DragonTrack: Transformer-Enhanced Graphical Multi-Person Tracking in Complex Scenarios

Bishoy Galoaa[1]★, Somaieh Amraee[1]★, Sarah Ostadabbas[1]*
[1]Augmented Cognition Lab (ACLab), Department of Electrical and Computer Engineering,
Northeastern University, Boston, MA, USA
★These authors contributed equally to this work.
*Corresponding author: ostadabbas@ece.neu.edu

Abstract

This paper introduces the dynamic robust adaptive graph-based tracker (DragonTrack), as a novel end-to-end framework for multi-person tracking (MPT) by integrating a detection transformer model for object detection and feature extraction with a graph convolutional network for re-identification. DragonTrack leverages encoded features from the transformer for precise subject matching and track maintenance, while the graphical component processes these features alongside geometric data to predict subsequent positions of tracked people. This methodology aims to enhance tracking accuracy and reliability, as evidenced by improvements in key metrics such as higher order tracking accuracy (HOTA) and multiple object tracking accuracy (MOTA). We quantitatively compare DragonTrack with state-of-the-art methods on MOT17, MOT20, and DanceTrack datasets, in which DragonTrack outperforms other methods. In challenging scenarios such as DanceTrack, DragonTrack achieves an impressive MOTA score of 93.4, significantly higher than the second-best SOTA method, ByteTrack, which achieves only 89.6. Similarly, on MOT17, DragonTrack scores 82.0 in MOTA, surpassing the closest competitor with a score of 80.3. On MOT20, DragonTrack attains a HOTA score of 63.2, outperforming the next best method scoring 62.6[1].

1. Introduction

Multi-person tracking (MPT), a pivotal component of computer vision, plays a crucial role in identifying and tracking individuals across video streams with distinct identifiers for each subject [2, 25]. Operating within the expansive domain of multiple object tracking (MOT), MPT finds extensive applications in areas such as surveillance, robotics, and autonomous vehicles [1, 2, 12, 25, 34], enabling the concurrent tracking of several entities. The essential processes of MOT—detection, localization, and association—work in tandem to ensure the accurate identification, positioning, and consistent tagging of objects across a video feed [20].

With the considerable progress in MOT techniques over the last decade [10,29,32,36], significant challenges remain, particularly in preserving temporal connections during frequent occlusions and periods of diminished visibility, as illustrated in Figure 1. This scenario, depicted through the output of the ByteTrack algorithm [36], highlights the pivotal challenge of sustaining consistent identity tracking over successive frames when direct line-of-sight to subjects is obstructed. The complexity increases when attempting to track individuals with similar features, requiring the development of more sophisticated methods for effective distinction [31].

Addressing these challenges, our paper presents the dynamic robust adaptive graph-based tracker named DragonTrack, as an innovative end-to-end approach. While graph neural networks have been previously integrated into MOT systems, such as MPNTrack [6] for 2D offline tracking and online graph representations for 3D multi-object tracking (OGR3MOT) [33], our approach uniquely combines the detection transformer (DETR) [8] with graphical convolutional networks (GCN) [7,30], enhanced by a multi-edge attribute graph and an attention mechanism. The contributions of our research are as follows:

- Introducing DragonTrack, an end-to-end tracking framework designed to enhance precision in multi-person scenarios. Unlike previous approaches, it leverages the detection transformer alongside a multi-edge attribute graph with an integrated attention mechanism, enabling granular analysis of individual contexts and interactions.

[1]The DragonTrack code is available at https://github.com/ostadabbas/DragonTrack.

Supported by NSF-CAREER Grant #2143882.

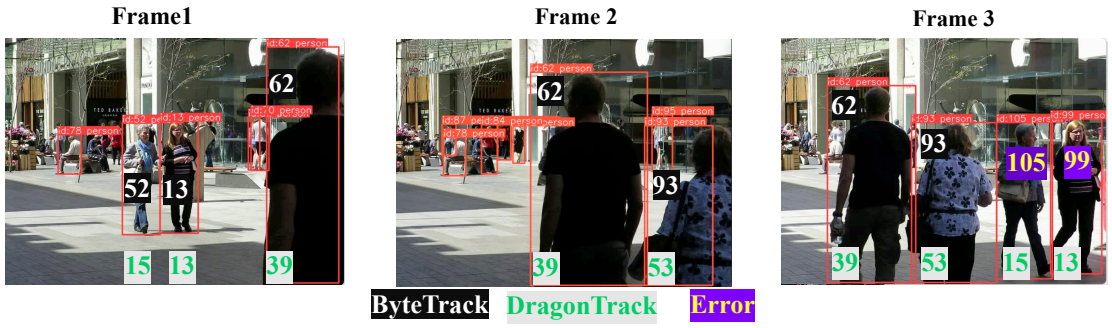

Figure 1. Illustration of occlusion-induced identity challenges in multiple person tracking across three frames: This sequence highlights a common issue in multi-object tracking, where occlusions can cause identity switching or generate new false identities for the same person. In Frame 1, multiple people are correctly identified by their respective IDs (e.g., person 52, 13, 15). As the sequence progresses to Frame 2, persons 52 and 13 are completely occluded by person 62, causing ByteTrack [36] to assign new IDs in the subsequent frame. In Frame 3, due to the occlusion, ByteTrack mistakenly assigns two new IDs (105 and 99) for persons 52 and 13, resulting in a tracking error. Our DragonTrack approach (shown with green IDs) successfully maintains the correct identities throughout the sequence, demonstrating superior tracking performance, particularly under occlusions. ByteTrack [36], while a top-performing method (ranked second in terms of MOTA according to Table 1), struggles in this scenario, whereas DragonTrack excels by mitigating identity switches even during challenging conditions.

