MART: MultiscAle Relational Transformer Networks for Multi-agent Trajectory Prediction

Seongju Lee[1], Junseok Lee[1], Yeonguk Yu[1], Taeri Kim[1], and Kyoobin Lee[1]*

School of Integrated Technology (SIT), Gwangju Institute of Science and Technology (GIST), Cheomdan-gwagiro 123, Buk-gu, Gwangju 61005, Republic of Korea
{lsj2121, junseoklee, yeon_guk, smterry0928, kyoobinlee}@gist.ac.kr

Abstract. Multi-agent trajectory prediction is crucial to autonomous driving and understanding the surrounding environment. Learning-based approaches for multi-agent trajectory prediction, such as primarily relying on graph neural networks, graph transformers, and hypergraph neural networks, have demonstrated outstanding performance on real-world datasets in recent years. However, the hypergraph transformer-based method for trajectory prediction is yet to be explored. Therefore, we present a **M**ultisc**A**le **R**elational **T**ransformer (**MART**) network for multi-agent trajectory prediction. MART is a hypergraph transformer architecture to consider individual and group behaviors in transformer machinery. The core module of MART is the encoder, which comprises a Pair-wise Relational Transformer (PRT) and a Hyper Relational Transformer (HRT). The encoder extends the capabilities of a relational transformer by introducing HRT, which integrates hyperedge features into the transformer mechanism, promoting attention weights to focus on group-wise relations. In addition, we propose an Adaptive Group Estimator (AGE) designed to infer complex group relations in real-world environments. Extensive experiments on three real-world datasets (NBA, SDD, and ETH-UCY) demonstrate that our method achieves state-of-the-art performance, enhancing ADE/FDE by 3.9%/11.8% on the NBA dataset. Code is available at https://github.com/gist-ailab/MART.

Keywords: Multi-agent Trajectory Prediction · Hypergraph Transformer

1 Introduction

Multi-agent trajectory prediction, which forecasts the future trajectories of agents based on their past movements, holds significance in understanding human behavior and has applications in several fields, including robotics [23, 62] and autonomous driving [10, 31]. This task is complex and challenging for several reasons. For instance, some pedestrians move in formation, often as part of a group, introducing interdependencies and shared decision-making. These group
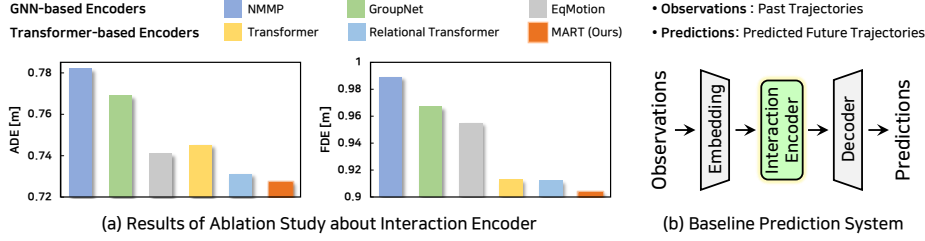
---

* Corresponding author.

Fig. 1: **(a) Results of ablation study about different interaction encoders on the NBA dataset. (b) Baseline prediction system used in ablation study.** We conduct an ablation study on interaction encoders to compare the MART encoder with state-of-the-art encoders. The average displacement error (ADE) and final displacement error (FDE) are assessed using the NBA dataset.

dynamics influence individual paths, making predicting the trajectory of a single agent without considering the broader context challenging. Consequently, researchers have explored sophisticated methods to capture individual interactions and group behavior [6, 43, 54, 57, 61].

Previous studies have focused on modeling pair-wise interactions in trajectory prediction using graph neural networks (GNNs) [4, 7, 20, 21, 26, 32, 38, 47, 49, 56] and transformer networks [17, 35, 50, 59, 60]. Some studies have employed graph convolutional networks [25] to account for pair-wise relations in crowded scenes [4, 38, 47, 49]. Other methods, such as those based on graph attention networks [51], have been proposed in [7, 21, 26, 32]. In addition, message-passing neural networks [15] have been employed in trajectory prediction tasks [20,54,57]. Recently, Xu *et al.* [56] introduced a motion prediction model under Euclidean geometric transformations based on equivariant GNNs [46]. In addition, Yu *et al.* [59] proposed a graph transformer network that considers pair-wise and temporal relations by incorporating an adjacency matrix into the attention mechanism.

To cooperate with individual and group behaviors, GroupNet [54], a multi-scale hypergraph message-passing neural network, was proposed for multi-agent trajectory prediction. As an extension of GroupNet [54], DynGroupNet [57] captures time-varying interactions at the pair and group scales. However, transformer machinery, an effective relationship-learning mechanism, is not fully incorporated into their methods. Bae *et al.* [6] proposed the GP-Graph, which considers intra- and inter-group relationships by partitioning the most likely behavioral agents in the same group. However, round-trip node-edge communication, crucial for relational reasoning [13], is ignored in GP-Graph.

In this study, we propose a **M**ultisc**A**le **R**elational **T**ransformer (**MART**) network, a hypergraph transformer architecture for multi-agent trajectory prediction. To address social relationships in hypergraph transformer architecture, we introduce the MART encoder (MARTE), comprising a pair-wise relational transformer and a Hyper Relational Transformer (HRT) to enhance the capabilities of the relational transformer [13]. We introduce Hyper Relational Attention (HRA) into HRT, incorporating hyperedge features into the attention mecha-

nism. As hyperedge features involve group information, HRT captures group behavior while promoting attention weights to focus on group-wise relations.

Since group relations are not explicitly defined in real-world scenarios, pioneering studies have introduced methods for reasoning group relations [6, 54]. However, they have limitations. In [6], the focus is on non-overlapping group relations that cannot handle complex relationships. For [54], the number of agents in a group requires a manual definition, and as the number of group scales increases, more encoders are required, resulting in higher computational costs. To address the aforementioned limitations, we propose an Adaptive Group Estimator (AGE). Inspired by the Straight-Through Estimation (STE) technique [8] used in binarized neural networks [22, 33, 53, 58], we utilize adaptive thresholding to include highly correlated agents within the same group. Our approach estimates overlapping group relations, recognizing that agents can be correlated with more than one group. Moreover, unlike [54], the manual definition of the number of agents in a group is unnecessary in our method.

To validate the effectiveness of MART, we conduct extensive experiments on three real-world datasets (NBA, SDD [42], and ETH-UCY [29, 41]) and demonstrate that our method achieves state-of-the-art (SOTA) performance. Notably, on the NBA dataset, which is the most challenging and complex, MART shows a subjective performance improvement by reducing the ADE by 3.9% and FDE by 11.8% compared to the previous SOTA method, EqMotion [56]. The ablation study on the interaction encoder demonstrates that hypergraph transformer architecture is beneficial for multi-agent trajectory prediction. The proposed MARTE, based on a hypergraph transformer, outperforms other SOTA GNN-based and transformer-based encoders, as illustrated in Figure 1. Additionally, MART exhibits computational efficiency compared to recent SOTA models. The main contributions are as follows:

- We introduce MART, a Multiscale RelAtional Transformer network for multi-agent trajectory prediction. MART captures relations at individual and group scales through a hypergraph transformer mechanism via MARTE. The proposed MARTE promotes the attention weights to emphasize interactions between agents, particularly focusing on group-wise relations.
- We propose the AGE module to infer group relations. The AGE module outperforms existing group reasoning methods in both performance and efficiency by considering overlapping group relations and dealing with complex interactions through adaptive thresholding.
- We experimentally demonstrate that MART achieves SOTA performance on three real-world datasets (NBA, SDD, and ETH-UCY). Moreover, MART is computationally efficient compared to the recent SOTA models.

2   Related Work

Trajectory Prediction. Traditional trajectory prediction methods are based on heuristics, energy potentials, and simple machine learning-based methods [9,

19,36,39]. Berndt *et al.* [9] proposed using the global vehicle position with a hidden Markov model to predict object trajectory in a road traffic environment. Subsequently, recurrent neural network (RNN)-based methods [2, 52, 63] have been proposed to capture the long-term behavior of agents. For instance, Vemula *et al.* [52] used an RNN-based prediction model with attention that captured the relative importance of each person to predict agents' trajectories in the crowd. Generative adversarial network (GAN)-based methods [12, 14, 18, 20, 44, 48] and variational autoencoder-based methods [27, 34, 45, 54, 57] have also been studied to generate future trajectory distribution. Similarly, [17, 35] proposed trajectory-prediction methods based on a diffusion model. In addition, transformer structures [16, 59, 60] have been used to capture the importance of the agents' spatial and temporal correlations. Our method is a hypergraph transformer network that captures individual and group behaviors for trajectory prediction.

Transformers with Graph. Recently, transformer-based structures have been proposed for representing graphs [11, 13, 37, 59]. Specifically, Chen *et al.* [11] addressed concerns regarding transformer architecture, in which the node representation generated by the transformer fails to capture the structural similarity between them, and solved it by incorporating structural information into the self-attention module. In addition, relational transformer [13], which uses an attention module to capture edge information effectively, has been demonstrated to perform diverse graph-structure tasks. For trajectory prediction, Li *et al.* [30] proposed a graph-based spatial transformer to capture the spatial interactions between pedestrians. Herein, we introduce MART to effectively capture individual and group behaviors via transformer mechanisms.

