

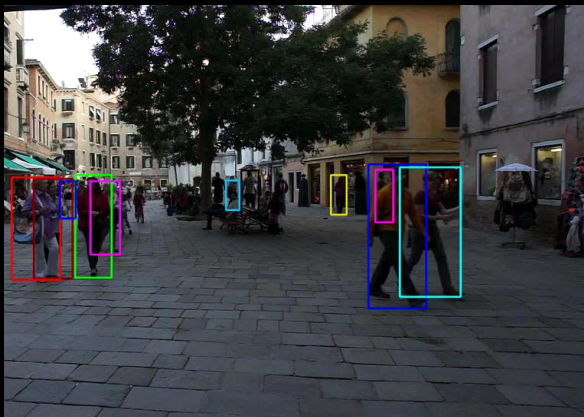
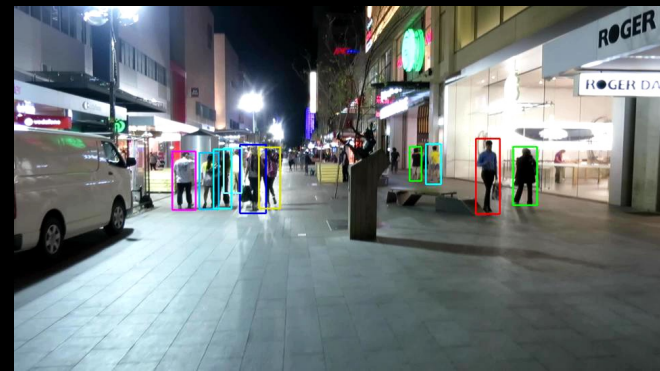
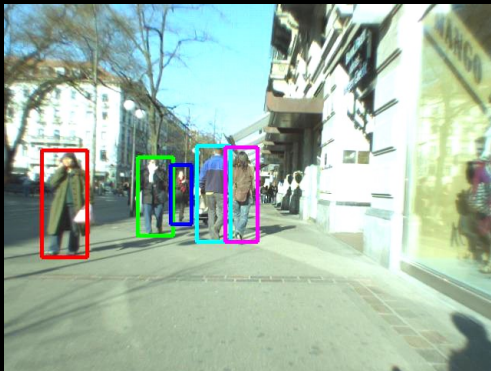


FROM HANDCRAFTED TO END-TO-END LEARNING AND BACK: A JOURNEY FOR MOT

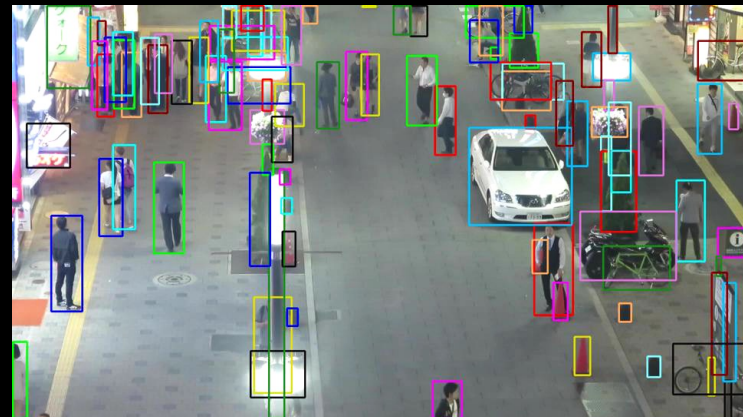
Laura Leal-Taixé

NVIDIA & Technical University of Munich

MULTI-OBJECT TRACKING

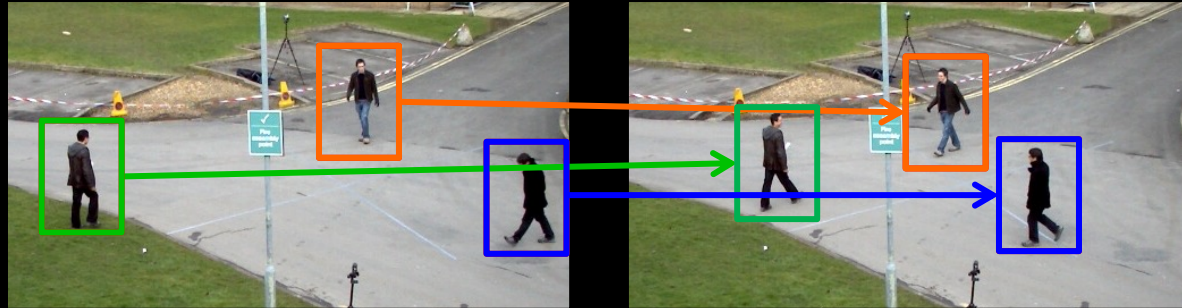


Goal: detect
and track all
objects in a
scene



PROBLEM STATEMENT

- Given a video, find out which parts of the image depict the same object in different frames
- Often we use detectors as starting points



WHY DO WE NEED TRACKING?

- To model objects when detection fails:
 - Occlusions
 - Viewpoint/pose/blur/illumination variations (in a few frames of a sequence)
 - Background clutter
- To reason about the dynamic world, e.g., trajectory prediction (is the person going to cross the street?)

TRACKING IS...

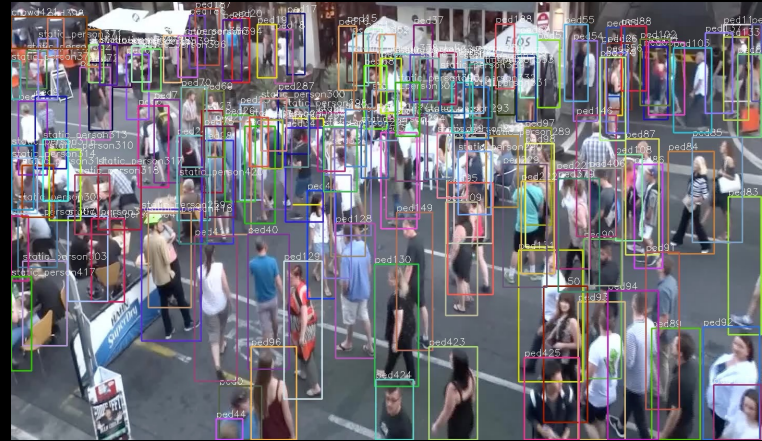
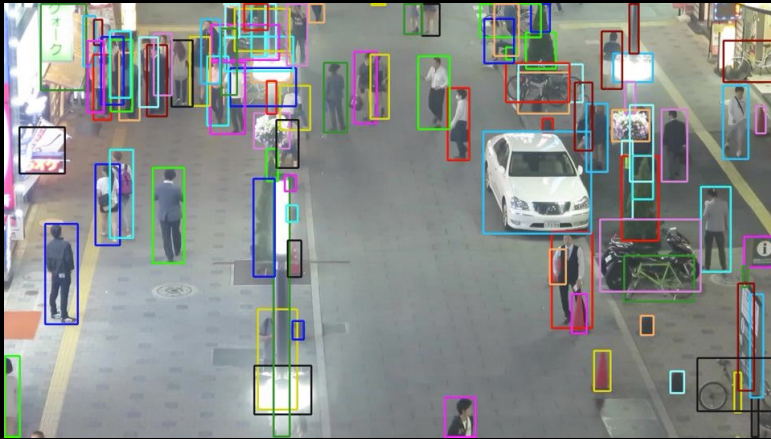
- Similarity measurement
- Correlation
- Correspondence
- Matching/retrieval
- Data association

TRACKING IS ALSO...

- Learning to model our target:
- **Appearance:** we need to know how the target looks like
 - Single object tracking
 - Re-identification
- **Motion:** to make predictions of where the targets goes
 - Trajectory prediction

CHALLENGES

- Multiple objects of the same type
- Heavy occlusions
- Appearance is often very similar



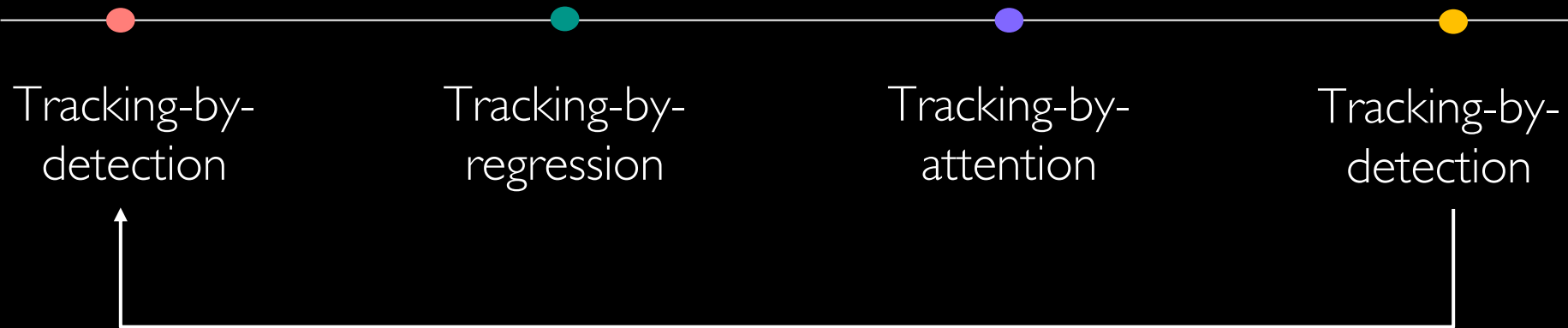
OFFLINE VS. ONLINE

- Online tracking
 - Processes frames as they become available
 - For real-time applications, e.g., autonomous driving, AR/VR
 - Prone to drifting → hard to recover from errors or occlusion
- Offline tracking
 - Processes a batch of frames
 - Good to recover from occlusions (short ones as we will see)
 - Not suitable for real-time applications
 - Suitable for video analysis, automatic labeling, video editing.

OFFLINE VS. ONLINE

- Online tracking
 - Tracking-by-regression, e.g., Tracktor, Centertrack.
 - Transformer-based trackers, e.g., Trackformer
 - GHOST: striving for simplicity
- Offline tracking
 - Tracking with graphical models
 - Learning to track with graph neural networks, e.g., MPNTrack

SHIFTING PARADIGMS IN MOT



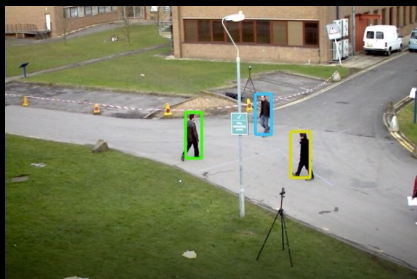
→ Towards unifying detection and tracking

→ Towards end-to-end learning

MOTCHALLENGE

- MOTChallenge: www.motchallenge.net
 - Multiple object tracking (from sparse to extremely crowded)

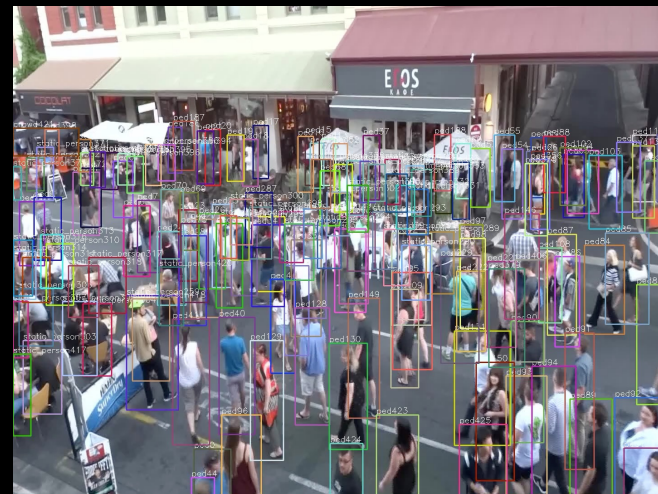
MOT15



MOT16/17



MOT20



TRACKING-BY- DETECTION

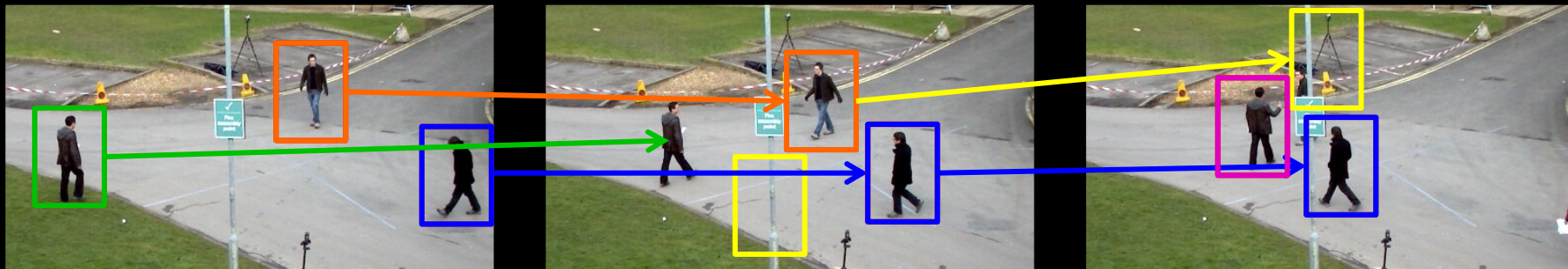
TRACKING-BY-DETECTION

- DETECTION: Detector on each frame to obtain a set of proposed locations



TRACKING-BY-DETECTION

- DETECTION: Detector on each frame to obtain a set of proposed locations
- DATA ASSOCIATION: Connect the detections in the temporal domain to create trajectories.



A SIMPLE ONLINE TRACKER

t



t+1



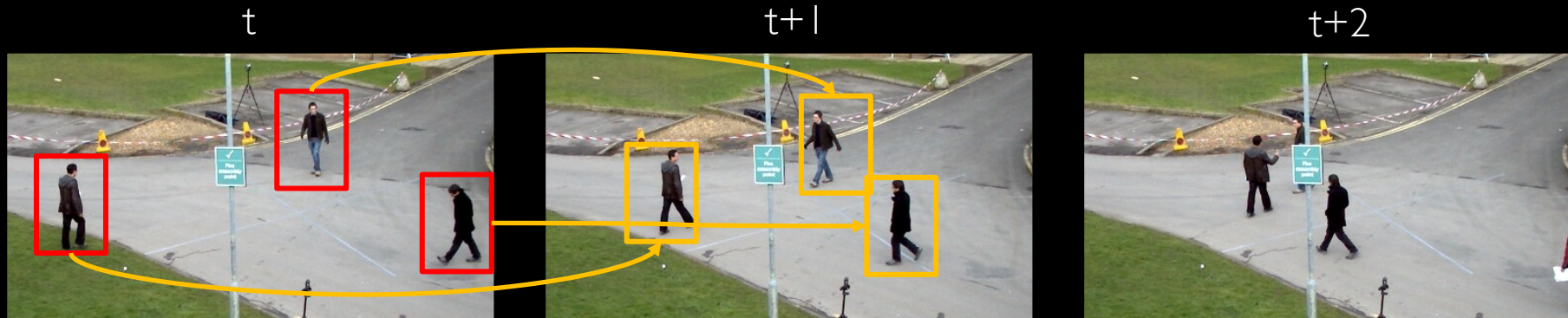
t+2



- 1. Track initialization (e.g. using a detector)