- Incorporating a novel graph attention mechanism that dynamically prioritizes critical attributes within the multi-edge graph structure. This synergy between graph representation and attention enhances the adaptability of the tracking approach across diverse scenarios, going beyond the static graph structures used in previous works.
- Integrating a combined loss function, utilizing weighted binary cross-entropy and contrastive loss, to address class imbalances and promote the learning of discriminative features between subjects, thereby improving tracking accuracy.
- Merging geometric features with appearance and positional data through a graphical network to accurately predict object positions based on spatial relationships and interactions. This leads to improved tracking accuracy and closes the gap between MOTA and HOTA performances, especially in high occlusion scenarios.
- Presenting a learnable Sinkhorn algorithm that represents a differentiable approach to optimizing track-detection assignments, leading to enhanced overall tracking performance. This is a novel application in the context of multi-person tracking.

These enhancements not only improve the effectiveness of multi-person tracking systems but also provide a foundation for future advancements in the field. Our approach addresses the limitations of previous methods, particularly in handling complex scenarios with high occlusion rates and similar-appearing subjects, as highlighted by the challenges illustrated in Figure 1.

2. Related Works

2.1. CNN-based Methods: Tracking by Detection

Most traditional multi-object tracking methodologies adopt the tracking-by-detection paradigm, where convolutional neural network (CNN) object detectors first localize objects in each frame. Subsequent track association across frames is then executed to generate tracking outcomes. For example, SORT [5] combines Kalman Filter and Hungarian algorithm for efficient track association, whereas DeepSORT [29] and Tracktor [3] introduce cosine distance to enhance appearance similarity measures in track association. ByteTrack [36] introduces a versatile data association strategy, BYTE, addressing missing detections through the inclusion of low-score candidates, such as occluded objects. Recently, StrongSORT [10] has advanced DeepSORT by improving object detection, feature embedding, and trajectory association, leveraging both motion and appearance-based similarity metrics. Generally, CNN-based MOT methods, with the paradigm of tracking by detection, are well-established and widely used. However, they may suffer from limitations such as scalability, dependency on accurate detection, and difficulties in handling occlusions and complex interactions. In these methods, the reliance on post-processing for association limits the full exploitation of temporal dynamics in video sequences.

2.2. End-to-End and Transformer-based Frameworks

A shift towards end-to-end MOT frameworks allows for the implicit learning of appearance and positional variances. These frameworks conceptualize MOT as sequential prediction tasks, integrating detection, localization, and identifica-

tion (association) in a cohesive process. Some approaches, such as MPNTrack [6], introduce neural message-passing for 2D offline tracking, enabling end-to-end optimization without relying on transformers.

Transformer-based methods like MOTR [35] employ deep learning to simultaneously optimize detection, tracking, and reidentification, thus eliminating the dependency on handcrafted features. Extending DETR [8], MOTR introduces a "track query" mechanism, evolving from "image query" in transformers, to model tracked instances across a video sequence, facilitating frame-by-frame iterative prediction. MOTRv2 [38] further refines this approach, employing a bootstrapping technique with pretrained detectors to improve tracking accuracy. In the 3D tracking domain, OGR3MOT [33] extend the concept to online scenarios using learnable graph representations.

Despite these advancements, MOT remains challenged by scenarios with high occlusion rates, as shown in Figure 1, affecting key accuracy metrics like HOTA and MOTA and limiting real-world applicability. This paper introduces DragonTrack, a novel framework that combines transformer-based detection with graph neural networks, designed for robust tracking in complex scenarios, including subjects' re-entries or occlusions. Notably, DragonTrack significantly enhances tracking accuracy, particularly in HOTA, outperforming contemporary MOT solutions.

3. Introducing DragonTrack Algorithm

Our end-to-end MPT method, DragonTrack, synergies the capabilities of graphical convolutional networks with an attention mechanism to adeptly handle complex data structures. DragonTrack is tailored to address the intricacies of integrating dynamic features from various dimensions—geometric, positional, and appearance—and ensuring temporal consistency in sequential data analysis. Our model employs a detection transformer for the initial extraction of features, which are further refined by specialized affinity networks, and then integrated and analyzed using a graphical network.

Figure 2 shows the block diagram of DragonTrack detailing its process flow from inputting past and current frames into the transformer for object detection, extracting geometric, appearance, and positional features of each detected subject, to integrating these features through individual multi-layer perceptrons (MLPs) and a graph convolutional attention network. This network dynamically weighs the importance of different attributes for robust tracking. The process culminates in the application of a learnable Sinkhorn algorithm and Hungarian matching to assign unique IDs to subjects and re-identify them accurately across frames. This addresses the challenges of occlusions, exit/enter events, and maintaining temporal associations, significantly improving multi-person tracking accuracy and robustness.