3   Background and Problem Formulation

3.1   Trajectory Prediction

Multi-agent trajectory prediction predicts the future trajectories of multiple agents based on their past trajectories. Assume that N agents exist in a multi-agent system. Specifically, let $\mathbf{X} \in \mathbb{R}^{N \times T_p \times 2}$, $\mathbf{Y} \in \mathbb{R}^{N \times T_f \times 2}$, and $\hat{\mathbf{Y}} \in \mathbb{R}^{N \times T_f \times 2}$ represent past trajectories, ground-truth future trajectories, and predicted future trajectories in 2D spatial coordinates, respectively. The objective is to minimize the error between the predicted and ground-truth future trajectories. We design a trajectory prediction model denoted by $\hat{\mathbf{Y}} = \mathcal{M}(\mathbf{X})$. Here, $\mathcal{M}$ produces predicted future trajectories that closely align with the ground-truth future trajectories.

3.2   Relational Transformer

The relational transformer (RT) [13], an extension of the transformer [50], incorporates edge vectors as fundamental components to achieve SOTA performance in various tasks involving graph-structured data. Mathematically, for a given input-directed graph, we denote an attributed graph as $\mathcal{G} = (\mathcal{N}, \mathcal{E})$. Here, $\mathcal{N}$ represents an unordered set of node vectors $\mathbf{n}_i \in \mathbb{R}^{d_n}$, where $\mathbf{n}_i$ is the i-th

node vector and d_n is the node dimension. $\mathcal{E}$ represents the set of edge vectors $\mathbf{e}_{ij} \in \mathbb{R}^{d_e}$, where $\mathbf{e}_{ij}$ represents the directed edge pointing from j-th to i-th node and d_e is the edge dimension.

Similar to message-passing neural networks [15], RT comprises two processes: node and edge updates. In the node update, RT introduces relational attention (RA), which incorporates edge vectors into the scaled dot-product attention. Formally, the query, key, and value vectors in RA are defined as follows:

$$\begin{aligned}
\mathbf{q}_{ij} &= (\mathbf{n}_i \mathbf{W}_n^Q + \mathbf{e}_{ij} \mathbf{W}_e^Q) \in \mathbb{R}^{d_n}, \\
\mathbf{k}_{ij} &= (\mathbf{n}_j \mathbf{W}_n^K + \mathbf{e}_{ij} \mathbf{W}_e^K) \in \mathbb{R}^{d_n}, \\
\mathbf{v}_{ij} &= (\mathbf{n}_j \mathbf{W}_n^V + \mathbf{e}_{ij} \mathbf{W}_e^V) \in \mathbb{R}^{d_n},
\end{aligned} \tag{1}$$

where $\mathbf{W}_n \in \mathbb{R}^{d_n \times d_n}$ and $\mathbf{W}_e \in \mathbb{R}^{d_e \times d_n}$ are the weight matrices used to project the node and edge vectors onto the query, key, and value vectors, respectively. The remainder of the node update is the same as the transformer mechanism. Formally, the updated node vector after $(l+1)$-th layer $\mathbf{n}_i^{(l+1)} \in \mathbb{R}^{d_n}$ is defined as follows:

$$\mathbf{n}_i^{(l+1)} = \mathcal{F}_n \left(\mathrm{softmax}_j \left(\frac{\mathbf{q}_{ij}^{(l)} \mathbf{k}_{ij}^{(l)\top}}{\sqrt{d_n}} \right) \mathbf{v}_{ij}^{(l)}, \mathbf{n}_i^{(l)} \right) \tag{2}$$

where $\mathcal{F}_n(\cdot)$ represents a transformer-like node update function of the form [Add & Norm]–[FeedForward]–[Add & Norm]. In this study, we omitted the equation for multi-head attention for simplicity.

For the edge update, the edge vector after $(l+1)$-th layer $\mathbf{e}_{ij}^{(l+1)} \in \mathbb{R}^{d_e}$ is produced by the edge update function $\mathcal{F}_e(\cdot)$ as follows:

$$\mathbf{e}_{ij}^{(l+1)} = \mathcal{F}_e \left(\mathbf{m}_{ij}^{(l)}, \mathbf{e}_{ij}^{(l)} \right). \tag{3}$$

Here, $\mathbf{m}_{ij}^{(l)} \in \mathbb{R}^{d_e}$ represents the message vector that aggregates the information from adjacent updated nodes and their edges. Function $\mathcal{F}_e(\cdot)$ denotes the transformer-like edge-update function, which has an identical operation as $\mathcal{F}_n(\cdot)$. The message vector $\mathbf{m}_{ij}^{(l)}$ is derived as follows:

$$\mathbf{m}_{ij}^{(l)} = \mathrm{ReLU}([\mathbf{e}_{ij}^{(l)}; \mathbf{e}_{ji}^{(l)}; \mathbf{n}_i^{(l+1)}; \mathbf{n}_j^{(l+1)}] \mathbf{W}_m), \tag{4}$$

where $[\cdot; \cdot]$ denotes the concatenation operation and $\mathbf{W}_m \in \mathbb{R}^{(2d_e + 2d_n) \times d_h}$ is a weight matrix for the message function.

4 Method

As illustrated in Figure 2, MART comprises four components: feature initialization, AGE, MARTE, and decoder. The subsequent sections offer a detailed description of the core components of MART—feature initialization, AGE module, and MARTE—as well as the overall process.
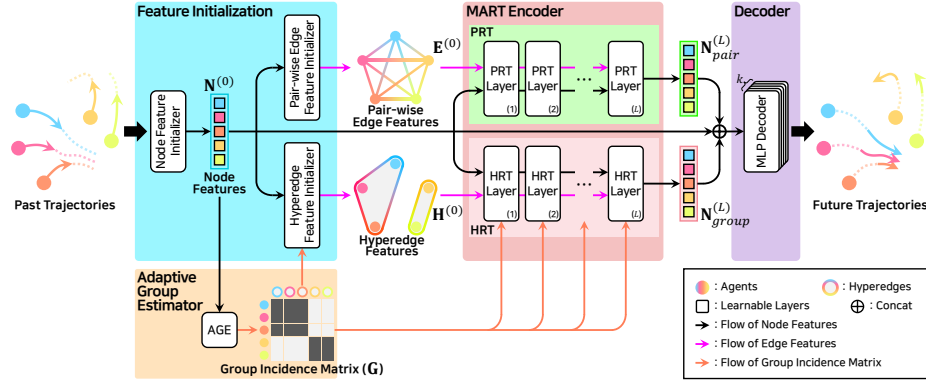
Fig. 2: Model architecture of the proposed MultiscAle Relational Transformer (MART) network. Our model has four components: feature initialization (sky blue), Adaptive Group Estimator (orange), MART encoder (red), and future trajectory decoder (Decoder) (purple). The MART encoder includes a pair-wise relational transformer (green) and a hyper relational transformer (pink).

4.1   Feature Initialization

We first transform the past input trajectories into agent embeddings to represent the initial node and edge features. Because edge relationships are often not explicitly defined in real-world environments, we represent edge features by aggregating the features of the adjacent nodes. In this paper, we consider pair-wise edge and hyperedge features.

Node Feature Initialization.   The past trajectory of i-th agent $\mathbf{X}_i \in \mathbb{R}^{T_p \times d_i}$ is embedded via $\mathbf{n}_i^{(0)} = \mathcal{F}_{\mathrm{NI}}(\mathbf{X}_i)$, where $\mathbf{n}_i^{(0)} \in \mathbb{R}^{d_n}$ is the initial node feature of the i-th agent and $\mathcal{F}_{\mathrm{NI}}(\cdot)$ represents two layers of multilayer perceptron (MLP). d_n denotes the node dimensions.

Edge Feature Initialization.   To consider pair- and group-wise relations, we initialize pair-wise edge and hyperedge features, respectively. To form pair-wise edge features, two adjacent nodes are concatenated and passed through the pair-wise edge feature initializer $\mathcal{F}_{\mathrm{PI}}(\cdot)$: $\mathbf{e}_{ij}^{(0)} = \mathcal{F}_{\mathrm{PI}}([\mathbf{n}_i^{(0)}; \mathbf{n}_j^{(0)}])$, where $\mathbf{e}_{ij}^{(0)}$ is the initial pair-wise edge feature from j-th to i-th agent and $\mathcal{F}_{\mathrm{PI}}(\cdot)$ is MLP.

The initial hyperedge feature is formed by averaging the initial node features of the corresponding hyperedge. The aggregated node features are fed to the hyperedge initializer $\mathcal{F}_{\mathrm{HI}}(\cdot)$: $\mathbf{h}_i^{(0)} = \mathcal{F}_{\mathrm{HI}}\left(\frac{1}{|\mathcal{N}_i|}\sum_{n_j \in \mathcal{N}_i} \mathbf{n}_j^{(0)}\right)$, where $\mathcal{F}_{\mathrm{HI}}$ is implemented by MLP, $\mathcal{N}_i$ is a set of node features of the i-th hyperedge, and $|\mathcal{N}_i|$ is its cardinality. In the following section, we propose a group reasoning method to address the lack of predefined hypergraph topology in real-world environments.

