AI Open 4 (2023) 145–153

Contents lists available at ScienceDirect

AI Open

journal homepage: www.keaipublishing.com/en/journals/ai-open

MOTT: A new model for multi-object tracking based on green learning paradigm

Shan Wu [a,*], Amnir Hadachi [a], Chaoru Lu [b], Damien Vivet [c]

[a] ITS Lab, Institute of Computer Science, University of Tartu, Narva mnt. 18, Tartu, 51009, Estonia
[b] Centre of Metropolitan Digitalization and Smartization (MetSmart), Department of Built Environment, Oslo Metropolitan University, Pilestredet 46, Oslo, 0167, Norway
[c] ISAE-SUPAERO, Université de Toulouse, 10 Av. Edouard Belin, Toulouse, 31400, France

ARTICLE INFO

Keywords:
Multi-object tracking
Pedestrian tracking
Green learning
Transformer
End-to-end

ABSTRACT

Multi-object tracking (MOT) is one of the most essential and challenging tasks in computer vision (CV). Unlike object detectors, MOT systems nowadays are more complicated and consist of several neural network models. Thus, the balance between the system performance and the runtime is crucial for online scenarios. While some of the works contribute by adding more modules to achieve improvements, we propose a pruned model by leveraging the state-of-the-art Transformer backbone model. Our model saves up to 62% FLOPS compared with other Transformer-based models and almost as twice as fast as them. The results of the proposed model are still competitive among the state-of-the-art methods. Moreover, we will open-source our modified Transformer backbone model for general CV tasks as well as the MOT system.

1. Introduction

As a cornerstone of diverse CV tasks, MOT is a significant prerequisite. For instance, public surveillance, autonomous vehicles, and video analysis, where MOT's performance is critical for the subsequent analysis.

Analogizing humans paying attention to the targets when tracking objects, Transformer's attention mechanism is an advisable solution for MOT (Vaswani et al., 2017). Some works have leveraged the novel Transformer-based detector for MOT, such as TransTrack (Sun et al., 2020) and TrackFormer (Meinhardt et al., 2022). It turns out that the key-query mechanism in the Transformer is capable of focusing attention on massive objects and simplifying the tracking systems from tracking-by-detection to jointly-detection-and-tracking.

Nonetheless, current Transformer-based solutions boost the overall model sizes by additional modules for tracking, as explained in Fig. 1. Indeed, it is beneficial to patch the existing solutions with novel techniques to make them more robust and reasonable. However, we also need to consider each module's purpose and effectiveness by breaking down the structure instead of using the whole model.

On the other hand, ResNet, which was proposed in 2015 (He et al., 2016), is an old but still prevalent and effective convolutional backbone for various CV tasks. Although it is famous for its innovative residual connection, it is arguable whether it is prominent over recent competitors.

This paper proposes a new standalone end-to-end tracking model to address the previous issues. Instead of following the conventional model paradigm and producing a mixed model, we create a new architecture by investigating the mechanism of Transformers deeply and keeping it compact. Firstly, we adopt an encoder-only Transformer. It works as a feature extraction backbone and an encoder to embed the object's spatial context. Next, only the decoder of the DETR model is used to decode the features and predict the object locations and classes (Zhu et al., 2021). In this way, we follow the general architecture of encoder–decoder Transformers since our problem is an iterative prediction of objects and removes the ambiguous module.

In our unified tracking Transformer, we keep the tracking-by-attention paradigm, which jointly detects and tracks objects using the same queries. As an online system, the encoder takes the image of the current time step, extracting the objects' features and encoding them into a memory embedding. Next, the decoder regresses the objects' locations and classes by the attention mechanism using learned object queries and memory. The regressed queries of detected objects will be added to the tracking queries in the next time step. Subsequently, the decoder will process all detection and tracking queries auto-regressively. Hence, jointly-detection-and-tracking is done within the system using an extended query mechanism.

We evaluate our model with the state-of-the-art Transformer-based models on the MOT17 dataset during the experiment. Our model

* Corresponding author.
E-mail addresses: Shan.Wu@ut.ee (S. Wu), hadachi@ut.ee (A. Hadachi), chaorulu@oslomet.no (C. Lu), damien.vivet@isae-supaero.fr (D. Vivet).

https://doi.org/10.1016/j.aiopen.2023.09.002
Received 19 July 2023; Received in revised form 12 September 2023; Accepted 28 September 2023
Available online 5 October 2023

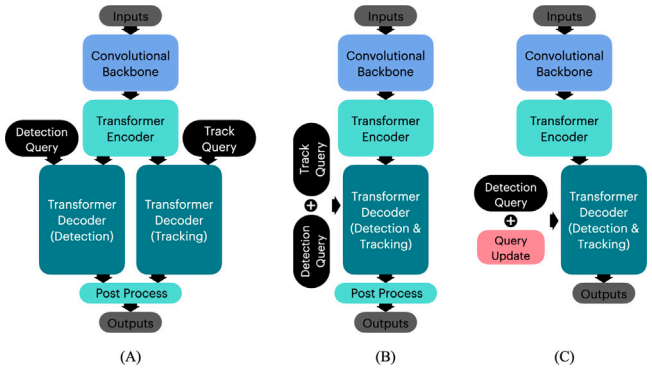

Fig. 1. A visualization of the architectures of the state-of-the-art Transformer-based MOT systems comprising a convolutional backbone, a Transformer encoder, and up to two Transformer decoders. (A) Sun et al. (2020) uses two decoders dedicated to detection and tracking, respectively. However, Transformer is a heavy module in terms of computation. Two decoders would be too bulky. (B) Meinhardt et al. (2022) is the most straightforward model where the detected queries are reused as tracking queries. (C) Zeng et al. (2022) and Xu and Vivet (2021) implemented a query update module for better-engineered query embedding. Still, (B) and (C) all combine two distinct models requiring more computational resources than ours.

achieves prominent results compared with the Transformer-based competitors while the model size shrunk by up to 30.7%. Ablation research is conducted to show the effectiveness of our new architecture. In this way, we hope the researchers can go further in this direction with the unified tracking model and exploit more *Transformers* and *green learning* for CV.

In summary, we listed our contributions as follows:

- A modified open-sourced[1] Transformer backbone, which can be a substitution of convolutional backbones in any CV tasks for arbitrary image sizes.
- A new unified MOT Transformer, namely *MOTT*, which is significantly efficient while maintaining competitive performance.
- We took a step towards a solely Transformer-based tracking model without relying on any convolutional model, which minimized both the size and the computational cost for green learning.

