arXiv:2105.03247v4 [cs.CV] 19 Jul 2022

MOTR: End-to-End Multiple-Object Tracking with Transformer

Fangao Zeng[1⋆], Bin Dong[1⋆], Yuang Zhang[2⋆], Tiancai Wang[1⋆⋆], Xiangyu Zhang[1], and Yichen Wei[1]

[1] MEGVII Technology
[2] Shanghai Jiao Tong University

Abstract. Temporal modeling of objects is a key challenge in multiple-object tracking (MOT). Existing methods track by associating detections through motion-based and appearance-based similarity heuristics. The post-processing nature of association prevents end-to-end exploitation of temporal variations in video sequence.
In this paper, we propose MOTR, which extends DETR [6] and introduces "track query" to model the tracked instances in the entire video. Track query is transferred and updated frame-by-frame to perform iterative prediction over time. We propose tracklet-aware label assignment to train track queries and newborn object queries. We further propose temporal aggregation network and collective average loss to enhance temporal relation modeling. Experimental results on DanceTrack show that MOTR significantly outperforms state-of-the-art method, ByteTrack [42] by **6.5%** on HOTA metric. On MOT17, MOTR outperforms our concurrent works, TrackFormer [18] and TransTrack [29], on association performance. MOTR can serve as a stronger baseline for future research on temporal modeling and Transformer-based trackers. Code is available at https://github.com/megvii-research/MOTR.

Keywords: Multiple-Object Tracking, Transformer, End-to-End

1 Introduction

Multiple-object tracking (MOT) predicts the trajectories of instances in continuous image sequences [39,2]. Most of existing methods separate the MOT temporal association into appearance and motion: appearance variance is usually measured by pair-wise Re-ID similarity [37,43] while motion is modeled via IoU [4] or Kalman Filtering [3] heuristic. These methods require similarity-based matching for post-processing, which becomes the bottleneck of temporal information flow across frames. In this paper, we aim to introduce a fully end-to-end MOT framework featuring joint motion and appearance modeling.

Recently, DETR [6,45] was proposed for end-to-end object detection. It formulates object detection as a set prediction problem. As shown in Fig. 1(a),

⋆ Equal contribution.
⋆⋆ Corresponding author. Email: wangtiancai@megvii.com

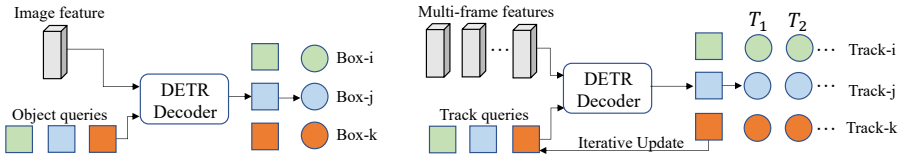

Fig. 1: (a) DETR achieves end-to-end detection by interacting object queries with image features and performs one-to-one assignment between the updated queries and objects. (b) MOTR performs set of sequence prediction by updating the track queries. Each track query represents a track. Best viewed in color.

object queries, served as a decoupled representation of objects, are fed into the Transformer decoder and interacted with the image feature to update their representation. Bipartite matching is further adopted to achieve one-to-one assignment between the object queries and ground-truths, eliminating post-processes, like NMS. Different from object detection, MOT can be regarded as a sequence prediction problem. The way to perform sequence prediction in the end-to-end DETR system is an open question.

Iterative prediction is popular in machine translation [30,31]. The output context is represented by a hidden state, and sentence features iteratively interact with the hidden state in the decoder to predict the translated words. Inspired by these advances in machine translation, we intuitively regard MOT as a problem of *set of sequence prediction* since MOT requires a set of object sequences. Each sequence corresponds to an object trajectory. Technically, we extend object query in DETR to *track query* for predicting object sequences. Track queries are served as the hidden states of object tracks. The representations of track queries are updated in the Transformer decoder and used to predict the object trajectory iteratively, as shown in Fig. 1(b). Specifically, track queries are updated through self-attention and cross-attention by frame features. The updated track queries are further used to predict the bounding boxes. The track of one object can be obtained from all predictions of one track query in different frames.

To achieve the goal above, we need to solve two problems: 1) track one object by one track query; 2) deal with newborn and terminated objects. To solve the first problem, we introduce tracklet-aware label assignment (TALA). It means that predictions of one track query are supervised by bounding box sequences with the same identity. To solve the second problem, we maintain a track query set of variable lengths. Queries of newborn objects are merged into this set while queries of terminated objects are removed. We name this process the entrance and exit mechanism. In this way, MOTR does not require explicit track associations during inference. Moreover, the iterative update of track queries enables temporal modeling regarding both appearance and motion.

To enhance the temporal modeling, we further propose collective average loss (CAL) and temporal aggregation network (TAN). With the CAL, MOTR takes

video clips as input during training. The parameters of MOTR are updated based on the overall loss calculated for the whole video clip. TAN introduces a shortcut for track query to aggregate the historical information from its previous states via the key-query mechanism in Transformer.

MOTR is a simple online tracker. It is easy to develop based on DETR with minor modifications on label assignment. It is a truly end-to-end MOT framework, requiring no post-processes, such as the track NMS or IoU matching employed in our concurrent works, TransTrack [29], and TrackFormer [18]. Experimental results on MOT17 and DanceTrack datasets show that MOTR achieves promising performance. On DanceTrack [28], MOTR outperforms the state-of-the-art ByteTrack [42] by **6.5%** on HOTA metric and **8.1%** on AssA.