4.2   Adaptive Group Estimator

Real-world environments often conceal group relationships, which are crucial for understanding group behavior. Thus, several works have proposed methods for reasoning about group relations [6, 54]; however, these methods have limitations, as mentioned in the Introduction. To overcome the aforementioned limitations, we propose the Adaptive Group Estimator (AGE) to effectively infer complex group relationships through adaptive thresholding, thereby grouping closely related agents.

Mathematically, given a hypergraph $\mathcal{G} = (\mathcal{N}, \mathcal{H})$, a group relation can be expressed as a group incident matrix $\mathbf{G} \in \mathbb{R}^{|\mathcal{N}| \times |\mathcal{H}|}$, where $\mathbf{G}_{ij} \in \{0, 1\}$. Here, $\mathcal{H}$ is an unordered set of hyperedge features, and $|\mathcal{H}|$ is its cardinality. When $\mathbf{G}_{ij} = 1$, the i-th agent belongs to the j-th hyperedge; however, when $\mathbf{G}_{ij} = 0$, it does not. In this study, we assume that the number of hyperedges is set to $|\mathcal{N}|$ to include agents highly correlated with the j-th agent (ego agent) in the same group. Thus, the (i, j)-th element of the group incidence matrix $\mathbf{G}$ is defined as follows:

$$\mathbf{G}_{ij} = \mathcal{U}(\mathbf{A}_{ij} - \Theta), \tag{5}$$

where $\mathbf{A}_{ij} = \mathbf{n}_i^\top \mathbf{n}_j / (||\mathbf{n}_i||_2 ||\mathbf{n}_j||_2) \in \mathbb{R}^{N \times N}$ is the (i, j)-th element of the affinity matrix, $\Theta \in (-1.0, 1.0)$ is a learnable threshold, and $\mathcal{U}(\cdot)$ is a unit step function defined as follows:

$$\mathcal{U}(x) = \begin{cases} 1, & \text{if } x \geq 0. \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

As the unit step function is not differentiable, we use the STE trick [8] inspired by binarized neural networks [22, 33, 53, 58]. We compute the group incidence matrix $\mathbf{G}$ using Eq. 5 for the forward path. In the backward path, we estimate the gradient of $\mathcal{U}(\cdot)$ to enable backpropagation. The estimated gradient of $\mathcal{U}(\cdot)$ is defined as Eq. 7. Note that this estimation is derived based on Eq. 8, which approximates $\mathcal{U}(\cdot)$. By introducing Θ into the AGE module, it selectively groups more correlated agents into the same group. The supplementary material describes the details of the function selection.

$$\frac{\partial \mathcal{U}(x)}{\partial x} \approx \begin{cases} 2 - 4|x|, & \text{if } |x| \leq 0.5, \\ 0, & \text{otherwise.} \end{cases} \tag{7} \quad \mathcal{U}(x) \approx \begin{cases} 0, & \text{if } x < -0.5, \\ 0.5 + 2x + 2x^2, & \text{if } -0.5 \leq x < 0, \\ 0.5 + 2x - 2x^2, & \text{if } 0 \leq x < 0.5, \\ 1, & \text{otherwise.} \end{cases} \tag{8}$$

4.3   Multiscale Relational Transformer Encoder

We introduce the MultiscAle Relational Transformer Encoder (MARTE) into our network to capture social interactions at multiple scales via hypergraph transformer architecture. The detailed architecture of MARTE is shown in Figure 4. PRT and HRT address individual and group behaviors, respectively.
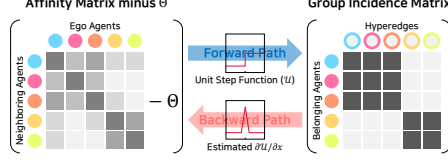
Fig. 3: Illustration of adaptive group estimator (AGE) module. The AGE module employs an adaptive thresholding inspired by STE trick [8]. The blue and red arrows represent the forward and estimated backward paths, respectively.
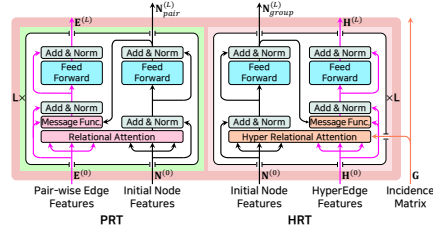
Fig. 4: Architecture of MARTE. The regions in green, pink, and red correspond to those in Figure 2. The black/purple/coral arrows denote the flow of node features/edge features/group incidence matrix.

Pair-wise Relational Transformer. PRT is based on RT [13], an effective relational learner, as described in Section 3.2. However, RT deals only with pair-wise relationships. Therefore, in the following section, we introduce HRT as an extension of RT to accommodate hypergraph structures, ensuring synergy between architectures.

Hyper Relational Transformer. We propose HRT to capture group behavior via a hypergraph transformer mechanism. To achieve this, we introduce Hyper Relational Attention (HRA) into HRT, extending the relational attention mechanism to accommodate hypergraph structures. Additionally, we introduce a message function to update the hyperedges in accordance with these structures.

(*Node update for HRT*) To extend the mechanism of relational attention to a hypergraph structure, we integrate hyperedge information into the query, key, and value by aggregating the hyperedge features to which the corresponding agent belongs. The average operation is used for aggregation for stable training. *The aggregated hyperedge features in query, key, and value work as the relative positional information that includes group information,* formulated as follows:

$$
\begin{aligned}
\mathbf{q}_i &= \left(\mathbf{n}_i \mathbf{W}_n^Q + \frac{1}{|\mathcal{H}_i|} \sum_{h_k \in \mathcal{H}_i} \mathbf{h}_k \mathbf{W}_h^Q \right) \in \mathbb{R}^{d_n}, \\
\mathbf{k}_j &= \left(\mathbf{n}_j \mathbf{W}_n^K + \frac{1}{|\mathcal{H}_j|} \sum_{h_k \in \mathcal{H}_j} \mathbf{h}_k \mathbf{W}_h^K \right) \in \mathbb{R}^{d_n}, \\
\mathbf{v}_j &= \left(\mathbf{n}_j \mathbf{W}_n^V + \frac{1}{|\mathcal{H}_j|} \sum_{h_k \in \mathcal{H}_j} \mathbf{h}_k \mathbf{W}_h^V \right) \in \mathbb{R}^{d_n},
\end{aligned}
\tag{9}
$$

where $\mathbf{W}_h \in \mathbb{R}^{d_e \times d_n}$ is the weight matrix used to project the hyperedge features onto the query, key, and value vectors, $\mathcal{H}_i$ is a set of hyperedges to which the i-th agent belongs, and $|\mathcal{H}_i|$ is its cardinality. These relations of belonging

are derived from $\mathbf{G}$. Consequently, the updated node feature $\mathbf{n}_i^{(l+1)}$ is formulated as follows:

$$\mathbf{n}_i^{(l+1)} = \mathcal{F}_n \left(\text{softmax}_j \left(\frac{\mathbf{q}_i^{(l)} \mathbf{k}_j^{(l)\top}}{\sqrt{d_n}} \right) \mathbf{v}_j^{(l)}, \mathbf{n}_i^{(l)} \right). \tag{10}$$

(*Hyperedge update for HRT*) The i-th hyperedge feature after the $(l+1)$-th HRT layer, denoted as $\mathbf{h}_i^{(l+1)} \in \mathbb{R}^{d_e}$, is defined as follows:

$$\mathbf{h}_i^{(l+1)} = \mathcal{F}_h(\mathbf{m}_i^{(l)}, \mathbf{h}_i^{(l)}) \tag{11}$$

where $\mathcal{F}_h(\cdot)$ denotes the hyperedge update function based on feedforward network, and $\mathbf{m}_i^{(l)}$ is a message vector. For the message vector $\mathbf{m}_i^{(l)}$, we formulate a message function to accommodate the hyperedge structure. More specifically, this function is formulated using the hyperedge feature and the mean of the updated node features belonging to the corresponding hyperedge. Mathematically, the message vector $\mathbf{m}_i^{(l)} \in \mathbb{R}^{d_e}$ is defined as follows:

$$\mathbf{m}_i^{(l)} = \text{ReLU} \left(\left[\mathbf{h}_i^{(l)}; \frac{1}{|\mathcal{N}_i|} \sum_{n_j \in \mathcal{N}_i} \mathbf{n}_j^{(l+1)} \right] \mathbf{W}_m \right), \tag{12}$$

where $\mathbf{W}_m \in \mathbb{R}^{(d_e + d_n) \times d_h}$ denotes the weight matrix to project post node-update features onto the message vector.

4.4   Overall Process of MART

We detail the trajectory prediction process of MART in the following sequence: feature initialization, encoding, decoding, and loss functions. The first three processes represent the forward path of MART *in matrix formula*, and the loss function serves as the objective function for model training.

