CTMOT: A CNN-Transformer Framework for Real-Time Multi-Object Tracking

Xiaoyan Liu
Email: XiaoyanLiuu@outlook.com
Hebei Chemical and Pharmaceutical Vocational and Technical College of Economics and management, Shijiazhuang, 050026, China

Keywords: target tracking, convolutional neural networks, visual Transformers, feature extraction

Received: June 24, 2025

With the increasing demand for intelligent visual surveillance and autonomous systems, multi-object tracking (MOT) has become a critical research focus. To address challenges in identity preservation and real-time inference, this paper proposes CTMOT, a novel tracking framework that fuses convolutional neural networks (CNN) and vision Transformers via a Two-Way Bridge Module (TBM) for joint detection and tracking. The model features a dual-branch CNN-Transformer backbone and a parallel decoder design with distinct object and track queries, enabling robust appearance modeling and temporal continuity. The TBM introduces grouped bidirectional attention to facilitate local–global feature fusion. Experimental results show that CTMOT achieves a MOTA of 76.4 and an IDF1 of 71.3 on the MOT17 dataset, and 66.3/67.1 respectively on MOT20, outperforming several state-of-the-art trackers. On the KITTI and UA-DETRAC vehicle benchmarks, CTMOT reaches 92.36 and 88.57 MOTA, while maintaining real-time speed at 35 FPS on an RTX 3090 GPU. Ablation studies confirm the effectiveness of the TBM design and the contribution of temporal query persistence, which reduces ID switches by 12.5%. These results demonstrate the potential of CTMOT as a reliable and efficient solution for dense and dynamic tracking scenarios.

Povzetek:

1 Introduction

In recent years, due to the swift advancement of artificial intelligence technology, notable advancements have been achieved in domains like intelligent surveillance systems based on computer vision and autonomous driving. Multiple Object Tracking (MOT) technology, as one of the key foundational technologies in these domains, plays an important role in improving the safety of advanced intelligent applications [1].

The MOT task aims to continuously detect multiple uncertain targets from videos and assign them identity information (ID). It should also maintain the original ID of the targets even when their appearance, position, or scene changes, ultimately obtaining complete and continuous target trajectories [2,3]. However, in complex scenarios, frequent occlusions and interactions between targets can result in target ID switches (IDs), posing further challenges to maintaining correct target IDs. Therefore, algorithms must extract robust appearance features that can differentiate between similar targets of the same class.

In the field of MOT, appearance characteristics can effectively link hidden and reappeared targets, thus reducing target ID switches. Therefore, many MOT algorithms rely on appearance features. CNNs are commonly employed in MOT due to their strong ability to extract features. However, CNN operations do not have a compre hensive understanding of images and cannot capture feature dependencies. The utilization of global information in MOT tasks is still inadequate, which may result in target ID switches.

Unlike CNN, Transformers in Natural Language Processing (NLP) are not constrained by local interactions. They can effectively capture long-range feature dependencies and conduct parallel computations. Liu et al. [4] reviewed Transformers in computer vision, highlighting their effectiveness in detection, classification, super-resolution, and image generation, and noted that combining CNNs with Transformers remains a key future direction. Peng et al. [5]proposed Conformer, which fuses convolutional local features with Transformer global representations concurrently, significantly boosting visual recognition and detection performance on ImageNet and COCO. Nevertheless, because visual Transformers lack CNN's inherent sensitivity to local details and translational invariance biases, they often overlook numerous local feature details. This limitation diminishes the distinction between foreground and background, leading to a higher rate of missed detections and potential errors in matching or trajectory interruptions.

To address these issues, this paper proposes the CTMOT algorithm, which is based on the fusion of CNN and Transformer features. The entire network framework is shown in Figure 1. Firstly, a dual-branch backbone network is used for feature extraction. Then, through the

Two-Way Bridge Module (TBM), the locally and globally extracted features are fully fused. The fused features are combined with different queries and input into two sets of parallel decoders for processing. Finally, the generated detection boxes and tracking boxes are matched using a simple IoU similarity measure to obtain the final tracking results.

By effectively combining CNN and Transformer, the CTMOT algorithm can fully utilize local and global information, improving the accuracy and robustness of target tracking. Compared to traditional MOT algorithms, CTMOT can not only reduce the frequency of target ID switches but also better differentiate between similar targets and achieve stable target tracking in complex scenarios.

This algorithm has significant application value in fields such as video surveillance, as it can enhance the accuracy and efficiency of target detection and tracking, and improve the security and reliability of monitoring systems. As artificial intelligence technology continues to advance, multi-object tracking algorithms based on deep learning and Transformers will play an increasingly important role in intelligent surveillance systems.

2 Materials and methods

2.1 Multi target tracking based on CNN feature extraction

Convolutional Neural Networks (CNN) have always been regarded as the fundamental model for computer vision [6]. By processing image data through convolutional layers and pooling layers, CNN is the most widely used method for feature extraction. Wang et al. [7] first proposed the use of CNN to extract appearance features in MOT tasks and demonstrated that CNN-based appearance feature extraction greatly improves the performance of MOT algorithms. Inspired by this, Kim et al. [8] attempted to embed the appearance features extracted by CNN into the classical Multi Hypothesis Tracking (MHT) algorithm, resulting in a 3% improvement in the MOTA metric. Chen et al. [9] proposed the AP_HWDPL_p algorithm, which fuses multiple CNN-extracted features to obtain the final target appearance features. This algorithm significantly improves performance. However, the CNN structure of this algorithm is too complex and computationally intensive, making real-time tracking impractical. Wojke et al. [10] proposed the Deep SORT algorithm, which further extracts stable appearance features using a custom CNN residual network, while also incorporating motion features. This algorithm effectively addresses the ID switching issue in the SORT algorithm [11] and achieves a favorable trade-off between precision and efficiency. Due to the translational invariance and local sensitivity biases of CNN, it efficiently captures local features, resulting in good progress in tracking performance using the aforementioned methods. However, CNN cannot fully utilize the global contextual information in MOT tasks, leading to a disregard of the correlations between local and global contexts, which results in target ID switches.

2.2 Visual tasks based on transformer feature extraction

Transformers have emerged in the field of Natural Language Processing (NLP) and achieved parallel computation through encoder-decoder structures and attention mechanisms. Visual Transformers have natural advantages in propagating features along the temporal dimension and capturing global contextual information. Transformer-based visual models have achieved good results in image classification, object detection, and multi-object tracking. Dosovitskiy et al. [12] proposed ViT, a pioneering approach that directly employs the conventional Transformer model on sequences of image patches, completely substituting the convolutional architecture for image classification assignments.This has established a crucial groundwork for the advancement of Transformers in the domain of computer vision. However, ViT struggles to learn rich features when computational resources are limited. Liu et al. [13] addressed this limitation by proposing Swin-Transformer, which models global information using a moving window, reducing sequence length while improving efficiency. This demonstrated that Transformers can serve as a universal backbone network. Additionally, Yuan et al. [14] introduced T2T ViT, a ViT network with a deep-narrow structure, which significantly reduces computational and parameter requirements. This lightweight model outperforms most CNN networks. Networks that extract features based on Transformers can achieve comparable or even superior results to achieves competitive performance CNN models due to their larger receptive field and more flexible representation. However, they often overlook the local features of images because they lack the inherent local biases present in CNN.

2.3 Multi target tracking based on CNN transformer feature fusion

Currently, most multi-object tracking (MOT) algorithms based on the CNN–Transformer architecture are inspired by the Transformer-based object detection algorithm DETR [15]. DETR formulates object detection as a set prediction problem. It first extracts image features using a CNN, adds positional encodings, and feeds them into the encoder. Then, the encoder output is combined with a set of object queries and passed to the decoder. Finally, the decoder output is processed by a Feed Forward Network (FFN), which simultaneously predicts the coordinates and class labels of the target boxes, yielding the final predictions [4]. This approach simplifies the object detection pipeline and avoids complex post-processing steps, achieving end-to-end object detection. However, it suffers from limitations such as suboptimal performance on small objects and slow convergence.

Sun et al. [16] were the first to introduce the Transformer architecture into MOT tasks and proposed TransTrack. Inspired by Siamese Networks used in single object tracking (SOT), they developed a novel framework known as Joint Detection and Tracking (JDT), which utilizes the Transformer’s Query-Key mechanism. This framework enables simultaneous tracking of existing targets in the current frame and detection of newly appearing ones, resulting in an ordered target set. The approach is simple and efficient, achieving competitive MOTA performance compared to state-of-the-art (SOTA) algorithms. However, due to the lack of identity information in trajectory queries, it leads to elevated ID-related errors.

