scale. In contrast to fully-supervised methods, self-supervised approaches seek to train a model that is able to temporally associate noisy detections, without requiring knowledge of the data association between them during training. Following this trend, Favyen *et al*. [1] designed a method that takes two modalities as input. While the first contains only bounding-box coordinates, the second contains appearance information solely. During training, the two inputs are forced to output the same association results. Similarly, Lu *et al*. [21] suggest to drop several detections within a track and utilize a path consistency constraint to train a network for association. This method achieves state-of-the-art performance in the self-supervised setting, but requires a number of heuristics and involves a complex removal strategy for training, which makes the training very expensive.

In this work, we introduce a novel framework to learn data association in a self-supervised manner. Motivated by the fact that object motion can be usually represented by a smooth Markov process, we propose an Expectation Maximization (EM) algorithm that finds associations, which rewards locally smooth trajectories over non-smooth associations. The core of our method relies on a neural Kalman filter [16, 18] and a differentiable assignment mechanism. The Kalman filter provides a principled way to model uncertainty in an efficient way since densities are evaluated in closed form. In particular, our Kalman filter's motion model is realized by a random walk process while the observation model is parameterized by a neural network that provides the assignment probabilities of the observations to states via a permutation matrix. In other words, the permutation matrix associates detections in a video to the corresponding tracks, as illustrated in Fig. 1. Since the permutation matrix should be doubly-stochastic, we propose to add a Sinkhorn layer into the system. Kalman smoothing [28] is then used to obtain the marginal probability of observations given inferred state trajectories, leading to a training objective that maximizes this marginal. As the Sinkhorn iterations merely involve normalization across rows and columns for several steps, the entire procedure is fully differentiable, allowing the underlying neural model to be trained in an end-to-end manner using gradient descent.

We further show how this approach can be used to successfully fine-tune an appearance model on the MOT17 [26] training set using association results produced by the trained

association model, such that it enables better association abilities. During inference, we follow an online Kalman filter tracking paradigm [32] where the detections within the current frame are matched to the predicted detections from the previous frame using the learned association network.

In comparison to prior works [1, 21], our approach provides a principled and probabilistic way for self-supervised learning. The model can be trained in just a few minutes, which is much faster than [1, 21], which require about 24 hours for training. We evaluate our approach on the MOT17, MOT20 [7, 26], and BDD100K [35] datasets, and we show that our approach achieves better or comparable results than existing self-supervised multi-object tracking approaches. Our contributions are summarized as follows:

- We propose a novel self-supervised framework that embraces the Expectation Maximization algorithm to learn data association for MOT.
- The framework enables us to learn motion as well as appearance affinity together for robust data association and the learned association network naturally fits into the online tracking paradigm.
- Our approach achieves state-of-the-art performance among existing self-supervised MOT methods on the challenging MOT17 and MOT20 datasets with public detections and on the BDD100K [35] dataset.

2 Related Work

Self-Supervised Multi-Object Tracking. Self-supervised MOT has the advantage that the training of data association models does not require expensive identity-level supervision, compared to its fully-supervised counterpart. SORT [4] adopts a simple Intersection-over-Union (IoU) as an affinity metric for data association. This approach, however, is sensitive to occlusions. Following this trend, Yang *et al*. [34] present a heuristic buffered IoU (BIoU) metric for data association. Karthik *et al*. [13] utilize SORT [4] to generate pseudo track labels for training an appearance model which is then used in an online tracking framework. Favyen *et al*. [1] propose an interesting input masking strategy that enforces mutual consistency between two given input modalities as an objective during training, an approach that achieves decent results for data association. Lin *et al*. [20] present a dynamical recurrent variational autoencoder architecture, but this approach requires pre-training of motion models, while the data association between states and observations admits a simple closed-form solution, it can only track a fixed number of objects. An interesting recent work is PKF [5], which assigns observations to a given track in a soft, probabilistic way, at the sacrifice of MOTA metirc. In contrast to these works, our approach provides a principled objective and it requires only minutes to train.

3 Neural Data Association using Expectation Maximisation

In this work, we propose a self-supervised learning approach for multi-object tracking. This means that only a set of unlabeled detections in a batch of frames are given and the goal is to train a neural network such that it can predict the associations of these detections accurately.