Feature Initialization. As described in Section 4.1, our model first initializes the node, pair-wise edge, and hyperedge features for N agents as:

$$\begin{aligned}
\mathbf{N}^{(0)} &= \{\mathbf{n}_1^{(0)}, \mathbf{n}_2^{(0)}, ..., \mathbf{n}_N^{(0)}\} \in \mathbb{R}^{N \times d_n}, \\
\mathbf{E}^{(0)} &= \{\mathbf{e}_{11}^{(0)}, \mathbf{e}_{12}^{(0)}, ..., \mathbf{e}_{NN}^{(0)}\} \in \mathbb{R}^{N^2 \times d_e}, \\
\mathbf{H}^{(0)} &= \{\mathbf{h}_1^{(0)}, \mathbf{h}_2^{(0)}, ..., \mathbf{h}_N^{(0)}\} \in \mathbb{R}^{N \times d_e}.
\end{aligned} \tag{13}$$

The pair-wise edge features $\mathbf{E}$ have $N^2 \times d_e$ dimensions because we consider a fully connected directed graph with self-loops.

Encoding Process. MARTE encodes pair- and group-wise node features from the initial features. Formally, the features produced by MARTE are described as follows:

$$\begin{aligned}
\mathbf{N}_{pair}^{(L)} &= \mathcal{PRT}(\mathbf{N}^{(0)}, \mathbf{E}^{(0)}), \\
\mathbf{N}_{group}^{(L)} &= \mathcal{HRT}(\mathbf{N}^{(0)}, \mathbf{H}^{(0)}, \mathbf{G}),
\end{aligned} \tag{14}$$

where $\mathbf{N}_{pair}^{(L)} \in \mathbb{R}^{N \times d_n}$ represents the node features at the pair-wise scale from the L-th PRT layer and $\mathbf{N}_{group}^{(L)} \in \mathbb{R}^{N \times d_n}$ denotes the node features at the group scale from the L-th HRT layer. We use $\mathcal{PRT}(\cdot)$ to refer to the PRT operation and $\mathcal{HRT}(\cdot)$ for the HRT operation.

Decoding Process. To predict future trajectories, we utilize multiple prediction heads as the decoder to obtain k predictions, as in [56]. Three types of features are concatenated: $\mathbf{N}^{(0)}$, $\mathbf{N}_{pair}^{(L)}$, and $\mathbf{N}_{group}^{(L)}$, then fed into the multi-head decoder $\mathcal{F}_\mathrm{D}$. Formally, the trajectories obtained from the k-th prediction head can be described as $\hat{\mathbf{Y}}_k = \mathcal{F}_{\mathrm{D},k}([\mathbf{N}^{(0)}; \mathbf{N}_{pair}^{(L)}; \mathbf{N}_{group}^{(L)}])$, where $\mathcal{F}_{\mathrm{D},k}(\cdot)$ denotes a decoder with three layers of MLP.

Loss Function. We trained our model using the variety loss proposed in social-GAN [18] to optimize the best prediction. With the given k-th predicted trajectories resulting from k-th prediction head $\hat{\mathbf{Y}}_k$, the minimum of the L2-norm among k predictions is utilized for backpropagation. Mathematically, the loss function is formulated as $\mathcal{L} = \frac{1}{NT_f} \sum_{n=1}^{N} \sum_{t=1}^{T_f} \min_k \left\| Y^{(n,t)} - \hat{Y}_k^{(n,t)} \right\|_2$, where (n,t) denotes the indices for the time step t for the n-th agent.

5 Experiments

Datasets. For the evaluation of MART, we utilize three datasets for multi-agent trajectory prediction: NBA SportVU (NBA), Stanford Drone (SDD) [42], and ETH-UCY [29,41]. Further details about the datasets and evaluation scheme are described in the supplementary material.

Implementation Details. The node and edge dimensions, denoted by d_n and d_e, respectively, are set to 64. Eight heads are used for multi-head attention. The number of layers in MARTE, denoted by L, is 4. For all datasets, our network is trained using the Adam optimizer [24]. A learning rate decay of 0.5 is applied every 100 epochs. The network is trained using an NVIDIA GeForce RTX 3090. All our implementations are on PyTorch 2.1.0 [40]. Further details about the implementation can be found in the supplementary material.

Evaluation Metrics. To assess the trajectory prediction performance, we employed two metrics, $\min \mathrm{ADE}_k$ and $\min \mathrm{FDE}_k$, as in [1, 18, 28]. ADE and FDE represent the Euclidean distances between the predicted and actual trajectories, considering the entire path and endpoints, respectively. Consequently, $\min \mathrm{ADE}_k$ and $\min \mathrm{FDE}_k$ are the smallest values among k computed ADEs and FDEs from the k predicted trajectories.

6 Results and Discussion

6.1 Comparison with State-Of-The-Art (SOTA) Methods

The performance of the proposed model is compared with the SOTA methods on the NBA, SDD, and ETH-UCY datasets using two metrics: $\min \mathrm{ADE}_{20}$ and $\min \mathrm{FDE}_{20}$, which are described in Section 5.

Table 1: Performance comparison on (a) NBA, (b) SDD, and (c) ETH-UCY datasets. The values denote $\min \text{ADE}_{20}/\min \text{FDE}_{20}$ in meters. **Bold**/<u>underline</u> indicates the best/second-best result. * denotes that values are reproduced using the official implementation

(a) NBA Dataset

Time	STAR [59]	GroupNet [54]	MemoNet [55]	MID [17]	NPSN [7]	DynGroupNet [57]	LED [35]	EqMotion* [56]	Ours
4.0s	1.13/2.01	0.96/1.30	1.25/1.47	0.96/1.27	1.31/1.79	0.89/1.13	0.81/1.10	<u>0.76</u>/<u>1.02</u>	**0.73**/**0.90**

(b) SDD Dataset

Time	PECNet [34]	GroupNet [54]	MemoNet [55]	MID [17]	NPSN [7]	DynGroupNet [57]	ET [5]	LED [35]	Ours
4.8s	9.96/15.88	9.31/16.11	8.56/12.66	9.73/15.32	8.56/11.85	8.42/13.58	<u>8.05</u>/13.25	8.48/**11.66**	**7.43**/<u>11.82</u>

(c) ETH-UCY Dataset

Subset	GroupNet [54]	MemoNet [55]	MID [17]	NPSN [7]	GP-Graph [6]	EqMotion [56]	ET [5]	LED [35]	Ours
ETH	0.46/0.73	0.40/0.61	0.39/0.66	<u>0.36</u>/0.59	0.43/0.63	0.40/0.61	<u>0.36</u>/<u>0.53</u>	0.39/0.58	**0.35**/**0.47**
HOTEL	0.15/0.25	**0.11**/<u>0.17</u>	0.13/0.22	0.16/0.25	0.18/0.30	<u>0.12</u>/<u>0.18</u>	<u>0.12</u>/0.19	**0.11**/**0.17**	0.14/0.22
UNIV	0.26/0.49	0.24/0.43	**0.22**/0.45	<u>0.23</u>/**0.39**	0.24/<u>0.42</u>	<u>0.23</u>/0.43	0.24/0.43	0.26/0.43	0.25/0.45
ZARA1	0.21/0.39	<u>0.18</u>/0.32	**0.17**/0.30	<u>0.18</u>/0.32	**0.17**/0.31	<u>0.18</u>/0.32	0.19/0.33	<u>0.18</u>/**0.26**	**0.17**/<u>0.29</u>
ZARA2	0.17/0.33	<u>0.14</u>/0.24	**0.13**/0.27	<u>0.14</u>/0.25	0.15/0.29	**0.13**/<u>0.23</u>	<u>0.14</u>/0.24	**0.13**/**0.22**	**0.13**/**0.22**
AVG	0.25/0.44	**0.21**/0.35	**0.21**/0.38	**0.21**/0.36	<u>0.23</u>/0.39	**0.21**/0.35	**0.21**/<u>0.34</u>	**0.21**/**0.33**	**0.21**/**0.33**

NBA Dataset. Table 1-(a) presents the performance comparison to eight SOTA methods on the NBA dataset. Compared to the previous best method EqMotion [56], our model reduces $\min \text{ADE}_{20}$ and $\min \text{FDE}_{20}$ at 4.0 s by 3.9% and 11.8%, respectively.

SDD Dataset. We compared our method with eight SOTA methods on the SDD dataset, as listed in Table 1-(b). Our method enhances the $\min \text{ADE}_{20}$ by 7.8% compared to EigenTrajectory (ET) [5] and shows comparable performance in $\min \text{FDE}_{20}$ compared to LED [35].

ETH-UCY Dataset. Performance comparison with eight SOTA methods on the ETH-UCY dataset is listed in Table 1-(c). Our model yields results comparable to those of a previous SOTA method, LED [35].

Computational Efficiency of MART. The computational efficiency of MART is evaluated in comparison with recent SOTA methods in terms of model parameters and multiple-add cumulation (MAC) for inference. These metrics are computed with 10 agents in the ETH-UCY setting. As shown in Table 2, our method demonstrates computational efficiency compared to previous SOTA methods such as LED [35] and EqMotion [56]. Despite STAR [59] having only 1.0M model parameters, it has a larger MAC count than ours due to its iterative architecture.

Discussion. MART demonstrates SOTA or comparable performance in trajectory prediction compared to previous methods. Our approach is more computationally efficient, with 7.3x fewer parameters and 347.3x fewer MACs than the previous SOTA method, LED [35]. These results indicate that MART is highly competitive for accurate trajectory prediction in real-time applications.