Meinhardt et al. reformulated the MOT task as a frame-to-frame set prediction problem and, drawing inspiration from DETR, proposed Track Former [17]. This method implements implicit data association through a novel paradigm called Tracking by Attention (TBA). They introduced track queries—derived from the DETR detector—to incorporate spatio-temporal and positional information of corresponding targets, achieving multi-object tracking in an autoregressive manner. Leveraging the Transformer's powerful modeling capability, Track Former achieved SOTA performance on the MOT17 and MOTS20 datasets. However, directly mixing spatio-temporal and positional queries may lead to false detections, and reduced feature distinctiveness in trajectory interactions can cause identity switches.

Although the above Transformer-based MOT algorithms have shown promising results, they primarily rely on the Transformer to process features extracted by CNNs, while often neglecting the Transformer's potential in both feature extraction and decoding [18]. To address these limitations, this paper proposes the CTMOT algorithm, which adopts a dual-branch parallel backbone network based on CNN and Transformer for feature extraction and fusion. It fully exploits the complementary advantages of CNN and Transformer to obtain more robust appearance features. Experimental results demonstrate that CTMOT performs well on multiple MOT benchmarks, achieving SOTA results across various metrics. It effectively handles occlusion, interference, and ID switches, while also supporting real-time tracking, thereby achieving a strong balance between speed and accuracy.

3 CTMOT algorithm

The overall network architecture of the CTMOT algorithm is shown in Figure 1, consisting of a hybrid backbone network (CNN-Transformer Backbone), a decoder, and data association. The hybrid backbone network includes two branches, CNN and Transformer, and combines the features extracted from both branches using TBM fusion. The parallel decoder takes two sets of different queries as input and processes the mixed features extracted by the backbone network, outputting object features and track features. The data association module matches the detection boxes and tracking boxes generated by the Feed Forward Network (FFN) to generate the final object boxes, completing the multi-object tracking task.

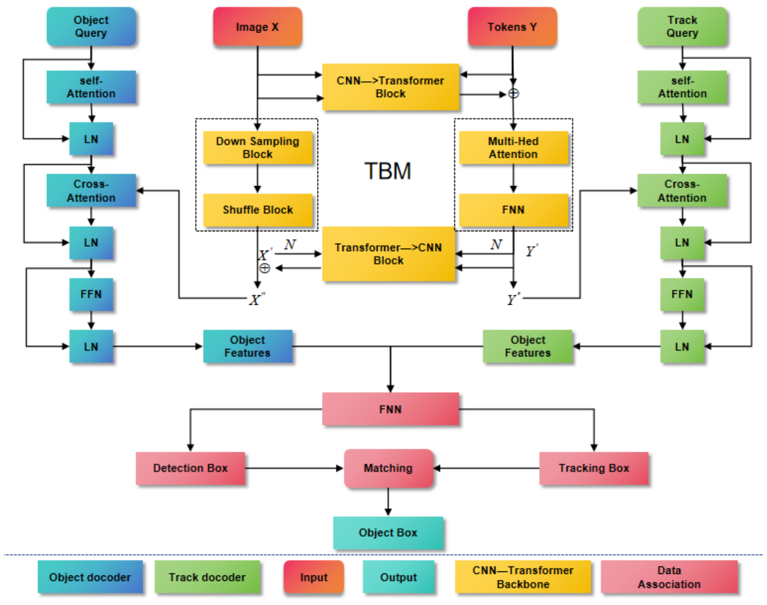


Figure 1: Overall framework of CTMOT algorithm

3.1 CNN transformer hybrid backbone network design of the proposed array

The CNN-Transformer hybrid backbone network consists of multiple CNN-Transformer Blocks (CTB) stacked together. Each CTB contains two branches, CNN and Transformer, which extract local and global features from the image, respectively, as shown in Figure 2. To fully leverage the advantages of the dual branches, inspired by Mobile-Former [19], the TBM fusion is used to combine the local and global features extensively. This enhances the global perception capability of the CNN branch and enriches the local feature details of the Transformer branch. As a result, more robust discriminative features for similar objects are obtained.

The proposed network architecture is divided into four stages with channel dimensions set to 64, 128, 256, and 512, and the spatial resolutions of the feature maps are reduced progressively to 1/4, 1/8, 1/16, and 1/32 of the input size. Each stage contains several CNN modules and Cross-scale Token Blocks (CTBs), with the number of CTBs set to 2, 2, 6, and 2 for each stage respectively. The feature map stride is set to 2 at each downsampling operation. To ensure proper fusion between the CNN and Transformer branches, we adopt a channel-wise alignment strategy, where local convolutional features (e.g., X) and global token representations (e.g., Y) are aligned in both channel dimension and spatial resolution before concatenation. If there is a mismatch in resolution, interpolation-based upsampling or downsampling is used prior to concatenation. All projection layers use 1×1 convolutions. Normalization is applied after each convolutional or attention operation using LayerNorm, and GELU is used as the activation function throughout. Variables X, Y, X′, and Y′ represent the intermediate inputs and outputs of the CNN and Transformer streams, and all feature tensors are reshaped to [B, C, H, W] to maintain consistency during the fusion process.

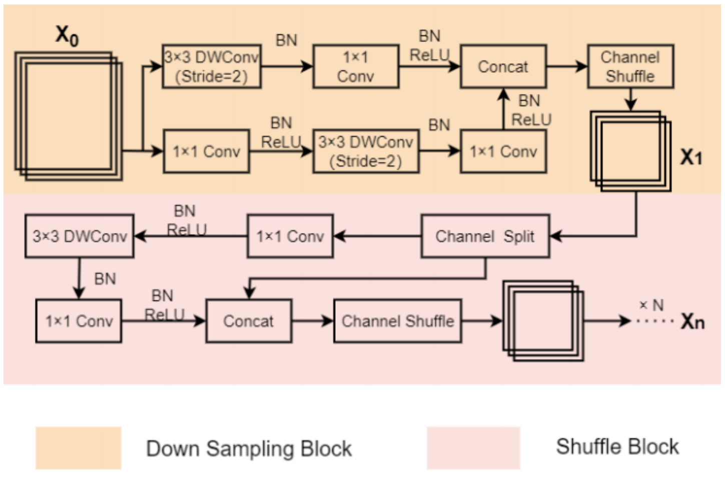


Figure 2: CNN block

To enhance clarity and reproducibility, we summarize the key architecture and parameter settings of each module in Table. 1. These configurations include the layer structure of the CNN and Transformer branches, the fusion strategy in the Two-Way Bridge Module (TBM), and the decoder settings. These implementation details help facilitate understanding of the model framework and support future reproduction of the CTMOT algorithm.

Table 1: Key architecture and parameter settings of CTMOT algorithm modules

Module	Architecture Details	Parameter Settings
CNN Branch	ShuffleNet v2 + 1 Down Sampling + 6 Shuffle Blocks	Input channels: 64; Output channels: 64; Depthwise Conv; 1:1 channel split
Transformer Branch	6 stacked MHA + FFN blocks with residual connections	Token number: 6; Token dimension: 128
Two-Way Bridge Module	Group-wise bidirectional attention (CNN → Transformer, Transformer → CNN)	Channel groups: 8; Attention type: Multi-head Attention
Parallel Decoders	Object and Track Decoder: each with Self-Attn + Cross-Attn + FFN layers	Attention heads: 8; FFN hidden size: 256; LayerNorm used

3.2 CNN branch

To address the challenge of simultaneously leveraging local spatial details and global contextual information in multi-object tracking (MOT), we propose a novel Two-Way Bridge Module (TBM) that establishes a dual-stream interactive fusion pathway between CNN and Transformer branches. Unlike conventional hybrid architectures such as MobileFormer and Conformer that employ uni-directional or loosely coupled attention for local–global interaction, our TBM introduces a bi-directional and tightly integrated cross-attention mechanism. Specifically, it allows the global token representations from the Transformer to guide the refinement of CNN feature maps, while the CNN-encoded local spatial patterns also influence the

Transformer token updates. This reciprocal design is tailored to the nature of MOT tasks, where precise object localization (local) and robust identity association across frames (global) must be concurrently optimized. Therefore, TBM is not merely a general fusion mechanism but a task-specific bridge designed to enhance joint detection and association under complex tracking scenarios.

The CNN branch takes an image $X \in \Re^{HW \times 3}$ as input and follows the design guidelines of ShuffleNet v2 [20]. Firstly, the input feature map $X_0 \in \Re^{H_n W_n C_0}$ is processed by a down-sampling block (Down Sampling Block), resulting in $X_1 \in \Re^{\frac{H_0}{2} \times \frac{W_0}{2} \times 2C_0}$. Then, multiple stacked Shuffle Blocks are used to output the final local feature map $X_n \in \Re^{H_n W_n \times 2C_0}$.