To summarize, our contributions are listed as below:

- We present a fully end-to-end MOT framework, named MOTR. MOTR can implicitly learn the appearance and position variances in a joint manner.
- We formulate MOT as a problem of *set of sequence prediction*. We generate track query from previous hidden states for iterative update and prediction.
- We propose tracklet-aware label assignment for one-to-one assignment between track queries and objects. An entrance and exit mechanism is introduced to deal with newborn and terminated tracks.
- We further propose CAL and TAN to enhance the temporal modeling.

2 Related Work

Transformer-based Architectures. Transformer [31] was first introduced to aggregate information from the entire input sequence for machine translation. It mainly involves self-attention and cross-attention mechanisms. Since that, it was gradually introduced to many fields, such as speech processing [13,7] and computer vision [34,5]. Recently, DETR [6] combined convolutional neural network (CNN), Transformer and bipartite matching to perform end-to-end object detection. To achieve the fast convergence, Deformable DETR [45] introduced deformable attention module into Transformer encoder and Transformer decoder. ViT [9] built a pure Transformer architecture for image classification. Further, Swin Transformer [16] proposed shifted windowing scheme to perform self-attention within local windows, bringing greater efficiency. VisTR [36] employed a direct end-to-end parallel sequence prediction framework to perform video instance segmentation.

Multiple-Object Tracking. Dominant MOT methods mainly followed the tracking-by-detection paradigm [3,12,22,24,39]. These approaches usually first employ object detectors to localize objects in each frame and then perform track association between adjacent frames to generate the tracking results. SORT [3] conducted track association by combining Kalman Filter [38] and Hungarian algorithm [11]. DeepSORT [39] and Tracktor [2] introduced an extra cosine distance and compute the appearance similarity for track association. Track-RCNN [26], JDE [37] and FairMOT [43] further added a Re-ID branch on top of object detector in a joint training framework, incorporating object detection and Re-ID

feature learning. TransMOT [8] builds a spatial-temporal graph transformer for association. Our concurrent works, TransTrack [29] and TrackFormer [18] also develop Transformer-based frameworks for MOT. For direct comparison with them, please refer to Sec. 3.7.

Iterative Sequence Prediction. Predicting sequence via sequence-to-sequence (seq2seq) with encoder-decoder architecture is popular in machine translation [30,31] and text recognition [25]. In seq2seq framework, the encoder network encodes the input into intermediate representation. Then, a hidden state with task-specific context information is introduced and iteratively interacted with the intermediate representation to generate the target sequence through the decoder network. The iterative decode process contains several iterations. In each iteration, hidden state decodes one element of target sequence.

3 Method

3.1 Query in Object Detection

DETR [6] introduced a fixed-length set of object queries to detect objects. Object queries are fed into the Transformer decoder and interacted with image features, extracted from Transformer encoder to update their representation. Bipartite matching is further adopted to achieve one-to-one assignment between the updated object queries and ground-truths. Here, we simply write the object query as "detect query" to specify the query used for object detection.

3.2 Detect Query and Track Query

When adapting DETR from object detection to MOT, two main problems arise: 1) how to track one object by one track query; 2) how to handle newborn and terminated objects. We extend detect queries to track queries in this paper. Track query set is updated dynamically, and the length is variable. As shown in Fig. 2, the track query set is initialized to be empty, and the detect queries in DETR are used to detect newborn objects (object 3 at T_2). Hidden states of detected objects produces track queries for the next frame; track queries assigned to terminated objects are removed from the track query set (object 2 at T_4).

3.3 Tracklet-Aware Label Assignment

In DETR, one detect (object) query may be assigned to any object in the image since the label assignment is determined by performing bipartite matching between all detect queries and ground-truths. While in MOTR, detect queries are only used to detect newborn objects while track queries predict all tracked objects. Here, we introduce the tracklet-aware label assignment (TALA) to solve this problem.

Generally, TALA consists of two strategies. For detect queries, we modify the assignment strategy in DETR as **newborn-only**, where bipartite matching is

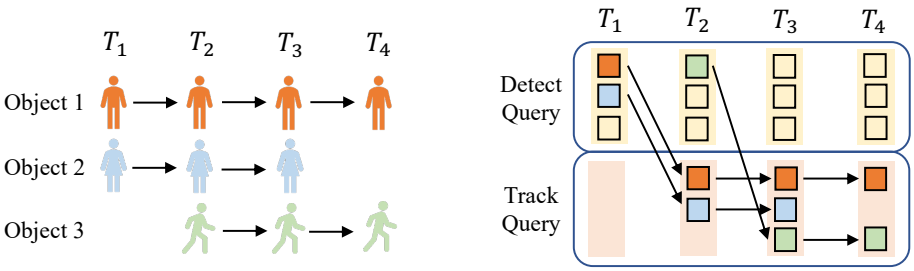

Fig. 2: Update process of detect (object) queries and track queries under some typical MOT cases. Track query set is updated dynamically, and the length is variable. Track query set is initialized to be empty, and the detect queries are used to detect newborn objects. Hidden states of all detected objects are concatenated to produce track queries for the next frame. Track queries assigned to terminated objects are removed from the track query set.

conducted between the detect queries and the ground-truths of newborn objects. For track queries, we design an **target-consistent** assignment strategy. Track queries follow the same assignment of previous frames and are therefore excluded from the aforementioned bipartite matching.