Table 2: Comparison of the computational efficiency of the proposed and state-of-the-art methods. **Bold**/underline indicate the best/second-best. All values are obtained using the official implementation. * denotes an improved version from their GitHub. R2O refers to Ratio-to-Ours.

Method	#Param.↓	R2O	#MAC↓	R2O
PECNet [34]	2.1M	1.4x	259.2M	6.0x
STAR [59]	**1.0M**	**0.7x**	12.0G	276.9x
MemoNet [55]	10.7M	7.1x	6.0G	137.8x
GroupNet [54]	2.2M	1.5x	411.5M	9.5x
MID* [17]	9.0M	6.0x	40.3G	931.7x
EqMotion [56]	3.0M	2.0x	147.1M	3.4x
LED [35]	10.9M	7.3x	15.0G	347.3x
MART (Ours)	1.5M	1.0x	**43.3M**	**1.0x**

Table 3: Ablation studies of interaction encoders and group estimation modules on the NBA dataset. **Bold**/underline indicate the best/second-best.

Interaction Encoder	min ADE$_{20}$	min FDE$_{20}$
NMMP [20]	0.782	0.989
GroupNet [54]	0.769	0.967
EqMotion [56]	0.741	0.955
Transformer [50]	0.745	0.913
Relational Transformer [13]	0.731	0.912
MARTE (ours)	**0.727**	**0.903**

Group Estimation	min ADE$_{20}$	min FDE$_{20}$
Module in GP-Graph [6]	0.740	0.913
Module in GroupNet [54]	0.732	0.912
AGE (ours)	**0.727**	**0.903**

6.2   Ablation Study

Effect of MARTE. Ablation studies are conducted on different interaction encoders to demonstrate the effectiveness of MARTE, which is a core component of MART. Five interaction encoders are evaluated in our prediction system for a fair comparison. The model specifications, such as the number of layers and heads for transformer [50] and relational transformer [13], are the same as those for MARTE. As illustrated at the top of Table 3, MARTE outperforms both transformer-based encoders (transformer and relational transformer) and GNN-based encoders (NMMP, GroupNet, and EqMotion). On the other hand, the results demonstrate that transformer-based encoders perform better than GNN-based encoders. Although the relational transformer exhibits commendable performance, it overlooks the importance of group-wise interaction, which is essential for accurate trajectory prediction. Thus, the findings underscore that MARTE enhances performance via a hypergraph transformer, marking a novel approach in trajectory prediction.

Effect of AGE. To validate our AGE module, it is compared to previously proposed group estimation methods [6, 54] by integrating them into MART. As shown at the bottom of Table 3, the AGE module performs better than the other methods. The group estimation of GP-Graph [6] leads to lower performance in complex scenes like the NBA datasets due to non-overlapping group estimation. GroupNet [54] performs similarly to our method in ADE but requires manual group size setting. Moreover, as the number of group scales increases, more encoders are required, resulting in higher computational costs.

6.3   Qualitative Results

Trajectory Prediction. In Figure 5, the predicted trajectories from MART (ours) are compared with two recent SOTA methods, LED [35] and EqMotion [56], along the ground truth (GT) trajectories on the NBA dataset. MART
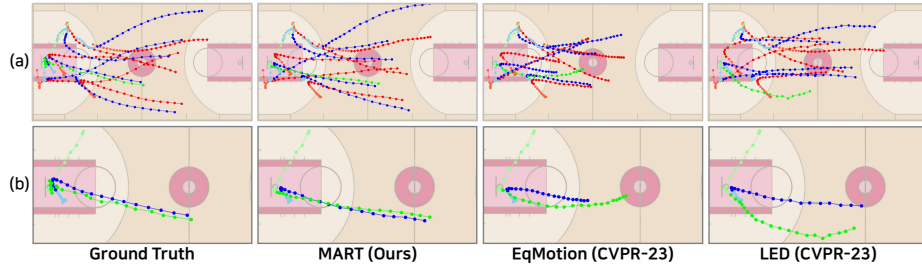
Fig. 5: Qualitative result comparison on NBA dataset. We qualitatively compare our method with two recent SOTA methods using the best of 20 predictions. (a) illustrates predictions for all agents, and (b) focuses on the ball-keeping player and basketball. Our method predicts more accurate future trajectories compared to LED [35] and EqMotion [56]. Notably, as illustrated in (b), MART provides robust predictions about the ball-keeping player and basketball, while other methods do not. These results demonstrate that MART can handle more complex interactions compared to state-of-the-art (SOTA) methods. (Light color represents the past trajectory, while blue/red/green indicate the two teams and basketball.)
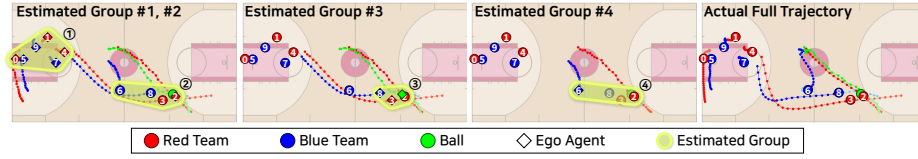

Fig. 6: Examples of group estimation results with predictions (first three figures) and actual full trajectories (last figure) on the NBA dataset. Duplicated groups are excluded, and the light color denotes the past trajectory. The left side of the court represents the blue team's region, while the right side of the court represents the red team's region. Group IDs are denoted as ①, ②, ③, and ④. Player IDs are provided in the figure on the far right (ID 0-4: red team, ID 5-9: blue team).

outperforms the previous methods, resulting in accurate trajectory predictions. Further visualization can be found in the supplementary material.

Group Estimation. Figure 6 displays examples of group estimation results on the NBA dataset, and Figure 7-(c) shows the matrix for unique groups. Each group reflects a specific strategy. For example, in Group #1, defenders employ man-to-man marking on offensive players during zone defense (See supplementary material for details).

Attention Weights Visualization. As illustrated in Figure 7-(a) and -(b), the attention weights of PRT and HRT focus on the pair-wise and group-wise relations, respectively. Notably, for HRT, the attention weights in HRT show the same accentuation as the group incidence matrix displayed on the right side of Figure 7-(b). Consequently, the visualization demonstrates that the proposed MARTE promotes the attention weights to emphasize interactions between agents, particularly focusing on group-wise relations.
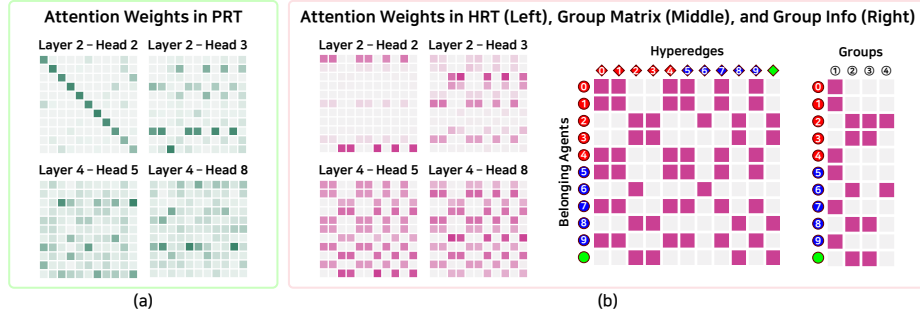
Fig. 7: (a) Visualization of attention weights in PRT. (b) Visualization of attention weights in HRT (left), the group incidence matrix G (middle), and the corresponding groups (right). The sample results from the identical test data sample used in Figure 6. For the attention visualization, min-max normalization is applied to ensure a clearer visual understanding, and darker colors indicate higher attention weights.

7   Conclusion

Conclusion. This study proposes MART, a multiscale relational transformer network for multi-agent trajectory prediction. MART considers interactions at both individual and group scales via the hypergraph transformer mechanism. Within the MART framework, we introduce the MARTE and AGE modules. MARTE extends the capabilities of the relational transformer and promotes attention weights to emphasize interactions between agents, particularly focusing on group-wise relations. The AGE module outperforms pioneering group estimation modules in terms of performance and efficiency. Our comprehensive evaluation across three real-world datasets demonstrates that MART achieves state-of-the-art performance in multi-agent trajectory prediction, showcasing its potential to advance the field.

Limitation and Future Work. Our model does not fully account for the temporal context of trajectories because it incorporates the past trajectory into the agent embedding. In future work, we plan to extend the model to incorporate spatio-temporal relationships. In addition, we aim to explore a trajectory prediction model with multi-modal inputs (e.g., trajectory inputs with scene context) because scene context, such as maps or images, is important in path prediction.