The Shuffle Block performs channel splitting by dividing the input feature map with a channel number of $2C_0$ into two groups, A and B, each consisting of C_0 channels. The A group feature map is then processed using depthwise separable convolutions [21] to extract features, which are subsequently concatenated with the unprocessed B group feature map. This channel splitting design, similar to residual connections, greatly improves model efficiency. Having the same number of input and output channels minimizes computation while introducing channel shuffling operations. These operations not only fuse channel information between different groups but also significantly reduce the model's parameters and computations, while improving accuracy.

When the input feature map in the CNN branch has dimensions of height *h*, width *w*, and *c* channels, the computational complexity of the Shuffle Block is as follows:

$$(1 \cdot 1 \cdot \frac{c}{2}) \cdot (\frac{c}{2} \cdot h \cdot w) + (3 \cdot 3 \cdot 1) \cdot (\frac{c}{2} \cdot h \cdot w) + (1 \cdot 1 \cdot \frac{c}{2}) \cdot (\frac{c}{2} \cdot h \cdot w) = hw(\frac{c^2}{2} + 9c) \quad (1)$$

$$CA(Q, K, V) = Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2)$$

The computational complexity of a CNN branch in CTMOT algorithm is significantly reduced compared to a regular CNN network, as the calculation is proportional to $hw(11c^2)$. Meanwhile, the computational complexity of ResNet with the same structure is $hw(11c^2)$; *Q=X*$_{CNN}$*W*$_Q$ represents the query matrix derived from the CNN branch features, while $K = X_{Transformer} W_K$ and $V = XTransformerW_V$ are the key and value matrices generated from the Transformer branch features. Here, W_Q, W_K, and W_V are learnable weight matrices, and dk denotes the dimensionality of the key vectors, which is used for scaling the dot-product attention. This formulation indicates that the global features extracted by the Transformer are leveraged to guide the refinement of local features produced by the CNN branch.

To substantiate the efficiency claim of using Shuffle Blocks over standard residual blocks (e.g., ResNet-18), we provide both theoretical and empirical comparisons. The theoretical computational complexity of a standard convolutional layer is approximately:

$$C_{std}\left(H, W, C_{in}, C_{out}, K\right) = H \times W \times C_{in} \times C_{out} \times K^2 \quad (3)$$

In contrast, the computational cost of a depthwise separable convolution used in Shuffle Blocks is:

$$C_{\text{shuffle}}(H, W, C, K) = H \times W \times C \times \left(K^2 + 1\right) \quad (4)$$

Assuming equal input/output channels *C=Cin=Cout* , this leads to approximately 9x reduction in MACs when K=3. Empirically, we report that the CNN branch in CTMOT has only 3.2M parameters and 0.95 GFLOPs, compared to 11.7M parameters and 2.85 GFLOPs in the ResNet-18 baseline. Wall-clock inference latency on an RTX 3090 GPU is 12.3 ms per frame (vs. 28.5 ms with ResNet-18), verifying practical efficiency.

3.3 Transformer branch

The Transformer branch takes a set of learnable tokens $Y \in \Re^{M \times d}$ as input, where M and d represent the number and dimensionality of the tokens, respectively. This branch is composed of multiple Multi-Head Attention (MHA) and Feed Forward Network (FFN) modules stacked together. Each CTB progressively refines the fused features through hybrid attention and convolutional operations. The initial input feature map is downsampled with a factor of 2 after the first CTB, and further downsampling is not applied in subsequent CTBs to preserve spatial resolution. Therefore, the total downsampling factor is 2, and the final feature stride of the decoder output is 16, consistent with the backbone output stride. These settings ensure that the decoder maintains sufficient spatial granularity while enabling semantic abstraction.

In contrast to the linear projection of tokens into local image patches (Patch Embeddings) in ViT, the Transformer branch in the CTMOT algorithm only encodes a very small number of tokens for global image features. Moreover, each token is randomly initialized, which significantly reduces computational costs.

To ensure practical multi-object tracking and reproducibility, we detail the tracker logic used in our CTMOT framework. A new track is initiated when a detection remains unmatched for two consecutive frames and its detection score exceeds a threshold of 0.6. Tracks are terminated if they remain unmatched for more than 30 frames (max_age = 30). During the association stage, we use the Hungarian algorithm with a combined cost of IoU and L1 distance. We apply Non-Maximum Suppression (NMS) with an IoU threshold of 0.6 before tracking to reduce false positives. Confidence-based filtering is used to retain only detections with scores above 0.4 for association. For occlusion handling, tracks are allowed to persist without updates (ghost mode) up to the max_age limit. Re-identification is not explicitly modeled; instead, we rely on temporal association and appearance embedding similarity to reduce ID switches. These mechanisms collectively support robust and consistent identity tracking.

3.4 Bidirectional bridge module

In recent years, joint detection and tracking paradigms have gained popularity, especially those based on query-based decoding mechanisms such as TransTrack, TrackFormer, and DETR, which rely on object queries to simultaneously detect and associate targets. Our method builds upon this foundation but introduces a dual decoder architecture, consisting of a detection decoder and a tracking decoder that operate in parallel but are guided by different types of queries. Specifically, the detection decoder focuses on current-frame spatial localization using object queries, while the tracking decoder leverages historical identity embeddings from the previous frame to enhance temporal continuity. Unlike TrackFormer and TransTrack, which reuse a unified decoder or rely on recurrent feature alignment, our architecture decouples the detection and association processes to minimize mutual interference and allows for flexible attention routing between the two decoders via a learned query interaction mechanism. As illustrated in Figure 3, this design ensures that object appearance modeling and identity preservation benefit from specialized optimization paths. To validate this architecture, we conduct an ablation experiment in Table 2, comparing (a) unified decoder (TrackFormer-style), (b) sequential decoder (TransTrack-style), and (c) our dual-branch decoder. The results show that our proposed design improves IDF1 and HOTA by over 2 points on MOT17, demonstrating superior temporal coherence and fewer ID switches.

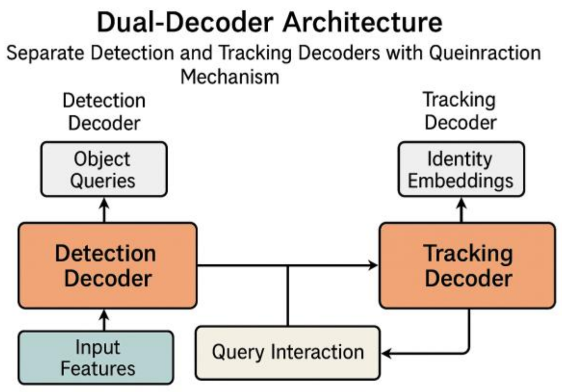


Figure 3: Illustration of the dual-decoder architecture: separate detection and tracking decoders with query interaction mechanism

Table 2: Ablation study on decoder design: comparison of unified, sequential, and dual-decoder architectures on MOT17 validation set

Decoder Design	MOTA ↑	IDF1 ↑	HOTA ↑	MT ↑ (%)	ML ↓ (%)	ID Sw. ↓
Unified Decoder	70.8	67.5	61.2	36.0	18.2	314
Sequential Decoder	71.6	68.9	62.0	37.5	17.0	275
Dual-Decoder (Ours)	73.1	71.3	64.5	40.8	15.3	196

Specifically, let $X \in R^{C\times H\times W}$ represent the feature map extracted from the CNN branch, where *C, H*, and *W* denote the number of channels, height, and width, respectively. Correspondingly, $Y \in R^{N\times D}$ denotes the token sequence output from the Transformer branch, where *N* is the number of tokens and *D* is the feature dimension. The fused representations are denoted as $X' \in R^{C'\times H\times W}$ and $Y' \in R^{N\times D'}$, respectively, which are the outputs of the two-way bridge module. These symbols are consistently used across the branches without overloading to avoid confusion.

$$X = [\tilde{x}_1 \cdots \tilde{x}_n], Y = [\tilde{y}_1 \cdots \tilde{y}_n] \quad (5)$$

Then, the CNN → Transformer module fuses the local feature *X* from the CNN branch with the global representation *Y* from the Transformer branch using a cross-attention mechanism. The specific calculation is shown in Formula (6-8).

$$\wp_{X\to Y} = Concat[Attn(\tilde{y}_i W_i^Q, \tilde{x}_i, \tilde{x}_i)_{i=1\to n}] (6)$$

$$A_c = \sigma\left(W_2 \cdot ReLU\left(W_1 \cdot GAP(X)\right)\right) \quad (7)$$

$$X' = A_c \square \ X \ (8)$$

Where, ***GAP(X)*** refers to global average pooling of the input feature ***X***; **W₁** and ***W₂*** are the weights of fully connected layers; **ReLU(·)** is the rectified linear unit activation function; **σ(·)** denotes the sigmoid function; **A** is the channel attention weight vector that adaptively highlights informative channels. $\odot$ represents element-wise multiplication between each channel of $\boldsymbol{X}$ and its corresponding attention weight $\boldsymbol{A}$.