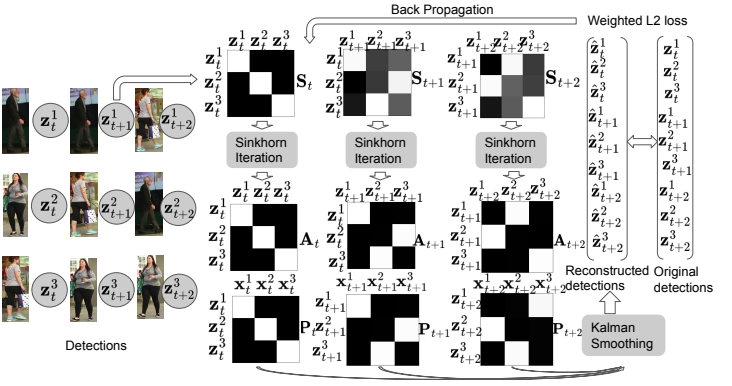


Figure 2: The proposed self-supervised learning framework. Given *unlabeled* detections $\mathbf{z}_{t:t+2}$, we train a neural network g_θ that predicts the score matrices $\mathbf{S}_{t:t+2}$ between detections from every pair of adjacent frames and for the first frame between all detections $\mathbf{z}_t$ of the same frame. For $\mathbf{S}_t$, the values should thus be high on the diagonal but low everywhere else. Sinkhorn normalization is then applied to $\mathbf{S}_{t:t+2}$ to produce the doubly stochastic association matrix $\mathbf{A}_{t:t+2}$, which associates detections between frames. The permutation matrices $\mathbf{P}_{t:t+2}$ then assign the detections to the states $\mathbf{x}$, where the state $\mathbf{x}^k_{t:t+2}$ represents a track of an object k. Kalman smoothing combined with the predicted $\mathbf{P}_{t:t+2}$ then reconstructs the detections, which are compared with original detections to define the loss in eq. (8). This loss is then back-propagated to update the parameters of the network g_θ. We only show three frames for simplicity, but the framework works on longer sequences.

We rephrase this problem as a Kalman filtering problem as shown in Fig. 1. The unknown states of K objects at the t-th time step are represented as $\mathbf{x}_t \in \mathbb{R}^{K\times d}$. Each object is represented by its d-dimensional state vector, containing its x, y central coordinates as well as its velocity in the image plane, *i.e.*$\mathbf{x} = (x, y, \dot{x}, \dot{y})$. The track of an object k over T frames is thus indicated by $\mathbf{x}^k_{1:T}$. $\mathbf{z}_t \in \mathbb{R}^{K\times d'}$ represents the observed detections where each detection at frame t is represented by a d'-dimensional observation vector, containing bounding box coordinates and appearance features. In contrast to a classical Kalman filtering problem, we do not know to which state $\mathbf{x}^k_t$ an observation $\mathbf{z}^l_t$ belongs to. This association is modeled by a permutation matrix $\mathbf{P}_t$ which assigns the observations $\mathbf{z}_t$ to the states $\mathbf{x}_t$. For training, we assume that all K objects are visible in all T frames and we will discuss occlusions or false positive detections later.

Given $\mathbf{P}_t$, we can update the states using a linear Kalman filter [18] with Gaussian noise:

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \mathbf{F}\mathbf{x}_{t-1}, \mathbf{Q}_t), \tag{1}$$

$$p(\mathbf{z}_t|\mathbf{x}_t, \mathbf{P}_t) = \mathcal{N}(\mathbf{z}_t; \mathbf{H}_t\mathbf{P}_t\mathbf{x}_t, \mathbf{R}_t), \tag{2}$$

where $\mathbf{Q}_t$ and $\mathbf{R}_t$ are Gaussian noise, $\mathbf{H}_t$ is the linear mapping from state space to observation space, and $\mathbf{x}_t = \mathbf{F}\mathbf{x}_{t-1}$ is the motion model. While we use a linear model, the approach can be extended to non-linear models or a learned dynamic model. $\mathbf{P}_t$ will be estimated by a neural network g_θ and we will learn the parameters of the network using expectation maximisation and back propagation.