Acknowledgements

References

1. Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S.: Social lstm: Human trajectory prediction in crowded spaces. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 961–971 (2016)
2. Altché, F., de La Fortelle, A.: An lstm network for highway trajectory prediction. In: 2017 IEEE 20th international conference on intelligent transportation systems (ITSC). pp. 353–359. IEEE (2017)
3. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016)
4. Bae, I., Jeon, H.G.: Disentangled multi-relational graph convolutional network for pedestrian trajectory prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 911–919 (2021)
5. Bae, I., Oh, J., Jeon, H.G.: Eigentrajectory: Low-rank descriptors for multi-modal trajectory forecasting. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10017–10029 (2023)
6. Bae, I., Park, J.H., Jeon, H.G.: Learning pedestrian group representations for multi-modal trajectory prediction. In: Proceedings of the European Conference on Computer Vision (2022)
7. Bae, I., Park, J.H., Jeon, H.G.: Non-probability sampling network for stochastic human trajectory prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6477–6487 (2022)
8. Bengio, Y., Léonard, N., Courville, A.: Estimating or propagating gradients through stochastic neurons for conditional computation. arXiv preprint arXiv:1308.3432 (2013)
9. Berndt, H., Emmert, J., Dietmayer, K.: Continuous driver intention recognition with hidden markov models. In: 2008 11th International IEEE Conference on Intelligent Transportation Systems. pp. 1189–1194. IEEE (2008)
10. Chai, Y., Sapp, B., Bansal, M., Anguelov, D.: Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. In: Conference on Robot Learning. pp. 86–99. PMLR (2020)
11. Chen, D., O'Bray, L., Borgwardt, K.: Structure-aware transformer for graph representation learning. In: International Conference on Machine Learning. pp. 3469–3489. PMLR (2022)
12. Dendorfer, P., Elflein, S., Leal-Taixé, L.: Mg-gan: A multi-generator model preventing out-of-distribution samples in pedestrian trajectory prediction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13158–13167 (2021)
13. Diao, C., Loynd, R.: Relational attention: Generalizing transformers for graph-structured tasks. In: The Eleventh International Conference on Learning Representations (2023), https://openreview.net/forum?id=cFuMmbWiN6
14. Fang, L., Jiang, Q., Shi, J., Zhou, B.: Tpnet: Trajectory proposal network for motion prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6797–6806 (2020)
15. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: International conference on machine learning. pp. 1263–1272. PMLR (2017)
16. Giuliari, F., Hasan, I., Cristani, M., Galasso, F.: Transformer networks for trajectory forecasting. In: 2020 25th international conference on pattern recognition (ICPR). pp. 10335–10342. IEEE (2021)

17. Gu, T., Chen, G., Li, J., Lin, C., Rao, Y., Zhou, J., Lu, J.: Stochastic trajectory prediction via motion indeterminacy diffusion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17113–17122 (2022)
18. Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., Alahi, A.: Social gan: Socially acceptable trajectories with generative adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2255–2264 (2018)
19. Hu, W., Xie, D., Tan, T., Maybank, S.: Learning activity patterns using fuzzy self-organizing neural network. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **34**(3), 1618–1626 (2004)
20. Hu, Y., Chen, S., Zhang, Y., Gu, X.: Collaborative motion prediction via neural motion message passing. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 6319–6328 (2020)
21. Huang, Y., Bi, H., Li, Z., Mao, T., Wang, Z.: Stgat: Modeling spatial-temporal interactions for human trajectory prediction. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 6272–6281 (2019)
22. Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., Bengio, Y.: Binarized neural networks. Advances in neural information processing systems **29** (2016)
23. Karnan, H., Nair, A., Xiao, X., Warnell, G., Pirk, S., Toshev, A., Hart, J., Biswas, J., Stone, P.: Socially compliant navigation dataset (scand): A large-scale dataset of demonstrations for social navigation. IEEE Robotics and Automation Letters **7**(4), 11807–11814 (2022)
24. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015), http://arxiv.org/abs/1412.6980
25. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (2017), https://openreview.net/forum?id=SJU4ayYgl
26. Kosaraju, V., Sadeghian, A., Martín-Martín, R., Reid, I., Rezatofighi, H., Savarese, S.: Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. Advances in Neural Information Processing Systems **32** (2019)
27. Lee, M., Sohn, S.S., Moon, S., Yoon, S., Kapadia, M., Pavlovic, V.: Muse-vae: multi-scale vae for environment-aware long term trajectory prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2221–2230 (2022)
28. Lee, N., Choi, W., Vernaza, P., Choy, C.B., Torr, P.H., Chandraker, M.: Desire: Distant future prediction in dynamic scenes with interacting agents. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 336–345 (2017)
29. Lerner, A., Chrysanthou, Y., Lischinski, D.: Crowds by example. In: Computer graphics forum. vol. 26, pp. 655–664. Wiley Online Library (2007)
30. Li, L., Pagnucco, M., Song, Y.: Graph-based spatial transformer with memory replay for multi-future pedestrian trajectory prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2231–2241 (2022)
31. Li, X., Ying, X., Chuah, M.C.: Grip: Graph-based interaction-aware trajectory prediction. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC). pp. 3960–3966. IEEE (2019)

32. Liang, J., Jiang, L., Murphy, K., Yu, T., Hauptmann, A.: The garden of forking paths: Towards multi-future trajectory prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10508–10518 (2020)

33. Liu, Z., Wu, B., Luo, W., Yang, X., Liu, W., Cheng, K.T.: Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In: Proceedings of the European conference on computer vision (ECCV). pp. 722–737 (2018)

34. Mangalam, K., Girase, H., Agarwal, S., Lee, K.H., Adeli, E., Malik, J., Gaidon, A.: It is not the journey but the destination: Endpoint conditioned trajectory prediction. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16. pp. 759–776. Springer (2020)

35. Mao, W., Xu, C., Zhu, Q., Chen, S., Wang, Y.: Leapfrog diffusion model for stochastic trajectory prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5517–5526 (2023)

36. Mehran, R., Oyama, A., Shah, M.: Abnormal crowd behavior detection using social force model. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 935–942. IEEE (2009)

37. Min, E., Chen, R., Bian, Y., Xu, T., Zhao, K., Huang, W., Zhao, P., Huang, J., Ananiadou, S., Rong, Y.: Transformer for graphs: An overview from architecture perspective. arXiv preprint arXiv:2202.08455 (2022)

38. Mohamed, A., Qian, K., Elhoseiny, M., Claudel, C.: Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 14424–14432 (2020)

39. Morris, B.T., Trivedi, M.M.: Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach. IEEE transactions on pattern analysis and machine intelligence **33**(11), 2287–2301 (2011)

40. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019)

41. Pellegrini, S., Ess, A., Schindler, K., Van Gool, L.: You'll never walk alone: Modeling social behavior for multi-target tracking. In: 2009 IEEE 12th international conference on computer vision. pp. 261–268. IEEE (2009)

42. Robicquet, A., Sadeghian, A., Alahi, A., Savarese, S.: Learning social etiquette: Human trajectory understanding in crowded scenes. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14. pp. 549–565. Springer (2016)

43. Rudenko, A., Palmieri, L., Lilienthal, A.J., Arras, K.O.: Human motion prediction under social grouping constraints. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 3358–3364. IEEE (2018)

44. Sadeghian, A., Kosaraju, V., Sadeghian, A., Hirose, N., Rezatofighi, H., Savarese, S.: Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1349–1358 (2019)

45. Salzmann, T., Ivanovic, B., Chakravarty, P., Pavone, M.: Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16. pp. 683–700. Springer (2020)

46. Satorras, V.G., Hoogeboom, E., Welling, M.: E (n) equivariant graph neural networks. In: International conference on machine learning. pp. 9323–9332. PMLR (2021)
47. Shi, L., Wang, L., Long, C., Zhou, S., Zhou, M., Niu, Z., Hua, G.: Sgcn: Sparse graph convolution network for pedestrian trajectory prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8994–9003 (2021)
48. Sun, H., Zhao, Z., He, Z.: Reciprocal learning networks for human trajectory prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7416–7425 (2020)
49. Sun, J., Jiang, Q., Lu, C.: Recursive social behavior graph for trajectory prediction. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 660–669 (2020)
50. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
51. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: International Conference on Learning Representations (2018), `https://openreview.net/forum?id=rJXMpikCZ`
52. Vemula, A., Muelling, K., Oh, J.: Social attention: Modeling attention in human crowds. In: 2018 IEEE international Conference on Robotics and Automation (ICRA). pp. 4601–4607. IEEE (2018)
53. Wang, P., He, X., Li, G., Zhao, T., Cheng, J.: Sparsity-inducing binarized neural networks. In: Proceedings of the AAAI conference on artificial intelligence. vol. 34, pp. 12192–12199 (2020)
54. Xu, C., Li, M., Ni, Z., Zhang, Y., Chen, S.: Groupnet: Multiscale hypergraph neural networks for trajectory prediction with relational reasoning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6498–6507 (2022)
55. Xu, C., Mao, W., Zhang, W., Chen, S.: Remember intentions: Retrospective-memory-based trajectory prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6488–6497 (2022)
56. Xu, C., Tan, R.T., Tan, Y., Chen, S., Wang, Y.G., Wang, X., Wang, Y.: Eqmotion: Equivariant multi-agent motion prediction with invariant interaction reasoning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1410–1420 (2023)
57. Xu, C., Wei, Y., Tang, B., Yin, S., Zhang, Y., Chen, S., Wang, Y.: Dynamic-group-aware networks for multi-agent trajectory prediction with relational reasoning. Neural Networks (2023). `https://doi.org/https://doi.org/10.1016/j.neunet.2023.11.005`, `https://www.sciencedirect.com/science/article/pii/S0893608023006299`
58. Xu, Z., Cheung, R.C.: Accurate and compact convolutional neural networks with trained binarization. In: 30th British Machine Vision Conference (BMVC 2019) (2019)
59. Yu, C., Ma, X., Ren, J., Zhao, H., Yi, S.: Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16. pp. 507–523. Springer (2020)
60. Yuan, Y., Weng, X., Ou, Y., Kitani, K.M.: Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9813–9823 (2021)