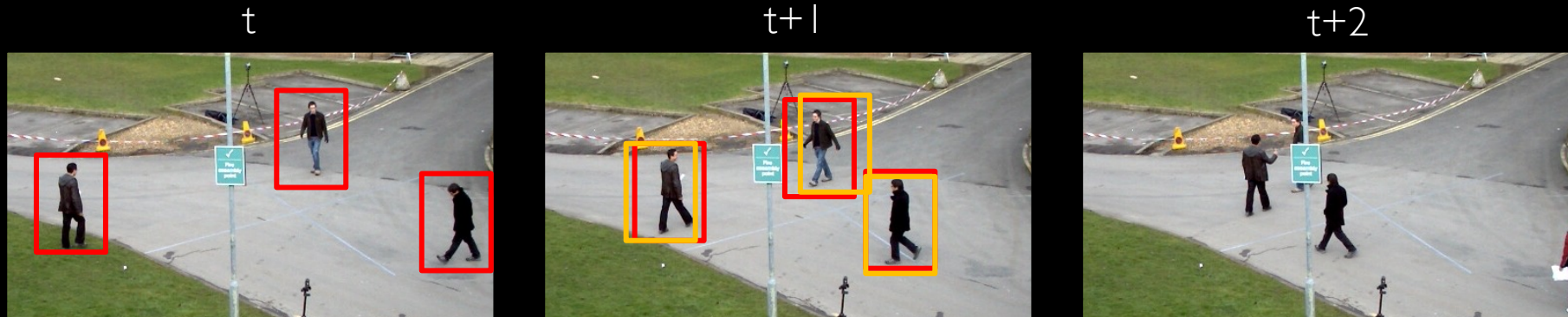
A SIMPLE ONLINE TRACKER



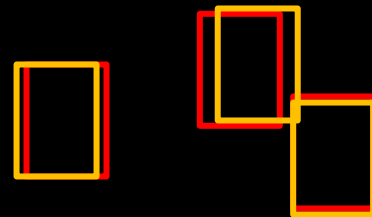
- 1. Track initialization (e.g. using a detector)
- 2. Prediction of the next position (motion model)



A SIMPLE ONLINE TRACKER



- 1. Track initialization (e.g. using a detector)
- 2. Prediction of the next position (motion model)
- 3. **Matching** predictions with detections (appearance model)

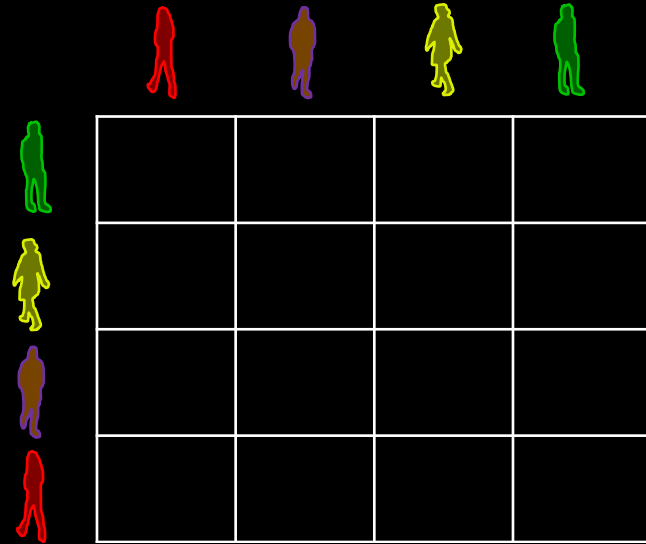


A SIMPLE ONLINE TRACKER

- 2. Prediction of the next position (motion model)
 - Classic: Kalman filter
 - Nowadays: Recurrent architecture
 - For now: we will assume a constant velocity model (spoiler alter: it works really well at high framerates and without occlusions!)









A SIMPLE ONLINE TRACKER

- 3. Matching predictions with detections



A SIMPLE ONLINE TRACKER

- Bipartite matching
 - Define distances between boxes (e.g., IoU, pixel distance, 3D distance, reID)

				
	0.9	0.8	0.8	0.1
	0.5	0.4	0.3	0.8
	0.2	0.1	0.4	0.8
	0.1	0.2	0.5	0.9

A SIMPLE ONLINE TRACKER

- Bipartite matching
 - Define distances between boxes (e.g., IoU, pixel distance, 3D distance, reID)
 - Solve the unique matching with e.g., the Hungarian algorithm*









0.9	0.8	0.8	0.1
0.5	0.4	0.3	0.8
0.2	0.1	0.4	0.8
0.1	0.2	0.5	0.9

*Demo: <http://www.hungarianalgorithm.com/solve.php>

A SIMPLE ONLINE TRACKER

- Bipartite matching
 - Define distances between boxes (e.g., IoU, pixel distance, 3D distance, reID)
 - Solve the unique matching with e.g., the Hungarian algorithm*
 - Solutions are the unique assignments that minimize the total cost

The diagram illustrates a bipartite matching problem. On the left, four tracked objects are shown as vertical silhouettes: a blue one at the top, a red one, a green one, and a blue one at the bottom. On the right, four current frame detections are shown as horizontal silhouettes: a blue one at the top, a green one, a red one, and a blue one at the bottom. A 4x4 cost matrix is positioned between them, with the minimum cost value in each row highlighted in red.

				
	0.9	0.8	0.8	0.1
	0.5	0.4	0.3	0.8
	0.2	0.1	0.4	0.8
	0.1	0.2	0.5	0.9

THE ROLE OF LEARNING

- 1. Track initialization (e.g. using a detector)
 - Deep Learning has provided us with better detectors
- 2. Prediction of the next position (motion model)
 - Trajectory prediction has evolved as a topic on its own
- 3. Matching predictions with detections (appearance model)
 - Improving appearance models → Re-Identification
 - Matching still happens separately from learning, we will see how to couple both steps

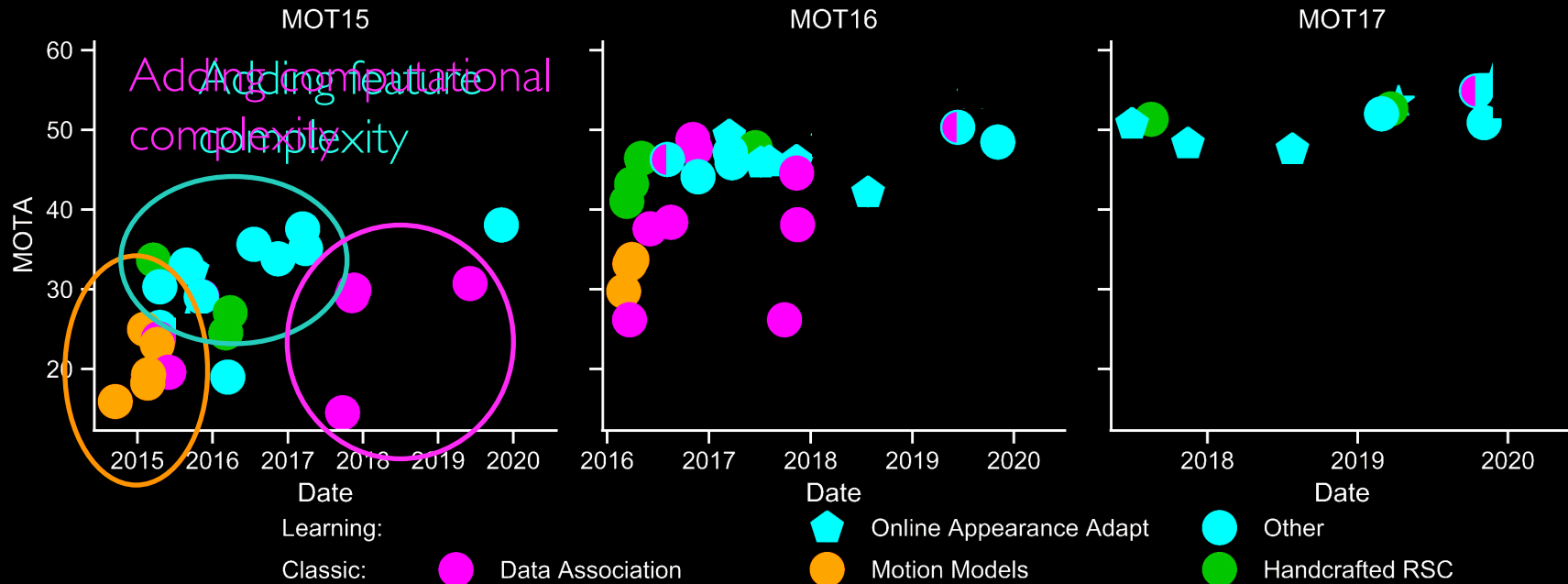


Adding temporal complexity

Adding feature complexity

Adding computational complexity

A HISTORIC VIEW



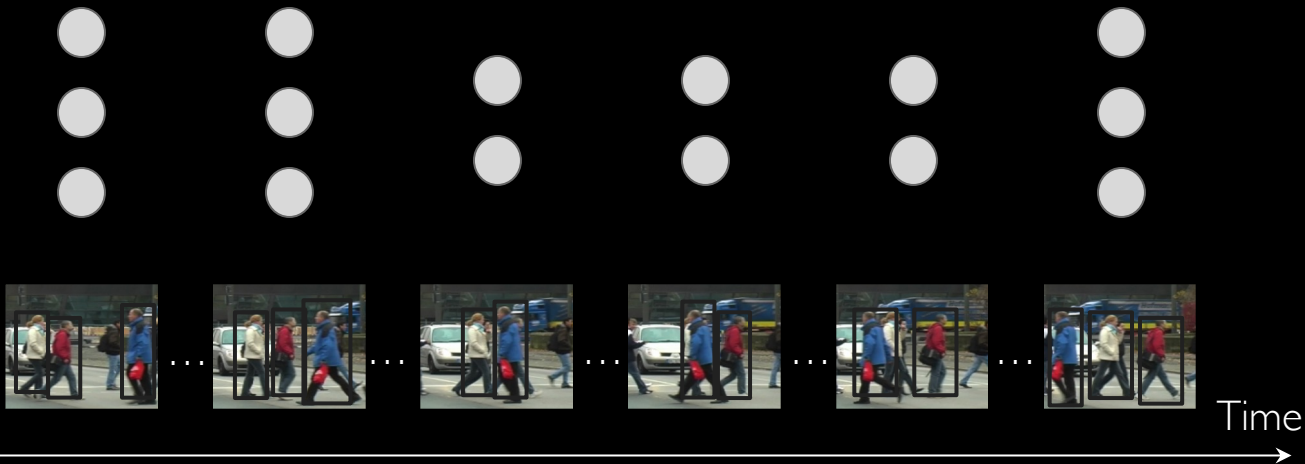
Adding temporal complexity



GRAPH-BASED ASSOCIATION

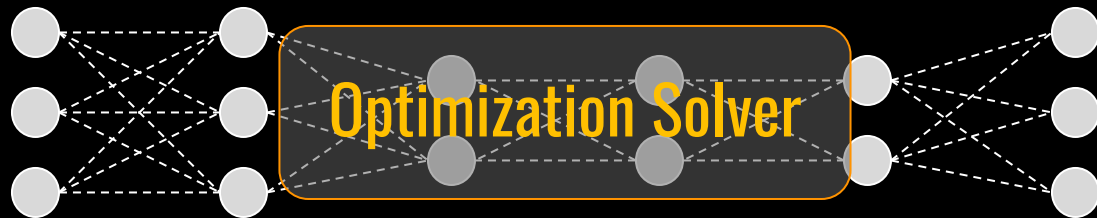
*Still tracking-by-detection.
Mostly focused on offline tracking.

Input frames and
object detections



GRAPH-BASED ASSOCIATION

- Pairwise edge costs can either be learned or handcrafted (same as for the Hungarian)
- Find trajectories with a solver, e.g., Simplex

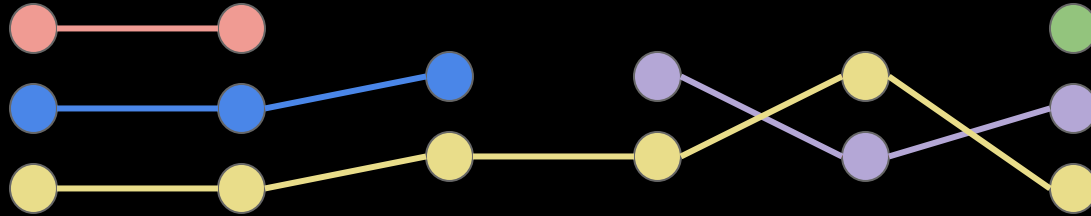


Input frames and
object detections



Time

GRAPH-BASED ASSOCIATION



Input frames and
object detections

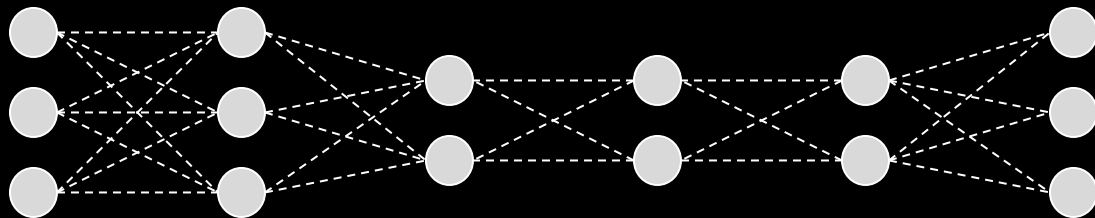


Time



GRAPH-BASED ASSOCIATION

- ✗ Feature extraction is done independently from the optimization problem
- ✗ Optimization can be expensive (depends on the graph connectivity)



Input frames and
object detections

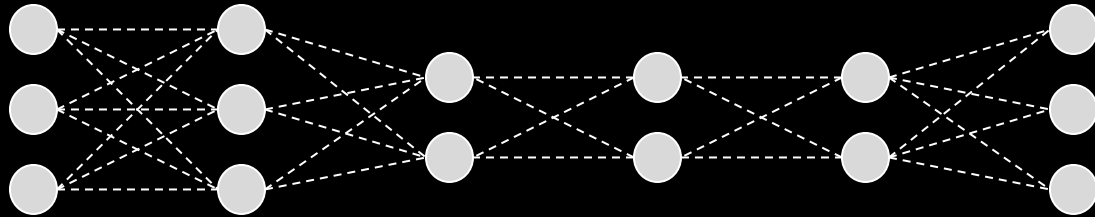


Time



GNN-BASED ASSOCIATION

- Solution: more machine learning!