3.1. Overview of the Proposed Algorithm

Given F_t the current frame of the video at time t, and a window of previous frames up to F_{t-N_i}, where N_i is a variable which is subsequently refined by specialized affinity networks, our model employs a detection transformer (DETR) for the initial extraction of features from all frames in the window. For each detection, the DETR outputs a bounding box directly in the text as $B = (x, y, h, w)$, where (x, y) is the top-left coordinates of the detected object, and (h, w) denotes its height and width. The positional embedding e_{ij}^{pos}, capturing the spatial relation between objects i and j, is computed as:

$$e_{ij}^{pos} = \begin{bmatrix} x_i - x_j \\ y_i - y_j \\ \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \end{bmatrix}. \quad (1)$$

These features are integrated and analyzed using a graph convolutional network (GCN), as depicted in Figure 2. The process involves feeding past and current frame data into the transformer for object detection, extracting geometric, appearance, and positional features, and integrating these through individual multi-layer perceptrons (MLPs) and the GCN. Post feature concatenation (h_{cat}), we construct a graph where nodes represent detected objects and edges are formed based on the affinity between positive and negative contrastive pairs. This graph structure underpins the interaction embeddings $h_{int} = \text{GCN}(h_{cat})$, which encode the complex interactions among different objects in the scene. The process culminates in the application of a learnable Sinkhorn algorithm and Hungarian matching to assign unique IDs to subjects and re-identify them accurately across frames. The entire network is trained end-to-end with a combined loss function balancing the weighted binary cross-entropy loss L_{WBCE} and the contrastive loss L_{CONT}. This loss combination optimizes tracking performance, ensuring temporal coherence and effectively handling challenges such as occlusions and entry/exit events.

3.2. Feature Extraction and Affinity Networks

Transformer-Based Encoder The adoption of a transformer-based encoder for initial feature extraction is grounded in the transformer's demonstrated efficacy in extracting rich, contextual information from visual data. The transformer-based architecture of the detection transformer processes images in their entirety, extracting comprehensive and context-aware features that serve as a robust basis for subsequent processing.

Affinity Networks After initial feature extraction, affinity networks refine features related to geometric, positional,

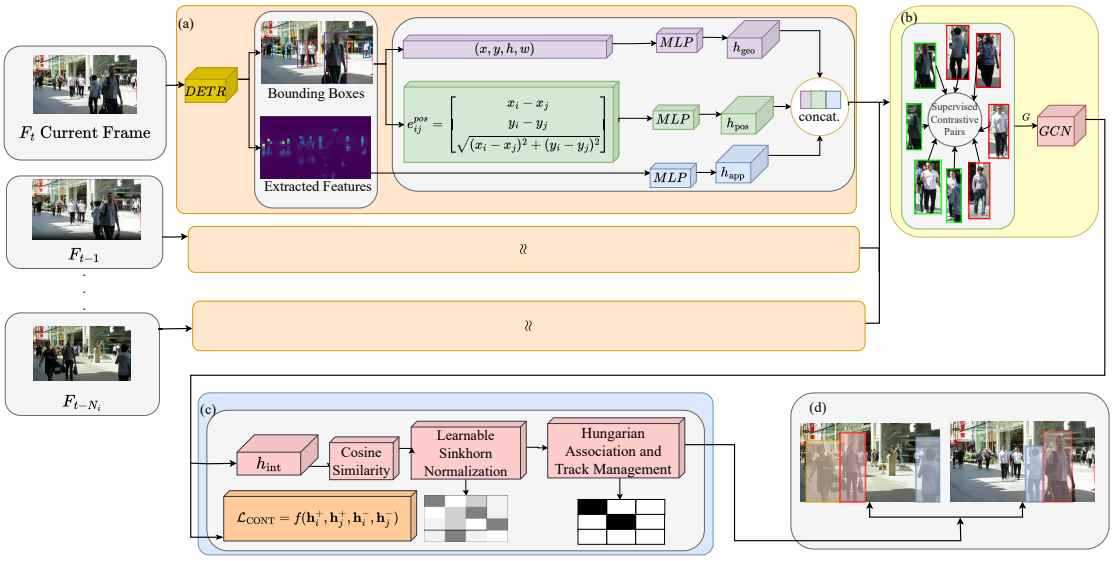

Figure 2. Schematic of the DragonTrack framework: (a) Input processing: The current frame (F_t) and a variable number of previous frames ($F_{t-1}, \ldots, F_{t-N_i}$) are processed by DETR to generate bounding boxes and feature embeddings. DETR extracts both spatial and appearance information. (b) Feature refinement: MLPs derive geometric (h_{geo}), positional (h_{pos}), and appearance (h_{app}) embeddings from the DETR output. These embeddings capture different aspects of object relationships. (c) Graph-based learning: A GCN further refines these features using supervised contrastive learning, enabling the model to capture complex inter-object relationships. (d) Track management: Cosine similarity is computed between object embeddings, followed by learnable Sinkhorn normalization. The Hungarian method is then applied for optimal track-to-detection assignment. The model's loss function combines weighted BCE with contrastive loss, optimizing for discriminative feature learning given the variable frame history N_i. This multi-stage process enables robust tracking across varying temporal contexts.

and appearance attributes. This multi-faceted approach addresses the unique contributions of each attribute type. Geometric and positional attributes help decode spatial relationships and dynamics within scenes. Incorporating positional encoding alongside geometric features enhances the model's performance by providing explicit spatial context, which is crucial for predicting object trajectories and relationships. Positional encodings also capture temporal dynamics, aiding in object tracking over time.

Refining appearance features is key for distinguishing entities, important for object recognition and classification. Combining positional encodings with appearance features adds a spatial dimension, improving predictions of object identities and movements. This integration is especially useful in scenes with multiple similar-looking objects, as positional encodings serve as disambiguation cues, enhancing the model's discriminative power.

The inclusion of positional encodings enriches the feature set, enabling the model to make more nuanced decisions based on a comprehensive understanding of the scene. This leads to improved tracking performance, highlighting the importance of integrating positional encodings with geometric and appearance features.