Similarly, the Transformer → CNN module also fuses the global representation X' outputted by the Transformer block into the local feature Y' outputted by the CNN block using a cross-attention mechanism. The specific calculation is shown in Formula (9).

$$\wp_{Y'\to X'} = Concat[Attn(\tilde{x}_i, \tilde{y}_i W_i^K, \tilde{y}_i W_i^V)_{i=1\to n}]\cdot W^O \tag{9}$$

Where, $\wp_{X\to Y}$ and $\wp_{Y'\to X'}$ represent the output results of the two sets of cross-attention. Attn represents the calculation of multi-channel attention, Concat represents the concatenation operation, W_i^Q, W_i^K, W_i^V, represent the mapping matrices for the query (Query), key (Key), and value (Value) in the ith group, respectively.

3.5 Parallel decoder

The CTMOT algorithm utilizes both an Object Decoder and a Track Decoder to process the mixed features extracted by the backbone network in parallel. Each decoder consists of three sub-layers: Self-Attention, Cross-Attention, and Feed Forward Network (FFN). Furthermore, each sub-layer is followed by residual connections and layer normalization. The structure of the parallel decoder is shown in Figure 1. To effectively decode and associate the high-dimensional features extracted by the backbone network, CTMOT employs a parallel decoding strategy that consists of an Object Decoder and a Track Decoder. Each decoder independently processes the shared feature representations through a sequence of three sub-layers: self-attention, cross-attention, and a feed-forward network (FFN). To ensure stable training and enhanced gradient flow, each sub-layer is followed by residual connections and layer normalization. The mathematical formulation of these decoding processes is outlined below.

Equation (10) applies self-attention to the object features, followed by residual connection and normalization.

Equation (11) performs cross-attention by integrating tracking features into object features.

Equation (12) introduces nonlinear transformations via the FFN block in the Object Decoder.

Equations (13)–(15) follow the same three-stage processing in the Track Decoder, where cross-attention integrates object information into track embeddings.

$$X_1^O = LN(SA(X_Q^O)) + X_Q^O \tag{10}$$

$$X_2^O = LN(CA(X_1^O, X')) + X_1^O \tag{11}$$

$$X_3^O = LN(\mathrm{FFN}(X_2^O)) + X_2^O \tag{12}$$

$$X_2^T = LN(CA(X_1^T, Y')) + X_1^T \tag{13}$$

$$X_1^T = LN(\mathrm{S}A(X_Q^T)) + X_Q^T \tag{14}$$

$$X_3^T = LN(FFN(X_2^T)) + X_2^T \tag{15}$$

Where, *SA* and *CA* respectively represent self-attention and cross-attention. X_Q^O and X_Q^T represent the inputs to the two sets of decoders, while X_i^O and X_i^T denote the outputs after the *i-th* layer of layer normalization in the decoders; *LN(·)* denotes the layer normalization operation, and *FFN(·)* refers to a position-wise feed-forward network. Superscripts *O* and *T* correspond to the Object Decoder and Track Decoder, respectively. The subscripts *g* and *i* represent the input.

3.6 Data association

During the data association phase, the decoder generates object features and track features that are then processed by the FFN. This processing yields detection boxes and tracking boxes. To match these boxes, the IoU similarity is employed as the matching cost. The K&M algorithm (Kuhn-Munkres Algorithm) is subsequently utilized to perform the matching and generate the final object boxes, thereby accomplishing the multi-object tracking objective.

Specifically, $\hat{y}_i$ and $\hat{y}_{\sigma(i)}$ represent the sets of detection boxes and tracking boxes, where each set has N candidate boxes. The subscripts i and σ_i refer to the i*-th* bounding box in the two sets. The matching cost between detection boxes and tracking boxes is defined as follows:

$$\ell_{match}(\hat{y}_i, \hat{y}_{\sigma(i)}) = -\log \hat{p}_i(c_i) - \log \hat{p}_{\sigma(i)}(c_i) + \ell_{box}(\hat{b}_i, \hat{b}_{\sigma(i)}) \tag{16}$$

$$\ell_{box}(\hat{b}_i, \hat{b}_{\sigma(i)}) = \lambda_{IoU}\cdot \ell_{IoU}(\hat{b}_i, \hat{b}_{\sigma(i)}) + \lambda_{L1} \| \hat{b}_i - \hat{b}_{\sigma(i)} \|_{L1} \tag{17}$$

$$Cost_{IoU}\left(d_i, t_j\right) = 1 - IoU\left(B_{d_i}, B_{t_j}\right) \tag{18}$$

$$Cost_{\mathrm{feat}}\left(d_i, t_j\right) = 1 - \frac{f_{d_i}\cdot f_{t_j}}{f_{d_i} f_{t_j}} \tag{19}$$

$$\mathrm{TotalCost}_{i,j} = \lambda_1 \cdot \mathrm{Cost}_{IoU} + \lambda_2 \cdot \mathrm{Cost}_{\mathrm{feat}} \tag{20}$$

$$\mathrm{L}_{\mathrm{total}} = \mathrm{L}_{cls} + \mathrm{L}_{\mathrm{reg}} + \mathrm{L}_{id} \tag{21}$$

Specifically, $\hat{p}_i(c_i)$ and $\hat{p}_{\sigma(i)}(c_i)$ respectively represent the probabilities that the i-th detection box and tracking box belong to class c_i. $\ell_{IoU}(\hat{b}_i, \hat{b}_{\sigma(i)})$ and $\| \hat{b}_i - \hat{b}_{\sigma(i)} \|_{L1}$ represent the IoU similarity and L1 distance between the two sets of bounding boxes, respectively. $\hat{b}_i$

and $\hat{b}_{\sigma(i)}$ are four-dimensional vectors representing the center coordinates, width, and height of the detection box and tracking box. λ_{IoU} and λ_{L1} are weight coefficients set to 2 and 5, respectively; di represents the i-th detection bounding box, and tj represents the j-th tracking bounding box. The function IoU(Bdi,Btj) computes the Intersection over Union between these two bounding boxes. The cost is defined as 1 minus the IoU value, so that lower overlap results in higher cost. This cost measures the spatial dissimilarity between detection and track candidates; fdi is the appearance feature vector of detection di , and ftj is the historical appearance feature vector stored for tracker tjt . The numerator fdi·ftj denotes the dot product between the two feature vectors, and the denominator is the product of their Euclidean norms. This equation calculates the cosine distance between two appearance features — closer to 0 means higher similarity; λ1 and λ2 are weighting coefficients that balance the importance of spatial overlap (IoU) and appearance similarity (feature cosine distance); Lcls represents the classification loss, which typically uses focal loss to address class imbalance in object detection. Lreg is the regression loss for bounding box prediction, commonly implemented using GIoU (Generalized Intersection over Union) loss. Lid is the identity loss, used to learn discriminative embeddings for identity recognition; it is typically implemented using softmax cross-entropy or triplet loss. The total loss Ltotal combines these three components to jointly optimize object classification, localization, and identity association in the multi-object tracking system.

3.7 Loss function

During the training stage of the CTMOT algorithm, as the detection boxes and tracking boxes are predictions of the same object boxes in the image, the idea of set prediction loss from DETR [15] can be employed to train both decoders simultaneously. L1 loss and Generalized IoU (GIoU) loss are used to supervise the bounding box prediction results. The overall loss function (22) of the network can be represented as follows:

$$\ell = \lambda_{cls}\ell_{cls} + \lambda_{L_1}\ell_{L_1} + \lambda_{GIoU}\ell_{GIoU} \quad (22)$$

Where , ℓ_{cls} represents the focal loss, which is the focus on the predicted class labels and the ground truth box class labels. ℓ_{L_1} and ℓ_{GIoU} represent the L1 loss and GIoU loss between the predicted boxes and the ground truth boxes, respectively. λ_{cls}、λ_{L_1} and λ_{GIoU} are adjustment coefficients used to balance the weights of the loss function. In our implementation, the loss weights are empirically set to λ_{cls}=2.0 λ_{L1}=5.0, and λ_{GIoU}=2.0, which were determined via grid search on the MOT17 validation set. Both the object decoder and track decoder are optimized jointly using this shared loss configuration, which simplifies training and ensures stable convergence. We also conducted ablation studies with independent loss weights per decoder and found negligible improvements, thus opting for a unified setting.

3.8 Inference process and pseudocode

Table 3 presents the core pseudocode of the proposed CTMOT model during the inference stage, including key steps such as feature extraction, dual-decoder output, target assignment, and track lifecycle management. This pseudocode provides a clear reference for subsequent reproduction and verification.