Input frames and
object detections

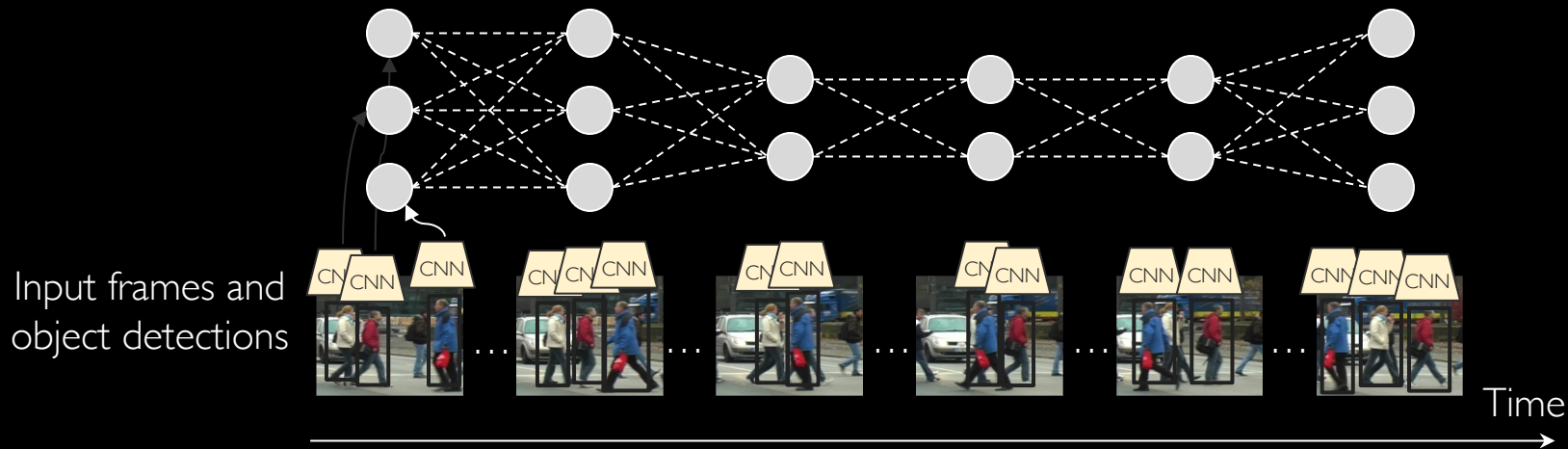


Time



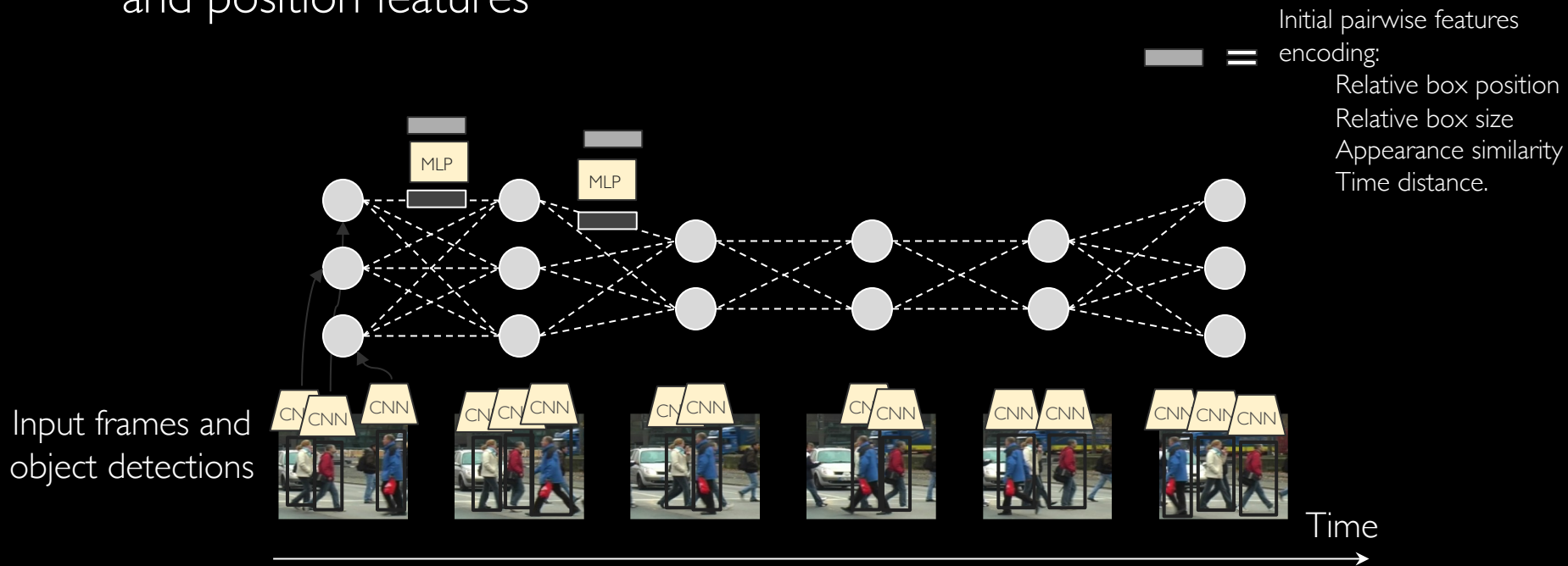
GNN-BASED ASSOCIATION

- Node embeddings are obtained from a CNN



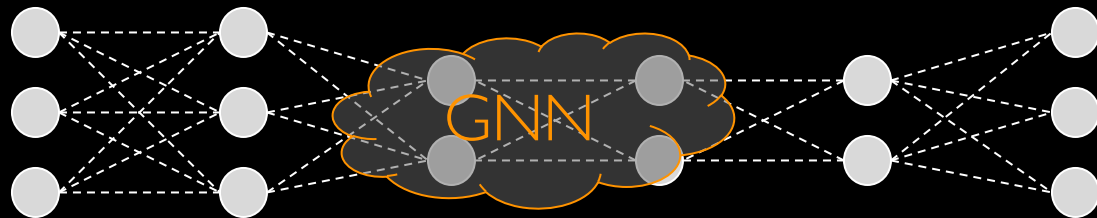
GNN-BASED ASSOCIATION

- Node embeddings are obtained from a CNN
- Edge embeddings are obtained from an MLP operating on appearance and position features



GNN-BASED ASSOCIATION

- A graph neural network (GNN) can be used to propagate node and edge embeddings over the graph



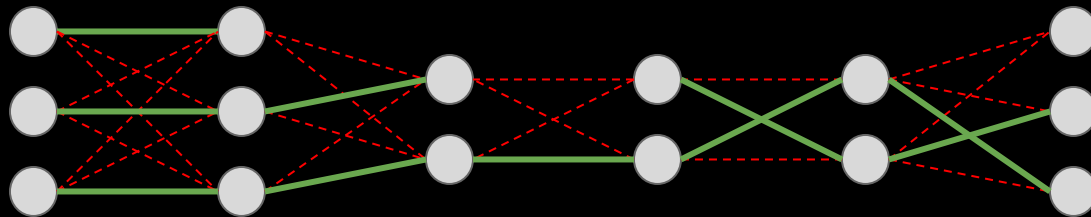
Input frames and
object detections



Time

GNN-BASED ASSOCIATION

After neural message passing, edge embeddings are classified into correct and incorrect track hypotheses



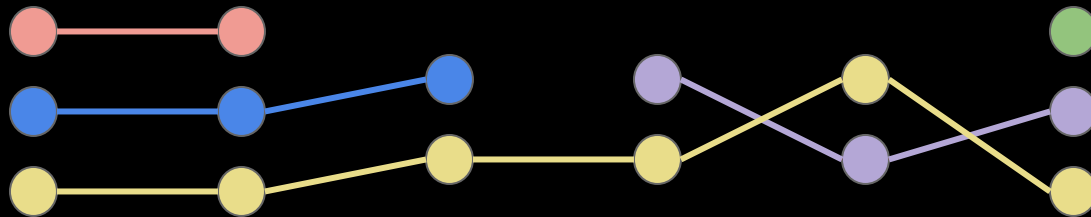
Input frames and
object detections



Time



GNN-BASED ASSOCIATION



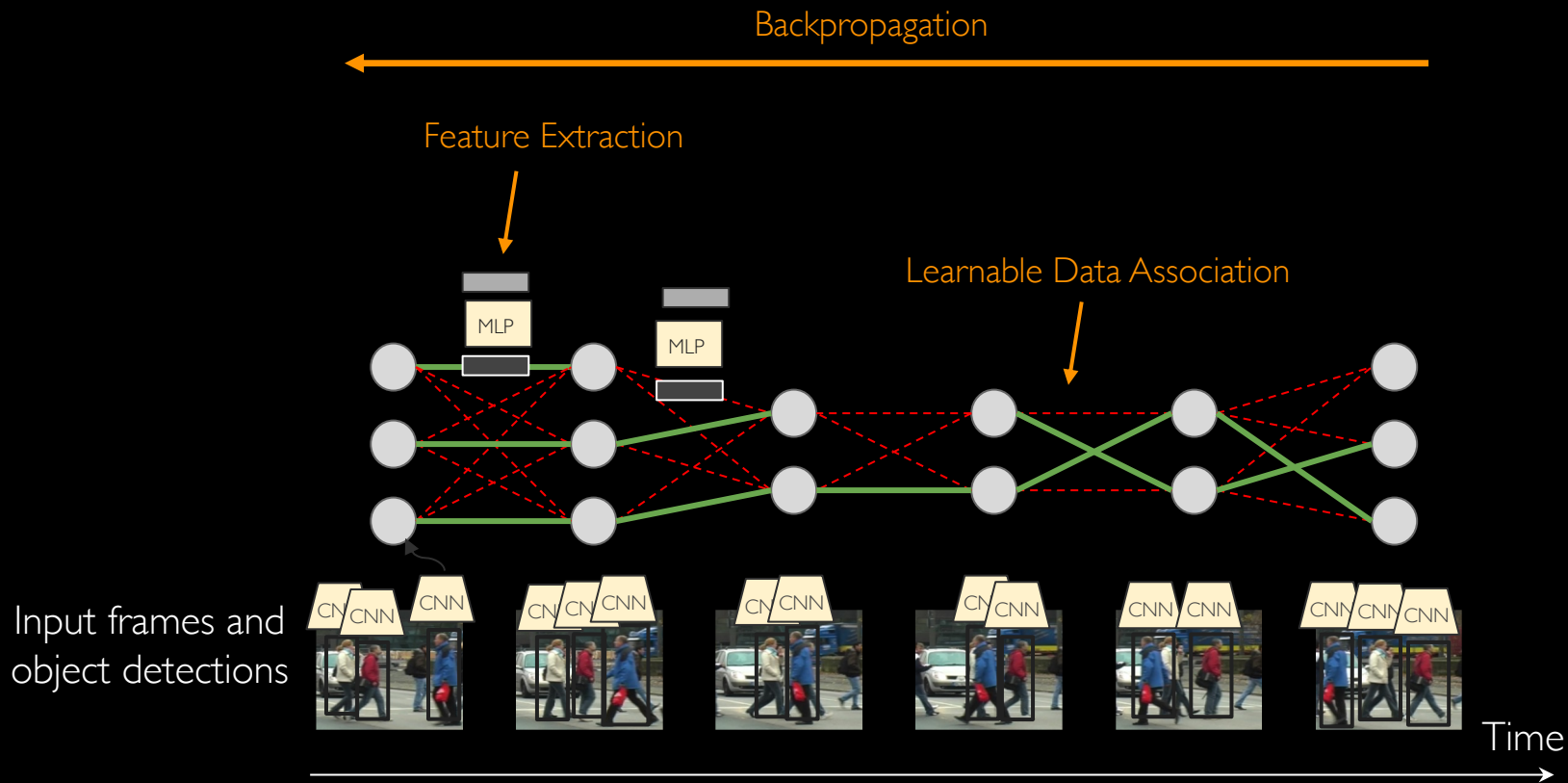
Input frames and
object detections



Time

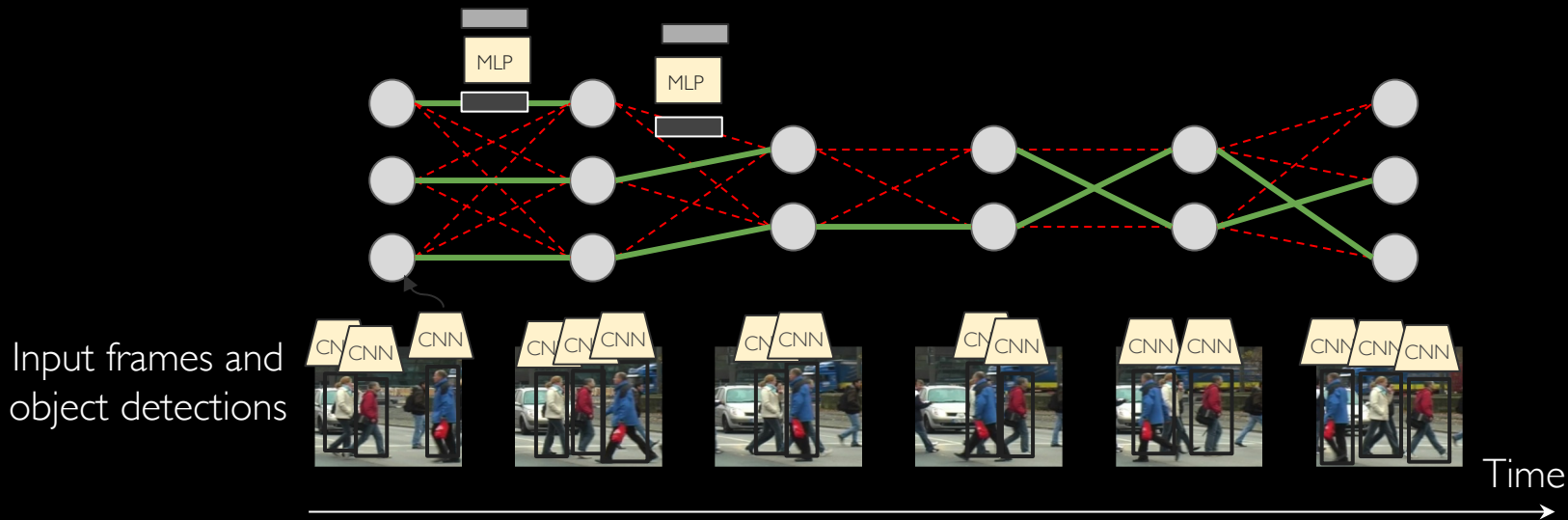


ADVANTAGES OF GNNS



ADVANTAGES OF GNNS

- We can directly work in the MOT domain (graph)
- Learn features specifically for the task and the graph structure
- Avoid the need of expensive optimization at test time

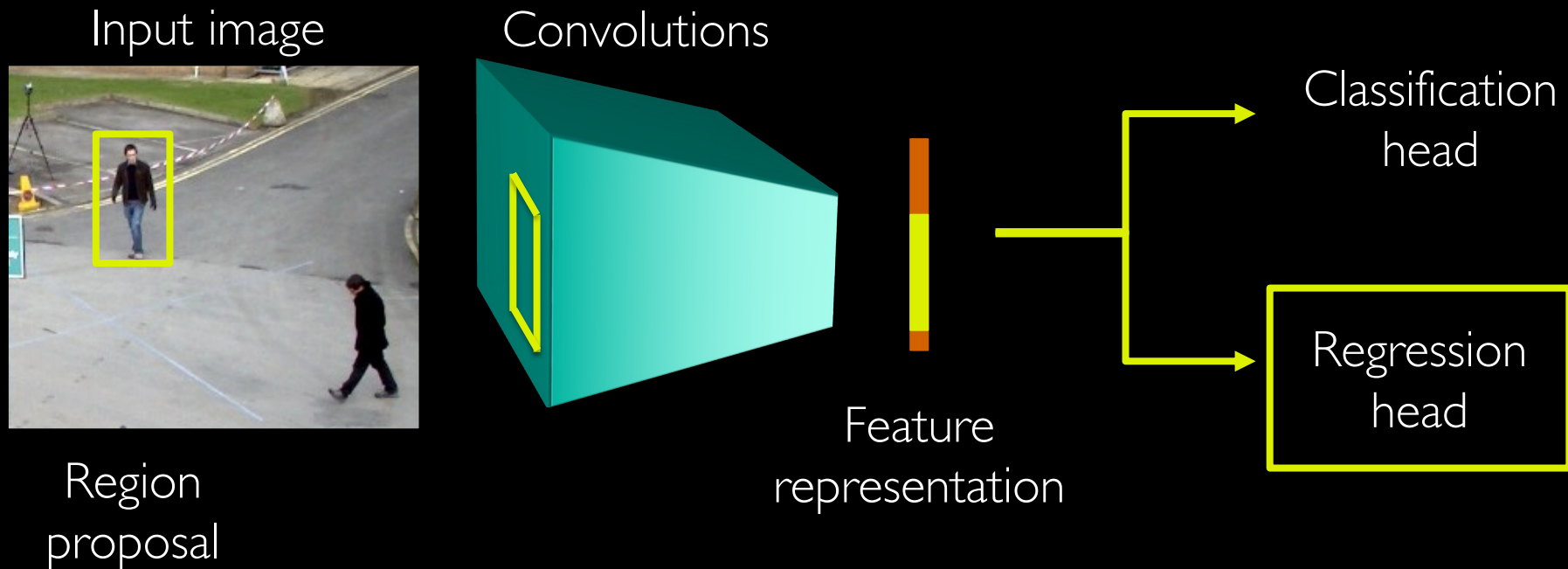


TRACKING-BY-DETECTION

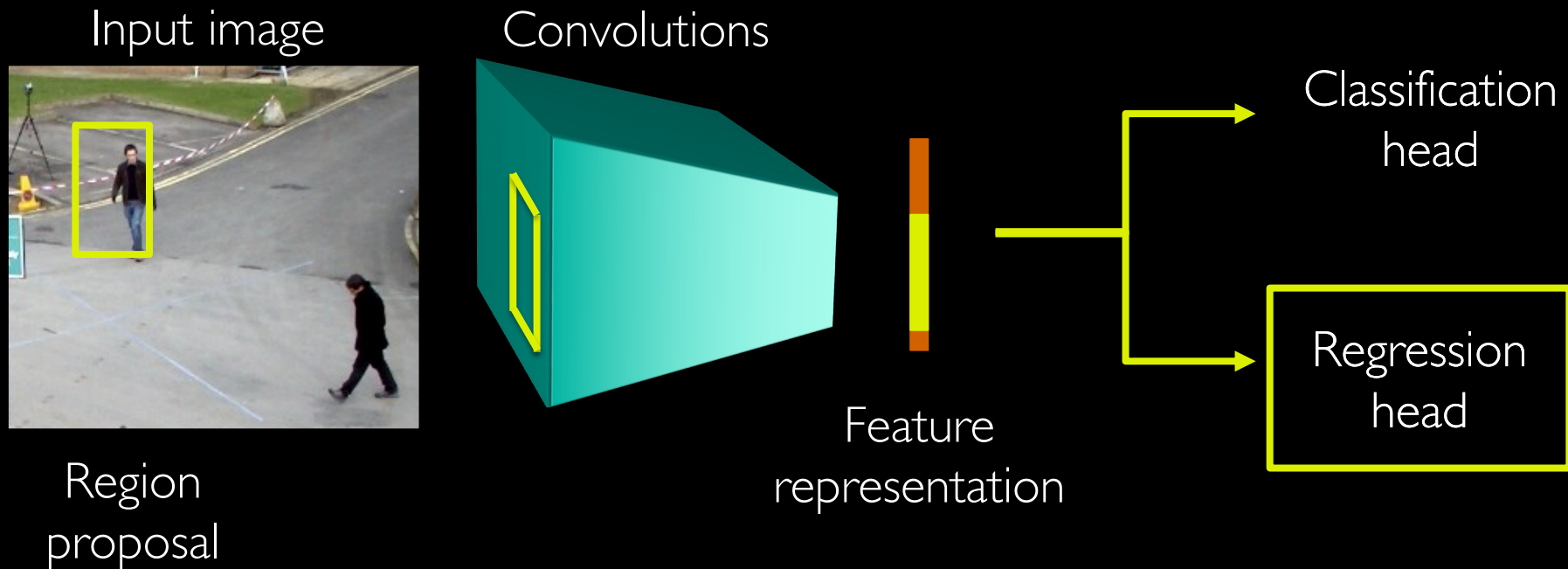
- Tracking-by-detection:
 - Most common paradigm until recently, leverages well the advances in object detection
 - It can be used online (Hungarian) + by batches (adding computational complexity)
+ learning-based solution with GNNs
- We still need detections to compute the graph → no end-to-end learning