3.3. Graphical Tracking with Attention Mechanism

In the GCN layer, as shown in Figure 2, each node's feature update is computed based on the features of its neighboring nodes, weighted by an attention mechanism. The attention weights are calculated based on the similarity between the edge features and the node features.

The updated feature for node i, denoted by h_i', is computed as follows:

$$h_i' = \sigma\left(\frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \alpha_{ij} \cdot W \cdot h_j\right), \tag{2}$$

where $\mathcal{N}(i)$ is the set of neighbors of node i, and α_{ij} represents the attention weight between nodes i and j. The weight α_{ij} depends on the edge features e_{ij} and the node features h_i and h_j. The matrix W is a learnable weight matrix, and $\sigma(\cdot)$ denotes a non-linear activation function (ReLU), which is applied to the sum of the weighted features.

Network Architecture The network backbone consists of a feature extractor for initial feature extraction, followed by affinity networks to process specific edge attributes, including appearance, geometric, and positional information. Each node in the graph represents a detected object in the

video frame, and these nodes are characterized by the features extracted via the feature extractor.

Edges between nodes capture the relationships between objects, including geometric proximity, appearance similarity, and positional dynamics. The affinity networks help evaluate and prioritize these attributes to facilitate accurate object association.

Attention Mechanism The core of our GCN layer is the attention mechanism, which updates the node features by weighting the neighboring nodes' features based on their relevance. The attention weights α_{ij} are derived from the similarity between the nodes and the attributes of the edges connecting them. These attention weights are normalized using a softmax function to ensure that they sum to 1 across all neighbors of node i:

$$\alpha_{ij} = \frac{\exp(f_{ij})}{\sum_{k \in \mathcal{N}(i)} \exp(f_{ik})}. \tag{3}$$

where the similarity score f_{ij} is computed as:

$$f_{ij} = \text{LeakyReLU}(a^T \cdot [Wh_i \| Wh_j \| e_{ij}]). \tag{4}$$

Here, a^T is a learnable weight vector, and the concatenation operation $\|$ combines the transformed feature vectors $W \cdot h_i$, $W \cdot h_j$, and the edge features e_{ij}. The LeakyReLU function introduces non-linearity, ensuring that the attention coefficients are computed effectively. This formulation allows the GCN to dynamically adjust the importance of neighboring nodes based on the edge and node features, improving the model's ability to focus on relevant information and enhance prediction accuracy.

3.4. Learnable Sinkhorn Algorithm

We introduce a learnable Sinkhorn algorithm for matrix normalization during object matching. Given input matrix X, the algorithm iteratively normalizes rows and columns:

$$R_i = \frac{X_i}{\sum_{j=1}^{m} X_{ij} \cdot \theta_r + \sigma}, \tag{5}$$

$$C_j = \frac{R_{ij}}{\sum_{i=1}^{n} R_{ij} \cdot \theta_c + \lambda}. \tag{6}$$

Here, θ_r and θ_c are learnable scaling factors for row and column normalization, respectively, allowing the algorithm to adapt to different tracking scenarios.

3.5. Construction of Positive and Negative Subject Pairs

An essential step in our training process involves the strategic construction of positive and negative subject pairs, which are crucial for the effective application of contrastive loss. This construction, as shown in Figure 2, is designed to ensure that the model learns to accurately differentiate between subjects by closely analyzing their features.

Positive Subject Pairs Positive subject pairs are formed by selecting two different instances of the same subject. These instances can be either from the same frame (in the case of multiple detections of the same subject within a frame) or across consecutive frames, where the subject's appearance may slightly vary due to motion or changes in viewpoint. The key criterion for a pair being considered positive is that both instances must belong to the same subject identity. This is represented as:

$$P = \{(i, j) | \text{subject}_i = \text{subject}_j, i \neq j\}, \tag{7}$$

where i and j are instance indices that belong to the same subject.

Negative Subject Pairs Negative subject pairs are generated by pairing instances of different subjects. This pairing is crucial for teaching the model the critical task of distinguishing between different subjects' features. Unlike positive pairs, these instances do not share the same identity and can be selected from the same frame or across frames. The selection process aims to include a diverse set of pairs to encapsulate various scenarios and differences in appearances. The criterion for a pair being considered negative is formulated as $N = \{(i, k) | \text{subject}_i \neq \text{subject}_k\}$, where i and j are instance indices that belong to different subjects.

Balancing and Sampling Strategy Given the potentially vast number of possible negative pairs compared to positive pairs, a balanced sampling strategy is employed. This strategy ensures that the model is not biased towards learning to only distinguish negative pairs, which are more abundant. We employ techniques such as random sampling, hard negative mining, or other heuristic-based approaches to select a representative and balanced set of positive and negative pairs for training. This careful construction and selection of subject pairs are instrumental in the effective application of the contrastive loss component of our loss function, ultimately enhancing the model's ability to discern between different subjects and contributing to its overall performance.

3.6. Loss Functions

The total loss function combines Weighted Binary Cross-Entropy (WBCE) to address class imbalance, and contrastive loss to enhance discriminative features. The WBCE loss is defined as:

$$L_{\text{WBCE}} = -\sum_{i=1}^{N} w_i \left[y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \right], \tag{8}$$

where w_i is the weight for the i-th sample, y_i is the true label, and $\hat{y}_i$ is the predicted probability. The contrastive loss for positive and negative pairs is given by:

$$L_{\text{CONT}} = \sum_{(i,j)\in P} d(h_i, h_j)^2 + \sum_{(i,k)\in N} \max(0, m - d(h_i, h_k))^2, \tag{9}$$

where m is the margin, a hyperparameter that defines the minimum distance between embeddings of different subjects. where $d(h_i, h_j)$ represents the distance between the embeddings of positive pairs (i, j), m is the margin for negative pairs (i, k), and P and N are the sets of positive and negative pairs, respectively.