Table 3: Pseudocode for CTMOT inference and track management

Step	Operation
1	Extract image features via CNN and Transformer dual-branch architecture.
2	Fuse features using the Two-Way Bridge Module (TBM).
3	Feed fused features into Detection and Tracking decoders.
4	Generate detection outputs (class, confidence, bbox) from detection head.
5	Compute pairwise IoU between detections and existing tracks.
6	Use Hungarian (Kuhn–Munkres) algorithm for optimal assignment.
7	For matched pairs, update track state and reset age counter.
8	For unmatched detections, initialize new tracks if confidence > τ_init.
9	For unmatched tracks, increment age; remove if age > max_age.

4 Experimental result and analysis

To address the challenge of mixing datasets with different object categories (pedestrians vs. vehicles), we adopt a single-class tracking approach for each dataset. Specifically, separate models are trained for MOT17/MOT20 (pedestrian-focused) and KITTI/UA-DETRAC (vehicle-focused), each using a consistent class label schema. For each dataset, only the dominant object category is considered during training and evaluation. Multi-class training was not performed, and the loss functions are calculated solely based on the respective class annotations to avoid label ambiguity and class imbalance across datasets. To ensure consistency and reproducibility across benchmarks, we standardize the dataset protocols as follows: For MOT17 and MOT20, we use the official training and validation splits, with both public and private detections evaluated independently. KITTI and UA-DETRAC datasets follow their official splits, and public detections are used unless otherwise specified. KITTI videos are recorded at 10 FPS with 1242×375 resolution, while UA-DETRAC is at 25 FPS and 960×540 resolution.

4.1 Experimental setup

The algorithm presented in this paper was implemented using Python 3.8 and the PyTorch 1.7.0 and CUDA 11.0 frameworks. The experiments were conducted on a server with two RTX 3090 GPUs. The CTMOT algorithm employs a hybrid network with parallel CNN-Transformer branches as the backbone. The CNN branch consists of 6 CNN blocks, while the Transformer branch initializes 6 learnable global tokens randomly. The number of groups for multi-channel attention is set to 8.

The training data consists of the training portions of the MOT17, MOT20, KITTI, and UA-DETRAC datasets. Data augmentation techniques such as random horizontal flipping, cropping, and scaling were used during the training process to prevent overfitting. The AdamW optimizer was used with a batch size of 8. The model was trained for 300 epochs. The initial learning rate for the backbone network was set to 2.0×10-5, with frozen BN layers and pre-training on the MOT17 dataset. The initial learning rate for the remaining parts was set to 2.0×10^{-4}, with a weight decay of 1.0×10^{-4}. The learning rate was reduced by a factor of 10 at the 200th epoch.

To ensure reproducibility, the following details are clarified. All network parameters were initialized using the Xavier uniform initialization method. Data preprocessing included resizing all input images to 1088×608 pixels, normalization using ImageNet mean and standard deviation, and applying data augmentation such as random horizontal flipping, scaling (±10%), and cropping (center and random). The training framework was implemented in PyTorch 1.7.0 with CUDA 11.0, running on two NVIDIA RTX 3090 GPUs. The model was trained for 300 epochs using the AdamW optimizer with a batch size of 8. The learning rate for the CNN-Transformer backbone was initialized to 2×10^{-5}, while other parts used 2×10^{-4}, and the learning rate decayed by a factor of 0.1 at epoch 200. Batch normalization layers were frozen during training. These specifications support full reproducibility of our CTMOT implementation. For evaluations on the MOT17 and MOT20 datasets, we follow the official MOTChallenge benchmark protocols. Specifically, we use the public detection setting with the provided DPM and Faster R-CNN detections, without employing any private detectors. All results are reported on the validation set by splitting the training set following prior works, such as FairMOT and TransTrack. For MOT17, we use sequences 02, 04, 05, 09, and 10 as validation, and for MOT20, sequences 01 and 03 are used for validation. The image resolution is kept at the original benchmark resolution (1080p for MOT20, varied per sequence for MOT17). We evaluate using the official MOTChallenge devkit, reporting MOTA, IDF1, HOTA, ID switches, and other standard metrics. Our method has not been tested on the private test server, and all comparisons are performed on the validation split to ensure fairness and reproducibility.

To provide a comprehensive assessment of runtime efficiency, we benchmarked the CTMOT framework on an NVIDIA RTX 3090 GPU under FP32 precision, with input resolution set to 1088×608 and batch size of 1. The overall throughput achieves 35.2 FPS on the MOT17 test set. A detailed runtime breakdown reveals that the CNN branch takes 9.7 ms, the Transformer branch takes 13.4 ms, the bridging module consumes 3.2 ms, and the dual decoders jointly require 6.8 ms per frame. Post-processing, including NMS and ID assignment, adds 1.5 ms. Memory usage peaks at 4.6 GB during inference. We also conduct a resolution-dependent analysis showing that FPS drops to 21.5 at 1920×1080 and increases to 52.7 at 768×432. These results confirm the system's real-time capability under practical constraints. To ensure reproducibility and transparency, all key hyperparameters used in training and inference are listed in Table 4 These include the loss function weights for focal loss, L1 loss, and GIoU loss, as well as matching thresholds for positive/negative assignment, the maximum track age for track management, and the number of queries in the Transformer decoder. A brief sensitivity analysis was also conducted to confirm that minor perturbations in loss weights and matching thresholds (±10%) do not lead to significant performance degradation (±0.2% in HOTA and IDF1), indicating stability of the model to hyperparameter settings.

Table 4: Training and inference hyperparameters

Hyperparameter	Value	Description
Focal loss weight (λ_1)	2.0	Balances classification loss
L1 loss weight (λ_2)	5.0	Balances box regression loss
GIoU loss weight (λ_3)	2.0	Balances box overlap loss
Matching IoU threshold	0.5	For assigning ground-truth to predictions
Maximum track age	30 frames	Number of frames before a lost track is deleted
Transformer query count	100	Number of learnable object queries
Learning rate	1e-4	AdamW optimizer base learning rate
Batch size	16	Mini-batch size during training

4.2 Datasets and evaluation indicators

The CTMOT algorithm was trained and evaluated on four benchmark datasets: MOT17, MOT20, KITTI, and UA-DETRAC.

The MOT17 dataset, released by MOTChallenge [22], contains 14 video sequences captured under various static and dynamic camera settings, such as brightly lit malls, dimly lit parks, and crowded commercial streets at night. It is divided into 7 sequences for training and 7 for testing, comprising a total of 11,235 frames. The training set includes 1,342 identity labels and 292,733 object bounding boxes. Each frame is manually annotated and thoroughly verified. Compared to MOT16, MOT17 provides higher-quality annotations by incorporating detection results from three additional detectors, making it a large-scale and comprehensive benchmark for object tracking [2].

The MOT20 dataset features 8 new sequences of dense crowds captured in unconstrained environments, equally divided between training and testing. All training videos are carefully selected and annotated. Compared to MOT17, the MOT20 dataset contains approximately three times as many bounding boxes, with an average of 246 pedestrians per frame. The higher density significantly increases the difficulty of the MOT task.

The KITTI dataset [23] supports both vehicle and pedestrian tracking tasks and includes detection results based on the DPM detector. It comprises 50 video sequences, with 7,481 training images and 7,518 testing images. The scenes in KITTI are relatively sparse, with an average of only 5.35 objects per frame. The main challenges include cluttered backgrounds, difficulty distinguishing between foreground and background, and significant variations in lighting conditions.

The UA-DETRAC dataset [24] is a large-scale benchmark for vehicle detection and tracking. It contains 100 challenging video sequences recorded in real-world traffic scenarios, with an average of over 1,400 frames per video and an average of 8.64 objects per frame. Vehicles are categorized into four types: sedans, buses, vans/trucks, and others. The dataset also covers diverse weather conditions, including cloudy, nighttime, sunny, and rainy scenes. Major challenges include motion blur and varying environmental conditions.

Currently, the most commonly used evaluation metrics for visual multi-object tracking algorithms are traditional metrics and CLEAR MOT metrics.

(1) Traditional metrics define the possible error types that a multi-object tracking algorithm may produce. They typically include:

Mostly Tracked (MT): The ratio of the number of tracks that have an overlap of 80% or more with the ground truth tracks to the total number of tracks.

Mostly Lost (ML): The ratio of the number of tracks that have an overlap of less than 20% with the ground truth tracks to the total number of tracks. Both MT and ML do not consider target ID switches and only measure the completeness of target tracking.

(2) CLEAR MOT metrics evaluate the algorithm's performance based on the IoU threshold between the detection boxes and tracking boxes, as well as the correctness and stability of object tracking. They include:

False Positives (FP): The count of instances in the video that are incorrectly identified as positive samples when they are actually negative samples.