3.1 Maximum Likelihood Learning Framework

Given the linear Gaussian motion and observation models (eqs. (1) and (2)), we can compute a predictive posterior, represented by state's mean $\hat{\mu}_t$ and covariance $\hat{\Sigma}_t$ at timestep t, in closed form using a Kalman filter prediction step conditioned on a series of permutations $\mathbf{P}_{1:t-1}$:

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{z}_{1:t-1},\mathbf{P}_{1:t-1}) &= \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1},\mathbf{P}_{1:t-1})d\mathbf{x}_{t-1} \\ &= \mathcal{N}(\mathbf{x}_t;\mathbf{F}\mu_{t-1},\mathbf{F}\Sigma_{t-1}\mathbf{F}^T+\mathbf{Q}_t) \\ &= \mathcal{N}(\mathbf{x}_t;\hat{\mu}_t,\hat{\Sigma}_t)\,. \end{aligned} \tag{3}$$

Here, $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1},\mathbf{P}_{1:t-1})$ is the Gaussian posterior at time $t-1$ from the last Kalman filter update step. Once the observations $\mathbf{z}_t$ are available at the current timestamp t, the update equation for the posterior is given by:

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{z}_{1:t},\mathbf{P}_{1:t}) &= \frac{p(\mathbf{z}_t|\mathbf{x}_t,\mathbf{P}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1},\mathbf{P}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1}\mathbf{P}_{1:t})} \\ &= \mathcal{N}(\mathbf{x}_t;\mu_t,\Sigma_t) \end{aligned} \tag{4}$$

$$\begin{aligned} p(\mathbf{z}_t|\mathbf{z}_{1:t-1},\mathbf{P}_{1:t}) &= \int p(\mathbf{z}_t|\mathbf{x}_t,\mathbf{P}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1},\mathbf{P}_{1:t-1})d\mathbf{x}_t \\ &= \mathcal{N}(\mathbf{z}_t;\mathbf{H}_t\mathbf{P}_t\hat{\mu}_t,(\mathbf{H}_t\mathbf{P}_t)\hat{\Sigma}_t(\mathbf{H}_t\mathbf{P}_t)^T+\mathbf{R}_t). \end{aligned} \tag{5}$$

The derivation above uses a Kalman filter that makes predictions based on a history of observations. During training, we use Kalman smoothing [28] to calculate the marginal as in eq. (6), where $\tilde{\mu}_t$ and $\tilde{\Sigma}_t$ denote the smoothed mean and covariance of the current state $\mathbf{x}_t$, respectively:

$$\begin{aligned} p(\mathbf{z}_t|\mathbf{z}_{1:T},\mathbf{P}_{1:T}) &= \int p(\mathbf{z}_t,\mathbf{x}_t|\mathbf{z}_{1:T},\mathbf{P}_{1:T})d\mathbf{x}_t \\ &= \int p(\mathbf{z}_t|\mathbf{x}_t,\mathbf{z}_{1:T},\mathbf{P}_{1:T})p(\mathbf{x}_t|\mathbf{z}_{1:T},\mathbf{P}_{1:T})d\mathbf{x}_t \\ &= \int \mathcal{N}(\mathbf{z}_t;\mathbf{H}_t\mathbf{P}_t\mathbf{x}_t,\mathbf{R}_t)\mathcal{N}(\mathbf{x}_t;\tilde{\mu}_t,\tilde{\Sigma}_t)d\mathbf{x}_t \\ &= \mathcal{N}(\mathbf{z}_t;\mathbf{H}_t\mathbf{P}_t\tilde{\mu}_t,(\mathbf{H}_t\mathbf{P}_t)\tilde{\Sigma}_t(\mathbf{H}_t\mathbf{P}_t)^T+\mathbf{R}_t). \end{aligned} \tag{6}$$

This uses both a forward and backward process to condition on all observations $\mathbf{z}_{1:T}$, rather than conditioning only on prior observations at a given time step $\mathbf{z}_{1:t}$. It not only reduces uncertainty but also helps to ensure forward and backward temporal consistency in the associations of $\mathbf{z}_{1:T}$, providing more robust training.

Estimating $\mathbf{P}_{1:T}$. We delineate the detailed implementation pipeline in Fig. 2. Given detections $\mathbf{z}_{t-1}^i$ and $\mathbf{z}_t^j$ from the frames $t-1$ and t, respectively, a network predicts their similarity $s_{ij} = g_\theta(\mathbf{z}_{t-1}^i,\mathbf{z}_t^j)$, with a higher value of s_{ij} indicating a higher similarity. By iterating over all detection pairs, we obtain a score matrix $\mathbf{S}_t$. From the score matrix, the Sinkhorn layer computes the association matrix $\mathbf{A}_t$ that associates detections between adjacent frames. As the permutation matrix $\mathbf{P}_t$ is required for training, we initialize the states in the first frame using the observations from the first frame, *i.e.*, $\mathbf{x}_1 = \mathbf{z}_1$, making $\mathbf{A}_1$ as the identity matrix. $\mathbf{P}_t$ is then obtained by $\mathbf{P}_t = \prod_{i=t}^{1}\mathbf{A}_i$.