The total loss function is a weighted combination of both losses:

$$L_{\text{total}} = \alpha L_{\text{WBCE}} + (1 - \alpha) L_{\text{CONT}}, \tag{10}$$

where α controls the trade-off between the WBCE and contrastive loss.

3.7. Track Handling and Occlusion Management

DragonTrack employs a sophisticated track handling mechanism to address challenges such as occlusions, re-entries, and exit/enter events. This process is crucial for maintaining temporal consistency and robust tracking performance in complex scenarios.

Temporal Window Our algorithm utilizes a sliding temporal window of 30 previous frames. This window allows DragonTrack to consider historical information when making association decisions, which is crucial for handling occlusions and re-entries.

Distance-Based Association DragonTrack uses a spatial constraint for potential associations. Objects within 200 pixels can be associated, helping to reduce false associations between distant objects.

Track Continuation and Termination DragonTrack maintains tracks even when objects are temporarily occluded or leave the scene. The algorithm continues to predict the position of occluded objects for a certain number of frames, allowing for re-association if the object reappears.

By integrating these mechanisms, DragonTrack effectively manages the challenges of occlusions, re-entries, and false positives. The combination of temporal windowing, distance-based association, confidence thresholding, and track continuation contributes to DragonTrack's robust performance across varied and complex tracking scenarios, as demonstrated by our results on challenging datasets like MOT20 and DanceTrack.

The affinity network uses a simple MLP structure with one hidden layer (input size 2, output size 1) and ReLU activation. The GCN layers use 512 and 128 units respectively. The learning rate is set to 1e-4 with Adam optimizer. Training is performed for 20 epochs with a batch size of 2.

4. Experimental Results

4.1. Implementation Details

Setting The results have been obtained on a setup with an Intel Xeon CPU E5 2.40GHz, T4 GPU, and 15 GB of RAM. These hardware specifications provided the necessary computational power and memory capacity to support the intensive training process of our machine learning model. The model was trained for a total of 20 epochs, which spanned over a duration of 2 days. This training period was essential to achieve convergence and to ensure that the model adequately learned from the training data across all the loss functions being compared.

Hyperparameters The affinity network uses a simple MLP structure with one hidden layer (input size 2, output size 1) and ReLU activation. The GCN layers use 512 and 128 units respectively. The learning rate is set to 1e-4 with Adam optimizer. Training is performed for 20 epochs with a batch size of 2.

Dataset We conduct experiments on MOT17 [18], MOT20 [9], and DanceTrack [23]. MOT17 consists of 7 training and 7 testing sequences, primarily featuring crowded street scenes. MOT20 includes 4 training and 4 testing sequences with more crowded scenes and higher occlusions. DanceTrack contains 100 sequences, focusing on tracking in scenarios with high inter-object similarity and complex motions.

Metrics We follow the standard evaluation protocols to evaluate our method. The common metrics include higher order tracking accuracy (HOTA) [16] to evaluate our method and analyze the contribution decomposed into association accuracy (AssA) and detection accuracy (DetA). We also list the iterative and discriminative frame- work1 (IDF1) [21], multiple object tracking accuracy(MOTA) [4], and identity switches (IDS) metrics.

4.2. State-of-the-Art Comparison on Benchmarks

We evaluate DragonTrack against CNN-based methods [3, 10, 19, 22, 26–28, 36, 37, 39] and transformer-based methods [17, 24, 35, 38]. As shown in Table 1, DragonTrack achieves superior accuracy scores (HOTA, AssA, DetA, IDF1, MOTA) and fewer identity switches (IDS) compared to both categories. DragonTrack outperforms StrongSORT [10] in HOTA by 1.8 (65.3 vs 63.5) and ByteTrack [36] in MOTA by 1.7 (82.0 vs 80.3). Although MPNTrack [6] has fewer IDS errors than our method, it struggles with overall tracking accuracy, whereas our method excels in metrics such as HOTA, MOTA, and IDF1, providing more accurate tracking.

We also compared DragonTrack with the top-ranked methods (from Table 1) on MOT20 and DanceTrack datasets. As shown in Table 2, on MOT20, DragonTrack achieves the highest HOTA (63.2), and IDF1 (78.6) scores,

Table 1. Performance comparison on the MOT17 [18] dataset. The table is partitioned into two sections: CNN-based and transformer-based methods. The best results for each metric are highlighted in bold, while the second-best performing metrics are indicated in blue. DragonTrack has demonstrated superior performance compared to other MOT methods in both categories.