2. Related work

Over the past few years, a plentiful of work has been done by researchers to enhance the performance of multi-object tracking (MOT) systems. Because MOT is a complex task involving both object detection and object tracking, we categorize the literature into two main domains: tracking-by-detection and joint-tracking-and-detection.

2.1. Tracking-by-detection

These approaches mainly focus on the tracking task using existing detections from other detectors.

In the early stages, works like Sanchez-Matilla et al. (2016) and Bewley et al. (2016) solely rely on the detection boxes and their probabilities. The tracking is achieved by data association using a particle filter framework or a bipartite matching between detected boxes and the estimated boxes. They perform moderately but suffer from low accuracy and identity switches because of the density of objects and the occlusions.

The evolution of machine learning and the involvement of image features in MOT systems remarkably boosted tracking performance. As many robust object detectors emerge (Ren et al., 2015; Duan et al., 2019; Redmon and Farhadi, 2018), some works (Wang et al., 2021; Sheng et al., 2018) integrate their data association techniques with existing detectors. For instance, Leal-Taixé et al. (2016) leveraged the Siamese network for auto-regressed data association. Wojke et al. (2017) integrated appearance information using a pre-trained network to mitigate the identity switch issue. It is important that a good detection result will contribute to the tracking performance. Nonetheless, tracking objects using several models would be less efficient and require more computational resources.

2.2. Joint-tracking-and-detection

By performing detection and tracking simultaneously, the tracking process could happen earlier. Zhang et al. (2021a) employed an encoder–decoder network for feature extraction. Both the detector and re-ID module take the feature embedding of the same origin simultaneously. However, an additional tracking step is needed using feature fusion and Kalman filters.

CTracker (Peng et al., 2020b) proposed an end-to-end neural network model to detect objects among three consecutive frames followed by an IoU matching for the boxes on the middle frame. Objects are chained by a sliding time window of three frames. This method facilitates the training process but requires considerable power to perform detection on multiple frames.

Due to the success of the Transformer architecture (Vaswani et al., 2017), many researchers find it a good model for MOT tasks because it simplifies the model structure, the data association process, and the training procedure. DETR (Carion et al., 2020) established the concept of object queries as a fixed number of learnable embeddings, which is acknowledged by most of the research works using Transformer in Computer Vision (CV). These queries will be transformed into output embeddings in the decoder, followed by a post-process decoding them into bounding boxes and class labels. The object query mechanism built in Transformers greatly facilitates the development of object detection and tracking.

TransTrack (Sun et al., 2020) is one of the first models implementing an end-to-end Transformer-based MOT architecture derived from deformable DETR (Zhu et al., 2021), a cornerstone of the Transformer detector. The TransTrack duplicates the decoder of deformable DETR, utilizing one for detection and another for tracking. This method is straightforward without additional data association. However, the backbone and the second decoder make it an enormous model.

Meinhardt et al. (2022), Zeng et al. (2022) and Xu and Vivet (2021) improved a step further by tracking-by-query mechanism, which only involves the backbone and a full deformable DETR in this paradigm. Object queries are categorized into detection queries and tracking queries. The deformable decoder will decode all types of queries at once. Hence, detection and tracking processes happen concurrently. Despite the simplification of the model, they are still a combination of two complete models.

2.3. The transformer

Since the Transformer came to CV, it developed into two directions as two well-known CV tasks: object classification and object detection. In object detection, Transformers keep its original architecture (Zhu et al., 2021; Carion et al., 2020), an encoder–decoder model, because object queries are necessary to populate instances. Zhu et al. (2021) is the state-of-the-art of this category, which reconstructs the multi-head attention and leverages multi-level features.

In object classification, the structure of Transformers (Dosovitskiy et al., 2020; Liu et al., 2021; Dong et al., 2022) are as plain as an encoder because attention is one of the substitutions of the receptive field as in the convolutions. Instead of attending features globally, these works exploit local attention, making them state-of-the-art backbones for feature extraction.

[1] GitHub link: https://github.com/simonwu53/MOTT.

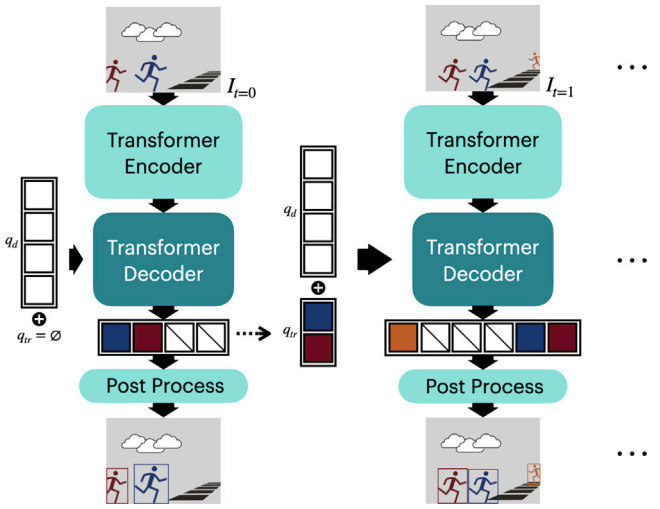

Fig. 2. Architecture of proposed *MOTT* model, where I is the input image, t is the time step, q_d and q_{tr} denote the detection queries and tracking queries, respectively.

Both feature extraction and object detection are crucial components in MOT systems. By leveraging Transformer encoders and decoders independently, we could further minimize the model by proposing *MOTT*, a single Transformer-only architecture for MOT. Consequently, we will reduce the power consumption of training and testing as a step forward to the green learning model (Strubell et al., 2019).

3. Methodology

It is known to all that a full encoder–decoder-based Transformer is good at sequence prediction problems by the interaction between object queries and memory embedding. We formulate MOT as a sequence prediction as other Transformer-based MOT systems. Based on the characteristics of different types of Transformers, we create a customized encoder–decoder architecture specialized in MOT, namely *MOTT*, as shown in Fig. 2. The model adheres to a general encoder–decoder Transformer architecture, where the encoder also functions as a backbone to extract features and encode information simultaneously. At every time step, an image I is fed into the model for feature extraction and encoding. The output of the encoder will be queried by learnable object queries comprised of fix-sized detection queries q_d and tracking queries q_{tr} in the decoder. In the post-process, Multi-Layer Perceptron (MLP) modules will process the decoded queries, which hold the target objects' appearance information to predict the locations and classes of objects. We will discuss all components in the following subsections.