Sinkhorn Layer. In general, the permutation $\mathbf{P}$ is a hard assignment matrix that makes it non-trivial to back-propagate the loss through it. It is therefore desirable to make it a "soft"

Algorithm 1 The proposed procedure for learning data association.

Input: Observations $\mathbf{z}_{1:T}$, learning rate α
Output: $g_\theta(\cdot)$

Initialize $\mathbf{A}_1$ with identity matrix, MLP with θ_0
for $n = 1$ to number of iterations N **do**
  for $t = 2$ to T **do**
    $\mathbf{S}_t = g_\theta(\mathbf{z}_{t-1}, \mathbf{z}_t)$
    Predict $\mathbf{A}_t$ using Sinkhorn iteration (eq. (7))
    $\mathbf{P}_t = \prod_{i=t}^{1} \mathbf{A}_i$
    Compute $p(\mathbf{x}_t|\mathbf{z}_{1:t-1}, \mathbf{P}_{1:t-1})$ using eq. (3)
    Compute $p(\mathbf{x}_t|\mathbf{z}_{1:t}, \mathbf{P}_{1:t})$ using eq. (4)
  for $t = T$ to 1 **do**
    Compute $p(\mathbf{z}_t|\mathbf{z}_{1:T}, \mathbf{P}_{1:T})$ using eq. (6)
  Compute $\mathcal{L} = -\sum_{t=1}^{T} \log p(\mathbf{z}_t|\mathbf{z}_{1:T}, \mathbf{P}_{1:T})$
  $\theta_{n+1} = \theta_n - \alpha \frac{\partial \mathcal{L}}{\partial \theta}$

Return Learned MLP $g_{\theta_N}(\cdot)$

version to enable gradient-based end-to-end training. The permutation matrix $\mathbf{P}$, however, needs to be doubly stochastic, *i.e.*, its rows and columns should sum to one. To address this, we propose to add a Sinkhorn layer [24, 31] to normalize the score matrix $\mathbf{S}$:

$$X^0(\mathbf{S}) = \exp(\mathbf{S}), \quad X^l(\mathbf{S}) = \mathcal{T}_{col}(\mathcal{T}_{row}(X^{l-1}(\mathbf{S}))), \quad \mathbf{A} = \lim_{l\to\infty} X^l(\mathbf{S}). \tag{7}$$

It applies the Sinkhorn operator [31] X to the square matrix $\mathbf{S}$. In particular, $\mathcal{T}_{row}(\cdot)$ and $\mathcal{T}_{col}(\cdot)$ indicate row-wise and column-wise normalization operations, respectively. Repeating this for several iterations creates a doubly stochastic data association matrix $\mathbf{A}$ between detections. By combining eq. (6) and $\mathbf{P}_{1:T}$, which depends on $\mathbf{S}_{1:T}$ and thus g_θ, we define our training objective for θ, which is optimized by gradient descent:

$$\arg\min_\theta -\sum_{t=1}^{T} \log \mathcal{N}(\mathbf{z}_t; \mathbf{H}_t\mathbf{P}_t\tilde{\mu}_t, (\mathbf{H}_t\mathbf{P}_t)\tilde{\Sigma}_t(\mathbf{H}_t\mathbf{P}_t)^T + \mathbf{R}_t). \tag{8}$$

The detailed training algorithm is shown in Algorithm 1. Intuitively, this can be seen as an expectation maximisation (EM) approach that alternates between inferring underlying state trajectories, and identifying permutations mapping observations to trajectories.

Learning the Appearance Model. The aforementioned procedure only involves learning association between detections from adjacent frames using relative geometrical features. However, it is also desirable to learn an appearance model in order to associate objects when geometrical information is unreliable due to abrupt camera motion. To this end, we use the inferred associations from the previous step to finetune an appearance model $\phi_\theta(\cdot)$ parameterized by the ImageNet pretrained ResNet-50 [9].

Given a training sample that contains $K \times T$ detections, we first calculate the permutation matrix $\mathbf{P}_T \in \mathbb{R}^{K\times K}$, such that the rows indicate detections at the T-th and the columns denote detections at the first frame. Each element p_{ij} represents the probability that detection i at frame T is associated to the detection j at frame 1. We also construct a similarity matrix $\mathbf{U}_T \in \mathbb{R}^{K\times K}$ using the cosine similarity of appearance features, *i.e.*,

$$u_{ij} = \frac{\phi_\theta(\mathbf{z}_T^i)^T \phi_\theta(\mathbf{z}_1^j)}{||\phi_\theta(\mathbf{z}_T^i)||^2 ||\phi_\theta(\mathbf{z}_1^j)||^2}, \tag{9}$$

where $\mathbf{U}_T$ is normalized row-wise through softmax.