Formally, we denote the predictions of track queries as $\widehat{Y}_{tr}$ and predictions of detect queries as $\widehat{Y}_{det}$. Y_{new} is the ground-truths of newborn objects. The label assignment results for track queries and detect queries can be written as ω_{tr} and ω_{det}. For frame i, label assignment for detect queries is obtained from bipartite matching among detect queries and newborn objects, i.e.,

$$\omega_{det}^{i} = \underset{\omega_{det}^{i} \in \Omega_i}{\arg\min}\ \mathcal{L}(\widehat{Y}_{det}^{i}|_{\omega_{det}^{i}}, Y_{new}^{i}), \tag{1}$$

where $\mathcal{L}$ is the pair-wise matching cost defined in DETR and Ω_i is the space of all bipartite matches among detect queries and newborn objects. For track query assignment, we merge the assignments for newborn objects and tracked objects from the last frame, i.e., for $i > 1$:

$$\omega_{tr}^{i} = \omega_{tr}^{i-1} \cup \omega_{det}^{i-1}. \tag{2}$$

For the first frame ($i = 1$), track query assignment ω_{tr}^{1} is an empty set $\emptyset$ since there are no tracked objects for the first frame. For successive frames ($i > 1$), the track query assignment ω_{tr}^{i} is the concatenation of previous track query assignment ω_{tr}^{i-1} and newborn object assignment ω_{det}^{i-1}.

In practice, the TALA strategy is simple and effective thanks to the powerful attention mechanism in Transformer. For each frame, detect queries and track queries are concatenated and fed into the Transformer decoder to update their representation. Detect queries will only detect newborn objects since query interaction by self-attention in the Transformer decoder will suppress detect queries that detect tracked objects. This mechanism is similar to duplicate removal in DETR that duplicate boxes are suppressed with low scores.

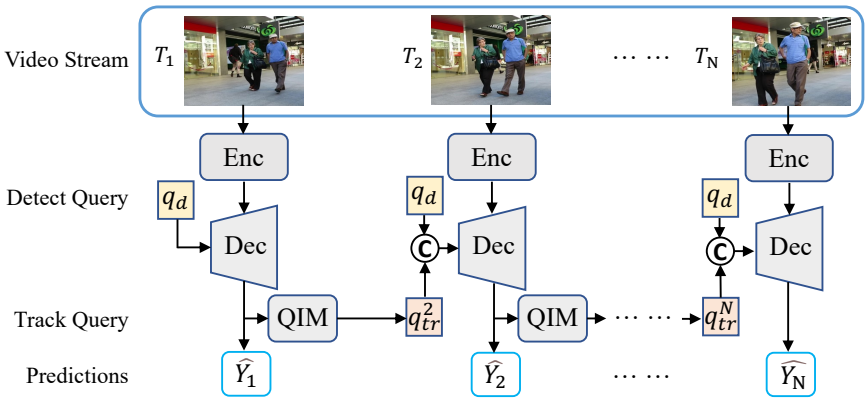

Fig. 3: The overall architecture of MOTR. "Enc" represents a convolutional neural network backbone and the Transformer encoder that extracts image features for each frame. The concatenation of detect queries q_d and track queries q_{tr} is fed into the Deformable DETR decoder (Dec) to produce the hidden states. The hidden states are used to generate the prediction $\widehat{Y}$ of newborn and tracked objects. The query interaction module (QIM) takes the hidden states as input and produces track queries for the next frame.

3.4 MOTR Architecture

The overall architecture of MOTR is shown in Fig. 3. Video sequences are fed into the convolutional neural network (CNN) (e.g. ResNet-50 [10]) and Deformable DETR [45] encoder to extract frame features.

For the first frame, there are no track query and we only feed the fixed-length learnable detect queries (q_d in Fig. 3) into the Deformable DETR [45] decoder. For successive frames, we feed the concatenation of track queries from the previous frame and the learnable detect queries into the decoder. These queries interact with image feature in the decoder to generate the hidden state for bounding box prediction. The hidden state is also fed into the query interaction module (QIM) to generate the track queries for the next frame.

During training phase, the label assignment for each frame is described in Sec. 3.3. All predictions of the video clip are collected into a prediction bank $\{\widehat{Y}_1, \widehat{Y}_2, \ldots, \widehat{Y}_N\}$, and we use the proposed collective average loss (CAL) described in Sec. 3.6 for supervision. During inference time, the video stream can be processed online and generate the prediction for each frame.

3.5 Query Interaction Module

In this section, we describe query interaction module (QIM). QIM includes object entrance and exit mechanism and temporal aggregation network (TAN).

Object Entrance and Exit. As mentioned above, some objects in video sequences may appear or disappear at intermediate frames. Here, we introduce the

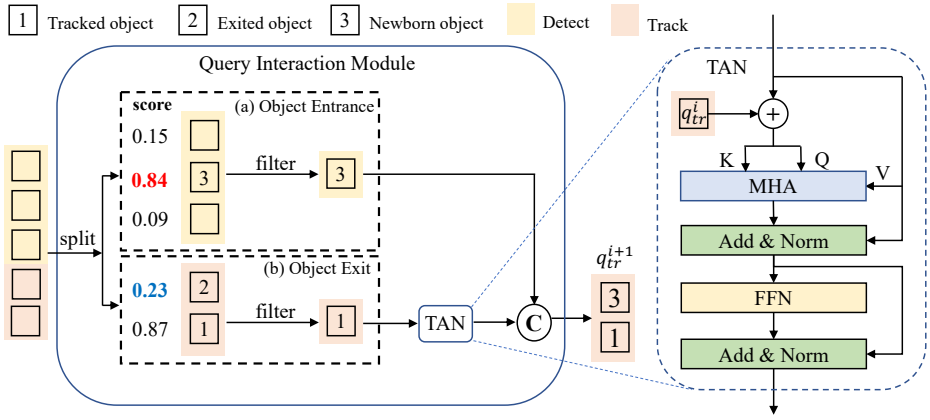

Fig. 4: The structure of query interaction module (QIM). The inputs of QIM are the hidden state produced by Transformer decoder and the corresponding prediction scores. In the inference stage, we keep newborn objects and drop exited objects based on the confidence scores. Temporal aggregation network (TAN) enhances long-range temporal modeling.

way we deal with the newborn and terminated objects in our method. For any frame, track queries are concatenated with the detect queries and input to the Transformer decoder, producing the hidden state (see the left side of Fig. 4).