3.1. Object queries

Transformers need a query for the sequence prediction. A query is a one-dimensional embedding with a hidden space of d_h. Similar to the work of Meinhardt et al. (2022) and Zeng et al. (2022), we leverage two types of object query q – detection query q_d and tracking query q_{tr} (shown in Eq. (1)). The decoder performs the track association on the concatenation of q_d and q_{tr} using deformable attentions (Zhu et al., 2021).

$$\begin{aligned} q &= \{q_{tr}, q_d\} \\ q_d &= \{q_t^i \mid \forall\, i \in N_{obj}\} \\ q_{tr} &= \{q_{t-1}^i \mid i \subset N_{obj}\} \end{aligned} \tag{1}$$

where t is the time step, N_{obj} is the length of detection query. q_{tr} is a subset of q_d from the previous time step.

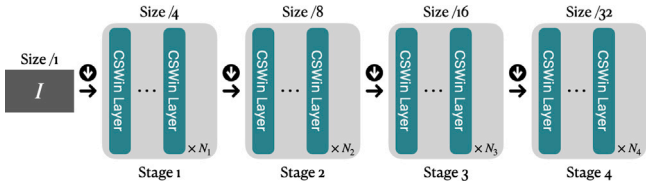

Fig. 3. Diagram shows the general architecture of CSWin model including four stages, where each stage has a predefined number of CSWin layers N_i. ↓ denotes a downsampling layer, where $/n$ means the size of the feature map compared with its original input size.

3.1.1. Detection query

Our model has a learnable query embedding N_{obj} of a fixed length, namely detection queries q_d. It is initialized by a normal distribution and gradually optimized to store the appearance attributes of target objects during the training process (white boxes in Fig. 2). The optimized q_d will be processed in the decoder to update its content to the new objects' attributes for every incoming image (colored boxes in Fig. 2).

In practice, the parameter N_{obj} defines how many variants of target objects to learn. It is much larger than the maximum number of objects in each frame to acquire as much detection as possible. Another parameter d_h determines the number of appearance attributes for every independent instance.

3.1.2. Tracking query

When one of the detection queries q_t^i at time step t detects an object, it will be kept as a tracking query for the next iteration (the boxes in the same color across frames as shown in Fig. 2). As previously mentioned, this query carries the appearance attributes of a specific object through attention operations. Hence, we leave all queries that have valid detection unaffected in the next frame.

tracking queries are always selected and concatenated with new detection queries to form the object queries at every iteration, as shown in Fig. 2. The model consumes a fluctuating number of object queries for all subsequent frames.

3.1.3. Query update

Let $g_t = \{b_{g,t}, s_{g,t}\}$ denotes the detection of an object at time step t with its bounding box $b_{g,t}$ and a score $s_{g,t}$. If the score of a detected object is larger than a threshold σ_{det}, the corresponding detection query is validated and kept for tracking. In the next time step, this tracking query will be responsible for g_{t+1}, where the score $s_{g,t+1}$ will be compared with a tracking threshold σ_{tr}. The model keeps updating this tracking query using the latest input frame until the score $s_{g,t+n}$ is lower than σ_{tr}, and the tracking query will be removed from the tracking queries.

There are occasions that a detection query does not find a valid object ($s_{g,t} < \sigma_{det}$), and the detection will be discarded (crossed boxes in Fig. 2). During the testing phase, the duplicated detection will be further reduced by a Non-Maximum Suppression (NMS) threshold σ_{nms}.

3.2. Transformer encoder

In this work, we leverage a novel Transformer encoder called CSWin (Dong et al., 2022), a general-purpose vision backbone. We select it for three reasons: it has a better performance compared with conventional ResNet backbone; it is a Transformer backbone, also an encoder, which significantly diminishes the model parameters and computations compared with the mixture of a backbone and an encoder; it is fully open-sourced.

The CSWin model, as shown in Fig. 3, is an encoder-only model with four stages like ResNet, where each stage has a different amount of encoder layers N_i, and the size of output feature maps will be shrunk by

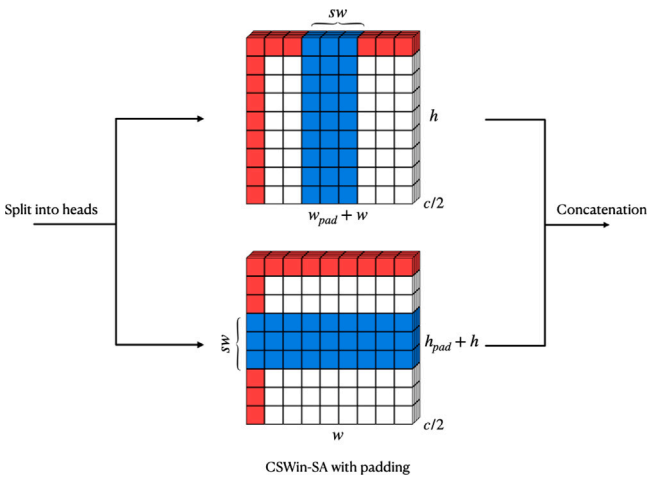

Fig. 4. Structure of the amended cross-shaped window self-attention (CSWin-SA) module in each CSWin layer. The red stripe is the zero-padding added to the feature map, while the blue stripe denotes the local range of the self-attention mechanism.

two after every stage. Within each layer is a cross-shaped window self-attention (CSWin-SA) module, dividing the feature maps into two heads by depth (as shown in Fig. 4). A dynamic stripe window sw divides the feature map into smaller stripes vertically and horizontally, followed by the CSWin-SA applied on each stripe in parallel. Hence, the module only attends to the local context within the stripes to extract features. sw is wider as the stage goes deeper because the receptive field is larger. Later, these features will be merged into global features using a linear layer.

However, this model is limited to specific datasets like ImageNet (Deng et al., 2009) and COCO (Lin et al., 2014) because of the split size sw in the CSWin-SA module. The module will not work when the sizes of feature maps cannot be divided by sw evenly. This constraint remarkably hinders the downstream CV tasks. Thus, we propose a padding mechanism to make it work with arbitrary images as shown in Fig. 4.