Intuitively, if the i-th detection at frame T is associated with the j-th detection at the first frame, then their appearance similarity should also be high. Since $\mathbf{P}_T$ is a soft permutation matrix, we propose to minimize the KL-divergence between $\mathbf{P}_T$ and $\mathbf{U}_T$ as a second loss:

$$D_{KL}(\mathbf{P_T}||\mathbf{U_T}) = \sum_{i=1}^{K}\sum_{j=1}^{K} p_{ij}\log\frac{p_{ij}}{u_{ij}}. \tag{10}$$

Minimizing this loss forces the appearance model to learn appearance features that agree with the learned associations. The rationale to choose appearance pairs that are temporally T frames apart is to capture appearance changes over a longer period instead of incremental changes between two frames.

Preprocessing Detections. In reality, objects frequently enter and exit the scene, and false positive and negative detections may occur. We therefore preprocess the detections such that each training batch contains exactly K objects throughout T, although an explicit object birth and death handling can be adopted [25].

Features. Given a pair of detections $\mathbf{z}_i = (x_i, y_i, w_i, h_i)$ and $\mathbf{z}_j = (x_j, y_j, w_j, h_j)$, where x, y w, h indicate bounding box's central coordinate and its width and height, the pairwise geometric feature is computed as: $\mathbf{f}_{ij} = \left(\frac{2(x_j-x_i)}{h_i+h_j}, \frac{2(y_j-y_i)}{h_i+h_j}, \log\frac{h_i}{h_j}, \log\frac{w_i}{w_j}, \text{IoU}\right)$. These pairwise features serve as the input to $g_\theta(\cdot)$ for regressing the score matrix $\mathbf{S}$. The network $g_\theta(\cdot)$ is implemented as two-layer MLP with ReLU non-linearity and is trained end-to-end using stochastic gradient descent. We provide further implementation and training details in the supplementary material.

3.2 Inference

For testing, we utilize Tracktor [2] to preprocess the provided raw detections as suggested in [1, 11, 21]. We use the learned network to associate detections with predicted objects using a Kalman filter. Following [37], we define state $\mathbf{x} = (x, y, w, h, \dot{x}, \dot{y}, \dot{w}, \dot{h})$ to denote bounding box central coordinates and its corresponding velocities. More details regarding the process and observation noise of the Kalman Filter are provided in the supplementary material.

Combining Motion and Appearance Cues. We use the learned g_θ and ϕ_θ to associate detections with the predicted objects. Suppose at frame t, we have N bounding boxes $\hat{\mathbf{z}}_t$ predicted by the motion model along with M detections $\mathbf{z}_t$, the cost matrix $\mathbf{C} \in \mathbb{R}^{N\times M}$ is then used for association:

$$c_{ij} = -g_\theta(\hat{\mathbf{z}}_t^i, \mathbf{z}_t^j) - \kappa\left(\frac{\phi_\theta(\hat{\mathbf{z}}_t^i)^T\phi_\theta(\mathbf{z}_t^j)}{||\phi_\theta(\hat{\mathbf{z}}_t^i)||^2||\phi_\theta(\mathbf{z}_t^j)||^2} - s_{\min}\right), \tag{11}$$

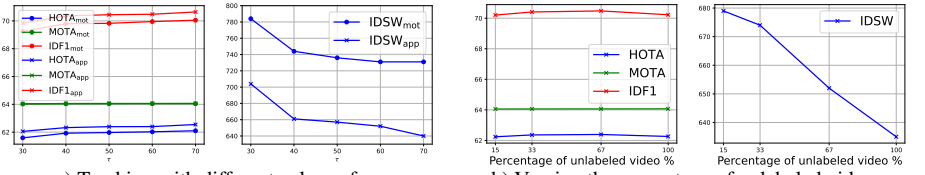
where $s_{\min}$ is the minimum cosine similarity threshold for two detections belonging to the same track and κ is the scaling factor for the cosine similarity between two detections.