During training, hidden states of terminated objects are removed if the matched objects disappeared in ground-truths or the intersection-over-union (IoU) between predicted bounding box and target is below a threshold of 0.5. It means that the corresponding hidden states will be filtered if these objects disappear at current frame while the rest hidden states are reserved. For newborn objects, the corresponding hidden states are kept based on the assignment of newborn object ω_{det}^{i} defined in Eq. 1.

For inference, we use the predicted classification scores to determine appearance of newborn objects and disappearance of tracked objects, as shown in Fig. 4. For object queries, predictions whose classification scores are higher than the entrance threshold τ_{en} are kept while other hidden states are removed. For track queries, predictions whose classification scores are lower than the exit threshold τ_{ex} for consecutive M frames are removed while other hidden states are kept.

Temporal Aggregation Network. Here, we introduce the temporal aggregation network (TAN) in QIM to enhance temporal relation modeling and provide contextual priors for tracked objects.

As shown in Fig. 4, the input of TAN is the filtered hidden state for tracked objects (object "1"). We also collect the track query q_{tr}^{i} from the last frame for temporal aggregation. TAN is a modified Transformer decoder layer. The track query from the last frame and the filtered hidden state are summed to be the key and query components of the multi-head self-attention (MHA). The

hidden state alone is the value component of MHA. After MHA, we apply a feed-forward network (FFN) and the results are concatenated with the hidden state for newborn objects (object "3") to produce the track query set q_{tr}^{i+1} for the next frame.

3.6 Collective Average Loss

Training samples are important for temporal modeling of track since MOTR learns temporal variances from data rather than hand-crafted heuristics like Kalman Filtering. Common training strategies, like training within two frames, fail to generate training samples of long-range object motion. Different from them, MOTR takes video clips as input. In this way, training samples of long-range object motion can be generated for temporal learning.

Instead of calculating the loss frame-by-frame, our collective average loss (CAL) collects the multiple predictions $\widehat{Y} = \{\widehat{Y}_i\}_{i=1}^{N}$. Then the loss within the whole video sequence is calculated by ground-truths $Y = \{Y_i\}_{i=1}^{N}$ and the matching results $\omega = \{\omega_i\}_{i=1}^{N}$. CAL is the overall loss of the whole video sequence, normalized by the number of objects:

$$\mathcal{L}_o(\widehat{Y}|_\omega, Y) = \frac{\sum_{n=1}^{N} (\mathcal{L}(\widehat{Y}_{tr}^i|_{\omega_{tr}^i}, Y_{tr}^i) + \mathcal{L}(\widehat{Y}_{det}^i|_{\omega_{det}^i}, Y_{det}^i))}{\sum_{n=1}^{N} (V_i)} \tag{3}$$

where $V_i = V_{tr}^i + V_{det}^i$ denotes the total number of ground-truths objects at frame i. V_{tr}^i and V_{det}^i are the numbers of tracked objects and newborn objects at frame i, respectively. $\mathcal{L}$ is the loss of single frame, which is similar to the detection loss in DETR. The single-frame loss $\mathcal{L}$ can be formulated as:

$$\mathcal{L}(\widehat{Y}_i|_{\omega_i}, Y_i) = \lambda_{cls}\mathcal{L}_{cls} + \lambda_{l_1}\mathcal{L}_{l_1} + \lambda_{giou}\mathcal{L}_{giou} \tag{4}$$

where $\mathcal{L}_{cls}$ is the focal loss [14]. $\mathcal{L}_{l_1}$ denotes the L1 loss and $\mathcal{L}_{giou}$ is the generalized IoU loss [21]. λ_{cls}, λ_{l_1} and λ_{giou} are the corresponding weight coefficients.

3.7 Discussion

Based on DETR, our concurrent works, TransTrack [29] and TrackFormer [18] also develop the Transformer-based frameworks for MOT. However, our method shows large differences compared to them:

TransTrack models a full track as a combination of several independent short tracklets. Similar to the track-by-detection paradigm, TransTrack decouples MOT as two sub-tasks: 1) detect object pairs as short tracklets within two adjacent frames; 2) associate short tracklets as full tracks by IoU-matching. While for MOTR, we model a full track in an end-to-end manner through the iterative update of track query, requiring no IoU-matching.

Table 1: Comparison with other MOT methods based on Transformer.

Method	IoU match	NMS	ReID
TransTrack [29]	✓		
TrackFormer [18]		✓	✓
MOTR (ours)			

Table 2: Statistics of chosen datasets for evaluation.