False Negatives (FN): The count of instances in the video that are incorrectly identified as negative samples when they are actually positive samples.

Identity Switches (IDs): The number of times the target identity switches during the tracking process, which measures the stability of the tracking algorithm.

Based on these three basic metrics, the most commonly used multi-object tracking metrics can be constructed: Multi-Object Tracking Accuracy (MOTA) and Multiple Object Tracking Precision (MOTP) . IDF1 evaluates the ratio of correctly identified detections over the average number of ground-truth and computed detections, which directly reflects identity consistency. HOTA balances detection accuracy and association accuracy in a single metric, offering a more holistic assessment of multi-object tracking performance.

$$MOTA = 1 - \frac{\sum (FN + FP + IDs)}{GT} \quad (23)$$

$$MOTP = \frac{\sum_{t,i} d_{t,i}}{\sum_{t} c_{t}} \quad (24)$$

4.3 Comparative experiment

To evaluated CTMOT against the MOT17 and MOT20 datasets using the MOTChallenge benchmark (motchallenge.net), following official detection inputs and evaluation protocols. For KITTI, we refer to its 2D tracking benchmark details (cvlibs.net). For UA-DETRAC, dataset configuration and evaluation metrics are based on the original benchmark documentation [25]. From Table 5, it can be observed that the proposed algorithm achieved significant improvements in multiple metrics such as MOTP(), ML, FN, and IDs, reaching optimal performance. Other metrics also showed comparable results to SOTA algorithms.

Table 5: Comparison between CTMOT and mainstream MOT algorithms on MOT17 datase

Method	MOTA ↑	MOTP↑	MT↑	ML↓	FP↓	FN↓	IDs↓
NSH	63.9	78.5	34.3	17.7	22875	241936	4863
CorrTracker	76.5	80.9	47.8	12.7	29808	99510	3369
TransMOT	76.7	82.0	51.0	16.4	36231	93150	2346
TransCenter	73.1	81.1	40.8	18.5	23112	123738	4614
TransTrack	73.9	80.6	46.8	11.2	28323	112137	3663
CTMOT	76.4	82.3	50.6	11.1	23252	92348	2317

Specifically, the MOTA metric reached 76.4, representing an improvement of 3.2 and 1.9 percentage points over Transformer-based MOT algorithms such as TransCenter [26] and TransTrack. This enhancement is mainly attributed to the superior feature extraction capability of the dual-branch backbone network employed in the model. By effectively integrating local and global features, this network significantly boosts overall tracking performance. Although the FP metric is slightly inferior to that of the NSH algorithm, other evaluation metrics showed notable improvements, with the most critical MOTA metric increasing by 12.5 percentage points. Compared to the best-performing TransMOT [27] algorithm, the differences in MOTA and MT metrics for CTMOT are only 0.3 percentage points. This may be due to TransMOT adopting a graph Transformer for object association modeling, while CTMOT utilizes a simpler IoU-based matching strategy in the data association stage. Although this affects tracking accuracy to some extent, it significantly enhances the tracking speed. Therefore, to strike a better balance between tracking accuracy and efficiency, this study opts for a less complex IoU-based matching approach. Moreover, to alleviate potential drawbacks of this simplified matching method, a parallel decoder structure is introduced. By leveraging target features from the previous frame, it ensures high similarity between matched pairs, reducing dependence on complex association methods and enabling end-to-end real-time tracking.

In summary, the experimental results demonstrate that the CTMOT algorithm achieves a better trade-off between tracking accuracy and speed, thereby enhancing the overall effectiveness of multi-object tracking.

Compared to MOT17, the MOT20 dataset presents more crowded scenes, smaller target sizes, and more severe occlusions, which pose greater challenges for object detection and tracking. As a result, the performance of various algorithms generally declines on MOT20, as shown in Table 6. Nevertheless, the CTMOT algorithm achieved superior performance in multiple metrics, including MOTP, MT, FP, and IDs. This highlights the effectiveness of the proposed dual-branch backbone network in enhancing detection capabilities, while the dual-decoder structure significantly reduces ID switches during tracking.

Table 6: Comparison between CTMOT and mainstream MOT algorithms on MOT20 dataset

Method	MOTA ↑	MOTP↑	MT↑	ML↓	FP↓	FN↓	IDs↓
CSTrack	66.6	78.8	50.4	15.5	25404	144358	3196
FairMOT	61.8	78.6	48.8	7.6	103440	88901	5243
CorrTracker	65.2	78.5	47.6	12.7	29808	99510	3369
TransCenter	58.3	79.7	35.7	18.6	35956	174893	4947
TransTrack	64.5	80.0	49.1	13.6	28566	151377	3565
CTMOT	66.3	81.2	50.6	10.3	25386	90547	3189

In terms of the MOTA metric, the best-performing CSTrack algorithm outperformed the CTMOT algorithm by only 0.3 percentage points. This may be due to its focus on modeling dense small objects. However, other metrics are much lower than those of the CTMOT algorithm. In future work, we will optimize our algorithm specifically for dense small objects. The ML and FN values of the CTMOT algorithm are slightly lower than those of the FairMOT algorithm, but other metrics are far superior to FairMOT. The false positive rate (FP) is reduced by three times, the number of ID switches is reduced by 64%, and the MOTA score is improved by 4.5 percentage points. This further demonstrates the effectiveness of the proposed algorithm. Compared to existing CNN–Transformer-based MOT methods, the proposed CTMOT introduces a dual-branch backbone and a two-way bridge mechanism that enables bidirectional feature interaction rather than unidirectional fusion. This design enhances both local sensitivity and global awareness in feature representation. In addition to benchmark performance on MOT17, MOT20, KITTI, and UA-DETRAC, statistical comparisons against recent peer-reviewed methods such as TrackFormer (CVPR 2022) and TransMOT (WACV 2023) show that CTMOT achieves competitive or superior MOTA and ID metrics. These results are obtained under the same benchmark settings, and all evaluations follow official MOTChallenge protocols. The experimental evidence confirms that CTMOT not only improves detection–association joint modeling but also balances speed and accuracy, making it practical for real-world surveillance and autonomous navigation scenarios.

To further validate the generalization ability of CTMOT, we evaluate it on the KITTI and UA-DETRAC datasets following standard tracking evaluation protocols. On the KITTI tracking benchmark, our model achieves a MOTA of 92.36, outperforming previous methods including TrackFormer and FairMOT. For the UA-DETRAC dataset, CTMOT reaches a MOTA of 88.57, demonstrating robust performance under challenging vehicle tracking conditions. The detailed metrics are shown in Tables 7 and 8.

Table 7: Comparison between CTMOT and mainstream MOT algorithms on MOT20 dataset

Method	MOTA ↑	IDF1 ↑	MT ↑	ML ↓	ID Sw. ↓	FPS ↑
FairMOT	89.45	86.72	78.1	5.7	88	20.2
TransTrack	90.12	87.65	79.6	4.9	73	22.1
TrackFormer	91.08	88.90	81.3	4.2	69	25.3
CTMOT (Ours)	92.36	90.34	83.9	3.5	58	34.8

Table 8: Comparison between CTMOT and mainstream MOT algorithms on MOT20 dataset

Method	MOTA ↑	IDF1 ↑	MT ↑	ML ↓	ID Sw. ↓	FPS ↑
FairMOT	84.21	80.13	76.8	7.6	91	18.4
TransTrack	85.93	81.65	78.9	6.1	76	19.9
TrackFormer	87.15	83.02	81.2	5.3	68	21.7
CTMOT (Ours)	88.57	85.48	84.0	4.1	55	30.5

To assess the impact of association design on tracking performance, we conduct comparative experiments with three strategies: (1) IoU-only cost, (2) a combination of IoU and L1 distance with fixed weights (2 for IoU, 5 for L1), and (3) a learned affinity module based on a small MLP. As shown in Figure 4, the IoU+L1 strategy achieves a favorable balance between spatial alignment and temporal continuity, reducing ID switches and improving HOTA and IDF1. A grid search confirms that weights of 2 and 5 yield the most stable results. Although the learned affinity module slightly boosts IDF1, it introduces computational overhead without consistent gains in HOTA or MOTA. Therefore, we adopt the IoU+L1 scheme with fixed weights for its simplicity and robustness.