TRACKING-BY- REGRESSION

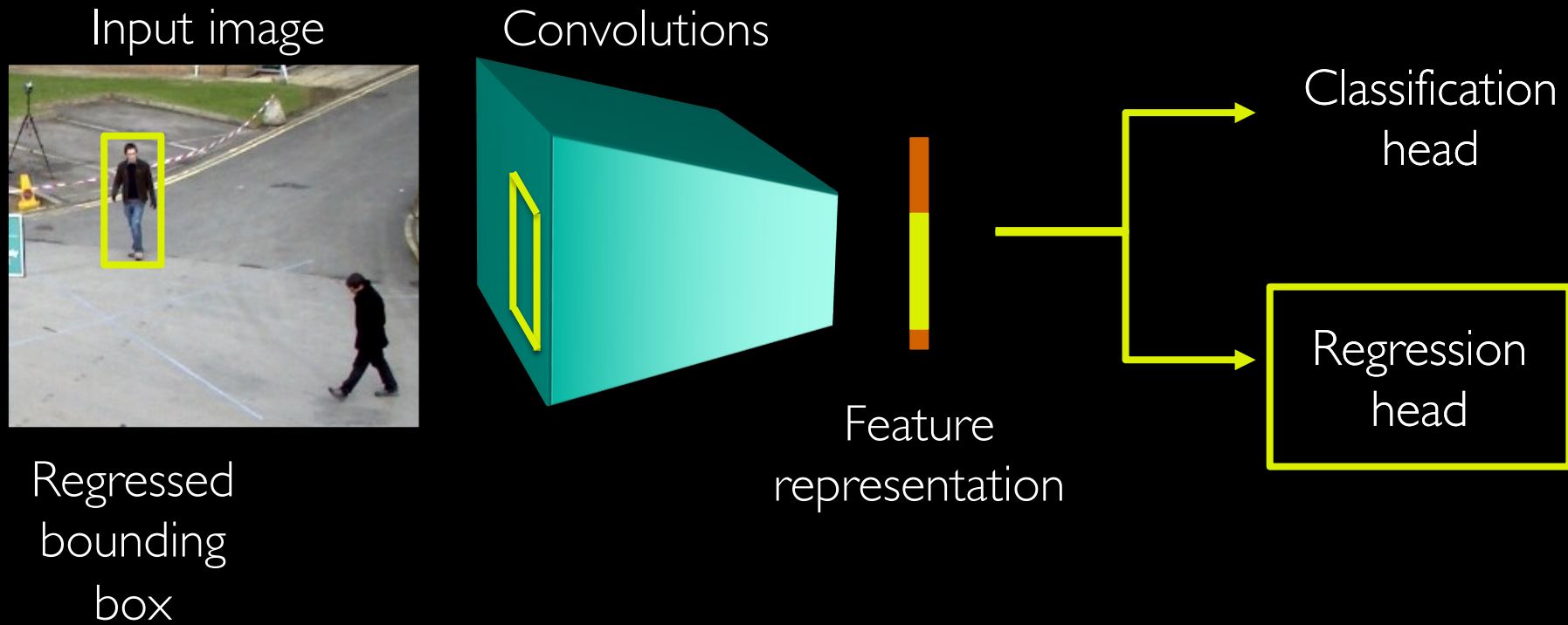
REGRESSION-BASED DETECTORS



REGRESSION-BASED DETECTORS



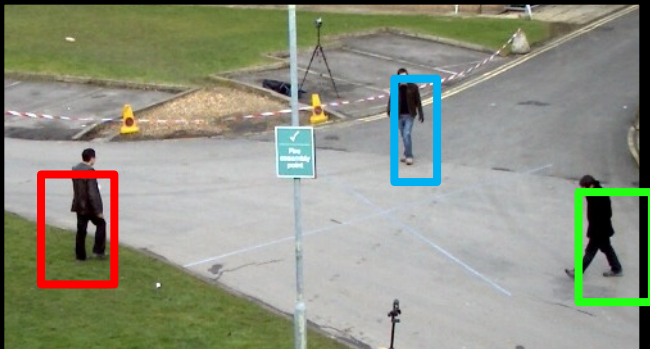
REGRESSION-BASED DETECTORS



FROM DETECTOR TO TRACKTOR

- This is very similar to what we want to do in online tracking
- **Tracktor**: a method trained as a detector but with tracking capabilities

FROM DETECTOR TO TRACKTOR

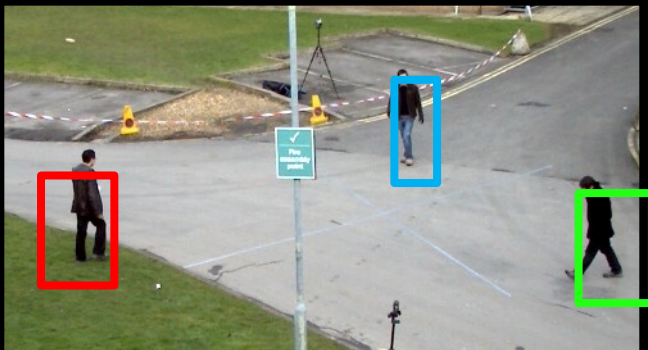


Frame $t+1$

Use detections of frame t as proposals

FROM DETECTOR TO TRACKTOR

Bounding box
regression



Where did the detection with ID 1 go in the next frame?

✓ Tracking!

PROS AND CONS

- **PRO** We can reuse an extremely well-trained regressor
 - We get well-positioned bounding boxes
- **PRO** We can train our model on still images → easier annotation!
- **PRO** Tracktor is online

TRACKING-BY-REGRESSION

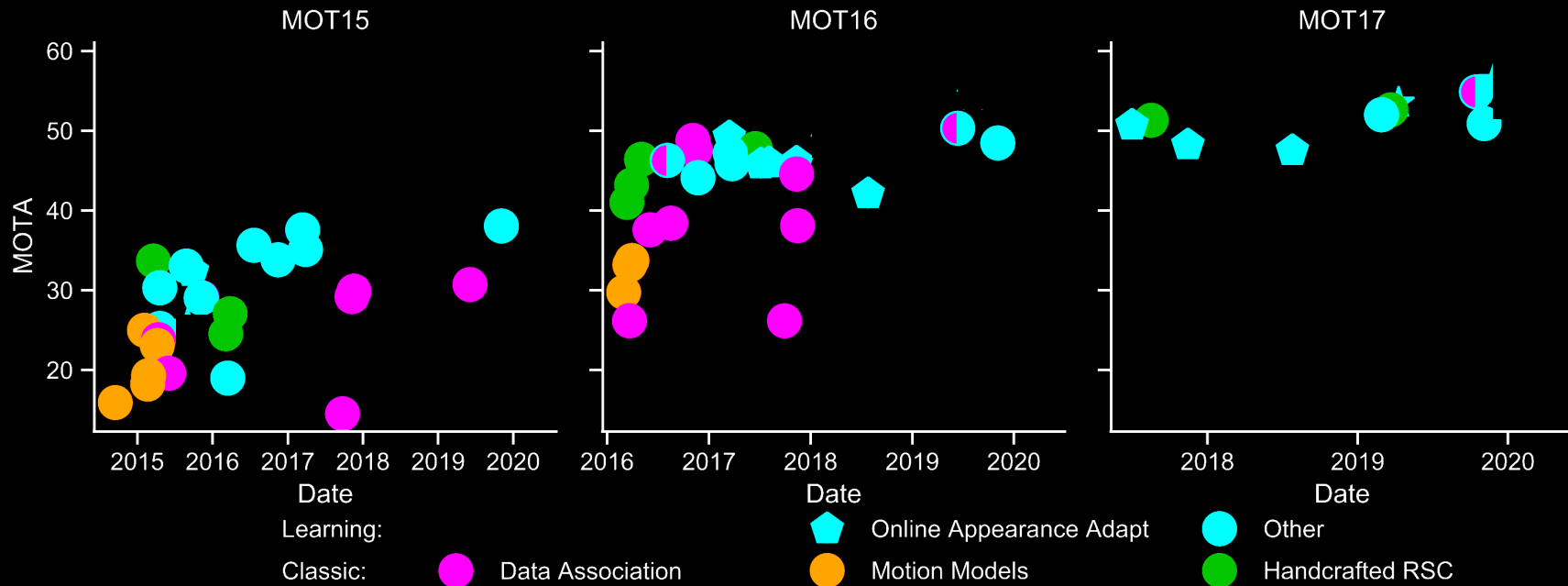


- A step towards merging tracking and detection tasks
- Exploit detection regression
 - a. Bounding boxes with Tracktor [1]
 - b. Center points with CenterTrack [2] – heatmap prediction
- Spatial > Appearance cues

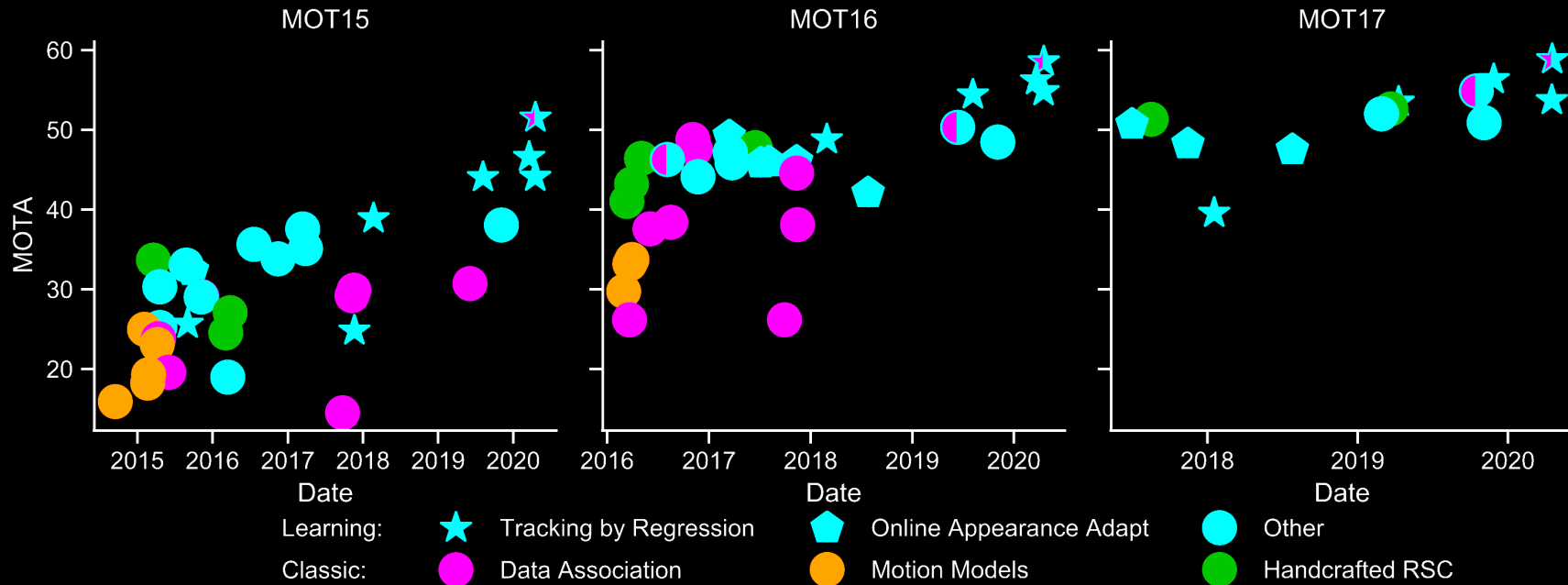
[1] Bergmann, Meinhardt and Leal-Taixé. *Tracking without bells and whistles*. ICCV 2019.

[2] Zhou et al. *Tracking Objects as Points*. ECCV 2020.