Methods	HOTA↑	AssA↑	DetA↑	IDF1↑	MOTA↑	IDS↓
CNN-based:						
Tracktor++ [3]	44.8	45.1	44.9	52.3	53.5	2072
MPNTrack [6]	49.0	51.1	47.3	61.7	58.8	**1185**
CenterTrack [39]	52.2	51.0	53.8	64.7	67.8	3039
TraDeS [19]	52.7	50.8	55.2	63.9	69.1	3555
QDTrack [19]	53.9	52.7	55.6	66.3	68.7	3378
GSDT [28]	55.5	54.8	56.4	68.7	66.2	3318
FairMOT [37]	59.3	58.0	60.9	72.3	73.7	3303
CorrTracker [26]	60.7	58.9	62.9	73.6	76.5	3369
GRTU [27]	62.0	62.1	62.1	75.0	74.9	1812
MAATrack [22]	62.0	60.2	64.2	75.9	79.4	1452
StrongSORT [10]	63.5	63.7	63.6	78.5	78.3	1446
ByteTrack [36]	63.1	62.0	64.5	77.3	80.3	2196
Transformer-based:						
TrackFormer [17]	-	-	-	63.9	65.0	3528
TransTrack [24]	54.1	47.9	61.6	63.9	74.5	3663
MOTR [35]	57.8	55.7	60.3	68.6	73.4	2439
MOTRv2 [38]	62.0	60.6	63.8	75.0	78.6	-
DragonTrack (Ours)	**65.3**	**66.2**	**65.8**	**79.2**	**82.0**	1313

Table 2. Performance metrics of the DragonTrack on MOT20 [9] and DanceTrack [23] datasets.

Method	MOT20			DanceTrack		
	HOTA	MOTA	IDF1	HOTA	MOTA	IDF1
MPNTrack [6]	46.8	57.6	59.1	-	-	-
ByteTrack [36]	61.3	**77.8**	75.2	47.7	89.6	53.9
StrongSORT [10]	62.6	73.8	77.0		-	
MOTRv2 [38]	61.0	76.2	73.1	69.9	91.9	71.7
DragonTrack (Ours)	**63.2**	77.2	**78.6**	**72.5**	**93.4**	**74.9**

outperforming MPNTrack [6], ByteTrack [36], StrongSORT [10], and MOTRv2 [38]. On DanceTrack, DragonTrack also leads with the highest HOTA (72.5), MOTA (93.4), and IDF1 (74.9) scores, surpassing MOTRv2 [38] and ByteTrack [36] by significant margins in all metrics.

We chose challenging datasets such as MOT20 and DanceTrack to further demonstrate DragonTrack's robustness in handling complex scenarios. The MOT20 dataset, known for its crowded scenes, contains 87,786 occlusion events, 4,303 exit events, and 22,365 entering events across its sequences [9]. Similarly, the DanceTrack dataset features 105,000 frames annotated for occlusion and motion events [23]. Based on these high scores across challenging datasets, we conclude that DragonTrack has a robust framework for handling occlusions and other complex tracking scenarios.

To further evaluate DragonTrack's performance, we compared several state-of-the-art algorithms in terms of their speed(FPS), accuracy (HOTA), and model complexity (number of parameters). These analyses are summarized in Section A.1 of the Supplementary Materials.

Furthermore, to evaluate our method in challenging real-world scenarios, we compared DragonTrack and ByteTrack [36] using an in-the-wild example of identical triplet toddlers found on YouTube in Section A.2 in the Supplementary Materials.

4.3. Ablation Study

Our ablation studies focus on evaluating the impact of various components and design choices in DragonTrack. We investigate the effects of different loss functions, the learnable Sinkhorn algorithm, the choice of detection backbone, and the use of the attention layer on the model's performance.

Table 3. The ablation study on the effect of different loss functions on DragonTrack performance. The symbols L_{WBCE}, L_{CONT}, T. A, O. A, and Z. A represent weighted binary cross-entropy loss, contrastive loss, total accuracy, ones accuracy, and zeros accuracy, respectively.

L_{WBCE}	L_{CONT}	T. A(%)	O. A(%)	Z. A(%)	HOTA (%)	MOTA (%)	IDF1 (%)
✓	✗	87.0	85.0	90.0	53.0	64.3	59.3
✗	✓	90.0	88.0	93.0	58.2	66.4	62.1
✓	✓	**95.0**	**93.0**	**97.5**	**65.3**	**82.0**	**79.2**

We begin by exploring the impact of different loss function combinations. Table 3 illustrates the effect of weighted binary cross-entropy (WBCE) and contrastive loss on the

model's performance. WBCE helps handle class imbalance by assigning different weights to classes, while contrastive loss aids in learning discriminative embeddings. The results clearly show that the combination of WBCE and contrastive loss yields superior outcomes across all metrics, demonstrating the synergistic effect of addressing both class imbalance and feature discrimination.

Next, we evaluate the impact of the learnable Sinkhorn algorithm, as described in Section 3.4. Table 4 demonstrates that incorporating the learnable Sinkhorn algorithm significantly improves the model's performance, particularly in terms of MOTA score. This improvement can be attributed to the algorithm's contribution to enhanced matching and association capabilities within the model.

Table 4. Ablation study on the effect of the learnable Sinkhorn algorithm on DragonTrack.

Ablation	HOTA	MOTA	IDF1
DragonTrack w/o Sinkhorn	62.2	78.7	75.4
DragonTrack w/ Sinkhorn	**65.3**	**82.0**	**79.2**

Table 5. Comparison of DETR, YOLOv8n, and Fast R-CNN backbones in DragonTrack on MOT17 dataset.