61. Zhou, B., Wang, X., Tang, X.: Understanding collective crowd behaviors: Learning a mixture model of dynamic pedestrian-agents. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 2871–2878. IEEE (2012)
62. Zhu, H., Claramunt, F.M., Brito, B., Alonso-Mora, J.: Learning interaction-aware trajectory predictions for decentralized multi-robot motion planning in dynamic environments. IEEE Robotics and Automation Letters **6**(2), 2256–2263 (2021)
63. Zyner, A., Worrall, S., Nebot, E.: A recurrent neural network solution for predicting driver intention at unsignalized intersections. IEEE Robotics and Automation Letters **3**(3), 1759–1764 (2018)

Supplementary Material

A   Detailed Model Architecture

We provide detailed descriptions of the components in the model architecture, which were briefly explained earlier. For clarity, the bias vectors are omitted in this study. The components are detailed as follows: the node/pair-wise edge/hyperedge feature initializer $\mathcal{F}_{\mathrm{NI}}(\cdot)$ / $\mathcal{F}_{\mathrm{PI}}(\cdot)$ / $\mathcal{F}_{\mathrm{HI}}(\cdot)$, the transformer-like node / pair-wise / hyperedge update function $\mathcal{F}_n(\cdot)$ / $\mathcal{F}_e(\cdot)$ / $\mathcal{F}_h(\cdot)$, and the decoder $\mathcal{F}_{\mathrm{D},k}(\cdot)$.

- $\mathcal{F}_{\mathrm{NI}}(\cdot)$: It comprises two fully connected layers. The sinusoidal positional encoding [50] blends the temporal information before the last layer. The equations are as follows:

$$\mathbf{n}_i^{(0)} = \mathrm{Flatten}(\mathrm{PosEnc}(\mathbf{X}_i\mathbf{W}_{\mathrm{NI},1}))\mathbf{W}_{\mathrm{NI},2}, \tag{15}$$

  where $\mathrm{Flatten}(\cdot)$ denotes the flattening operation that transforms a multi-dimensional tensor into a one-dimensional tensor. $\mathrm{PosEnc}(\cdot)$ denotes the sinusoidal positional encoding [50] along the temporal dimension. $\mathbf{W}_{\mathrm{NI},1} \in \mathbb{R}^{d_{in} \times d_n}$ and $\mathbf{W}_{\mathrm{NI},2} \in \mathbb{R}^{T_p d_n \times d_n}$ are weight matrices for node feature initialization.
- $\mathcal{F}_{\mathrm{PI}}(\cdot)$ & $\mathcal{F}_{\mathrm{HI}}(\cdot)$: Two layers of MLP with ReLU activation and the hidden dimension d_h.
- $\mathcal{F}_n(\cdot)$: Assume that the output of multi-head relational or hyper-relational attention for the i-th agent is $\mathbf{a}_i^{(l)}$, and given the node feature for the i-th agent $\mathbf{n}_i^{(l)}$, the updated node feature $\mathbf{n}_i^{(l+1)}$ via $\mathcal{F}_n(\cdot)$ is described as follows:

$$\begin{aligned}
\mathbf{u}_i^{(l)} &= \mathrm{LayerNorm}(\mathbf{a}_i^{(l)}\mathbf{W}_1 + \mathbf{n}_i^{(l)}) \in \mathbb{R}^{d_n}, \\
\mathbf{z}_i^{(l)} &= \mathrm{FeedForward}(\mathbf{u}_i^{(l)}) \in \mathbb{R}^{d_n}, \\
\mathbf{n}_i^{(l+1)} &= \mathrm{LayerNorm}(\mathbf{z}_i^{(l)} + \mathbf{u}_i^{(l)}) \in \mathbb{R}^{d_n},
\end{aligned} \tag{16}$$

  where $\mathbf{W}_1 \in \mathbb{R}^{d_n \times d_n}$ is a weight matrix, $\mathrm{LayerNorm}(\cdot)$ denotes the layer normalization [3], and $\mathrm{FeedForward}(\cdot)$ represents two layers of MLP with ReLU activation and a hidden dimension of d_h.
- $\mathcal{F}_e(\cdot)$ & $\mathcal{F}_h(\cdot)$: Given the message vector $\mathbf{m}_i^{(l)}$ for the i-th edge feature $\mathbf{e}_i^{(l)}$, transformer-like edge update function is described as follows:

$$\begin{aligned}
\mathbf{u}_i'^{(l)} &= \mathrm{LayerNorm}(\mathbf{m}_i^{(l)}\mathbf{W}_2 + \mathbf{e}_i^{(l)}) \in \mathbb{R}^{d_e}, \\
\mathbf{z}_i'^{(l)} &= \mathrm{FeedForward}(\mathbf{u}_i'^{(l)}) \in \mathbb{R}^{d_e}, \\
\mathbf{e}_i^{(l+1)} &= \mathrm{LayerNorm}(\mathbf{z}_i'^{(l)} + \mathbf{u}_i'^{(l)}) \in \mathbb{R}^{d_e},
\end{aligned} \tag{17}$$

  where $\mathbf{W}_2 \in \mathbb{R}^{d_h \times d_e}$ is a weight matrix.
- $\mathcal{F}_{\mathrm{D},k}(\cdot)$: Three layers of MLP with ReLU activation and hidden dimensions of $[d_{\mathrm{D}}, d_{\mathrm{D}}/2]$, where d_{D} denotes the decoder dimension.

B   Further Experimental Details

B.1   Datasets

NBA SportVU Dataset (NBA). The NBA collected the NBA SportVU dataset using the SportVU tracking system during the 2015–2016 NBA season. The raw data is in JSON format and contains frame information, such as the IDs and positions of the players on the court, team IDs, and the game period. Consequently, the trajectories of the 10 players and the ball during actual basketball games were recorded and included in the dataset. Following [35, 57], we used 32.5k samples for training, 5.0k samples for validation, and 12.5k samples for testing. Additionally, we utilized the past 2.0 seconds (10 frames) of trajectories to forecast the next 4.0 seconds (20 frames).

ETH-UCY dataset. The ETH-UCY dataset [29, 41] includes five subsets: ETH, HOTEL, UNIV, ZARA1, and ZARA2, each comprising various pedestrian motion scenes captured at a frame rate of 2.5Hz. Based on previous studies [1, 6, 18], our model predicts the future trajectories of multiple agents for 4.8 s (12 frames) using the given 3.2 s (8 frames) of past trajectories. For evaluation, we train on four sets and assess the performance on the unused set, utilizing the leave-one-out method.

Stanford Drone Dataset (SDD). The SDD dataset [42] is a large-scale pedestrian motion dataset, recorded from a bird's eye view on a university campus and consisting of 20 scenes. We utilize the standard train-test split based on a previous study [34]. Then, given the 3.2 seconds (8 frames) of past trajectories, we predict the future trajectories for 4.8 seconds (12 frames).

B.2   Baseline Methods

We describe the baselines compared in Section 6.1.

- **NMMP** [20]: This method introduces neural motion message-passing to model interactions among traffic actors for motion prediction in vehicles and social robots.
- **STAR** [59]: This model is a spatio-temporal graph transformer focusing on trajectory prediction for crowd motion dynamics by leveraging attention mechanisms.
- **PECNet** [34]: The model introduces the predicted endpoint-conditioned network for human trajectory prediction. This method models the stochastic endpoints and proposes a novel non-local social pooling layer.
- **GroupNet** [54]: This method proposes a multiscale hypergraph message-passing neural network for multi-agent trajectory prediction to capture individual and group behaviors.
- **MemoNet** [55]: This model is a novel instance-based trajectory prediction framework designed to predict future intentions by establishing direct connections between present scenarios and previously observed instances.

Table 4: Hyperparameter candidates within the model architecture. **Bold** denotes the selected hyperparameters.

Parameter Name	Notation	Considered Values
node dimension	d_n	$\{\mathbf{64}, 128, 192\}$
edge dimension	d_e	$\{\mathbf{64}, 128, 192\}$
hidden dimension	d_h	$\{64, \mathbf{128}, 192\}$
decoder dimension	d_{D}	$\{\mathbf{128}, 256, 384, 512\}$
number of heads	-	$\{1, 2, 4, \mathbf{8}\}$

- **MID** [17]: This approach, termed motion indeterminacy diffusion (MID), employs a parameterized Markov chain and a transformer-based diffusion model to progressively refine trajectory predictions.
- **NPSN** [7]: This method addresses stochastic pedestrian trajectory prediction variability by introducing the Quasi-Monte Carlo method for improved sampling and proposing the Non-Probability Sampling Network (NPSN).
- **GP-Graph** [6]: GP-Graph is a pedestrian trajectory prediction model that captures intra-group and inter-group interactions by utilizing a group assignment module to cluster pedestrians into proximity-based groups.
- **DynGroupNet** [57]: This method, an extension of GroupNet [54], is a dynamic-group-aware network designed to enhance the modeling of interactions among agents for trajectory prediction, capable of capturing time-varying interactions at both the pair-wise and group scales.
- **ET** [5]: The paper presents EigenTrajectory (ET), a method for enhancing pedestrian trajectory prediction by creating a compact ET space for pedestrian movement representations.
- **EqMotion** [56]: EqMotion is a motion prediction model that is equivariant under Euclidean geometric transformations while considering invariant interaction reasoning.
- **LED** [35]: LED is a diffusion-based trajectory prediction model that offers real-time, accurate, and diverse predictions. This method accelerates inference speed by bypassing numerous denoising steps with a leapfrog initializer.