A HISTORIC VIEW

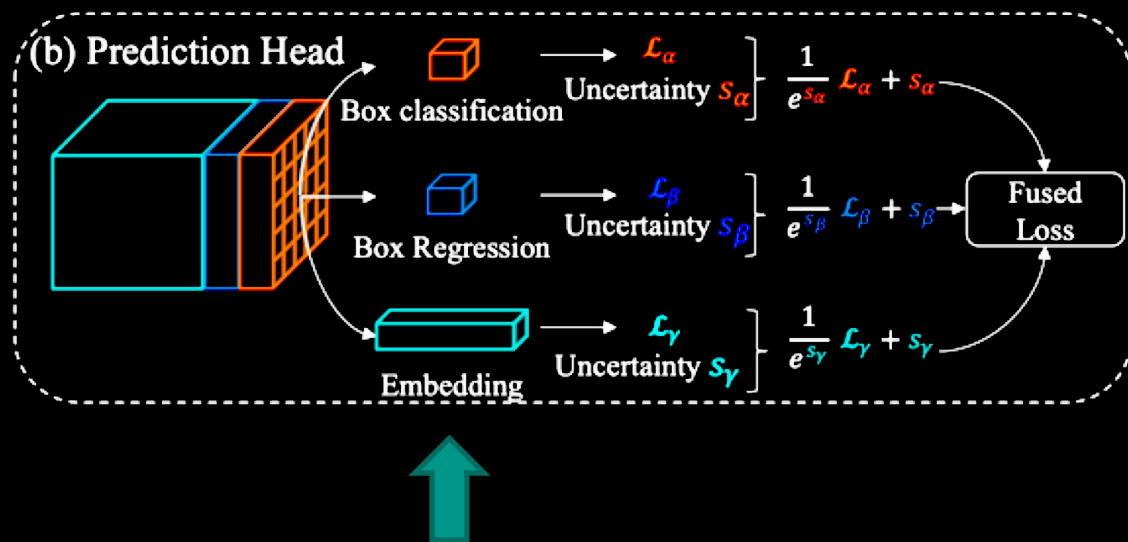


A HISTORIC VIEW



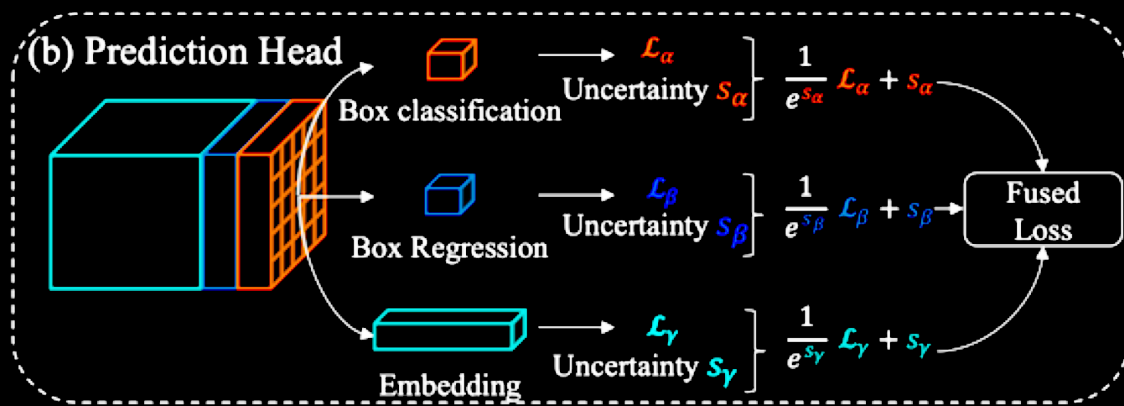
TOWARDS UNIFYING DETECTION AND TRACKING

- Option 1: Joint detection and association embedding prediction (JDE)



TOWARDS UNIFYING DETECTION AND TRACKING

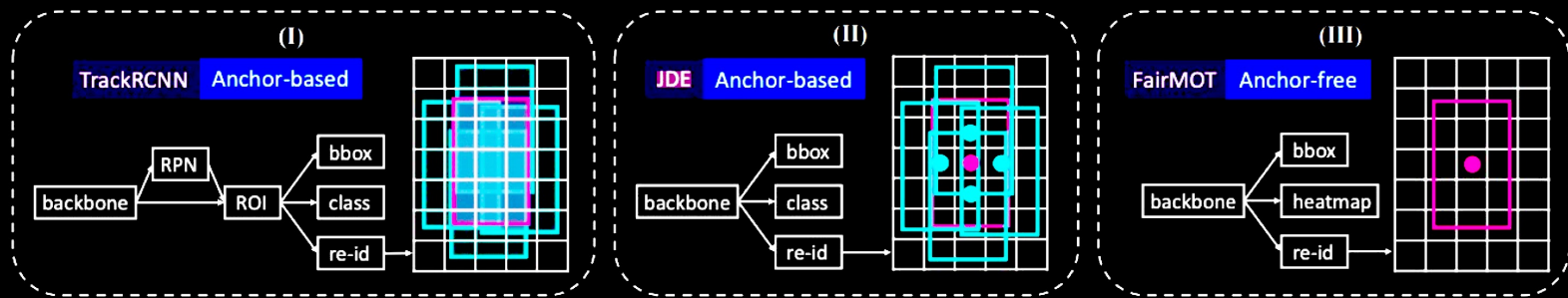
- Option 1: Joint detection and association embedding prediction (JDE)



- Heuristic association via embedding distance
- Near real-time (shared backbone)
- Jointly training for detection and tracking but tasks still separate in different heads

TOWARDS UNIFYING DETECTION AND TRACKING

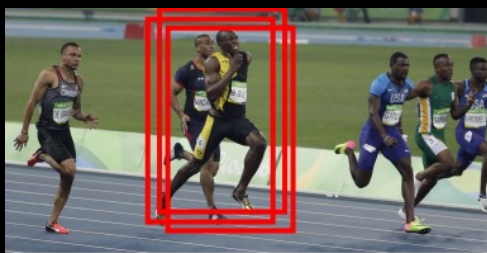
- Option I: Anchor-free JDE (FairMOT) based on CenterNet



(a) Comparison of the existing one-shot trackers and FairMOT



(b) One anchor contains multiple identities



(c) Multiple anchors response for one identity



(d) One point for one identity

TOWARDS UNIFYING DETECTION AND TRACKING

- Option 1: Joint detection and association embedding prediction
 - Anchor-based: JDE
 - Anchor-free: FairMOT
- Option 2?

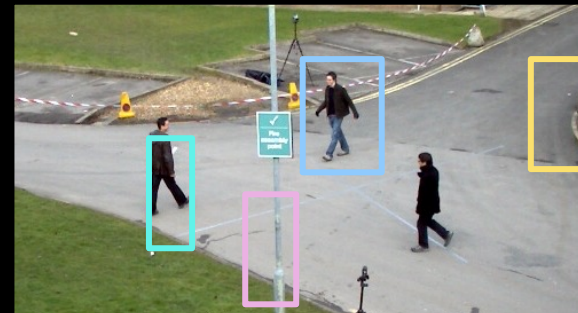
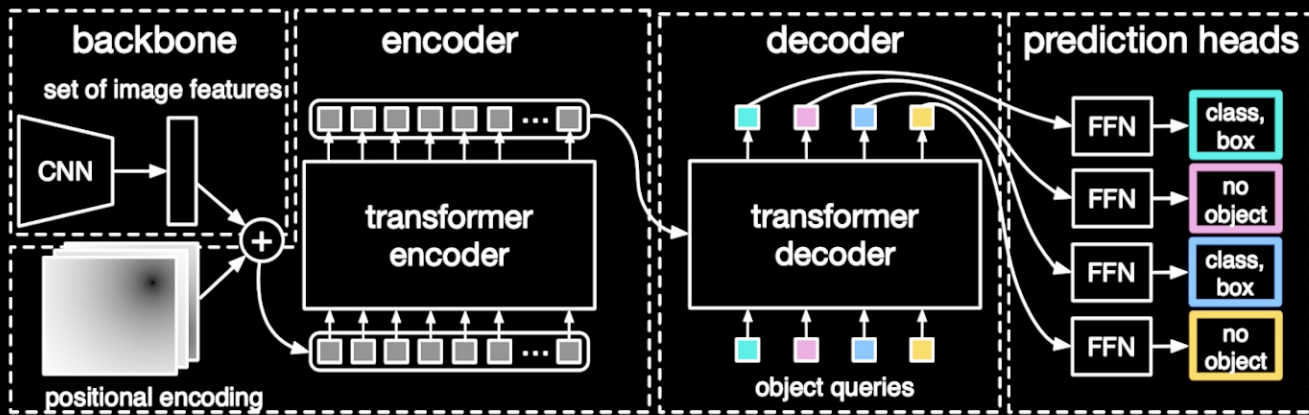
TOWARDS UNIFYING DETECTION AND TRACKING



TRACKING-BY- ATTENTION*

*Attention jointly solves the detection and tracking task.

DETECTION WITH TRANSFORMERS



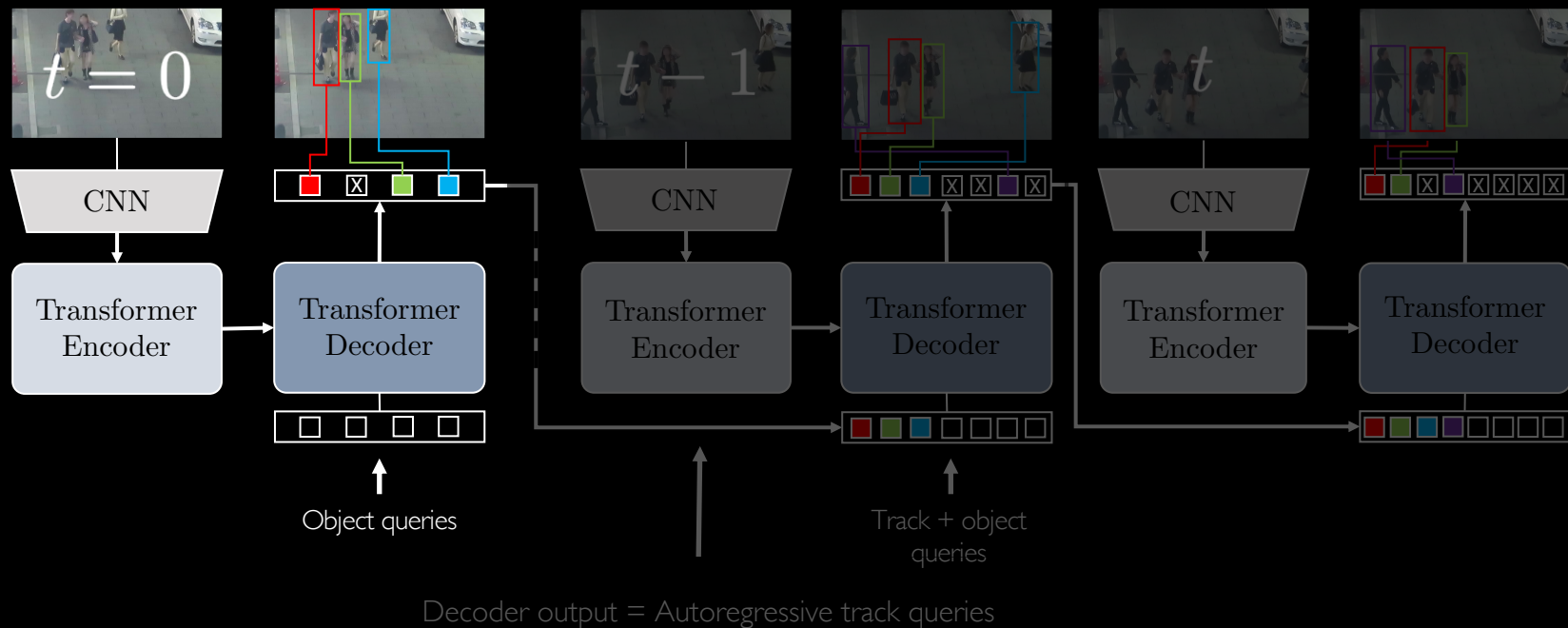
- Object detection a set prediction problem [1, 2]
- Transformer decoder
 - Object query self-attention ($\{\text{class, box}\}$ or no-object)
 - Encoded image feature and object query cross attention

[1] Carion et al. *End-to-End Object Detection with Transformers*. ECCV, 2020.

[2] Zhu et al. *Deformable DETR: Deformable transformers for end-to-end object detection*. ICLR 2021.

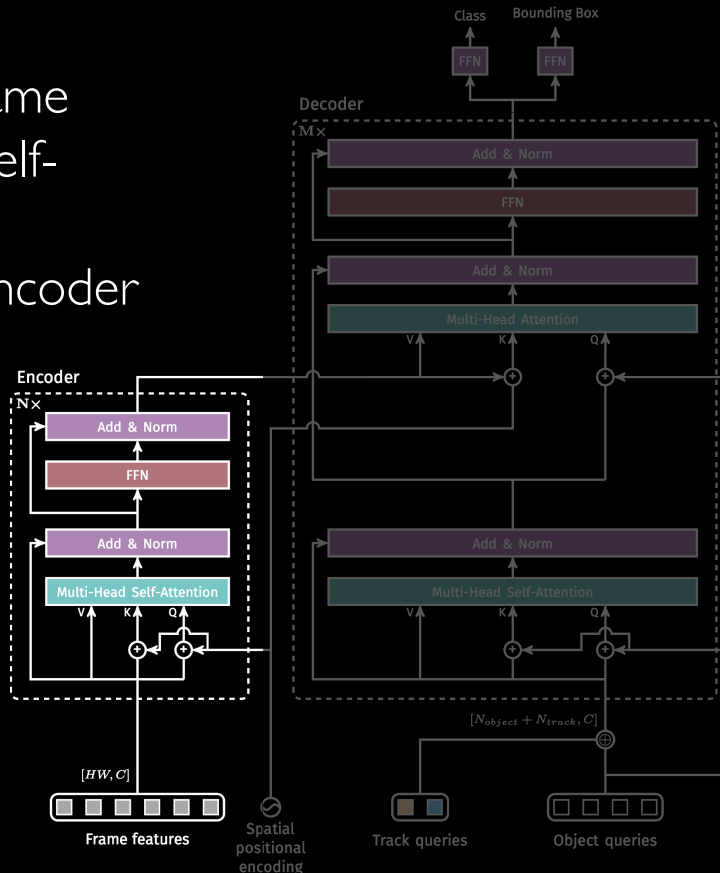
TRACKFORMER

- MOT as a frame-to-frame set prediction problem



ENCODER-DECODER TRANSFORMERS

Encoding of frame features with self-attention in a Transformer encoder



ENCODER

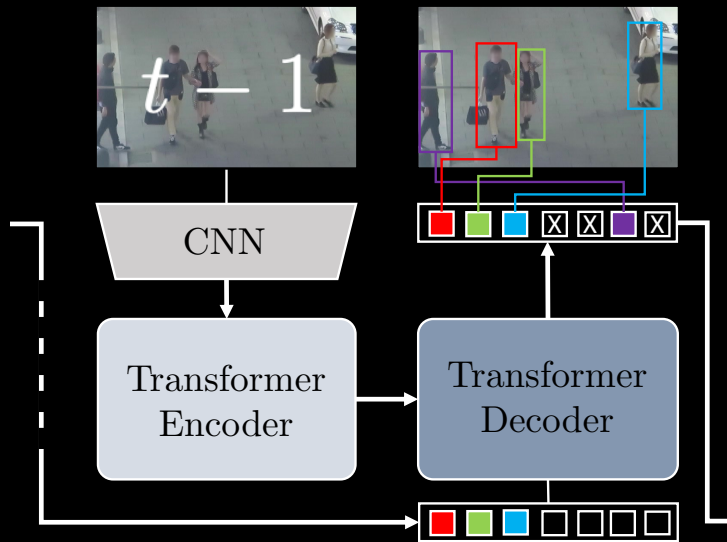
DECODER

Mapping of queries to box and class predictions using MLPs

Self- and encoder-decoder attention

Concatenation of object and track queries.

TRANSFORMER QUERY DECODING



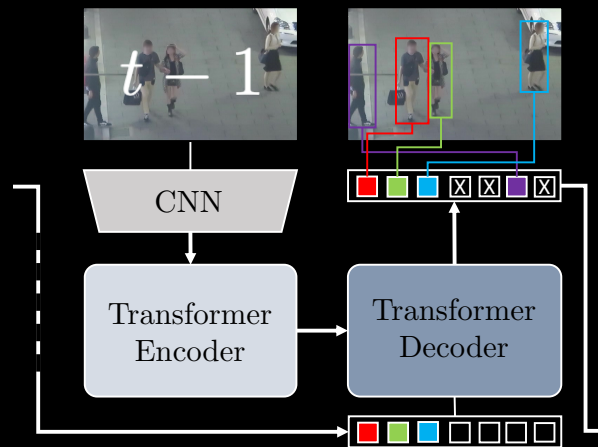
1. Self-attention between queries
 - a. Initialize new track (object query)
 - b. Terminate occluded track
2. Encoder-decoder attention
 - a. Find new object in frame
 - b. Adjust to changed position of tracks

CAN I RECOVER FROM OCCLUSIONS?

👍 Just keep track queries active for a time window.

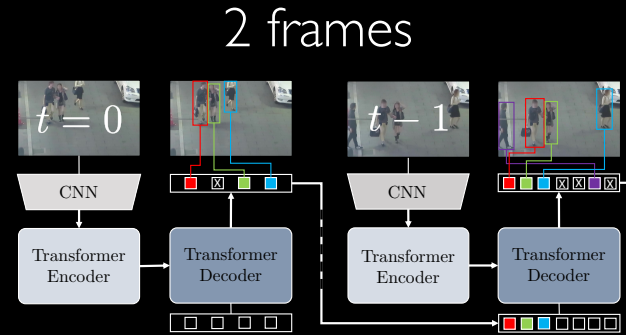
👍 No need to an extra re-ID head.

👉 The spatial information embedded into each track query prevents their application for long-term occlusions.