Backbone	HOTA↑	AssA↑	DetA↑	IDF1↑	MOTA↑	IDS↓	FPS↑	Params↓
Fast R-CNN [11]	52.1	53.3	52.8	65.7	68.2	2345	6	25 M
YOLOv8n [14]	63.8	64.5	64.2	77.6	80.5	1425	**45**	**3.2** M
DETR [8]	**65.3**	**66.2**	**65.8**	**79.2**	**82.0**	**1313**	12	41 M

To assess the effectiveness of our DETR-based feature extractor, we compare it with YOLOv8n [14], a state-of-the-art object detector known for its efficiency, and Fast R-CNN [11], a traditional object detection model. Table 5 presents the results of this comparison on the MOT17 dataset. The results reveal an interesting trade-off between tracking accuracy and computational efficiency. DETR [8] demonstrates superior performance across all tracking metrics, achieving a 1.5% improvement in HOTA and a 1.5% increase in MOTA compared to YOLOv8n, and a substantial 13.2% improvement in HOTA and 13.8% increase in MOTA compared to Fast R-CNN [11]. It also shows better performance in terms of identity switches (IDS). However, YOLOv8n offers significant advantages in terms of inference speed and model size. The YOLOv8n-based model operates at approximately 3.75 times the speed of the DETR-based model (45 FPS vs. 12 FPS) and has a substantially smaller model size (3.2M vs. 41M parameters). Fast R-CNN [11], despite its historical significance, underperforms both DETR and YOLOv8n across all metrics. Furthermore, when comparing these results to Table 3, it is evident that our method with YOLOv8n still outperforms the second-best methods in terms of HOTA (63.8 vs 63.5), ASSA (64.5 vs 63.7), MOTA (80.5 vs 80.3) and IDS (1425 vs 1446), establishing it as a more accurate approach despite the computational trade-offs.

Table 6. Ablation study on the effect of the attention layer in DragonTrack.

Model Variant	HOTA↑	AssA↑	DetA↑	IDF1↑	MOTA↑	IDS↓
Without Attention	62.8	63.5	63.1	76.4	79.1	1542
With Attention	**65.3**	**66.2**	**65.8**	**79.2**	**82.0**	**1313**

Finally, we examine the impact of the attention layer in our approach. Table 6 compares the performance of DragonTrack with and without the attention mechanism. The results demonstrate that incorporating the attention layer leads to substantial improvements across all metrics. Specifically, we observe a 2.5% increase in HOTA, a 2.7% improvement in AssA, and a 2.9% boost in MOTA when using the attention layer. Additionally, the number of identity switches (IDS) is reduced by 229, highlighting the attention mechanism's role in improving identity consistency throughout the tracking process. These improvements underscore the importance of the attention layer not only in focusing on the most relevant features and relationships but also in effectively combining geometric, positional, and appearance features. By dynamically weighting these features, the attention mechanism enhances DragonTrack's ability to maintain accurate and robust multi-object tracking performance.

5. Discussion, Conclusion, and Future Work

This paper presented DragonTrack, a transformer-enhanced graphical method for multi-person tracking in complex scenarios. By utilizing multiple affinity networks for edge attribute processing, DragonTrack significantly enhances tracking accuracy, particularly in challenging scenarios with high occlusion or similar-appearing subjects. The integration of a learnable Sinkhorn algorithm introduces a differentiable technique for tackling the assignment problem, improving tracking continuity and precision.

The ablation studies collectively demonstrate the effectiveness of DragonTrack's design choices, including the combination of loss functions, the incorporation of the learnable Sinkhorn algorithm, the use of DETR as the detection backbone, and the integration of the attention layer. These findings suggest that while DragonTrack achieves state-of-the-art performance, future work can focus on improving computational efficiency and further enhancing accuracy, especially in models using YOLOv8n as a detection backbone. Balancing accuracy and efficiency in multi-person tracking systems is a promising direction for future research.

While DragonTrack represents a considerable advancement in tracking, several areas for future work emerge. The sophisticated architecture, including the detection transformer and graph convolutional networks, is computationally intensive, presenting opportunities for optimization in real-time applications. Current evaluations focus on

MOT17, MOT20, and DanceTrack datasets, and extending validation to more diverse datasets will confirm DragonTrack's broad applicability. Additionally, enhancing the model's capability to accurately track and match features of distant subjects will expand its utility across a wider range of scenarios.

These considerations underscore DragonTrack's innovative contributions to MOT and outline clear paths for future research. By addressing computational efficiency, broader dataset validation, and long-distance tracking capabilities, DragonTrack can further solidify its position as a leading solution in multi-person tracking, adapting to an even broader range of real-world scenarios and applications.

References

[1] Imran Ahmed, Sadia Din, Gwanggil Jeon, Francesco Piccialli, and Giancarlo Fortino. Towards collaborative robotics in top view surveillance: A framework for multiple object tracking by detection using deep learning. *IEEE/CAA Journal of Automatica Sinica*, 8(7):1253–1270, 2021. 1

[2] Somaieh Amraee, Bishoy Galoaa, Matthew Goodwin, Elaheh Hatamimajoumerd, and Sarah Ostadabbas. Multiple toddler tracking in indoor videos. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 11–20, 2024. 1

[3] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 941–951, 2019. 2, 6, 7, 11

[4] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008. 6

[5] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE, 2016. 2

[6] Guillem Brasó and Laura Leal-Taixé. Learning a neural solver for multiple object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6247–6257, 2020. 1, 3, 6, 7, 11

[7] Pingping Cao, Zeqi Zhu, Ziyuan Wang, Yanping Zhu, and Qiang Niu. Applications of graph convolutional networks in computer vision. *Neural Computing and Applications*, 34(16):13387–13405, 2022. 1

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 1, 3, 8

[9] Patrick Dendorfer, Hamid Rezatofighi, Anton Milan, Javen Shi, Daniel Cremers, Ian Reid, Stefan Roth, Konrad Schindler, and Laura Leal-Taixé. Mot20: A benchmark for multi object tracking in crowded scenes. *arXiv preprint arXiv:2003.09003*, 2020. 6, 7