Detection Noise Handling. In practice, a predicted box at frame t might not be matched to any detection due to occlusion or a missing detection. Vice versa, a detection at frame t may not be matched to any of the predicted boxes as it could be a false positive or start of a new track. To handle this, we propose to augment $\mathbf{C}$ with an auxiliary row and column containing a learned cost c_{miss} for a missing association, such that $\mathbf{C} \in \mathbb{R}^{(N+1)\times(M+1)}$. We then obtain the optimal data association $\mathbf{A}^*$ by solving

$$\mathbf{A}^* = \arg\min_{\mathbf{A}\in\mathcal{A}} \sum_{i=1}^{N+1}\sum_{j=1}^{M+1} c_{ij}a_{ij} \quad \text{s.t.} \sum_{j=1}^{M+1} a_{ij} = 1, \forall i \in \{1,\cdots,N+1\}, \sum_{i=1}^{N+1} a_{ij} = 1, \forall j \in \{1,\cdots,M+1\} \tag{12}$$

Association	HOTA ↑	MOTA ↑	IDF1 ↑	IDSW ↓
mot	62.0	64.0	69.9	731
mot + app	**62.4**	**64.1**	**70.5**	**652**

Table 1: Results on the MOT17 training set using **public** detections.



a) Tracking with different values of τ b) Varying the percentage of unlabeled videos

Figure 3: Ablation studies on MOT17 training set.

using the Hungarian algorithm [17]. In this way, tracks that are unmatched will only be updated by the motion model and detections that are unmatched to predictions initialize a new track if the detection confidence is high enough, so that our approach can deal with newly entering objects. Tracks that remain unmatched for more than τ frames are terminated.

4 Experiments

4.1 Datasets

Our experiments are conducted on the MOT17/20 and BDD100K [35] datasets. MOT17 contains 7 videos for training and 7 for testing. For all videos, detections from DPM [8], FRCNN [29] and SDP [34] are provided, resulting in 21 videos in total for training and testing. For MOT20, 4 videos are provided for training and testing under crowded scenarios. We report CLEAR [3] metrics such as MOTA and number of identity switches (IDSW), IDF1 and the HOTA metric [22]. BDD100K [35] is a large-scale autonomous driving dataset that has 1400 and 200 videos in the training and validation set, respectively. The videos contain fast camera ego-motion and frequent occlusions. 8 classes including cars and pedestrians are included. mHOTA and mIDF1 are averaged across all classes, whereas IDF1 and IDSW are summed over all classes.

4.2 Ablation Study

Impact of Appearance Cost. To show the effectiveness of fine-tuning the appearance model, we conduct experiments on the MOT17 training set. As can be seen in Table 1, adding appearance cost can further boost the model's tracking performance on all metrics. In particular, it reduces the number of identity switches (IDSW) by 11%.

Impact of τ. We study how the tracking performance is affected by the number of frames used for re-identification after occlusion. From Fig. 3(a), we observe a consistent gain among tracking metrics by increasing τ. By default, we use $\tau = 60$.

Impact of Amount of Unlabeled Data. We study how the tracking performance varies when training with different amount of unlabeled videos. We vary the training corpus by using 100%, 67%, 33%, and 15% of MOT17 and report the results on the training set with public detections. Results are shown in Fig. 3(b). While the difference in HOTA, MOTA, and IDF1 metrics is small, we observe that IDSW decreases with more training data, indicating the improved association with more unlabeled training videos.

Impact of Number of Sinkhorn Iterations. Our method is not sensitive to the number of Sinkhorn iterations as long as it exceeds 5. We used 20 iterations in our experiments.

Method	Sup.	HOTA ↑	MOTA ↑	IDF1 ↑	IDSW ↓
MHT_BiLSTM [15]	✓	41.0	47.5	51.9	2069
Tracktor++ [1]	✓	44.8	56.3	55.1	1987
TrackFormer [23]	✓	-	57.6	62.3	4018
SUSHI [3]	✓	54.6	62.0	71.5	1041
SORT [1]	✗	-	43.1	39.8	4852
UNS20regress [1]	✗	46.4	56.8	58.3	1320
UnsupTrack [13]	✗	46.9	**61.7**	58.1	1864
Lu *et al.* [21]	✗	<u>49.0</u>	58.8	<u>61.2</u>	**1219**
Ours	✗	**50.3**	<u>60.3</u>	**63.4**	<u>1266</u>

Table 2: Benchmark results on MOT17 test set using **public** detections, ✓ indicates fully-supervised and ✗ means self-supervised methods. The best and second best self-supervised performances are shown in bold and underlined numbers, respectively