TRAINING

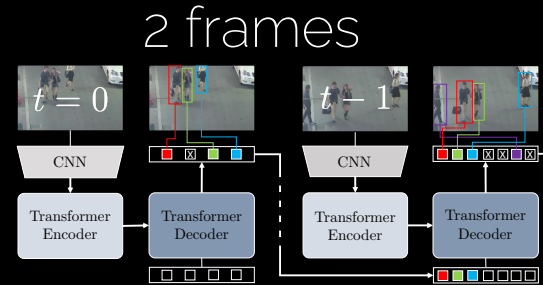
1. Object detection on frame $t - 1$ with N_{object} object queries
2. Tracking of objects from (1.) and detection of new objects on frame t with all $N = N_{\text{object}} + N_{\text{track}}$ queries
3. Assign N predictions to ground truth objects in t
4. Compute set-prediction loss:
 - a. Classification (pedestrian or no-object)
 - b. Bounding box



TRAINING

Prediction-ground truth bipartite matching:

- Box at t comes from a track query, hence, we assign the same ID as in $t - 1$
- Query of $t - 1$ is matched to background class
- New objects never matched with $t - 1$ are matched by classification and box score.



Objects are occluded or leave the scene

New objects enter the scene

ABLATION

Method	MOTA \uparrow Δ	IDF1 \uparrow Δ
TrackFormer	51.4	55.3
————— w/o —————		
Pretraining on CrowdHuman	42.8 -8.6	45.2 -10.1
Track query re-identification	42.7 -0.1	43.6 -1.6
Track augmentations (FP)	40.1 -2.6	42.9 -0.7
Track augmentations (Range)	38.1 -2.0	41.0 -1.9
Track queries	37.8 -0.3	27.4 -13.6

Simulated tracking data

$t - 1$

t



TRACKFORMER 🍌

- Elegant formulation of tracking which naturally merges detection and data association
- Good performance with partial occlusions
- Good performance where detectors are weak
- State-of-the-art results (with some data and some tricks)
- Similar concurrent papers: MeMOT (ECCV22) and MOTR (CVPR22)

TRACKFORMER 🙄

- Training such a model is not straightforward and requires A LOT of data → MOTChallenge is not enough
- Unclear how much do these methods generalize, e.g., not seeing any MOTChallenge data hurts performance significantly.









TRACKING-BY-DETECTION

- A tracker that generalizes to diverse tracking conditions
- That does not require vast amount of training data
- To the rescue comes...

A SIMPLE ONLINE TRACKER

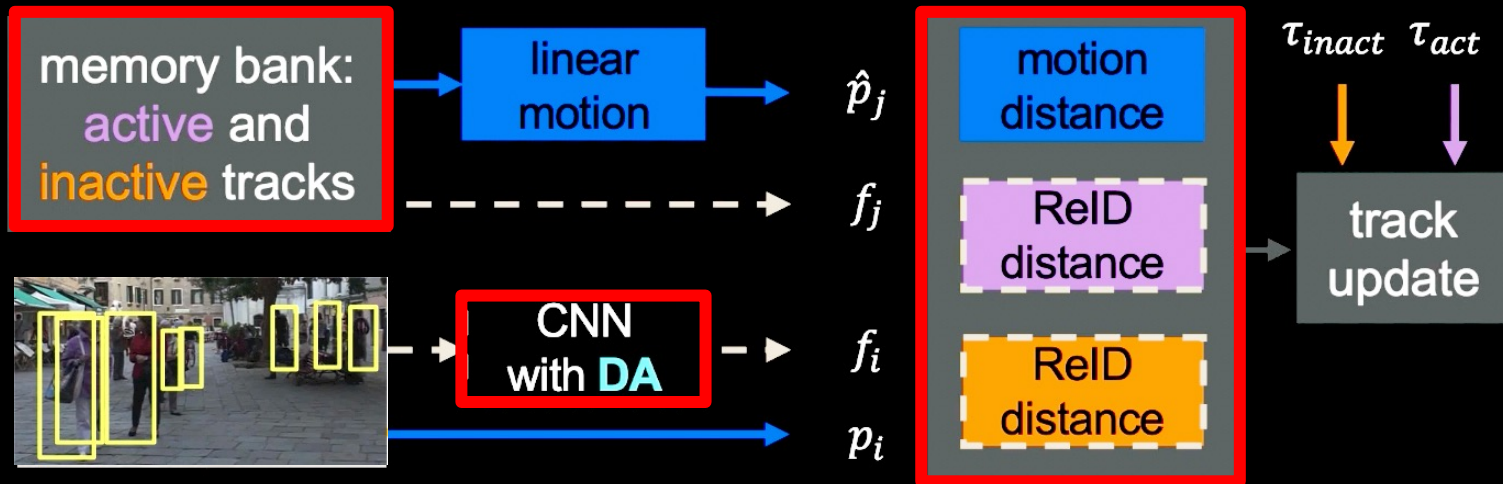
- Bipartite matching
 - Define distances between boxes (e.g., IoU, pixel distance, 3D distance, reID)
 - Solve the unique matching with e.g., the Hungarian algorithm



				
	0.9	0.8	0.8	0.1
	0.5	0.4	0.3	0.8
	0.2	0.1	0.4	0.8
	0.1	0.2	0.5	0.9

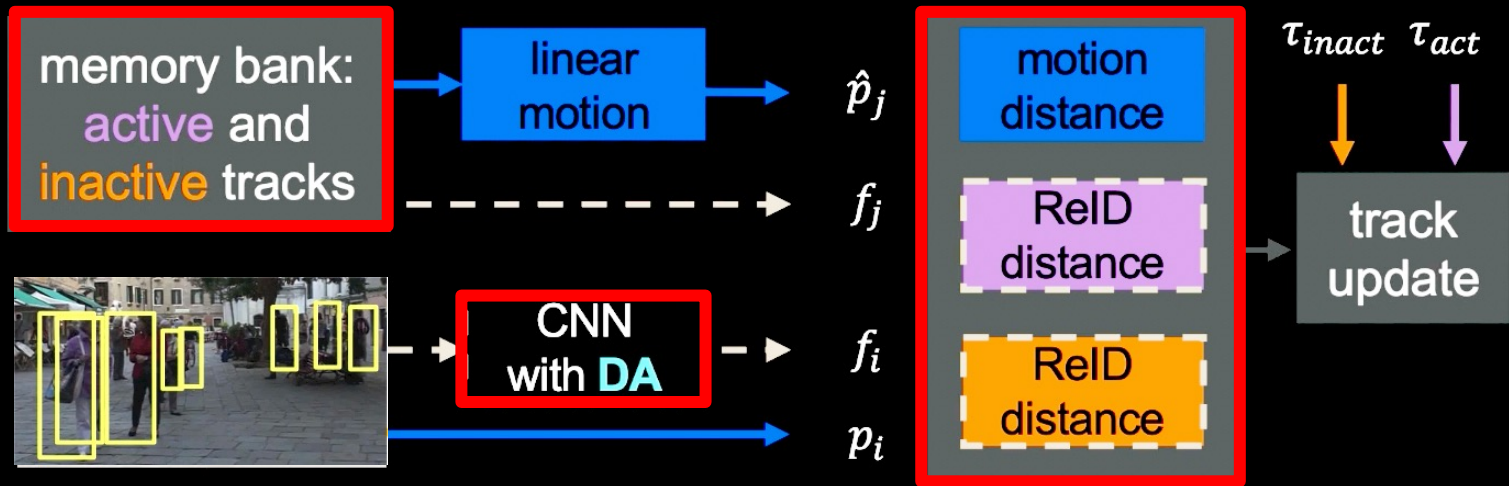
A SIMPLE ONLINE TRACKER

- We need to pay attention to details



A SIMPLE ONLINE TRACKER

- We need to pay attention to details

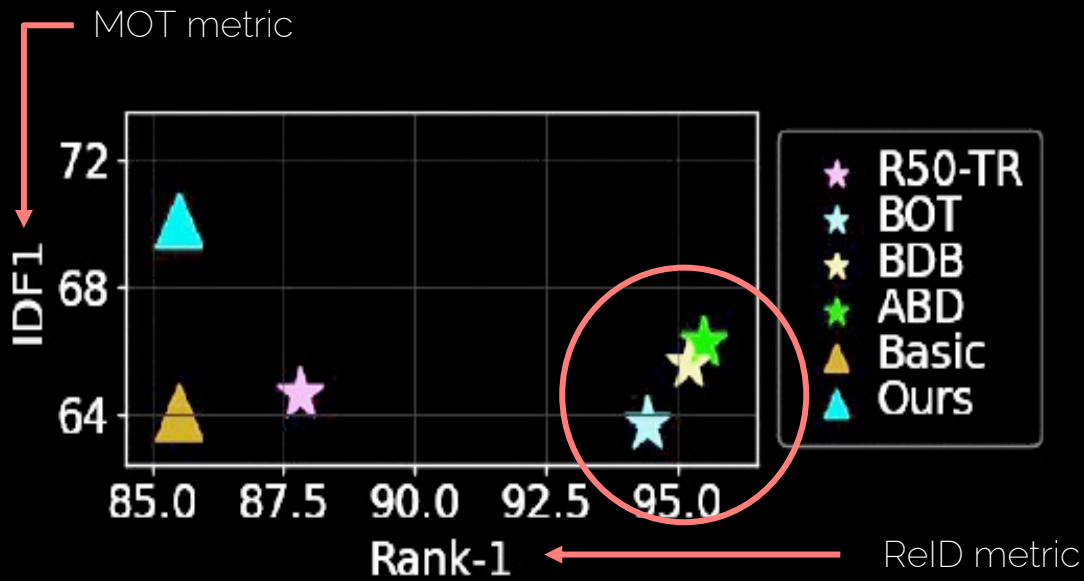


- GHOST: Good Old Hungarian Simple Tracker



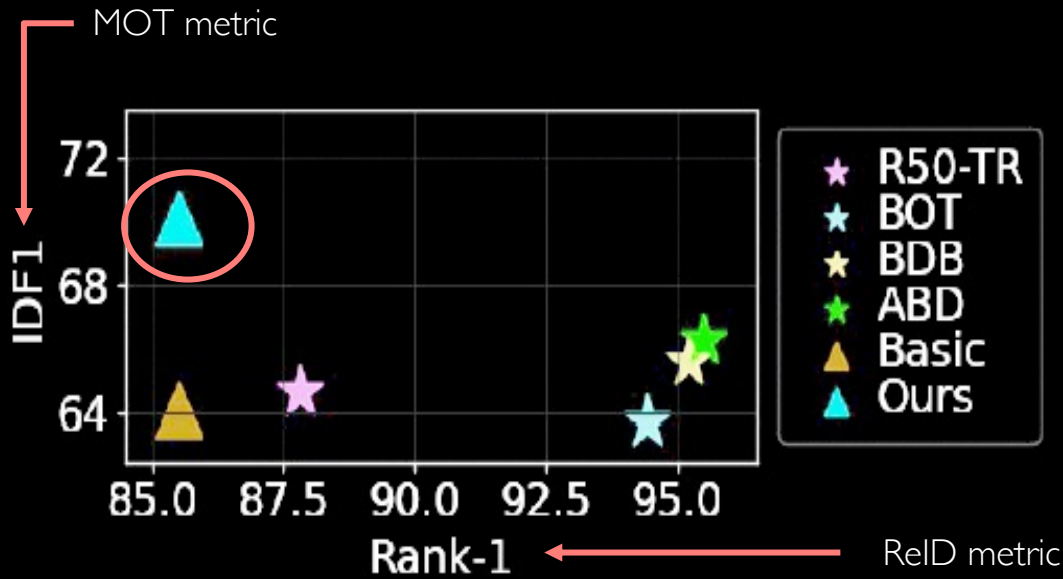
ReID FEATURES DO NOT WORK FOR MOT

- Appearance networks are typically trained on ReID datasets, but the distribution of appearances for MOT is not the same



ReID FEATURES DO NOT WORK FOR MOT

- Appearance networks are typically trained on ReID datasets, but the distribution of appearances for MOT is not the same

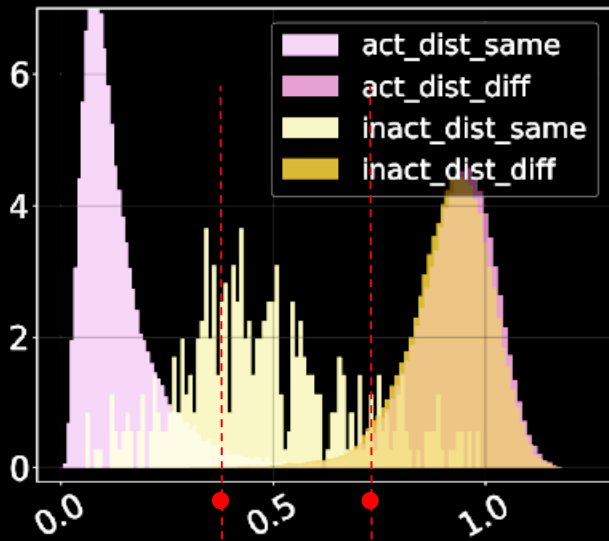


ReID FEATURES DO NOT WORK FOR MOT

- Appearance networks are typically trained on ReID datasets, but the distribution of appearances for MOT is not the same
- Solution: adapt to the MOT statistics by re-weighting the Batch Normalization layers of our ReID network.
- Mean and variance of the features obtained from the detections of the current frame.

ACTIVE AND INACTIVE TRACKS

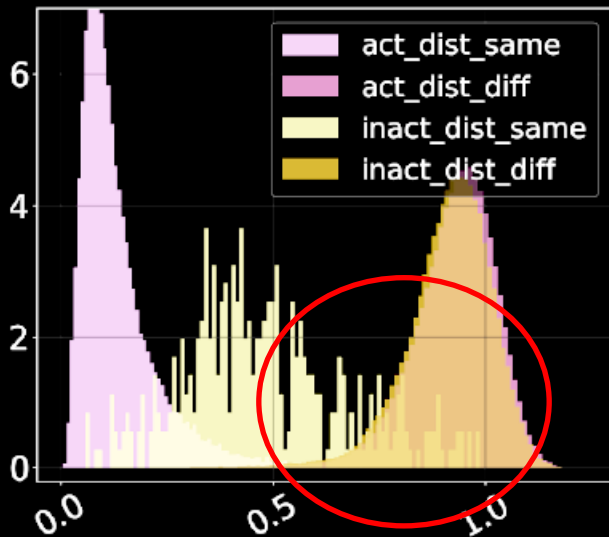
- Pink: active tracks . Yellow inactive tracks.
 - Light pink/yellow (left): reID distance between two boxes with the same ID
 - Dark pink/yellow (right) reID distance between two boxes with different IDs



Different intersection points for active and inactive tracks → different matching thresholds

ACTIVE AND INACTIVE TRACKS

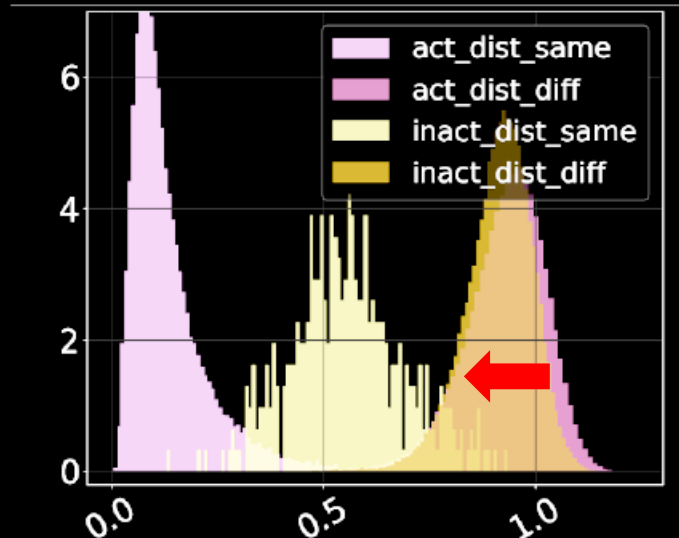
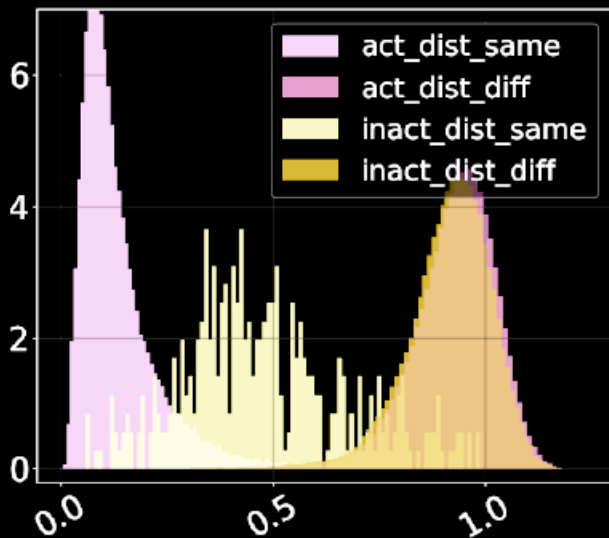
- Pink: active tracks . Yellow inactive tracks.
 - Light pink/yellow (left): reID distance between two boxes with the same ID
 - Dark pink/yellow (right) reID distance between two boxes with different IDs