[10] Yunhao Du, Zhicheng Zhao, Yang Song, Yanyun Zhao, Fei Su, Tao Gong, and Hongying Meng. Strongsort: Make deepsort great again. *IEEE Transactions on Multimedia*, 2023. 1, 2, 6, 7, 11

[11] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 8

[12] Diego Gragnaniello, Antonio Greco, Alessia Saggese, Mario Vento, and Antonio Vicinanza. Benchmarking 2d multi-object detection and tracking algorithms in autonomous vehicle driving scenarios. *Sensors*, 23(8):4024, 2023. 1

[13] Xiaofei Huang, Lingfei Luan, Elaheh Hatamimajoumerd, Michael Wan, Pooria Daneshvar Kakhaki, Rita Obeid, and Sarah Ostadabbas. Posture-based infant action recognition in the wild with very limited data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 4912–4921, June 2023. 11

[14] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics YOLOv8. https://github.com/ultralytics/ultralytics, 2023. Accessed: 2024-09-09. 8

[15] Marco Leo, Giuseppe Massimo Bernava, Pierluigi Carcagnì, and Cosimo Distante. Video-based automatic baby motion analysis for early neurological disorder diagnosis: State of the art and future directions. *Sensors*, 22(3), 2022. 11

[16] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *International journal of computer vision*, 129:548–578, 2021. 6

[17] T. Meinhardt, A. Kirillov, L. Leal-Taixe, and C. Feichtenhofer. Trackformer: Multi-object tracking with transformers. *arXiv preprint arXiv:2101.02702*, 2021. 6, 7

[18] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016. 6, 7

[19] J. Pang, L. Qiu, X. Li, H. Chen, Q. Li, T. Darrell, and F. Yu. Quasi-dense similarity learning for multiple object tracking. In *CVPR*, 2021. 6, 7, 11

[20] Ricardo Pereira, Guilherme Carvalho, Luís Garrote, and Urbano J Nunes. Sort and deep-sort based multi-object tracking for mobile robotics: Evaluation with new data association metrics. *Applied Sciences*, 12(3):1319, 2022. 1

[21] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European conference on computer vision*, pages 17–35. Springer, 2016. 6

[22] D. Stadler and J. Beyerer. Modelling ambiguous assignments for multi-person tracking in crowds. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 133–142, 2022. 6, 7

[23] Peize Sun, Jinkun Cao, Yi Jiang, Zehuan Yuan, Song Bai, Kris Kitani, and Ping Luo. Dancetrack: Multi-object tracking in uniform appearance and diverse motion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20993–21002, 2022. 6, 7

[24] P. Sun, Y. Jiang, R. Zhang, E. Xie, J. Cao, X. Hu, T. Kong, Z. Yuan, C. Wang, and P. Luo. Transtrack: Multiple-object

tracking with transformer. *arXiv preprint arXiv:2012.15460*, 2020. 6, 7

[25] Thangaswamy Judi Vennila and Vanniappan Balamurugan. A rough set framework for multi-human tracking in surveillance video. *IEEE Sensors Journal*, 2023. 1

[26] Q. Wang, Y. Zheng, P. Pan, and Y. Xu. Multiple object tracking with correlation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3876–3886, 2021. 6, 7

[27] S. Wang, H. Sheng, Y. Zhang, Y. Wu, and Z. Xiong. A general recurrent tracking framework without real data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13219–13228, 2021. 6, 7

[28] Y. Wang, K. Kitani, and X. Weng. Joint object detection and multi-object tracking with graph neural networks. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13708–13715. IEEE, 2021. 6, 7

[29] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017. 1, 2

[30] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020. 1

[31] Fan Yang, Shigeyuki Odashima, Shoichi Masui, and Shan Jiang. Hard to track objects with irregular motions and similar appearances? make it easier by buffering the matching space. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4799–4808, 2023. 1

[32] Mingzhan Yang, Guangxin Han, Bin Yan, Wenhua Zhang, Jinqing Qi, Huchuan Lu, and Dong Wang. Hybrid-sort: Weak cues matter for online multi-object tracking. *arXiv preprint arXiv:2308.00783*, 2023. 1

[33] Jan Niklas Zaech, Alexander Liniger, Dengxin Dai, Martin Danelljan, and Luc Van Gool. Learnable online graph representations for 3d multi-object tracking. *IEEE Robotics and Automation Letters*, 7(2):5103–5110, 2022. 1, 3

[34] Ilham Ari Elbaith Zaeni, Siti Sendari, Dyah Lestari, Yogi Dwi Mahandi, Mahfud Jiono, and M Syaifuddin. An implementation of multi-object tracking using omnidirectional camera for trash picking robot. In *2018 Electrical Power, Electronics, Communications, Controls and Informatics Seminar (EECCIS)*, pages 154–158. IEEE, 2018. 1

[35] Fangao Zeng, Bin Dong, Yuang Zhang, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. Motr: End-to-end multiple-object tracking with transformer. In *European Conference on Computer Vision*, pages 659–675. Springer, 2022. 3, 6, 7, 11

[36] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box. In *European Conference on Computer Vision*, pages 1–21. Springer, 2022. 1, 2, 6, 7, 11, 12

[37] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *IJCV*, pages 1–19, 2021. 6, 7, 11

[38] Yuang Zhang, Tiancai Wang, and Xiangyu Zhang. Motrv2: Bootstrapping end-to-end multi-object tracking by pre-trained object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22056–22065, 2023. 3, 6, 7

[39] X. Zhou, V. Koltun, and P. Krähenbühl. Tracking objects as points. In *ECCV*, 2020. 6, 7, 11