For an arbitrary feature map $f_i^{c_i \times h_i \times w_i}$ of stage $i \in [1,4]$ and the sw_i of stage i, if any side l of the feature map cannot be divided by sw_i evenly (e.g. $h_i \bmod sw_i \neq 0$), we add zero-paddings to that side using Eq. (2) before the CSWin-SA module.

$$\begin{aligned} l_{pad} &= sw_i - (l \bmod sw_i) \\ \forall l \in \{h_i, w_i\} \ &\wedge \ l \bmod sw_i \neq 0 \\ l_a &= l_{pad} \mid 2 \\ l_b &= l_{pad} - l_a \end{aligned} \tag{2}$$

where l_{pad} is the total padding size for both extremities of a side l. l_a and l_b are the padding sizes of each extremity.

Local feature extraction is convolutional networks' specialty as they can easily extract features like eyes, noses, and ears by limiting their receptive field. This differs for the Transformers because the attention mechanism focuses on the whole input. CSWin-SA controls its receptive field by dynamic sw strengthening local feature extraction while drawing the global features from other heads.

Besides, our modified backbone provides multi-level feature maps collected from the last layer of every stage for CV tasks requiring feature pyramids. The feature pyramid comprises two-dimensional feature maps $\chi = \{f_i^{c_i \times h_i \times w_i} \mid i \in [1,4]\}$.

In this work, we leverage the last three stages' feature maps ($i \in \{2,3,4\}$) along with an additional feature map downsampled by a convolution layer ($kernel = 3,\ stride = 2,\ pad = 1$) using the last stage's feature map f_4 to enrich the spatial context in different scales. Eq. (3) summarizes the output features we use in all models where CSWin is embedded.

$$\begin{aligned} \chi_e &= \{f_i^{c_i \times h_i \times w_i} \mid i \in \{2,3,4,5\}\} \\ \text{where } f_5^{c_5 \times h_5 \times w_5} &= \text{Conv}(f_4^{c_4 \times h_4 \times w_4}) \end{aligned} \tag{3}$$

3.3. Transformer decoder

Before the interaction with queries in the decoder, multi-level feature maps need to be preprocessed. As shown in Eq. (4), feature maps will be projected into d_h channels to match the queries. Next, two spatial dimensions of the feature maps will be merged, followed by a concatenation of all flattened features generating a memory embedding χ_{enc} for the decoder.

$$\begin{aligned} \chi_p &= \{f_i^{c_i \times h_i \times w_i} \rightarrow f_i^{d_h \times h_i \times w_i} \mid \forall f_i \in \chi_e\} \\ \chi_f &= \{f_i^{d_h \times h_i \times w_i} \rightarrow f_i^{d_h \times hw_i} \mid \forall f_i \in \chi_p\} \\ \chi_{enc} &= \text{Concat}(\chi_f) \end{aligned} \tag{4}$$

Decoders are intended for performing detection and tracking simultaneously using object queries and memory embedding. In order to decode massive queries while keeping the computational cost low, we only select the decoder from the deformable DETR (Zhu et al., 2021). This decoder can regress the bounding boxes using its efficient Multi-scale Deformable Attention module (MSDAttn), whose computational complexity is irrelevant to the input size.

MSDAttn is capable of attending memory embedding χ_{enc} containing multi-level features, where each level's feature embedding f_i will go through deformable attention (DAttn) as shown in Eq. (5). DAttn comprises m heads of attention, which only attends to a subset of the feature embedding $\hat{f}$ sifted by reference points p, and its sampling offsets Δp. Both Reference points and sampling offsets are derived from query embedding q', and object queries q using linear layers, respectively. Query embedding q' is a trainable positional encoding (Vaswani et al., 2017) dedicated to the decoder.

$$\begin{aligned} \text{DAttn}(q, q', f) &= \sum_m^M W_m \text{Attn}(q, q', f) + b_m \\ \text{Attn}(q, q', f) &= A \cdot \hat{f} \\ A &= \text{Softmax}(W_a q + b_a) \\ \hat{f} &= (W_v f + b_v)(\pi(p) + \Delta p) \\ p &= W_{q'} q' + b_{q'} \\ \Delta p &= W_q q + b_q \end{aligned} \tag{5}$$

where A is the attention weight, W and b are the projection weights and bias. π is a function rescaling the reference points to its feature maps' scales. The final output will become the new object queries q with updated appearance attributes of the current frame.

3.4. Post process

For every query from the decoder, the bounding box is predicted by a three-layer MLP module, while the class label is generated from a fully-connected layer directly. Because the MSDAttn utilizes reference points for attention, the predicted bounding boxes are the offsets relative to the center of reference points.

In this implementation, N_{dec} cloned MLP modules are created for iterative bounding box refinement, where N_{dec} is the number of decoder layers. These MLP modules will be injected into the decoders to update the coordinates of reference points on every decoder layer.

3.5. Re-Identification (Re-ID)

An object usually will not appear continuously among the frames. It may be occluded by another object or temporarily out of the frame. Thus, we do not terminate a tracking query if the score $s_{g,t}$ at time

step t is lower than the track threshold σ_{tr}. Instead, this tracking query is marked inactive, and we keep it in the list of tracking queries for another T_{reid} time steps. Thus, the model can still re-identify an inactive track object within a small time window if the track score $s_{g,t+n}$ is larger than a Re-ID threshold σ_{reid}.

Because inactive tracking queries cannot be updated effectively in the decoder when targeting objects are not visible, the model cannot recover a long-term inactive tracking query or inactive tracking queries with drastic appearance changes. Nonetheless, the model acquires Re-ID ability because of the Transformer's attention mechanism on object queries without additional techniques and training.

3.6. Training

Initially, the model only understands the detection and tracking queries once it was trained in simulated scenarios of tracking objects between two consecutive frames. Thus, we train the model using supervised learning as proposed in Meinhardt et al. (2022). A training cycle consists of two model forward propagations:

1. The model performs purely object detection with detection queries q_d on frame $t-1$ ($t=0$ in Fig. 2). Select partial of the outputs to be the track targets.
2. The model performs joint object detection and tracking with both detection queries q_d and tracking queries q_{tr} on frame t ($t=1$ in Fig. 2).

The loss is calculated for all predictions in the second forward propagation using bipartite matching (Hungarian algorithm (Kuhn, 1955)), which sets up one-to-one matching between predictions and ground truths. In addition, there are three cases when assigning ground truth targets to the predictions. Let $\mathcal{P}_t = \{y\}_{j=1}^{N}$ be a collection of N ground truth targets y for frame t.