Inactive tracks do not have a good separation → need to compute another type of distance


ACTIVE AND INACTIVE TRACKS

- Pink: active tracks . Yellow inactive tracks.
 - Light pink/yellow (left): reID distance between two boxes with the same ID
 - Dark pink/yellow (right) reID distance between two boxes with different IDs

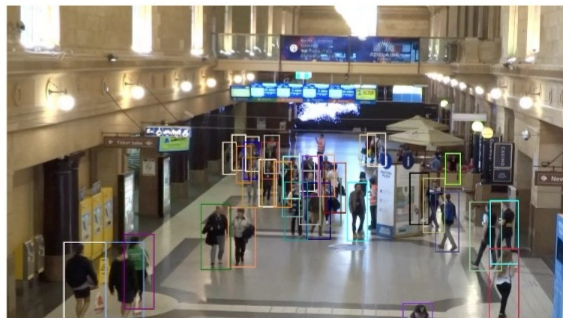
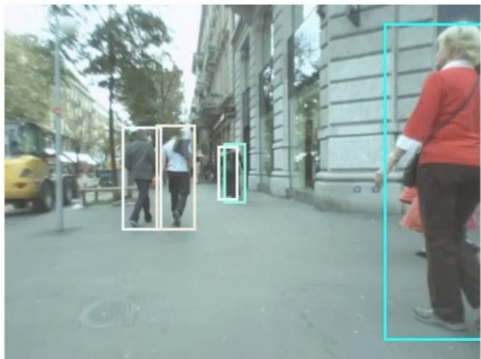
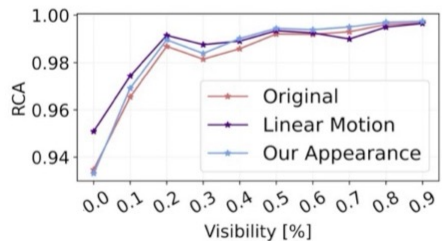
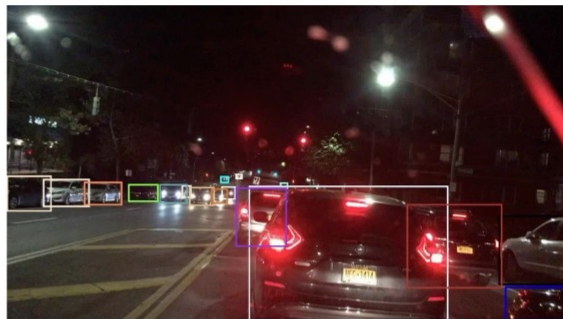
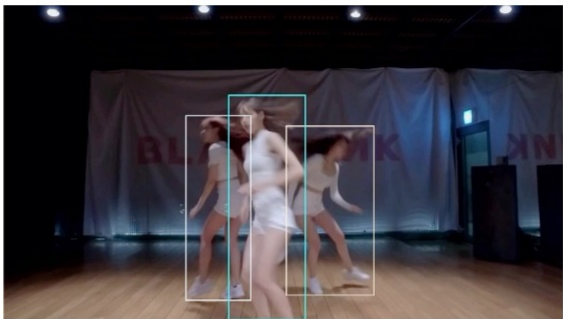
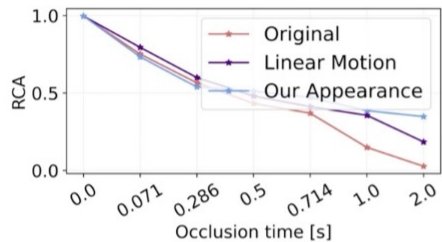


Proxy

A SIMPLE ONLINE TRACKER

- GHOST: Good Old Hungarian Simple Tracker 
- Simplicity strikes back:
 - Frame-by-frame tracker → FAST
 - No need to train on ANY tracking sequence!
 - Generalization to 4 benchmarks

GHOST Excels at Long-Term and Occlusion



SHIFTING PARADIGMS



Tracking-by-detection ^{+GNN}

Tracking-by-regression

Tracking-by-attention

Tracking-by-detection

- TbD for online tracking + offline graph tracking (optimization or learned)
- Show that a detector can be turned into a tracker
- Merging detection and tracking for an end-to-end solution: Trackformer
- GHOST: show that Hungarian-based tracker can still rule all

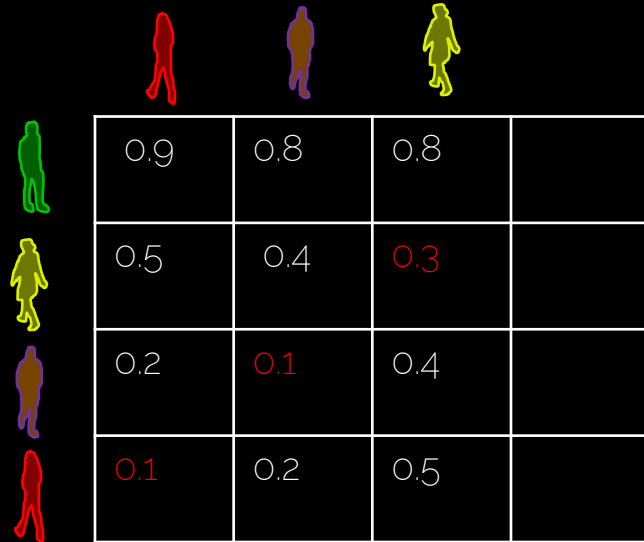
An aerial view of a city street with various objects and people. The image is overlaid with numerous colorful rectangular bounding boxes of different colors (red, green, blue, yellow, purple, black) that represent object detection results. The text "THANK YOU (& QUESTIONS!)" is centered in the image. The URL "https://dvl.in.tum.de" is located below the text.








THANK YOU
(& QUESTIONS!)

<https://dvl.in.tum.de>

A SIMPLE ONLINE TRACKER

- Bipartite matching
 - What happens if we are missing a prediction?



				
	0.9	0.8	0.8	
	0.5	0.4	0.3	
	0.2	0.1	0.4	
	0.1	0.2	0.5	

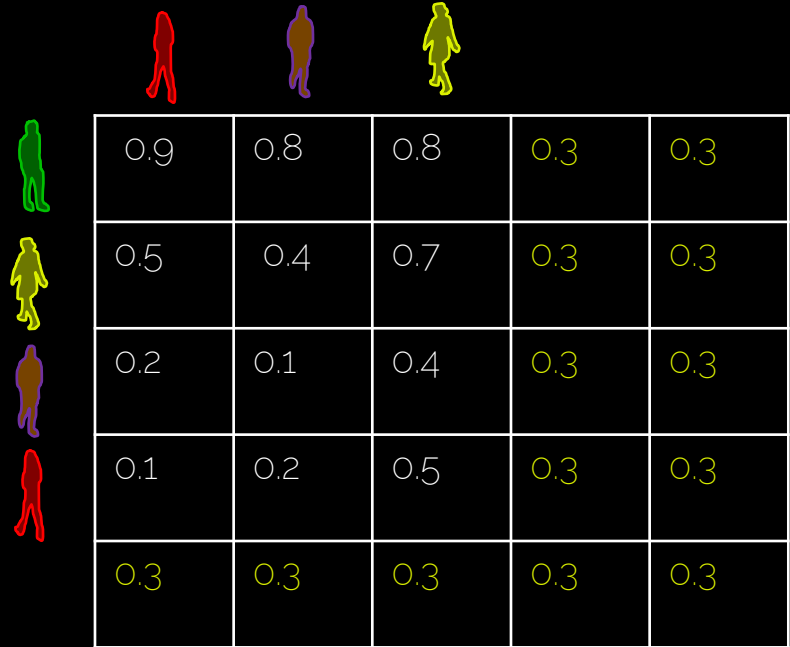
A SIMPLE ONLINE TRACKER

- Bipartite matching
 - What happens if we are missing a prediction?
 - What happens if no prediction is suitable for the match?

0.9	0.8	0.8	
0.5	0.4	0.7	
0.2	0.1	0.4	
0.1	0.2	0.5	

A SIMPLE ONLINE TRACKER

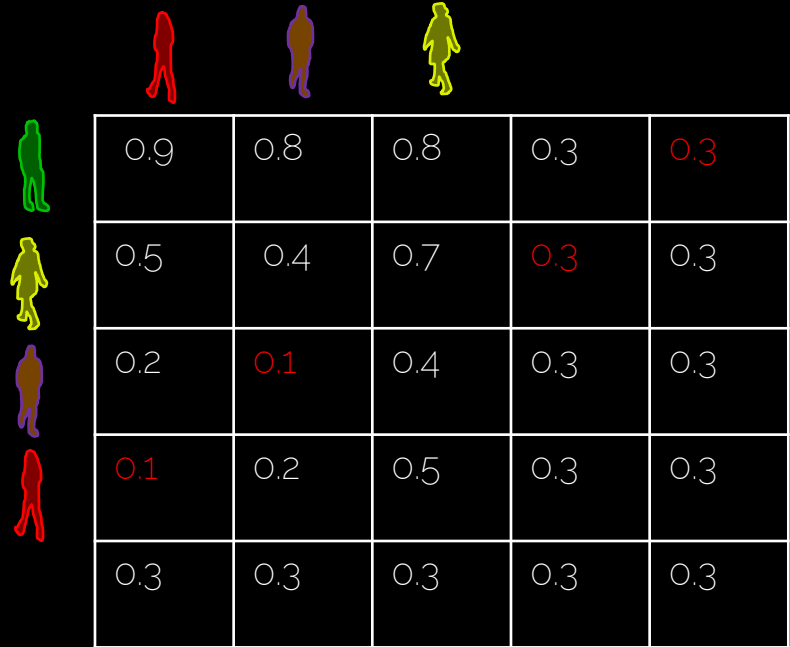
- Bipartite matching
 - What happens if we are missing a prediction?
 - What happens if no prediction is suitable for the match?
 - Introducing extra nodes with a threshold cost



0.9	0.8	0.8	0.3	0.3
0.5	0.4	0.7	0.3	0.3
0.2	0.1	0.4	0.3	0.3
0.1	0.2	0.5	0.3	0.3
0.3	0.3	0.3	0.3	0.3

A SIMPLE ONLINE TRACKER

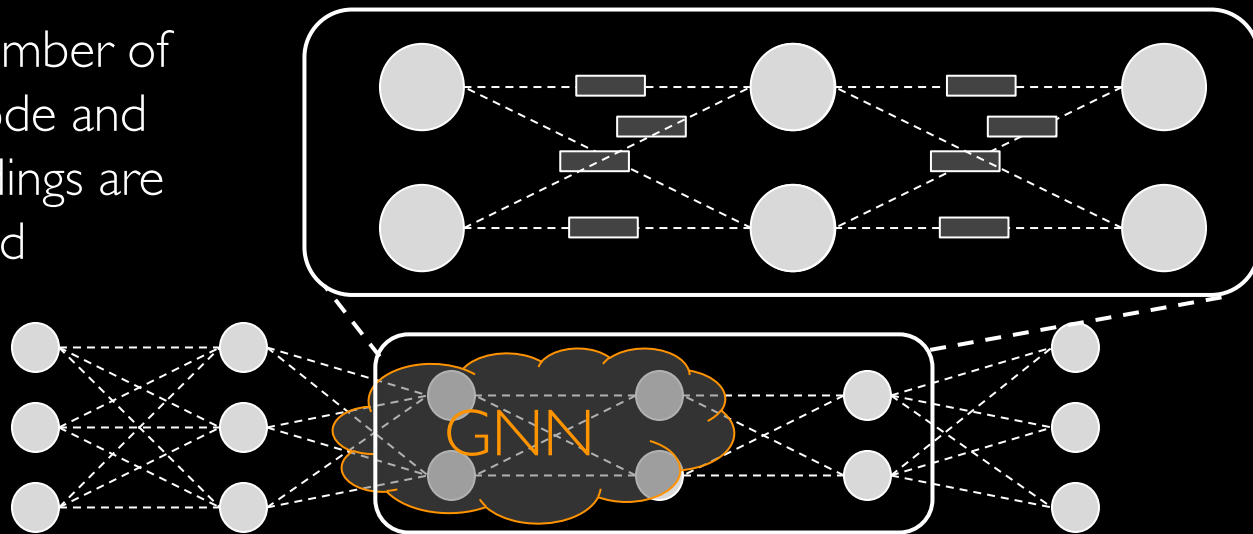
- Bipartite matching
 - What happens if we are missing a prediction?
 - What happens if no prediction is suitable for the match?
 - Introducing extra nodes with a threshold cost
 - Apply Hungarian
 - Result: two detections have no matched prediction



0.9	0.8	0.8	0.3	0.3
0.5	0.4	0.7	0.3	0.3
0.2	0.1	0.4	0.3	0.3
0.1	0.2	0.5	0.3	0.3
0.3	0.3	0.3	0.3	0.3