Datasets	Class	Frame	Video	ID
DanceTrack [28]	1	106k	100	990
MOT17 [19]	1	11k	14	1342
BDD100K [41]	8	318k	1400	131k

TrackFormer shares the idea of track query with us. However, TrackFormer still learns within two adjacent frames. As discussed in Sec. 3.6, learning within short-range will result in relatively weak temporal learning. Therefore, TrackFormer employs heuristics, such as Track NMS and Re-ID features, to filter out duplicate tracks. Different from TrackFormer, MOTR learns stronger temporal motion with CAL and TAN, removing the need of those heuristics. For direct comparison with TransTrack and TrackFormer, please refer to the Table 1.

Here, we clarify that we started this work independently long before TrackFormer and TransTrack appear on arXiv. Adding that they are not formally published, we treat them as *concurrent and independent* works instead of *previous* works on which our work is built upon.

4 Experiments

4.1 Datasets and Metrics

Datasets. For comprehensive evaluation, we conducted experiments on three datasets: DanceTrack [28], MOT17 [19], and BDD100k [41]. MOT17 [19] contains 7 training sequences and 7 test sequences. DanceTrack [28] is a recent multi-object tracking dataset featuring uniform appearance and diverse motion. It contains more videos for training and evaluation thus providing a better choice to verify the tracking performance. BDD100k [41] is an autonomous driving dataset with an MOT track featuring multiple object classes. For more details, please refer to the statistics of datasets, shown in Table 2.

Evaluation Metrics. We follow the standard evaluation protocols to evaluate our method. The common metrics include Higher Order Metric for Evaluating Multi-object Tracking [17] (HOTA, AssA, DetA), Multiple-Object Tracking Accuracy (MOTA), Identity Switches (IDS) and Identity F1 Score (IDF1).

4.2 Implementation Details

Following the settings in CenterTrack [44], MOTR adopts several data augmentation methods, such as random flip and random crop. The shorter side of the input image is resized to 800 and the maximum size is restricted to 1536. The inference speed on Tesla V100 at this resolution is about 7.5 FPS. We sample keyframes with random intervals to solve the problem of variable frame rates. Besides, we erase the tracked queries with the probability p_{drop} to generate more

samples for newborn objects and insert track queries of false positives with the probability p_{insert} to simulate the terminated objects. All the experiments are conducted on PyTorch with 8 NVIDIA Tesla V100 GPUs. We also provide a memory-optimized version that can be trained on NVIDIA 2080 Ti GPUs.

We built MOTR upon Deformable-DETR [45] with ResNet50 [10] for fast convergence. The batch size is set to 1 and each batch contains a video clip of 5 frames. We train our model with the AdamW optimizer with the initial learning rate of $2.0 \cdot 10^{-4}$. For all datasets, we initialize MOTR with the official Deformable DETR [45] weights pre-trained on the COCO [15] dataset. On **MOT17**, we train MOTR for 200 epochs and the learning rate decays by a factor of 10 at the 100^{th} epoch. For state-of-the-art comparison, we train on the joint dataset (MOT17 training set and CrowdHuman [23] val set). For ∼5k static images in CrowdHuman val set, we apply random shift as in [44] to generate video clips with pseudo tracks. The initial length of video clip is 2 and we gradually increase it to 3,4,5 at the 50^{th},90^{th},150^{th} epochs, respectively. The progressive increment of video clip length improves the training efficiency and stability. For the ablation study, we train MOTR on the MOT17 training set without using the CrowdHuman dataset and validate on the 2DMOT15 training set. On **DanceTrack**, we train for 20 epochs on the train set and learning rate decays at the 10^{th} epoch. We gradually increase the clip length from 2 to 3,4,5 at the 5^{th},9^{th},15^{th} epochs. On **BDD100k**, we train for 20 epochs on the train set and learning rate decays at the 16^{th} epoch. We gradually increase the clip length from 2 to 3 and 4 at the 6^{th} and 12^{th} epochs.

4.3 State-of-the-art Comparison on MOT17

Table 3 compares our approach with state-of-the-art methods on MOT17 test set. We mainly compare MOTR with our concurrent works based on Transformer: TrackFormer [18] and TransTrack [29]. Our method gets higher IDF1 scores, surpassing TransTrack and TrackFormer by 4.5%. The performance of MOTR on the HOTA metric is much higher than TransTrack by 3.1%. For the MOTA metric, our method achieves much better performance than TrackFormer (71.9% *vs.* 65.0%). Interestingly, we find that the performance of TransTrack is better than our MOTR on MOTA. We suppose the decoupling of detection and tracking branches in TransTrack indeed improves the object detection performance. While in MOTR, detect and track queries are learned through a shared Transformer decoder. Detect queries are suppressed on detecting tracked objects, limiting the detection performance on newborn objects.

If we compare the performance with other state-of-the-art methods, like ByteTrack [42], it shows that MOTR is **frustratingly inferior** to them on the MOT17 dataset. Usually, state-of-the-art performance on the MOT17 dataset is dominated by trackers with good detection performance to cope with various appearance distributions. Also, different trackers tend to employ different detectors for object detection. It is pretty difficult for us to fairly verify the motion performance of various trackers. Therefore, we argue that the MOT17 dataset alone is **not enough** to fully evaluate the tracking performance of MOTR. We

Table 3: Performance comparison between MOTR and existing methods on the MOT17 dataset under the private detection protocols. The number is marked in bold if it is the best among the Transformer-based methods.