Method	Sup.	HOTA ↑	MOTA ↑	IDF1 ↑	IDSW ↓
Tracktor++V2 [1]	✓	42.1	52.6	52.7	1648
ArTist [30]	✓	-	53.6	51.0	1531
ApLift [12]	✓	46.6	58.9	56.5	2241
SUSHI [3]	✓	55.4	61.6	71.6	1053
SORT20 [1]	✗	36.1	42.7	45.1	4470
Ho *et al.* [10]	✗	-	41.8	-	5918
UnsupTrack [13]	✗	<u>41.7</u>	<u>53.6</u>	<u>50.6</u>	<u>2178</u>
Ours	✗	**47.4**	**59.5**	**58.5**	**1656**

Table 3: Benchmark results on MOT20 test set using **public** detections, ✓ indicates fully-supervised and ✗ means self-supervised methods. The best and second best self-supervised performances are shown in bold and underlined numbers, respectively

Method	Sup.	mHOTA ↑	mIDF1 ↑	IDF1 ↑	IDSW ↓
MOTR *et al.* [34]	✓	-	43.5	-	-
Yu *et al.* [35]	✓	-	44.5	66.8	8315
QDTrack [27]	✓	41.7	50.8	71.5	6262
ByteTrack [31]	✓	-	54.8	70.4	9140
SORT [1]	✗	27.9	33.8	56.4	**9647**
Ours	✗	**34.5**	**42.2**	**62.3**	23143

Table 4: Benchmark results on BDD100K [35] validation set. The best self-supervised performances are shown in bold numbers. We use the same detections as in [37].

4.3 Comparison with State of the Art

MOT17. We compare our method with several other approaches in Table 2. SORT [1] relies on heuristic IoU matching with a Kalman filter. UnsupTrack [13] uses SORT to generate pseudo labels for learning an appearance model. UNS20regress [1] utilizes motion and appearance consistency trained with an RNN for association. Lu *et al.* [21] impose a heuristic path consistency constraint with several loss terms to train a matching network. Our method uses exactly the same input detections with [1, 21] preprocessed with Tracktor, and outperforms their approaches in terms of HOTA and IDF1. UnsupTrack [13] achieves better MOTA, but it uses a better detector, *i.e.* CenterNet, to improve public detections.

It is worth noting that our method performs even better than several fully-supervised approaches like MHT_BiLSTM [15], TrackFormer [23]. Tracking results using private de-

Figure 4: Qualitative results of our tracking method on the MOT17/20 and BDD100K dataset. Our method is able to track objects of different classes under occlusions, camera ego-motion and crowded scenarios. Best viewed in color.

tections are provided in the supplementary material.

MOT20. Table 3 shows our results. Our method outperforms the strongest self-supervised baseline UnsupTrack [13] in all tracking metrics by a large margin.

BDD100K. We train our model on the BDD100K training set and report the tracking results on the validation set in Table 4, using the same YOLOX detector as in [37]. For completeness, we evaluated SORT [4] as well. Our approach outperforms SORT for all metrics, only IDSW is lower for SORT. This, however, can be explained by the low recall of SORT (26.0 IDentity Recall). SORT therefore tracks only the simple cases and misses many tracks, which results in a low IDSW. Our approach has a much higher recall (36.5 IDR), which comes at the cost of more IDSWs. The results show that our approach also performs well on large-scale MOT datasets.

Qualitative Results. Figure 4 shows some qualitative results for the MOT17/20 and BDD100K test set. Our method can track objects with similar appearances under occlusion, camera motion and crowded scenarios, despite learning data associations in a self-supervised manner.

5 Conclusion

In this work, we introduced a maximum likelihood learning framework for self-supervised multi-object tracking. It learns a network for associating detections between adjacent frames and an appearance model by associating detections to states of a Kalman filter. Our method does not require expensive identity-level annotations for training, it enables online inference and achieves state-of-the-art performances on the MOT17 and MOT20 datasets among existing self-supervised approaches given public detections. It also achieves promising results on the large-scale BDD100K dataset.

Acknowledgements

The work has been supported by the project iBehave (receiving funding from the programme "Netzwerke 2021", an initiative of the Ministry of Culture and Science of the State of Northrhine Westphalia) and the ERC Consolidator Grant FORHUE (101044724). Shuai would like to thank Andreas Döring for technical discussions and Sicong Pan for proofreading the draft. The sole responsibility for the content of this publication lies with the authors.

References

[1] Favyen Bastani, Songtao He, and Samuel Madden. Self-supervised multi-object tracking with cross-input consistency. *Advances in Neural Information Processing Systems*, 34:13695–13706, 2021.

[2] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 941–951, 2019.

[3] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008: 1–10, 2008.

[4] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE, 2016.