GNN-BASED ASSOCIATION

For a fixed number of iterations, node and edge embeddings are updated



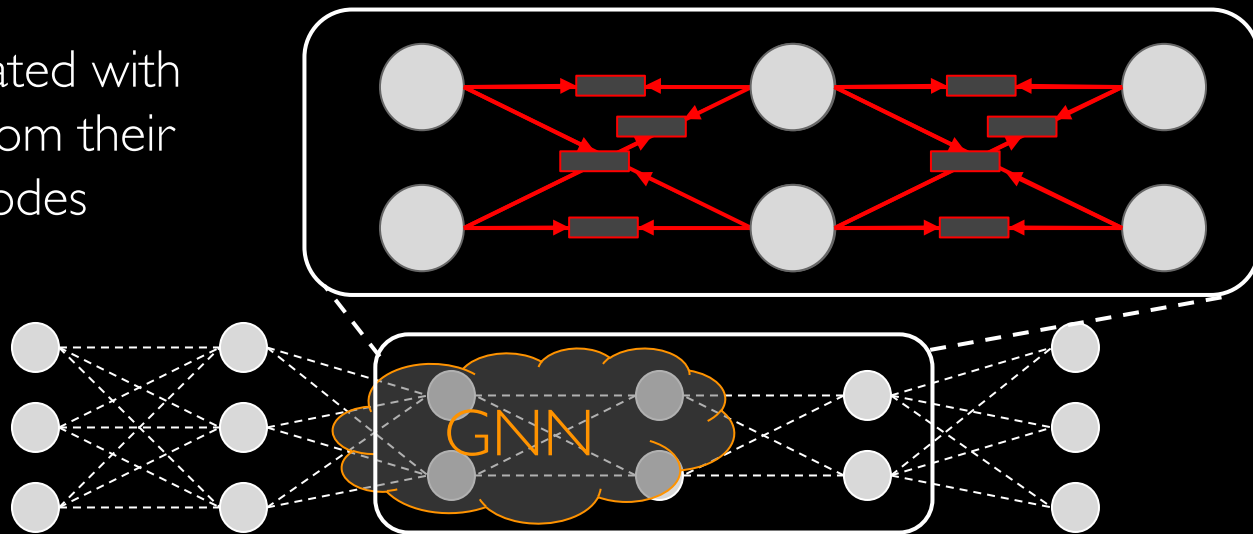
Input frames and object detections



Time

GNN-BASED ASSOCIATION

Edges are updated with embeddings from their incident nodes



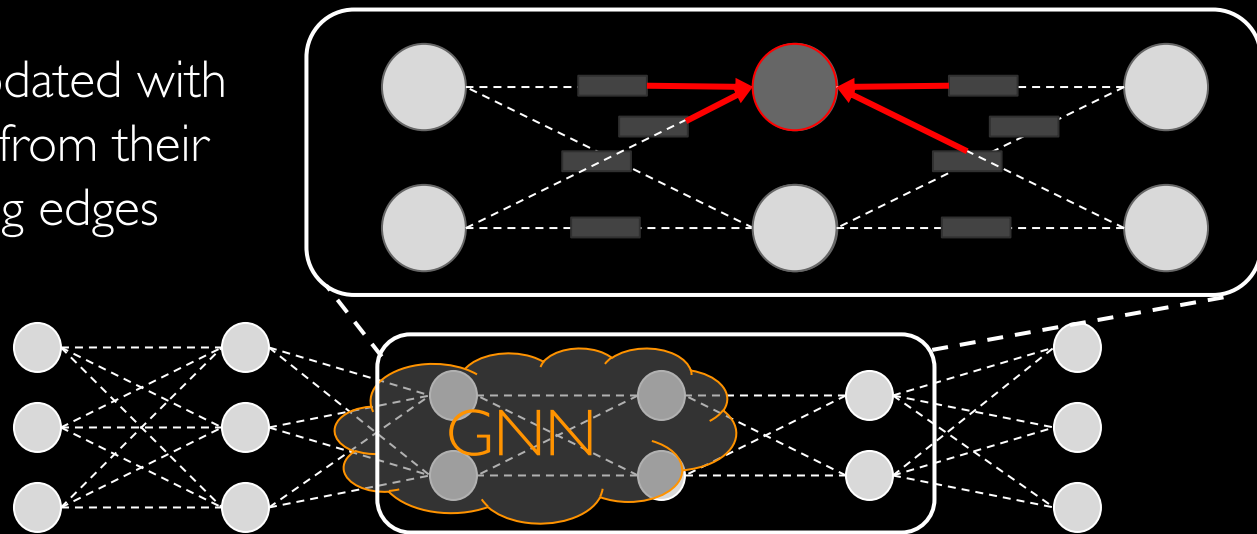
Input frames and object detections



Time

GNN-BASED ASSOCIATION

Nodes are updated with embeddings from their neighboring edges



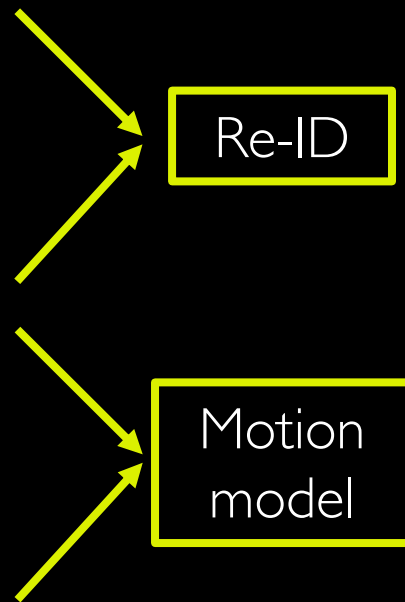
Input frames and object detections



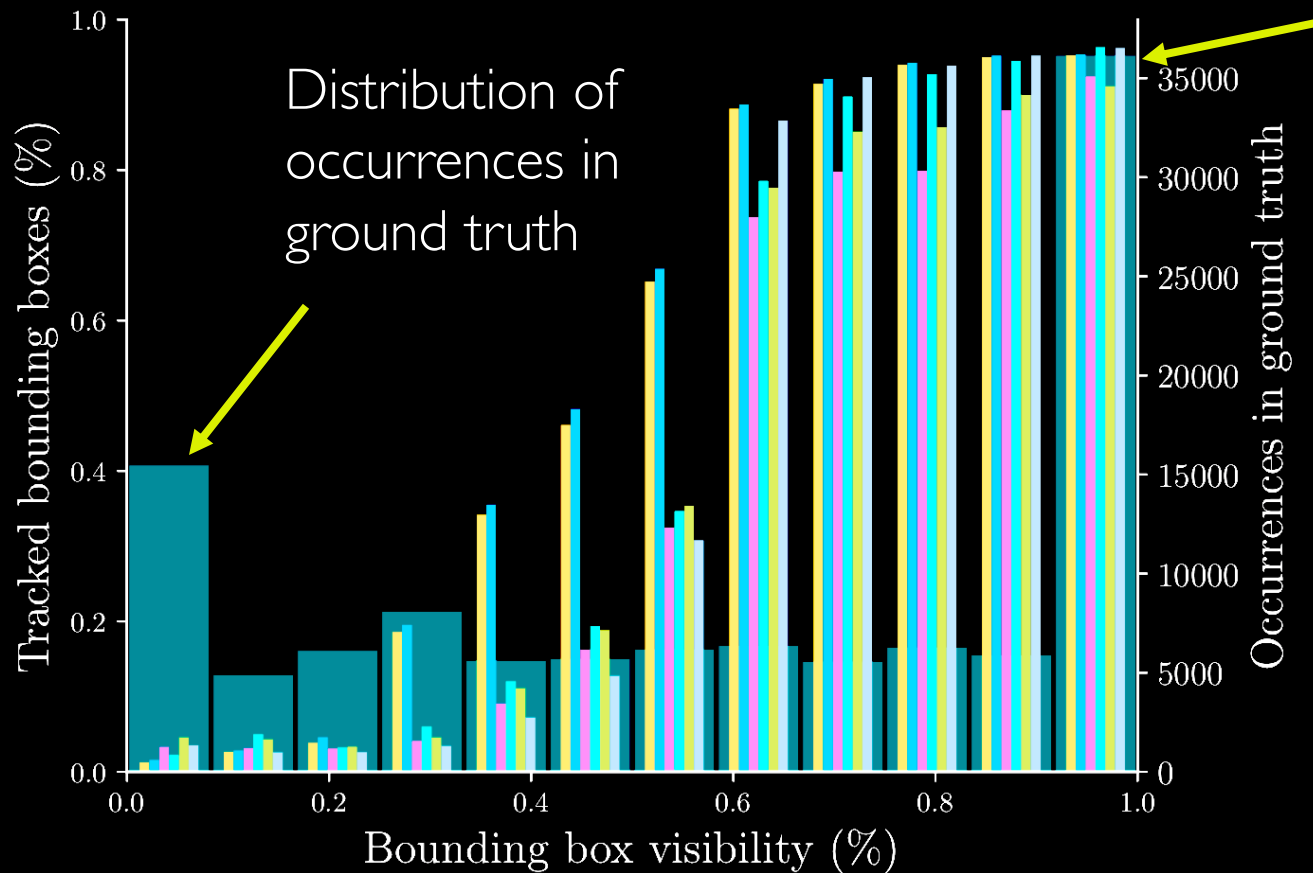
Time

PROS AND CONS

- **CON** There is no notion of “identity” in the model
 - Confusion in crowded spaces
- **CON** As any online tracker, the track is killed if the target becomes occluded
 - Need to close small gaps and occlusions
- **CON** The regressor only shifts the box by a small quantity
 - Large camera motions
 - Large displacements due to low framerate

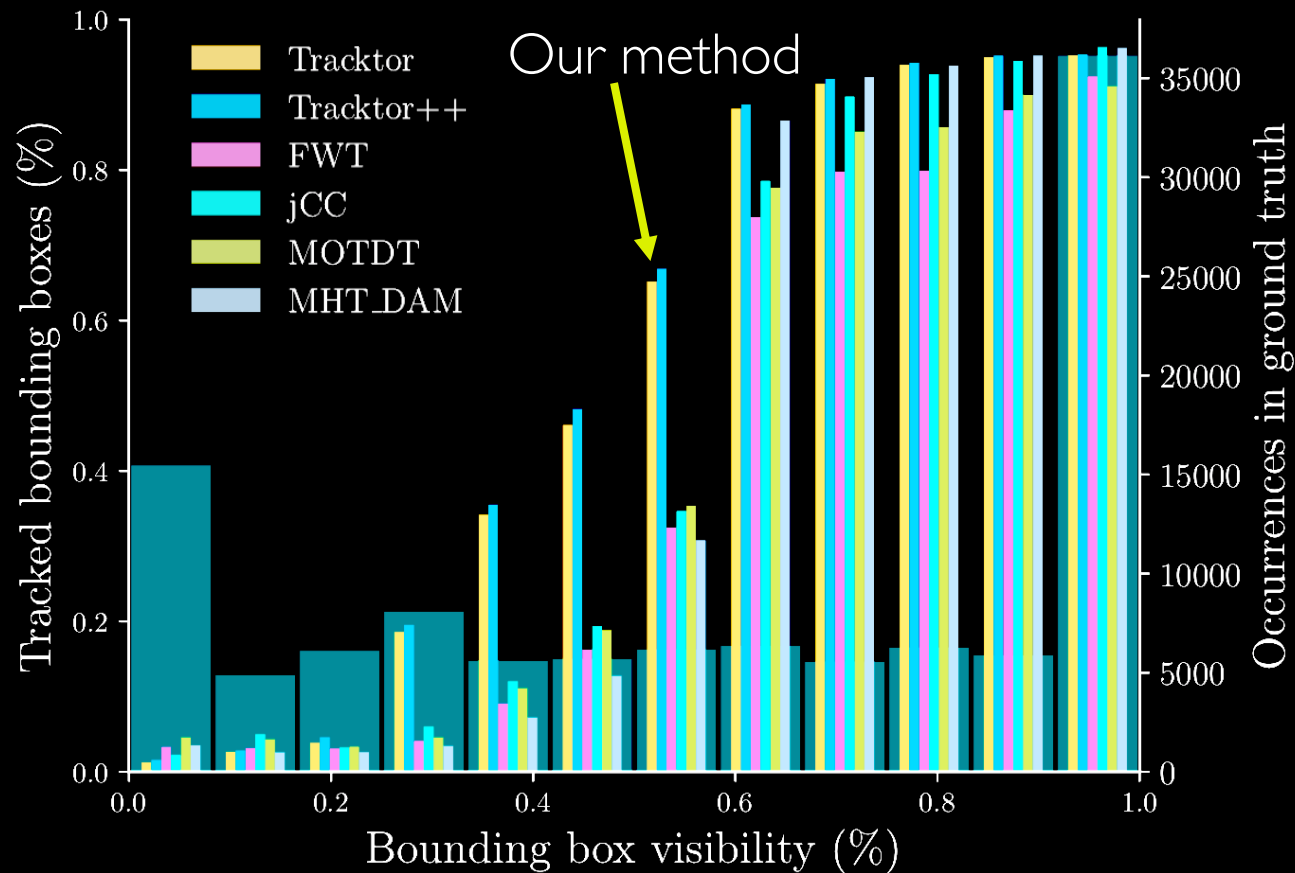


ANALYZING THE RESULTS

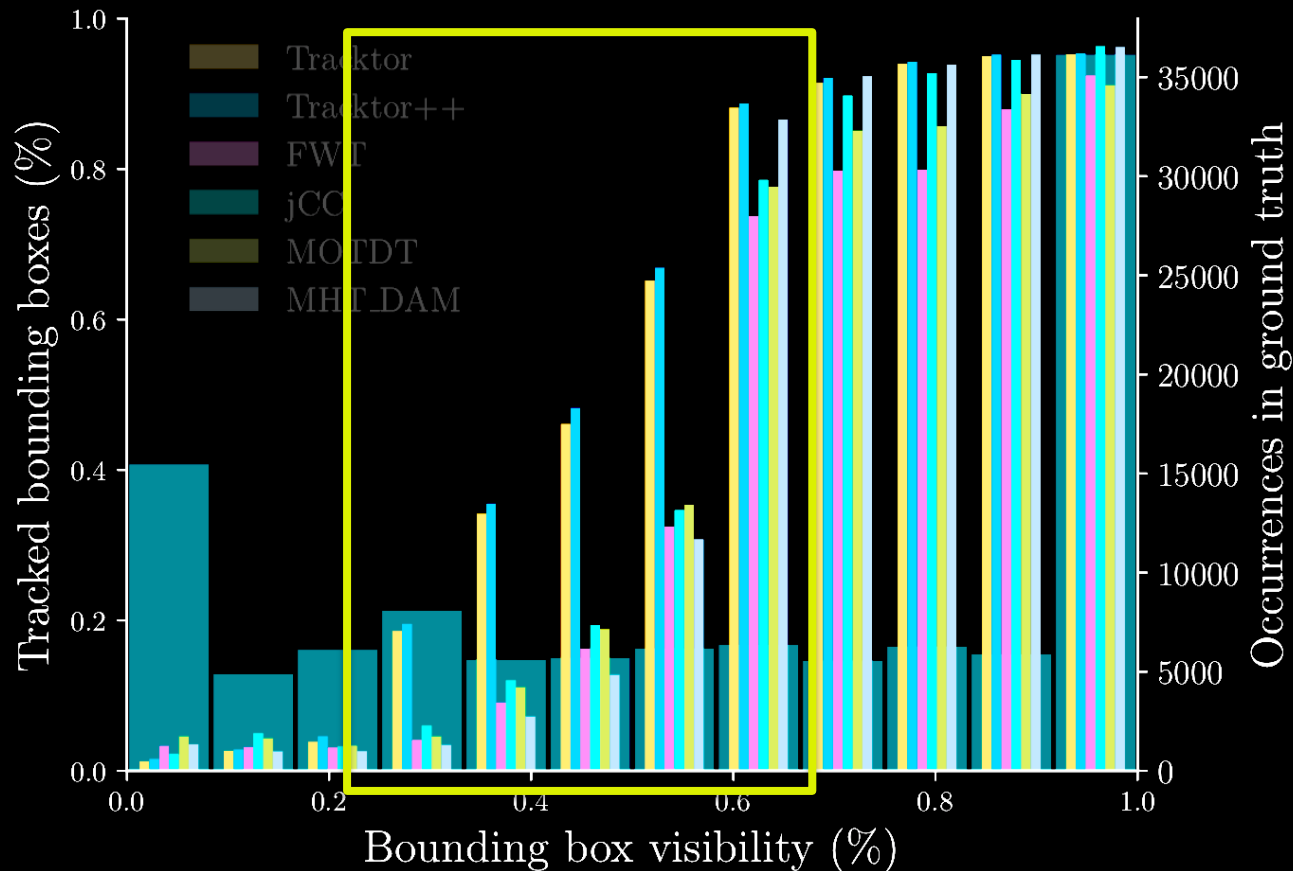


Most of the boxes have 100% visibility

ANALYZING THE RESULTS



ANALYZING THE RESULTS



None of the tracking-by-detection graph methods are more capable to recover from (partial) occlusions

ANALYZING THE RESULTS

- Hard problems in tracking are left **unsolved**
 - coverage of large gaps in detections
 - recovering from partial occlusions
 - tracking of small targets
- All other methods are just marginally improving “easy” scenes
- In fact, accuracy in tracking has only increased by 2.4 percentage points between 2017 and 2019 for MOT16 in MOTChallenge

ABLATION

Method	MOTA \uparrow	Δ	IDF1 \uparrow	Δ
TrackFormer	51.4		55.3	
———— w/o ————				
Pretraining on CrowdHuman	42.8	-8.6	45.2	-10.1
Track query re-identification	42.7	-0.1	43.6	-1.6
Track augmentations (FP)	40.1	-2.6	42.9	-0.7
Track augmentations (Range)	38.1	-2.0	41.0	-1.9
Track queries	37.8	-0.3	27.4	-13.6

Track augmentations

1. Improve track termination by adding false positive (FP) track queries
2. Train on challenging tracking scenarios by sampling $t - 1$ from a range of frames

ABLATION

Method	MOTA \uparrow	Δ	IDF1 \uparrow	Δ
TrackFormer	51.4		55.3	
<hr/>				
w/o				
Pretraining on CrowdHuman	42.8	-8.6	45.2	-10.1
Track query re-identification	42.7	-0.1	43.6	-1.6
Track augmentations (FP)	40.1	-2.6	42.9	-0.7
Track augmentations (Range)	38.1	-2.0	41.0	-1.9
Track queries	37.8	-0.3	27.4	-13.6



Track queries

1. Train only object queries for detection
2. Tracking-by-detection with (greedy) center distance matching as in CenterTrack^[1].