Methods	HOTA↑	AssA↑	DetA↑	IDF1↑	MOTA↑	IDS↓
CNN-based:						
Tracktor++[2]	44.8	45.1	44.9	52.3	53.5	2072
CenterTrack[44]	52.2	51.0	53.8	64.7	67.8	3039
TraDeS [40]	52.7	50.8	55.2	63.9	69.1	3555
QDTrack [20]	53.9	52.7	55.6	66.3	68.7	3378
GSDT [35]	55.5	54.8	56.4	68.7	66.2	3318
FairMOT[43]	59.3	58.0	60.9	72.3	73.7	3303
CorrTracker [32]	60.7	58.9	62.9	73.6	76.5	3369
GRTU [33]	62.0	62.1	62.1	75.0	74.9	1812
MAATrack [27]	62.0	60.2	64.2	75.9	79.4	1452
ByteTrack [42]	63.1	62.0	64.5	77.3	80.3	2196
Transformer-based:						
TrackFormer [18]	/	/	/	63.9	65.0	3528
TransTrack[29]	54.1	47.9	**61.6**	63.9	**74.5**	3663
MOTR (ours)	**57.8**	**55.7**	60.3	**68.6**	73.4	**2439**

further evaluate the tracking performance on DanceTrack [28] dataset with uniform appearance and diverse motion, as described next.

4.4 State-of-the-art Comparison on DanceTrack

Recently, DanceTrack [28], a dataset with uniform appearance and diverse motion, is introduced (see Tab. 2). It contains much more videos for evaluation and provides a better choice to verify the tracking performance. We further conduct the experiments on the DanceTrack dataset and perform the performance comparison with state-of-the-art methods in Tab. 4. It shows that MOTR achieves much better performance on DanceTrack dataset. Our method gets a much higher HOTA score, surpassing ByteTrack by 6.5%. For the AssA metric, our method also achieves much better performance than ByteTrack (40.2% *vs.* 32.1%). While for the DetA metric, MOTR is inferior to some state-of-the-art methods. It means that MOTR performs well on temporal motion learning while the detection performance is not that good. The large improvements on HOTA are mainly from the temporal aggregation network and collective average loss.

4.5 Generalization on Multi-Class Scene

Re-ID based methods, like FairMOT [43], tend to regard each tracked object (e.g., person) as a class and associate the detection results by the feature similarity. However, the association will be difficult when the number of tracked objects is very large. Different from them, each object is denoted as one track query in MOTR and the track query set is of dynamic length. MOTR can easily deal with

Table 4: Performance comparison between MOTR and existing methods on the DanceTrack[28] dataset. Results for existing methods are from DanceTrack [28].

Methods	HOTA	AssA	DetA	MOTA	IDF1
CenterTrack [44]	41.8	22.6	**78.1**	86.8	35.7
FairMOT [43]	39.7	23.8	66.7	82.2	40.8
QDTrack [20]	45.7	29.2	72.1	83.0	44.8
TransTrack [29]	45.5	27.5	75.9	88.4	45.2
TraDes [40]	43.3	25.4	74.5	86.2	41.2
ByteTrack [42]	47.7	32.1	71.0	**89.6**	**53.9**
MOTR (ours)	**54.2**	**40.2**	73.5	79.7	51.5

Table 5: Performance comparison between MOTR and existing methods on the BDD100k[41] validation set.

Methods	mMOTA	mIDF1	IDSw
Yu *et al.* [41]	25.9	**44.5**	8315
DeepBlueAI [1]	26.9	/	13366
MOTR (ours)	**32.0**	43.5	**3493**

the multi-class prediction problem, by simply modifying the class number of the classification branch. To verify the performance of MOTR on multi-class scenes, we further conduct the experiments on the BDD100k dataset (see Tab. 5). Results on bdd100k validation set show that MOTR performs well on multi-class scenes and achieves promising performance with fewer ID switches.

4.6 Ablation Study

MOTR Components. Table 6a shows the impact of integrating different components. Integrating our components into the baseline can gradually improve overall performance. Using only object query of as original leads to numerous IDs since most objects are treated as entrance objects. With track query introduced, the baseline is able to handle tracking association and improve IDF1 from 1.2 to 49.8. Further, adding TAN to the baseline improves MOTA by 7.8% and IDF1 by 13.6%. When using CAL during training, there are extra 8.3% and 7.1% improvements in MOTA and IDF1, respectively. It demonstrates that TAN combined with CAL can enhance the learning of temporal motion.

Collective Average Loss. Here, we explored the impact of video sequence length on the tracking performance in CAL. As shown in Table 6b, when the length of the video clip gradually increases from 2 to 5, MOTA and IDF1 metrics are improved by 8.3% and 7.1%, respectively. Thus, multi-frame CAL can greatly boost the tracking performance. We explained that multiple frames CAL can help the network to handle some hard cases such as occlusion scenes. We observed that duplicated boxes, ID switches, and object missing in occluded scenes are significantly reduced. To verify it, we provide some visualizations in Fig. 5.

Table 6: Ablation studies on our proposed MOTR. All experiments use the single-level C5 feature in ResNet50.

(a) The effect of our contributions. TrackQ: track query. TAN: temporal aggregation network. CAL: collective average loss.

TrackQ	TAN	CAL	MOTA↑	IDF1↑	IDS↓
			-	1.2	33198
✓			37.1	49.8	562
✓	✓		44.9	63.4	257
✓		✓	47.5	56.1	417
✓	✓	✓	**53.2**	**70.5**	**155**

(b) The impact of increasing video clip length in Collective Average Loss during training on tracking performance.