- $\mathcal{P}_t - \mathcal{P}_{t-1}$: these are new objects appears in frame t and bipartite matching is used for those objects.
- $\mathcal{P}_{t-1} - \mathcal{P}_t$: these are terminated objects which become invisible or occluded in frame t. Thus, the background class is assigned to these predictions.
- $\mathcal{P}_t \cap \mathcal{P}_{t-1}$: these are objects in both frames. The connection between the prediction and the ground truth is hard linked by its track id.

Following Meinhardt et al. (2022), Sun et al. (2020) and Carion et al. (2020), we calculate the loss based on cases whether a prediction is matched by a ground truth as shown in Eq. (6):

$$\mathcal{L}(\hat{y}, y) = \sum_{i=1}^{N_{obj}} \left[\lambda_{cls}\mathcal{L}_{cls} + \mathbb{1}_{j=\varphi(i)}\mathcal{L}_{box}\right] \tag{6}$$

where $\hat{y}$ and y are the prediction and ground truth objects, respectively. λ is a factor adjusting the balance between class loss $\mathcal{L}_{cls}$ and box loss $\mathcal{L}_{box}$. φ is bipartite matching performed by Hungarian algorithm and $\mathbb{1}_{j=\varphi(i)} = 1$ if $\hat{y}_i$ is matched to y_j. The class loss and box loss are defined as Eq. (7):

$$\begin{aligned} \mathcal{L}_{cls} &= -\log \hat{p}_i(c_{\varphi(i)}) \\ \mathcal{L}_{box} &= \lambda_{\ell 1}\|b_{\varphi(i)} - \hat{b}_i\|_1 + \lambda_{iou}\mathcal{C}_{iou}(b_{\varphi(i)}, \hat{b}_i) \end{aligned} \tag{7}$$

where $\hat{p}_i(c_{\varphi(i)})$ is the predicted class probability of matched class $c_{\varphi(i)}$. If $\hat{y}_i$ has no matched ground truth, we give it a background label ($c_{\varphi(i)=\varnothing} = 0$). For the bounding boxes, a combination of $\ell 1$ loss and generalized IoU loss (Rezatofighi et al., 2019) is used.

In addition to the regular supervised training procedure mentioned above, several data augmentations techniques are used as in Meinhardt et al. (2022) and Sun et al. (2020), such as tracking with larger time steps, false positive/negative for object queries, and conventional image augmentations.

4. Experiments

4.1. Dataset

We conducted our experiments on the MOT17 dataset (Milan et al., 2016) provided by MOTChallenge Benchmark to compare with other competitors. It is a multi-object tracking dataset favored by most of the works we researched. The dataset contains 14 videos in total, where seven are for training and the remaining seven are for testing. Noted that there are only annotations for training sequences. In order to get evaluation results on testing sequences, one must create a new entry on their benchmark and submit the tracking results to their server to get the scores. Many works include additional training data and tuning to get a competitive score. Hence, in our experiments, we use data differently in order to control variables:

- Benchmark: All training data, including additional ones (mention below).
- Comparison: Similar to Sun et al. (2020) and Zhou et al. (2020), we split the training data into two halves — one for training and the other part for validation and comparison.

In addition to data splits, MOTChallenge also provides two tracks for competition: public and private detections. Methods based on public detection only associate spatiotemporal tracklets using the detections provided in the dataset. Private-detection methods perform detection and tracking by themselves. Hence, our method is an online private one that performs one-shot prediction for every incoming frame.

As mentioned before, CrowdHuman dataset (Shao et al., 2018) is another dataset we used for benchmarking. Because the MOT17 dataset is relatively small, CrowdHuman could be a great complement. CrowdHuman dataset focuses on human detection, which has 15 000, 4370, and 5000 images for training, validation, and testing, respectively. Although it is not a sequence-based dataset, we can imitate two consecutive frames by cropping and shifting the original image.

To test the capability of the proposed *MOTT*, we also evaluated it with the MOT20 dataset (Dendorfer et al., 2020) and DanceTrack (Sun et al., 2022). MOT20 features a tremendous number of people in a single image, while DanceTrack focuses on different people's poses.

4.2. Metrics

All tracking performances are measured by MOT metrics (Bernardin and Stiefelhagen, 2008) widely acknowledged among other works. Specifically, there are Multi-Object Tracking Accuracy (MOTA), ID F1 score (IDF1), False Positives (FP), False Negatives (FN), Mostly Tracked targets (MT), Mostly Lost targets (ML), and the number of Identity Switches (IDs).

Among them, MOTA (shown in Eq. (8)) measures the overall accuracy of the tracking, taking into account FP, FN, and IDs. IDF1 ranks all methods on the same scale showing the balance of identification precision and recall.

$$\text{MOTA} = 1 - \frac{\sum_t(\text{FP}_t + \text{FN}_t + \text{IDs}_t)}{\sum_t y_t} \tag{8}$$

where y_t is the number of ground truth boxes at time step t.

4.3. Implementation details

We trained several models for a comprehensive comparison. Firstly, we trained vanilla TrackFormer (Meinhardt et al., 2022) and TransTrack (Sun et al., 2020) as two online private competitors. Secondly, we substitute the ResNet backbone in TrackFormer with our modified CSWin model to test the ability of the state-of-the-art Transformer backbone (denoted as TrackFormer-CSWin). Thirdly, our proposed efficient model *MOTT*.

Table 1
Four models are compared in terms of the number of parameters (#Params), total CUDA time used, and averaged FLOPS.

Model	#Params (M) ↓	CUDA time total (s) ↓	Avg. FLOPS (G) ↓
TransTrack	46.9	8.17	428.69
TrackFormer	44.0	13.67	674.92
TrackFormer-CSWin	38.3	16.26	714.83
MOTT	32.5	6.76	255.74

In this paper, we use the tiny variant of CSWin in all corresponding models, which features 25 layers with an initial embedding size of 64 and produces a three-level feature pyramid from stages 2, 3, and 4. The encoder is pre-trained with COCO dataset (Lin et al., 2014). For the decoder, we leverage deformable DETR's decoder, featuring 6 layers with an embedding size of 288 and 8 attention heads. It takes a four-level feature pyramid, where the last level is derived from the result of the last stage of the CSWin model by a convolutional layer. Besides, we set the number of detection queries to 500. The total number of object queries varies based on how many objects are tracked among frames.