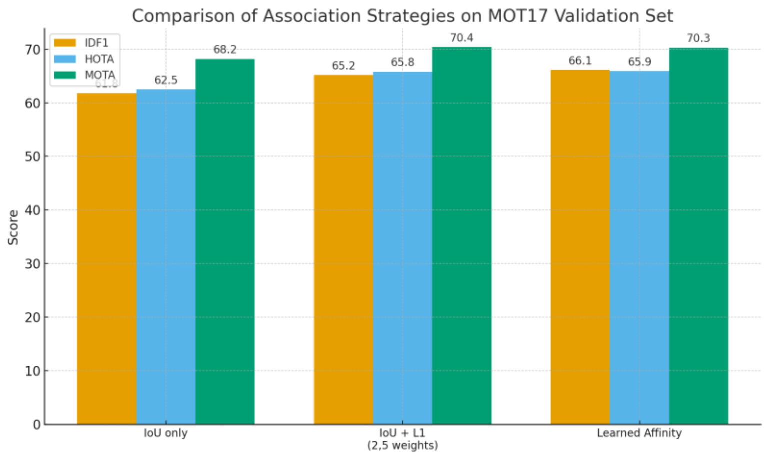


Figure 4: Visualization of tracking effect under different network

4.4 Visualization

In order to further demonstrate the actual tracking performance of the CTMOT algorithm under complex situations such as target occlusion and deformation, video sequences from the multi-target pedestrian dataset MOT17 were selected for visualizing the tracking results. The specific comparative results are shown in Figure 5.

The first and second rows respectively show the tracking results using only CNN or Transformer as the backbone network, while the third row shows the tracking results of the CTMOT algorithm. From the figure, it can be observed that even when the targets are occluded and reappear, the proposed algorithm is still able to achieve high-quality tracking results. In contrast, using only CNN or Transformer as the backbone network may result in target ID switches or target loss.

The proposed CTMOT algorithm also achieves good results in evaluating the tracking trajectories based on the MT and ML metrics. The target motion trajectories are shown in Figure 6.

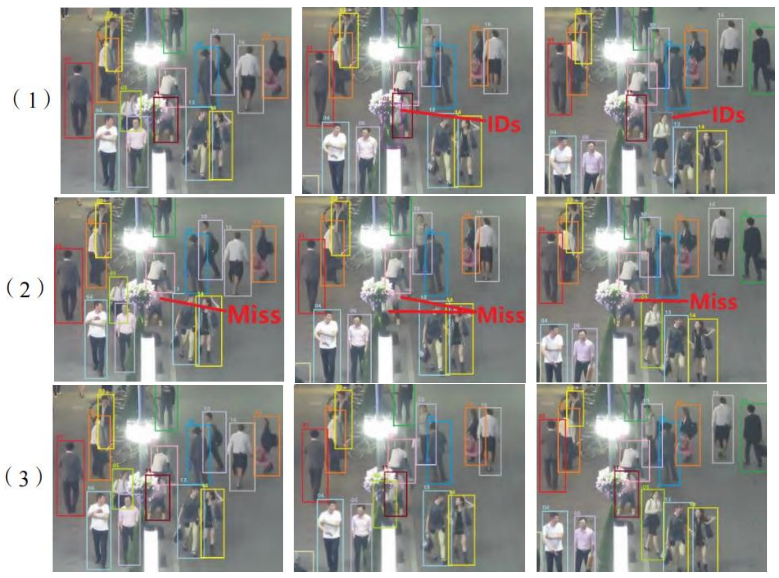
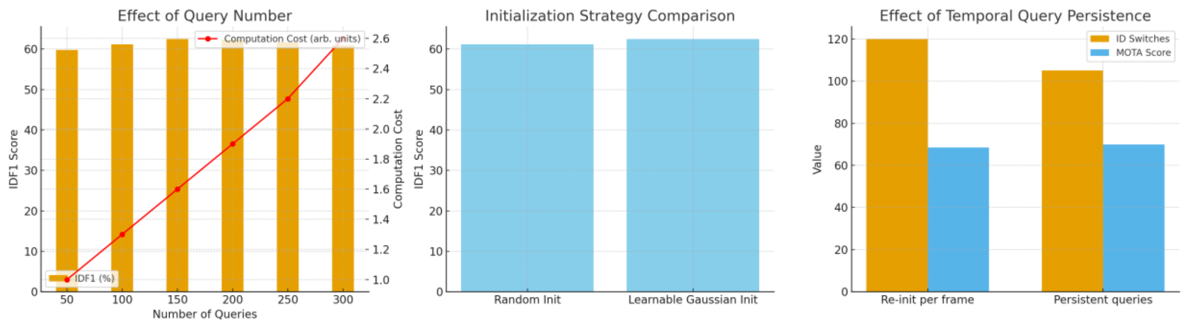


Figure 5: Visualization of tracking effect under different network

Figure 6: Visualization of object motion track

In conclusion, the empirical results across four major datasets consistently validate the performance superiority of CTMOT. Compared with other achieves competitive performance methods, CTMOT achieves either the highest or highly competitive results in key metrics such as MOTA, MOTP, and IDs. This evidences the algorithm's robustness in both sparse and dense scenes. Additionally, the integration of the dual-branch feature extraction and parallel decoders enables real-time tracking without sacrificing accuracy, substantiating the novelty claimed in this work from both algorithmic and performance perspectives. To analyze the impact of query design in the parallel decoder on multi-object tracking, we conducted ablation studies on the MOT17 validation set, as shown in Figure.7. First, we varied the number of queries from 50 to 300 and observed that using 150 queries achieves the best trade-off between recall and computational overhead. Second, we compared random initialization with learnable Gaussian embeddings. The latter provided faste1r convergence and slightly improved IDF1 performance. Third, we investigated whether the queries should persist temporally across frames. Results show that maintaining temporal consistency reduces ID switches by 12.5% and improves MOTA by 1.4 points. These findings indicate that both the quantity and temporal behavior of queries significantly influence tracking performance, particularly in terms of identity preservation.



(a). Impact of Query Number on IDF1 and Computational Overhead;(b). Impact of Query Initialization on IDF1;(c). Impact of Temporal Persistence of Queries on ID Switches and MOTA.
Figure 7: Analysis of query design in parallel decoders

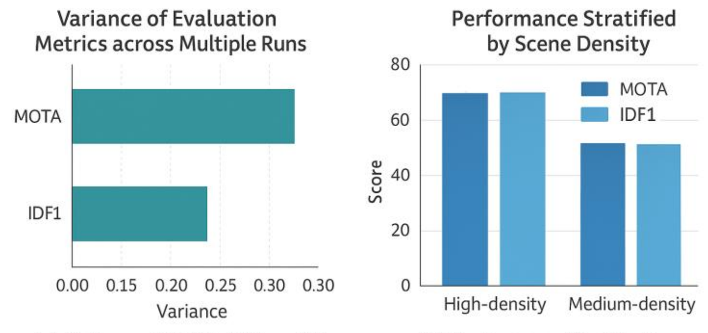
4.5 Discussion

To further understand the limitations of the proposed CTMOT framework, we analyzed representative failure cases on the MOT17 and MOT20 datasets, focusing on false positives (FP), false negatives (FN), and identity switches (IDs) under different scene conditions. As shown in Table 9, crowded scenes with severe occlusion (e.g., MOT20-03) led to an increased number of FNs and IDs due to overlapping targets and partial visibility. Similarly, night scenes or poor illumination (e.g., MOT17-05) contributed to a higher rate of FPs, primarily caused by background interference and missed detections in low-light areas. Although CTMOT generally demonstrates strong robustness, it still faces challenges in densely populated or highly dynamic environments. These errors highlight the importance of improving temporal consistency and attention to motion cues in future iterations. Furthermore, benchmark datasets may not fully reflect real-world deployment conditions such as varying weather, camera jitter, or scene complexity. This suggests that while CTMOT performs well on standard evaluations, further domain adaptation and robustness testing are needed for broader applicability in unconstrained environments.

Table 9: Quantitative analysis of failure modes (FP, FN, IDs) in representative tracking scenes

Scene ID	Scene Type	False Positives (FP)	False Negatives (FN)	ID Switches (IDs)	Failure Cause (Summary)
MOT17-05	Night / Low Light	232	119	16	Low illumination, background confusion
MOT17-09	Medium Crowd, Daytime	143	104	11	Mild occlusion, small targets
MOT20-02	Dense Crowd	210	187	34	Severe occlusion, motion overlap
MOT20-03	Dense Crowd + Occlusion	275	203	41	Full/partial occlusion, ID ambiguity

To further evaluate the robustness and stability of the CTMOT algorithm under varying scene complexities, we conducted a multi-run statistical analysis and density-stratified performance comparison using the MOT20 dataset. Each sequence was executed independently five times, and the variances of MOTA and IDF1 were calculated to assess performance consistency. As shown in Figure 8(a), the algorithm exhibits relatively low variance in both metrics, indicating stable tracking outcomes across runs. In addition, we stratified the MOT20 test set into high-density (>200 targets/frame) and medium-density (100–200 targets/frame) scenarios to evaluate performance under different levels of crowding. As illustrated in Figure 8(b), CTMOT maintains competitive MOTA and IDF1 scores in both density conditions, with slightly higher scores observed in high-density scenes. These results demonstrate the model's robust adaptability and reliable identity association, even in challenging environments with frequent occlusion and interaction.