Length	MOTA↑	IDF1↑	IDS↓
2	44.9	63.4	257
3	51.6	59.4	424
4	50.6	64.0	314
5	**53.2**	**70.5**	**155**

(c) Analysis on random track query erasing probability p_{drop} during training.

p_{drop}	MOTA↑	IDF1↑	IDS↓
5e-2	49.0	60.4	411
0.1	**53.2**	**70.5**	**155**
0.3	51.1	69.0	180
0.5	48.5	62.0	302

(d) Effect of random false positive inserting probability p_{insert} during training.

p_{insert}	MOTA↑	IDF1↑	IDS↓
0.1	51.2	**71.7**	**148**
0.3	**53.2**	70.5	155
0.5	52.1	62.0	345
0.7	50.7	57.7	444

(e) The exploration of different combinations of τ_{ex} and τ_{en} in QIM network.

τ_{ex}	0.6	0.6	0.6	0.5	0.6	0.7
τ_{en}	0.7	0.8	0.9	0.8	0.8	0.8
MOTA↑	52.7	**53.2**	53.1	**53.5**	53.2	52.8
IDF1↑	69.8	**70.5**	70.1	**70.5**	70.5	68.3
IDS↓	181	155	**142**	**153**	155	181

(f) The effect of random sampling interval on tracking performance.

Intervals	MOTA↑	IDF1↑	IDS↓
3	53.2	64.8	218
5	50.8	62.8	324
10	**53.2**	**70.5**	**155**
12	53.1	69	158

Erasing and Inserting Track Query. In MOT datasets, there are few training samples for two cases: entrance objects and exit objects in video sequences. Therefore, we adopt track query erasing and inserting to simulate these two cases with probability p_{drop} and p_{insert}, respectively. Table 6c reports the performance using different value of p_{drop} during training. MOTR achieves the best performance when p_{drop} is set to 0.1. Similar to the entrance objects, track queries transferred from the previous frame, whose predictions are false positives, are inserted into the current frame to simulate the case of object exit. In Table 6d, we explore the impact on tracking performance of different p_{insert}. When progressively increasing p_{insert} from 0.1 to 0.7, our MOTR achieves the highest score on MOTA when p_{insert} is set to 0.3 while the IDF1 score is decreasing.

Object Entrance and Exit Threshold. Table 6e investigates the impact of different combination of object entrance threshold τ_{en} and exit threshold τ_{ex} in QIM. As we vary the object entrance threshold τ_{en}, we can see that the performance is not that sensitive to τ_{en} (within 0.5% on MOTA) and using an

(a) (b)

Fig. 5: The effect of CAL on solving (a) duplicated boxes and (b) ID switch problems. Top and bottom rows are the tracking results without and with CAL, respectively.

entrance threshold of 0.8 produces relatively better performance. We also further conduct experiments by varying the object exit threshold τ_{ex}. It is shown that using a threshold of 0.5 results in slightly better performance than that of 0.6. In our practice, τ_{en} with 0.6 shows better performance on the MOT17 test set.

Sampling Interval. In Table 6f, we evaluate the effect of random sampling interval on tracking performance during training. When the sampling interval increases from 2 to 10, the IDS decreases significantly from 209 to 155. During training, the network is easy to fall into a local optimal solution when the frames are sampled in a small interval. Appropriate increment on sampling interval can simulate real scenes. When the random sampling interval is greater than 10, the tracking framework fails to capture such long-range dynamics, leading to relatively worse tracking performance.

5 Limitations

MOTR, an online tracker, achieves end-to-end multiple-object tracking. It implicitly learns the appearance and position variances in a joint manner thanks to the DETR architecture as well as the tracklet-aware label assignment. However, it also has several shortcomings. First, the performance of detecting newborn objects is far from satisfactory (the result on the MOTA metric is not good enough). As we analyzed above, detect queries are suppressed on detecting tracked objects, which may go against the nature of object query and limits the detection performance on newborn objects. Second, the query passing in MOTR is performed frame-by-frame, limiting the efficiency of model learning during training. In our practice, the parallel decoding in VisTR [36] fails to deal with the complex scenarios in MOT. Solving these two problems above will be an important research topic for Transformer-based MOT frameworks.

Acknowledgements: This research was supported by National Key R&D Program of China (No. 2017YFA0700800) and Beijing Academy of Artificial Intelligence (BAAI).

References

1. CodaLab Competition - CVPR 2020 BDD100K Multiple Object Tracking Challenge (Jul 2022), https://competitions.codalab.org/competitions/24910, [Online; accessed 19. Jul. 2022] 12
2. Bergmann, P., Meinhardt, T., Leal-Taixe, L.: Tracking without bells and whistles. In: ICCV (2019) 1, 3, 11
3. Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B.: Simple online and realtime tracking. In: ICIP (2016) 1, 3
4. Bochinski, E., Eiselein, V., Sikora, T.: High-speed tracking-by-detection without using image information. In: AVSS (2017) 1
5. Camgoz, N.C., Koller, O., Hadfield, S., Bowden, R.: Sign language transformers: Joint end-to-end sign language recognition and translation. In: CVPR (2020) 3
6. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: ECCV (2020) 1, 3, 4
7. Chang, X., Zhang, W., Qian, Y., Le Roux, J., Watanabe, S.: End-to-end multi-speaker speech recognition with transformer. In: ICASSP (2020) 3
8. Chu, P., Wang, J., You, Q., Ling, H., Liu, Z.: Transmot: Spatial-temporal graph transformer for multiple object tracking. arXiv preprint arXiv:2104.00194 (2021) 4
9. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021) 3
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016) 6, 10
11. Kuhn, H.W.: The hungarian method for the assignment problem. Naval research logistics quarterly **2**(1-2), 83–97 (1955) 3
12. Leal-Taixé, L., Canton-Ferrer, C., Schindler, K.: Learning by tracking: Siamese cnn for robust target association. In: CVPRW (2016) 3
13. Li, N., Liu, S., Liu, Y., Zhao, S., Liu, M.: Neural speech synthesis with transformer network. In: AAAI (2019) 3
14. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: ICCV (2017) 8
15. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014) 10
16. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. arXiv preprint arXiv:2103.14030 (2021) 3
17. Luiten, J., Osep, A., Dendorfer, P., Torr, P., Geiger, A., Leal-Taixé, L., Leibe, B.: Hota: A higher order metric for evaluating multi-object tracking. IJCV **129**(2), 548–578 (2021) 9
18. Meinhardt, T., Kirillov, A., Leal-Taixe, L., Feichtenhofer, C.: Trackformer: Multi-object tracking with transformers. arXiv preprint arXiv:2101.02702 (2021) 1, 3, 4, 8, 9, 10, 11