For the benchmark model, the model is pre-trained on the CrowdHuman dataset for 80 epochs. Then, we train the model on a mixed dataset combining MOT17 and CrowdHuman for 40 epochs. For the comparison models, we pre-train the models as before. Next, all models are trained on the first half of the MOT17 training set for 40 epochs and evaluated on the second half. During all training, we use AdamW optimizer (Loshchilov and Hutter, 2019) with the initial learning rate at 2×10^{-4}. The learning rate drops by a factor of 10 at 50 epochs for the pretraining stage; Then, it drops again at 10 epochs for the second stage. Because the encoder is initialized with pre-trained weights, we reduce its learning rate to 2×10^{-5}. The input samples are randomly cropped and resized so that the shorter edges range from 480 to 800 pixels and the longer edges not exceeding 1333 pixels.

4.4. Computing efficiency

As a backbone, CSWin (the tiny variant we adopted) requires slightly more floating point operations per second (FLOPS) with 4.3×10^9 compared to ResNet50 with 3.8×10^9 used in all competitors on the ImageNet dataset. Nonetheless, it surpasses ResNet50 in object detection and segmentation tasks by a large margin on the COCO dataset in the experiment conducted by Dong et al. (2022).

It is known to all that the computation cost of the attention mechanism is extremely high in the image domain because of the high-resolution images (Zhu et al., 2021). The computational complexity of multi-head attention is $O(N_q C^2 + N_k C^2 + N_q N_k C)$, where N_q, N_k, and C are the length of (object) query, key (flattened feature map), and depth (channel), respectively. In the DETR encoder (Carion et al., 2020), where only self-attention is applied, the complexity will grow quadratically regarding the image size (H, W) to $O(HWC^2 + H^2W^2C)$ where $N_q = N_k = HW$.

CSWin self-attention, on the other hand, only requires $O(HWC^2 + swH^2WC + swHW^2C)$ due to its paralleled cross-shaped attention window. The computation complexity can be adjusted dynamically by the stripe size sw. We can select small sw in shallow layers when HW is large and gradually increase sw in deeper layers while HW is shrinking through stages. Consequently, the receptive field varies from small to large, akin to stacked convolution layers.

In addition, we measured the FLOPS of different models on sequence 2 of MOT17 using PyTorch's profiler (Paszke et al., 2017) in testing mode. All frames are reshaped into 800 pixels in width.

As shown in Table 1, *MOTT* has the least trainable parameters, which is the most lightweight among the Transformer-based models. Besides, it requires around 60% of the FLOPS compared with TransTrack and even less than 38% of FLOPS than TrackFormer. During the evaluation, *MOTT* is the fastest in CUDA execution time, achieving the best efficiency.

In summary, *MOTT* surpasses other Transformer-based models regarding computing efficiency, which benefits carbon footprint reduction and green learning. Besides, as shown in the following subsection, we only use one encoder to substitute the backbone-encoder combination and gain extra performance.

4.5. Ablation study

In this section, we reveal the effects of each module in *MOTT* along with the TrackFormer, which is our baseline model. Table 3 presents the performance impact of each module. The base model consists of a ResNet-50 backbone, a deformable encoder, and a deformable decoder. When we change only the backbone to CSWin, the model achieves a considerable performance boost from 66.8% to 72.7% in MOTA and from 70.7% to 72.9% in IDF1, which shows the capability of the novel Transformer-based backbone.

However, CSWin is more computationally-intensive than ResNet, as we previously reviewed, making the system even slower. Since CSWin and the deformable encoder are pursuing the same objective, we remove the deformable encoder entirely and get on par performance at 71.9% in MOTA and 72.6% in IDF1. Furthermore, the speed of the final model almost doubled.

The reason we remove the deformable encoder module is that it is not efficient. We can barely get any improvements from such a big module. Tracking happens inside the decoder by transforming object queries into embeddings holding information from tracked objects. The key, or flattened feature map, is required by object queries in a decoder. However, we cannot use the feature map from the ResNet backbone directly because it has to be encoded. *MOTT* is different since our backbone is an encoder itself. Thus, we choose to remove the deformable encoder. The performance shows that CSWin is capable of extracting features as well as encoding by the cross-shaped self-attention mechanism.

Additionally, the deformable encoder still puts attention over the entire image, demonstrating a deficiency in local attention. Conversely, local attention learns local relations and structure information (Han et al., 2021; Wu et al., 2023), which is essential in CV. Still, the deformable decoder's complexity is irrelevant to the spatial size and makes it an efficient module for decoding object queries.

4.6. Evaluation on different datasets

Firstly, we run the benchmark on the MOT17 testing set with our *MOTT* following the training steps in 4.3. The results from the MOTChallenge leaderboard are reported in Table 2. Our method takes a single image at each time step, and outputs detected and tracked objects. Thus, it is categorized as a private online method in the table.

MOTT achieves competitive results with the state-of-the-art methods. The highest score in MOTA means an overall better capability in MOT. MT and ML show the ability to track objects among frames. Compared to the second-best results, our method gets 4.9% more in MT and 9.1% less in ML. FN is another essential criterion because catastrophic situations may happen if the system misses an object in an unmanned scenario. Our model keeps a low FN result by providing much more detections, which also contributes to the occasions of the identity switches. Another reason for frequently making identity switches is that it cannot memorize an inactive object long-term with the current simple tracking query mechanism.

In Table 4, we compare our method with the novel Transformer-based methods in the same training procedure on the same MOT17 half-training set. The *MOTT* outperforms the other two methods by a noticeable margin. Due to the local and global strip-shaped attention mechanism of the new Transformer encoder, the model can track more objects precisely than other attention structures. The deformable

Fig. 5. Visualization of *MOTT* on MOT17, MOT20, and DanceTrack datasets.

Table 2
Benchmark results on MOT17 testing set. We list both public and private detection methods published on the benchmark leaderboard. All the methods are categorized into online and offline methods. Bold numbers are the best results.