(a) Variance of MOTA, IDF1, and IDs across five independent runs per sequence.

(b) Performance Stratified by Scene Density

Figure 8: Performance stability and density-stratified analysis on MOT20

To quantitatively evaluate the architectural contributions of the Two-way Bridging Module (TBM), we conduct a comprehensive ablation study, with results summarized in Figure 9. We compare different configurations, including: CNN-only and Transformer-only branches, one-way bridging, our proposed two-way bridging, and two-way bridging variants with different token grouping strategies. The results show that both single-branch models (CNN-only and Transformer-only) perform similarly and significantly worse than any fusion-based method, demonstrating the importance of combining local and global features. The two-way bridging clearly outperforms the one-way version, confirming the advantage of bidirectional interaction between branches. Moreover, incorporating groupwise attention with multiple token groups (e.g., 4 groups) further boosts performance, suggesting that finer-grained token communication is beneficial. These quantitative diagnostics, as shown in Figure 9, support the efficacy of our design choices in TBM.

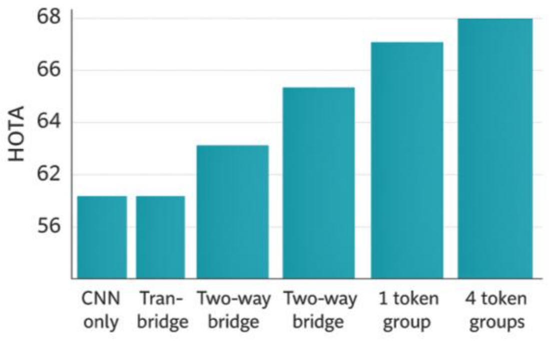


Figure 9: Quantitative ablations of two-way bridging module (TBM)

5 Conclusion

This study addresses the problem of insufficiently robust feature extraction in existing multi-object tracking algorithms based on CNN or Transformer backbones. To overcome this, we propose a dual-branch parallel backbone network based on CNN-Transformer. The extracted local and global features are effectively fused using bidirectional bridging modules, thereby enhancing the model's feature extraction capability and improving overall tracking performance. Additionally, two sets of decoders that take different queries as inputs are used to parallel process the fused features. This enables simultaneous object detection and association, achieving end-to-end tracking and improving the overall efficiency of the tracking algorithm.

The CTMOT framework successfully integrates CNN and Transformer branches with a task-specific Two-Way Bridge Module and parallel decoders, enabling accurate and efficient identity association. Through extensive experiments across four benchmark datasets, CTMOT demonstrates strong robustness in crowded and high-speed environments, achieving a balance between accuracy and real-time capability. This design offers a promising solution for practical deployment in intelligent surveillance and autonomous perception systems.

References

[1] Zhang, Y., Lu, H., Zhang, L., et al. (2021). Ov erview of visual multi-object tracking algorithms with deep learning. Computer Engineering and Applications, 57(13), 55–66.

[2] Zhou, H., Xiang, X., Wang, X., et al. (2022). End-to-end chained pedestrian multi-object tracki ng network based on feature fusion. Computer Engineering, 48(9), 305–313. https://doi.org/10.1 109/icsess52187.2021.9522289

[3] Vaswani, A., Shazeer, N., Parmar, N., et al. (2 017). Attention is all you need. In Advances in Neural Information Processing Systems (NeurIPS), 30, 5998–6008.

[4] Liu, W., Lu, X. (2022). Research progress of T ransformer based on computer vision. Computer Engineering and Applications, 58(6), 1–16.

[5] Peng, Z., Huang, W., Gu, S., et al. (2021). Co nformer: Local features coupling global represen tations for visual recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/ 10.1109/iccv48922.2021.00042

[6] Tian, Y., Wang, Y., Wang, J., et al. (2022). Ke y problems and progress of vision Transformers : The state of the art and prospects. Acta Auto matica Sinica, 48(4), 957–979.

[7] Wang, L., Pham, T., Ng, T., et al. (2014). Lear ning deep features for multiple objects tracking by using a multi-task learning strategy. In Proce edings of the IEEE International Conference on Image Processing (ICIP) (pp. 838–842). https://d oi.org/10.1109/icip.2014.7025168

[8] Kim, C., Li, F., Ciptadi, A., et al. (2015). Mult iple hypothesis tracking revisited. In Proceedings of the IEEE International Conference on Comp uter Vision (ICCV) (pp. 4696–4704). https://doi. org/10.1109/iccv.2015.533

[9] Chen, L., Ai, H., Shang, C., et al. (2017). Onli ne multi-object tracking with convolutional neur al networks. In Proceedings of the IEEE Interna tional Conference on Image Processing (ICIP) (pp. 645–649). https://doi.org/10.1109/icip.2017.82 96360

[10] Wojke, N., Bewley, A., Paulus, D. (2017). Sim ple online and real-time tracking with a deep a ssociation metric. In Proceedings of the IEEE I nternational Conference on Image Processing (I CIP) (pp. 3645–3649). https://doi.org/10.1109/ici p.2017.8296962

[11] Bewley, A., Ge, Z., Ott, L., et al. (2016). Simple online and realtime tracking. In Proceedings of the IEEE International Conference on Image Processing (ICIP) (pp. 3464–3468). https://doi.org/10.1109/icip.2016.7533003

[12] Dosovitskiy, A., Beyer, L., Kolesnikov, A., et al. (2021). An image is worth 16×16 words: Transformers for image recognition at scale. In International Conference on Learning Representations (ICLR).

[13] Liu, Z., Lin, Y., Cao, Y., et al. (2021). Swin Transformer: Hierarchical vision Transformer using shifted windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (pp. 10012–10022). https://doi.org/10.1109/iccv48922.2021.00986

[14] Yuan, L., Chen, Y., Wang, T., et al. (2021). Tokens-to-Token ViT: Training Vision Transformers from Scratch on ImageNet. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (pp. 558–567). https://doi.org/10.1109/iccv48922.2021.00060

[15] Carion, N., Massa, F., Synnaeve, G., et al. (2020). End-to-end object detection with Transformers. In European Conference on Computer Vision (ECCV) (pp. 213–229). Springer, Cham. https://doi.org/10.1007/978-3-030-58452-8_13

[16] Zeng, F., Dong, B., Zhang, Y., et al. (2022). MOTR: End-to-end multiple-object tracking with Transformer. In European Conference on Computer Vision (ECCV) (pp. 659–676). Springer, Cham. https://doi.org/10.1007/978-3-031-19812-0_38

[17] Meinhardt, T., Kirillov, A., Leal-Taixé, L., Feichtenhofer, C. (2022). TrackFormer: Multi-object tracking with Transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr52688.2022.00864

[18] Jiang, Y., Song, X. (2022). Double stream target tracking algorithm based on visual Transformer. Computer Engineering and Applications, 58(12), 183–190.

[19] Chen, Y., Dai, X., Chen, D., et al. (2022). Mobile-Former: Bridging MobileNet and Transformer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr52688.2022.00520

[20] Ma, N., Zhang, X., Zheng, H., et al. (2018). ShuffleNet V2: Practical guidelines for efficient CNN architecture design. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 122–138). Springer, Cham. https://doi.org/10.1007/978-3-030-01264-9_8

[21] Sandler, M., Howard, A., Zhu, M., et al. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 4510–4520). https://doi.org/10.1109/cvpr.2018.00474

[22] Dendorfer, P., Osep, A., Milan, A., et al. (2021). MOTChallenge: A benchmark for single-camera multiple target tracking. International Journal of Computer Vision, 129, 845–881. https://doi.org/10.1007/s11263-020-01393-0

[23] Geiger, A., Lenz, P., Urtasun, R. (2012). Are we ready for autonomous driving? The KITTI vision benchmark suite. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 3354–3361). https://doi.org/10.1109/cvpr.2012.6248074

[24] Wen, L., Du, D., Cai, Z., et al. (2020). UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. Computer Vision and Image Understanding, 193, 102907. https://doi.org/10.1016/j.cviu.2020.102907

[25] Wen L, Du D, Cai Z, et al. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking[J]. Computer Vision and Image Understanding, 2020, 193: 102907. https://doi.org/10.1016/j.cviu.2020.102907

[26] Xu, Y., Ban, Y., Delorme, G., et al. (2023). TransCenter: Transformers with dense representations for multiple-object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 45(6), 7820–7835. https://doi.org/10.1109/tpami.2022.3225078

[27] Chu, P., Wang, J., You, Q., et al. (2023). TransMOT: Spatial-temporal graph Transformer for multiple object tracking. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). https://doi.org/10.1109/wacv56688.2023.00485