19. Milan, A., Leal-Taixé, L., Reid, I., Roth, S., Schindler, K.: Mot16: A benchmark for multi-object tracking. arXiv preprint arXiv:1603.00831 (2016) 9
20. Pang, J., Qiu, L., Li, X., Chen, H., Li, Q., Darrell, T., Yu, F.: Quasi-dense similarity learning for multiple object tracking. In: CVPR (2021) 11, 12
21. Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., Savarese, S.: Generalized intersection over union: A metric and a loss for bounding box regression. In: CVPR (2019) 8
22. Schulter, S., Vernaza, P., Choi, W., Chandraker, M.: Deep network flow for multi-object tracking. In: CVPR (2017) 3
23. Shao, S., Zhao, Z., Li, B., Xiao, T., Yu, G., Zhang, X., Sun, J.: Crowdhuman: A benchmark for detecting human in a crowd. arXiv preprint arXiv:1805.00123 (2018) 10
24. Sharma, S., Ansari, J.A., Murthy, J.K., Krishna, K.M.: Beyond pixels: Leveraging geometry and shape cues for online multi-object tracking. In: ICRA (2018) 3
25. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. TPAMI **39**(11), 2298–2304 (2016) 4
26. Shuai, B., Berneshawi, A.G., Modolo, D., Tighe, J.: Multi-object tracking with siamese track-rcnn. arXiv preprint arXiv:2004.07786 (2020) 3
27. Stadler, D., Beyerer, J.: Modelling ambiguous assignments for multi-person tracking in crowds. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 133–142 (2022) 11
28. Sun, P., Cao, J., Jiang, Y., Yuan, Z., Bai, S., Kitani, K., Luo, P.: Dancetrack: Multi-object tracking in uniform appearance and diverse motion. arXiv preprint arXiv:2111.14690 (2021) 3, 9, 11, 12
29. Sun, P., Jiang, Y., Zhang, R., Xie, E., Cao, J., Hu, X., Kong, T., Yuan, Z., Wang, C., Luo, P.: Transtrack: Multiple-object tracking with transformer. arXiv preprint arXiv: 2012.15460 (2020) 1, 3, 4, 8, 9, 10, 11, 12
30. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: NeurIPS (2014) 2, 4
31. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017) 2, 3, 4
32. Wang, Q., Zheng, Y., Pan, P., Xu, Y.: Multiple object tracking with correlation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3876–3886 (2021) 11
33. Wang, S., Sheng, H., Zhang, Y., Wu, Y., Xiong, Z.: A general recurrent tracking framework without real data. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13219–13228 (2021) 11
34. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: CVPR (2018) 3
35. Wang, Y., Kitani, K., Weng, X.: Joint object detection and multi-object tracking with graph neural networks. In: 2021 IEEE International Conference on Robotics and Automation (ICRA). pp. 13708–13715. IEEE (2021) 11
36. Wang, Y., Xu, Z., Wang, X., Shen, C., Cheng, B., Shen, H., Xia, H.: End-to-end video instance segmentation with transformers. In: CVPR (2021) 3, 14
37. Wang, Z., Zheng, L., Liu, Y., Li, Y., Wang, S.: Towards real-time multi-object tracking. In: ECCV (2020) 1, 3
38. Welch, G., Bishop, G., et al.: An introduction to the kalman filter (1995) 3
39. Wojke, N., Bewley, A., Paulus, D.: Simple online and realtime tracking with a deep association metric. In: ICIP (2017) 1, 3

40. Wu, J., Cao, J., Song, L., Wang, Y., Yang, M., Yuan, J.: Track to detect and segment: An online multi-object tracker. In: CVPR (2021) 11, 12
41. Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., Darrell, T.: Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020) 9, 12
42. Zhang, Y., Sun, P., Jiang, Y., Yu, D., Yuan, Z., Luo, P., Liu, W., Wang, X.: Bytetrack: Multi-object tracking by associating every detection box. arXiv preprint arXiv:2110.06864 (2021) 1, 3, 10, 11, 12
43. Zhang, Y., Wang, C., Wang, X., Zeng, W., Liu, W.: Fairmot: On the fairness of detection and re-identification in multiple object tracking. IJCV pp. 1–19 (2021) 1, 3, 11, 12
44. Zhou, X., Koltun, V., Krähenbühl, P.: Tracking objects as points. In: ECCV (2020) 9, 10, 11, 12
45. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. In: ICLR (2020) 1, 3, 6, 10