Public detection								
	Method	MOTA ↑	IDF1 ↑	MT (%) ↑	ML (%) ↓	FP ↓	FN ↓	IDs ↓
offline	jCC (Keuper et al., 2018)	51.2%	54.5%	20.9%	37.0%	25,937	247,822	**1802**
	TPM (Peng et al., 2020a)	54.2%	52.6%	22.8%	37.5%	13,739	242,730	1824
	Sp_Con (Wang et al., 2022)	61.5%	63.3%	26.4%	32.0%	14,056	200,655	2478
	HTracker (Zhang et al., 2021b)	66.9%	**70.4%**	28.3%	20.8%	30,704	151,001	4806
online	Tracktor++ (Bergmann et al., 2019)	53.5%	52.3%	19.5%	36.6%	12,201	248,047	2072
	MPTC (Stadler and Beyerer, 2021)	62.6%	65.8%	26.6%	31.8%	**8824**	198,338	4074
	HUGMOT (Wan et al., 2021)	64.8%	62.8%	31.3%	27.4%	16,174	180,371	2102
	PixelGuide (Boragule et al., 2022)	69.7%	68.4%	38.3%	26.1%	26,871	140,457	3639
Private detection								
online	Tube_TK (Pang et al., 2020)	63.0%	58.6%	31.2%	19.9%	27,060	177,483	4137
	CTracker (Peng et al., 2020c)	66.6%	57.4%	32.2%	24.2%	22,284	160,491	5529
	CenterTrack (Zhou et al., 2020)	67.8%	64.7%	34.6%	24.6%	18,489	160,332	3039
	QuasiDense (Pang et al., 2021)	68.7%	66.3%	40.6%	21.9%	26,589	146,643	3378
	MOTT (ours)	**73.2%**	66.0%	**45.5%**	**10.8%**	35,859	**111,711**	3651
	Method	MOTA ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDs ↓

Table 3
The ablation study shows the performance differences when gradually removing the components. Notations: Res = ResNet50, CSWin = CSWin-tiny, DE = Deformable Encoder, DD = Deformable Decoder.

Modules aval.	MOTA ↑	IDF1 ↑	Hz ↑
Res+DE+DD (TrackFormer)	66.8%	70.7%	5.39
CSWin+DE+DD	72.7%	72.9%	4.73
CSWin+DD (**MOTT**)	71.9%	72.6%	9.09

Table 4
Comparison among Transformer-based methods. All models are trained using the same dataset and procedure.

Method	MOTA ↑	MOTP ↑	IDF1 ↑	MT ↑	ML ↓
TransTrack	66.5%	83.4%	66.8%	134	61
TrackFormer	67.0%	84.1%	69.5%	152	57
MOTT (ours)	71.6%	84.5%	71.7%	166	41

decoder is also efficient and great at decoding the queries based on selected key embedding. To be mentioned, the proposed model is much lighter than the others considering the number of parameters and FLOPS.

Next, we evaluated *MOTT* on the testing sets of MOT20 and DanceTrack in addition to MOT17. Table 5 unveils more insights into our model from different perspectives. The MOT17 and MOT20 results are obtained from the model trained by corresponding datasets, while the DanceTrack results are derived from the MOT20 model without fine-tuning. As mentioned, MOT20 contains much more pedestrians per frame. Hence, the model gets a lower MOTA score due to increasing false negatives (higher ML score). *MOTT* misses a few of them but detects most pedestrians successfully (high MOTP score). The reason could be the non-maximum suppression (NMS) deployed after detection to eliminate duplicating bounding boxes as many pedestrians are occluded. DanceTrack comprises dancing video sequences. It is relevantly easy to detect people on the stage (higher MT and MOTA scores), but ID switching happens frequently (lower IDF1 score) because of various dance formations. Nonetheless, the evaluation of DanceTrack is

Table 5
MOTT performance on MOT17, MOT20, and DanceTrack dataset. MT and ML are calculated in percentages instead of absolute values.

Dataset	MOTA ↑	MOTP ↑	IDF1 ↑	MT ↑	ML ↓
DanceTrack	85.4%	81.9%	33.7%	81.5%	0.3%
MOT20	66.5%	81.1%	57.9%	52.1%	13.8%
MOT17	71.6%	84.5%	71.7%	49.0%	12.1%

achieved by the MOT20 model, which means *MOTT* can be generalized in various use cases.

Finally, we show the model outputs in Fig. 5, corroborating the statistics in Table 5. In MOT17, the model manages to track most people but has low confidence in riding people due to insufficient training data in this case. MOT20 shows the capacity of *MOTT*. Because of NMS and occlusion, few bounding boxes among the crowd are eliminated. Furthermore, *MOTT* detects people in various poses, demonstrating invariance to translation, rotation, and scale by learning from the MOT20 dataset.

5. Conclusion

In this paper, we proposed a new Transformer-based MOT architecture, namely *MOTT*, which could save much on the hardware cost and energy while retaining the state-of-the-art MOT performance. By only leveraging the effective modules based on its specification, the new model only contains an encoder and a decoder for the challenging MOT task. Our model achieves a competitive score in MOTA at 73.2% with up to 62% fewer FLOPS than a typical Transformer-based MOT model. The model shows the potential of moving towards a green learning paradigm in CV tasks. In future work, we will focus on improving the object queries to solve the issues with re-identification. We hope this new architecture could enlighten others on the balance of performance and computational cost.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Bergmann, P., Meinhardt, T., Leal-Taixe, L., 2019. Tracking without bells and whistles. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 941–951.

Bernardin, K., Stiefelhagen, R., 2008. Evaluating multiple object tracking performance: the clear mot metrics. EURASIP J. Image Video Process. 2008, 1–10.

Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B., 2016. Simple online and realtime tracking. In: 2016 IEEE International Conference on Image Processing (ICIP). IEEE, pp. 3464–3468.

Boragule, A., Jang, H., Ha, N., Jeon, M., 2022. Pixel-guided association for multi-object tracking. Sensors 22 (22), 8922.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S., 2020. End-to-end object detection with transformers. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16. Springer, pp. 213–229.

Dendorfer, P., Rezatofighi, H., Milan, A., Shi, J., Cremers, D., Reid, I., Roth, S., Schindler, K., Leal-Taixé, L., 2020. Mot20: A benchmark for multi object tracking in crowded scenes. arXiv preprint arXiv:2003.09003.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L., 2009. ImageNet: A Large-Scale Hierarchical Image Database. In: CVPR09.

Dong, X., Bao, J., Chen, D., Zhang, W., Yu, N., Yuan, L., Chen, D., Guo, B., 2022. Cswin transformer: A general vision transformer backbone with cross-shaped windows. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12124–12134.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al., 2020. An image is worth 16 × 16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., Tian, Q., 2019. Centernet: Keypoint triplets for object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6569–6578.