[5] Hanwen Cao, George J Pappas, and Nikolay Atanasov. Pkf: Probabilistic data association kalman filter for multi-object tracking. *arXiv preprint arXiv:2411.06378*, 2024.

[6] Orcun Cetintas, Guillem Brasó, and Laura Leal-Taixé. Unifying short and long-term tracking with graph hierarchies. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22877–22887, 2023.

[7] P Dendorfer, H Rezatofighi, A Milan, J Shi, D Cremers, I Reid, S Roth, K Schindler, and L Leal-Taixe. Cvpr19 tracking and detection challenge: How crowded can it get? arxiv 2019. *arXiv preprint arXiv:1906.04567*, 2019.

[8] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Kalun Ho, Amirhossein Kardoost, Franz-Josef Pfreundt, Janis Keuper, and Margret Keuper. A two-stage minimum cost multicut approach to self-supervised multiple person tracking. In *Proceedings of the Asian conference on computer vision*, 2020.

[11] Andrea Hornakova, Roberto Henschel, Bodo Rosenhahn, and Paul Swoboda. Lifted disjoint paths with application in multiple object tracking. In *International Conference on Machine Learning*, pages 4364–4375. PMLR, 2020.

[12] Andrea Hornakova, Timo Kaiser, Paul Swoboda, Michal Rolinek, Bodo Rosenhahn, and Roberto Henschel. Making higher order mot scalable: An efficient approximate solver for lifted disjoint paths. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6330–6340, 2021.

[13] Shyamgopal Karthik, Ameya Prabhu, and Vineet Gandhi. Simple unsupervised multi-object tracking. *arXiv preprint arXiv:2006.02609*, 2020.

[14] Chanho Kim, Fuxin Li, Arridhana Ciptadi, and James M Rehg. Multiple hypothesis tracking revisited. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4696–4704, 2015.

[15] Chanho Kim, Fuxin Li, and James M Rehg. Multi-object tracking with neural gating using bilinear lstm. In *Proceedings of the European conference on computer vision (ECCV)*, pages 200–215, 2018.

[16] Rahul G Krishnan, Uri Shalit, and David Sontag. Deep kalman filters. *arXiv preprint arXiv:1511.05121*, 2015.

[17] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[18] Rudolf E. Kálmán. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[19] Shuai Li, Yu Kong, and Hamid Rezatofighi. Learning of global objective for network flow in multi-object tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8855–8865, 2022.

[20] Xiaoyu Lin. Unsupervised multi-object tracking via dynamical vae and variational inference. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 6910–6914, 2022.

[21] Zijia Lu, Bing Shuai, Yanbei Chen, Zhenlin Xu, and Davide Modolo. Self-supervised multi-object tracking with path consistency. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19016–19026, 2024.

[22] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *International journal of computer vision*, 129:548–578, 2021.

[23] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8844–8854, 2022.

[24] Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. Learning latent permutations with gumbel-sinkhorn networks. *International Conference on Learning Representations*, 2018.

[25] Mehdi Miah, Guillaume-Alexandre Bilodeau, and Nicolas Saunier. Learning data association for multi-object tracking using only coordinates. *Pattern Recognition*, 160: 111169, 2025.

[26] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016.

[27] Jiangmiao Pang, Linlu Qiu, Xia Li, Haofeng Chen, Qi Li, Trevor Darrell, and Fisher Yu. Quasi-dense similarity learning for multiple object tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 164–173, 2021.

[28] Herbert E Rauch, F Tung, and Charlotte T Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA journal*, 3(8):1445–1450, 1965.

[29] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*, 28:91–99, 2015.

[30] Fatemeh Saleh, Sadegh Aliakbarian, Hamid Rezatofighi, Mathieu Salzmann, and Stephen Gould. Probabilistic tracklet scoring and inpainting for multiple object tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14329–14339, 2021.

[31] Richard Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The annals of mathematical statistics*, 35(2):876–879, 1964.

[32] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017.

[33] Fan Yang, Wongun Choi, and Yuanqing Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2129–2137, 2016.

[34] Fan Yang, Shigeyuki Odashima, Shoichi Masui, and Shan Jiang. Hard to track objects with irregular motions and similar appearances? make it easier by buffering the matching space. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4799–4808, 2023.

[35] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2636–2645, 2020.

[36] Fangao Zeng, Bin Dong, Yuang Zhang, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. Motr: End-to-end multiple-object tracking with transformer. In *European conference on computer vision*, pages 659–675. Springer, 2022.

[37] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box. In *European conference on computer vision*, pages 1–21. Springer, 2022.