Han, K., Xiao, A., Wu, E., Guo, J., Xu, C., Wang, Y., 2021. Transformer in transformer. Adv. Neural Inf. Process. Syst. 34, 15908–15919.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778.

Keuper, M., Tang, S., Andres, B., Brox, T., Schiele, B., 2018. Motion segmentation & multiple object tracking by correlation co-clustering. IEEE Trans. Pattern Anal. Mach. Intell. 42 (1), 140–153.

Kuhn, H.W., 1955. The hungarian method for the assignment problem. Nav. Res. Logist. Q. 2 (1–2), 83–97.

Leal-Taixé, L., Canton-Ferrer, C., Schindler, K., 2016. Learning by tracking: Siamese CNN for robust target association. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 33–40.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft coco: Common objects in context. In: European Conference on Computer Vision. Springer, pp. 740–755.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B., 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10012–10022.

Loshchilov, I., Hutter, F., 2019. Decoupled weight decay regularization. In: International Conference on Learning Representations. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Meinhardt, T., Kirillov, A., Leal-Taixe, L., Feichtenhofer, C., 2022. Trackformer: Multi-object tracking with transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8844–8854.

Milan, A., Leal-Taixé, L., Reid, I., Roth, S., Schindler, K., 2016. MOT16: A benchmark for multi-object tracking. arXiv preprint arXiv:1603.00831.

Pang, B., Li, Y., Zhang, Y., Li, M., Lu, C., 2020. Tubetk: Adopting tubes to track multi-object in a one-step training model. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6308–6318.

Pang, J., Qiu, L., Li, X., Chen, H., Li, Q., Darrell, T., Yu, F., 2021. Quasi-dense similarity learning for multiple object tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 164–173.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A., 2017. Automatic differentiation in pytorch. In: NIPS-W.

Peng, J., Wang, T., Lin, W., Wang, J., See, J., Wen, S., Ding, E., 2020a. TPM: Multiple object tracking with tracklet-plane matching. Pattern Recognit. 107, 107480.

Peng, J., Wang, C., Wan, F., Wu, Y., Wang, Y., Tai, Y., Wang, C., Li, J., Huang, F., Fu, Y., 2020b. Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking. In: Proceedings of the European Conference on Computer Vision.

Peng, J., Wang, C., Wan, F., Wu, Y., Wang, Y., Tai, Y., Wang, C., Li, J., Huang, F., Fu, Y., 2020c. Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16. Springer, pp. 145–161.

Redmon, J., Farhadi, A., 2018. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

Ren, S., He, K., Girshick, R., Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. Adv. Neural Inf. Process. Syst. 28.

Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., Savarese, S., 2019. Generalized intersection over union: A metric and a loss for bounding box regression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 658–666.

Sanchez-Matilla, R., Poiesi, F., Cavallaro, A., 2016. Online multi-target tracking with strong and weak detections. In: Hua, G., Jégou, H. (Eds.), Computer Vision – ECCV 2016 Workshops. Springer International Publishing, Cham, pp. 84–99.

Shao, S., Zhao, Z., Li, B., Xiao, T., Yu, G., Zhang, X., Sun, J., 2018. CrowdHuman: A benchmark for detecting human in a crowd. arXiv preprint arXiv:1805.00123.

Sheng, H., Zhang, Y., Chen, J., Xiong, Z., Zhang, J., 2018. Heterogeneous association graph fusion for target association in multiple object tracking. IEEE Trans. Circuits Syst. Video Technol. 29 (11), 3269–3280.

Stadler, D., Beyerer, J., 2021. Multi-pedestrian tracking with clusters. In: 2021 17th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS). IEEE, pp. 1–10.

Strubell, E., Ganesh, A., McCallum, A., 2019. Energy and policy considerations for deep learning in NLP. arXiv preprint arXiv:1906.02243.

Sun, P., Cao, J., Jiang, Y., Yuan, Z., Bai, S., Kitani, K., Luo, P., 2022. DanceTrack: Multi-object tracking in uniform appearance and diverse motion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

Sun, P., Cao, J., Jiang, Y., Zhang, R., Xie, E., Yuan, Z., Wang, C., Luo, P., 2020. Transtrack: Multiple object tracking with transformer. arXiv preprint arXiv:2012.15460.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. Adv. Neural Inf. Process. Syst. 30.

Wan, X., Zhou, S., Wang, J., Meng, R., 2021. Multiple object tracking by trajectory map regression with temporal priors embedding. In: Proceedings of the 29th ACM International Conference on Multimedia. pp. 1377–1386.

Wang, G., Wang, Y., Gu, R., Hu, W., Hwang, J.-N., 2022. Split and connect: A universal tracklet booster for multi-object tracking. IEEE Trans. Multimed..

Wang, Q., Zheng, Y., Pan, P., Xu, Y., 2021. Multiple object tracking with correlation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3876–3886.

Wojke, N., Bewley, A., Paulus, D., 2017. Simple online and realtime tracking with a deep association metric. In: 2017 IEEE International Conference on Image Processing (ICIP). IEEE, pp. 3645–3649.

Wu, S., Hadachi, A., Lu, C., Vivet, D., 2023. Transformer for multiple object tracking: Exploring locality to vision. Pattern Recognit. Lett. 170, 70–76.

Xu, Z., Vivet, D., 2021. Instance sequence queries for video instance segmentation with transformers. Sensors 21 (13), http://dx.doi.org/10.3390/s21134507, URL https://www.mdpi.com/1424-8220/21/13/4507.

Zeng, F., Dong, B., Zhang, Y., Wang, T., Zhang, X., Wei, Y., 2022. Motr: End-to-end multiple-object tracking with transformer. In: European Conference on Computer Vision. Springer, pp. 659–675.

Zhang, Y., Wang, C., Wang, X., Zeng, W., Liu, W., 2021a. Fairmot: On the fairness of detection and re-identification in multiple object tracking. Int. J. Comput. Vis. 129, 3069–3087.

Zhang, X., Zhao, L., Gu, F., 2021b. Boosting the speed of real-time multi-object trackers. In: 2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/IOP/SCI). IEEE, p. 487.

Zhou, X., Koltun, V., Krähenbühl, P., 2020. Tracking objects as points. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV. Springer, pp. 474–490.

Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J., 2021. Deformable {DETR}: Deformable transformers for end-to-end object detection. In: International Conference on Learning Representations. URL https://openreview.net/forum?id=gZ9hCDWe6ke.