B.3   Model Input

We utilize the absolute and relative positions as model inputs for the NBA dataset, resulting in an input dimension d_{in} of 4. For the ETH-UCY and SDD datasets, the relative positions are used as input, establishing the input dimension d_{in} at 2.

B.4   Model Hyperparameter Search

Table 4 lists the hyperparameters considered within the model architecture. Utilizing these candidates, we conducted a hyperparameter search on the NBA dataset, with the number of layers (L) fixed at 1. Subsequently, the optimal value for L was established, as detailed in Section C.1.

Table 5: Learning rate, batch size, and number of training epochs used in this study.

Dataset	Subset	Learning Rate	Batch Size	Training Epochs
NBA	-	5.0×10^{-4}	64	100
SDD	-	1.0×10^{-3}	256	300
ETH-UCY	ETH HOTEL UNIV ZARA1 ZARA2	1.0×10^{-3} 1.8×10^{-3} 1.0×10^{-3} 1.2×10^{-3} 1.2×10^{-3}	64	300

Table 6: Effect of the number of layers (L) in MARTE on the NBA datsaet. The reported values are in meters.

L	min ADE_{20}	min FDE_{20}
1	0.761	0.943
2	0.738	0.916
3	0.733	0.908
4	**0.727**	**0.903**
5	0.741	0.919

Table 7: Ablation on the NBA dataset for the estimated gradient within the AGE module. Function types (a), (b), and (c) match with Figure 8(a), (b), and (c), respectively.

Func. Type	min ADE_{20}	min FDE_{20}
(a)	0.733	0.906
(b)	**0.727**	**0.903**
(c)	0.731	0.907

B.5 Model Training

We used an Adam optimizer [24] to train our network. The specific training hyperparameters, including the learning rate, batch size, and the number of training epochs, are detailed in Table 5. We applied a learning rate decay of 0.5 per 100 epochs for all datasets. Our training and evaluation were conducted on an NVIDIA GeForce RTX 3090. Our implementations were on PyTorch 2.1.0 [40].

C Supplementary Experiments

C.1 Effect of the Number of Layers

Table 6 displays the effect of the number of layers (L) in MART. This experiment was conducted on the NBA dataset using the same condition described in Section B.5. The performance of MART gradually improves until $L = 4$, after which it begins to decline. This experiment illustrates why we selected four layers.

C.2 Ablation for Estimated Gradient within AGE

Figure 8 illustrates the candidates for the estimated gradient within AGE. Table 7 lists the ablations for the estimated gradient within AGE. The modified triangular-shaped estimator performs better than others. The original STE [22] and the triangle-shaped estimator [33] approximate the gradient of $\mathrm{Sign}(\cdot)$. Because we estimate the unit step function $\mathcal{U}(\cdot)$, we have made slight modifications to them, as illustrated in Figures 8-(a) and -(b).
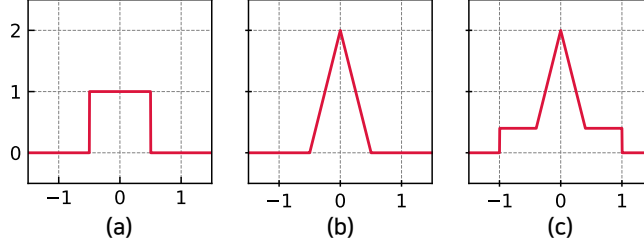
Fig. 8: Candidates for the estimated gradient in AGE. (a) Modified STE [22]. (b) Modified triangle-shaped estimator [33]. (c) Long-tailed estimator [58].

D   Further Qualitative Results

D.1   Trajectory Prediction

Figure 9 presents additional qualitative result comparisons on the NBA dataset. As depicted in the figure, our method yields more accurate predictions than recent state-of-the-art (SOTA) methods, including LED [35] and EqMotion [56], across a larger number of samples. Figure 10 shows qualitative results of MART on the ETH-UCY dataset. In some cases, we observed that the predicted trajectory does not satisfy the scene context, such as obstacles and maps.

D.2   Group Estimation

This section provides a detailed description of the group estimation results presented in Section 6.3. Figure 11 displays the visualization of group estimation results on the NBA dataset, while Figure 12 shows the visualization of the group incidence matrix $\mathbf{G}$ and the unique groups corresponding to Figure 11. Each group reflects a specific strategy as follows:

- **Group #1 (Zone Defense Group)**: Blue team defenders with player IDs 5, 7, and 9 employ man-to-man marking on red team offensive players with player IDs 0, 4, and 1, respectively.
- **Group #2 (Global Att&Def Group)**: The red team player with ID 2 dribbles the ball while being marked by the blue team player with ID 6. The player with ID 3 from the red team is connected to the player with ID 2 through an attacking link. Blue team players with IDs 6 and 8 are marking the red team players with IDs 2 and 3, respectively.
- **Group #3 (Local Att&Def Group)**: The red team player with ID 2 is dribbling with the ball. The blue team player with ID 8 is marking the attacking red team player with ID 3.
- **Group #4 (Local Att&Def Group)**: The blue team player with ID 6 defends against the attacking red team player with ID 2.
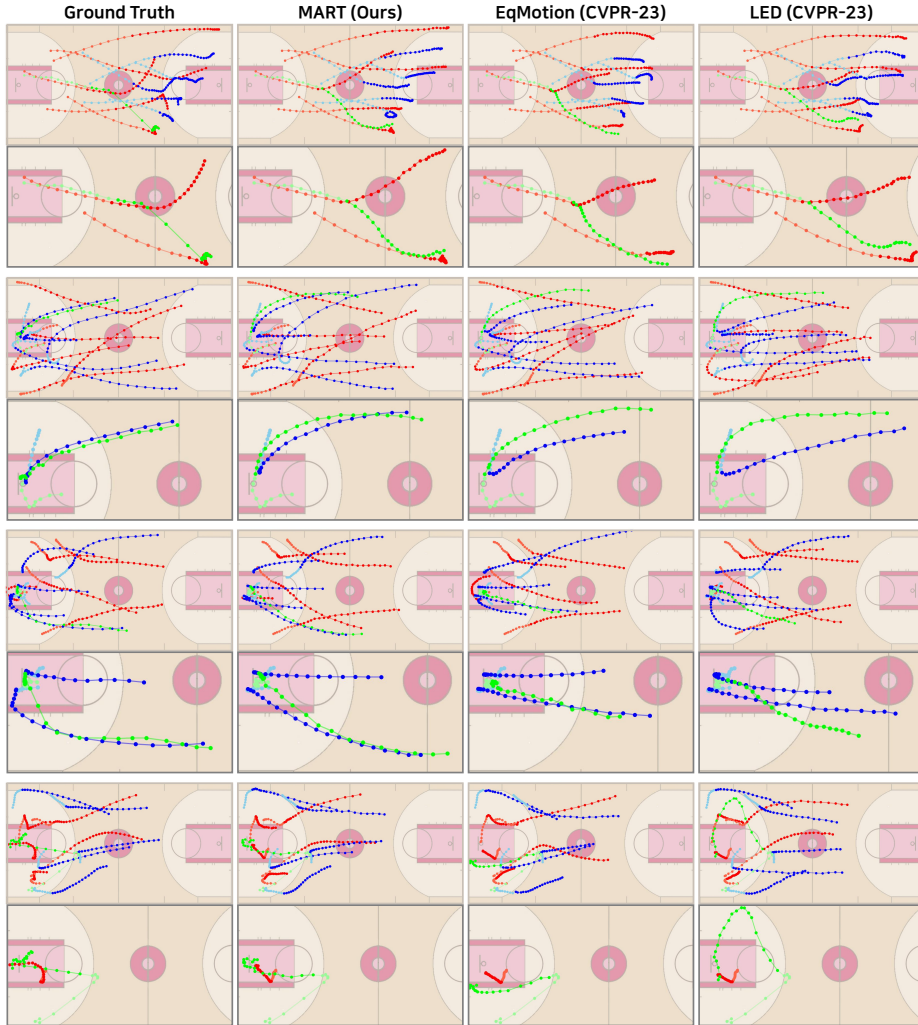
Fig. 9: Further qualitative result comparisons on NBA dataset. We qualitatively compare our method with two recent state-of-the-art methods using the best of 20 predictions. The khaki border illustrates predictions for all agents, while the gray border focuses on the ball-keeping player and the basketball. When comparing our method across more samples, it predicts more accurate future trajectories compared to LED [35] and EqMotion [56]. Notably, as illustrated in figures with a khaki border, MART provides robust predictions about the ball-keeping player and the basketball, whereas other methods do not. These results demonstrate that MART can handle more complex interactions compared to state-of-the-art (SOTA) methods. (Light color represents the past trajectory, while blue/red/green indicate the two teams and basketball.)
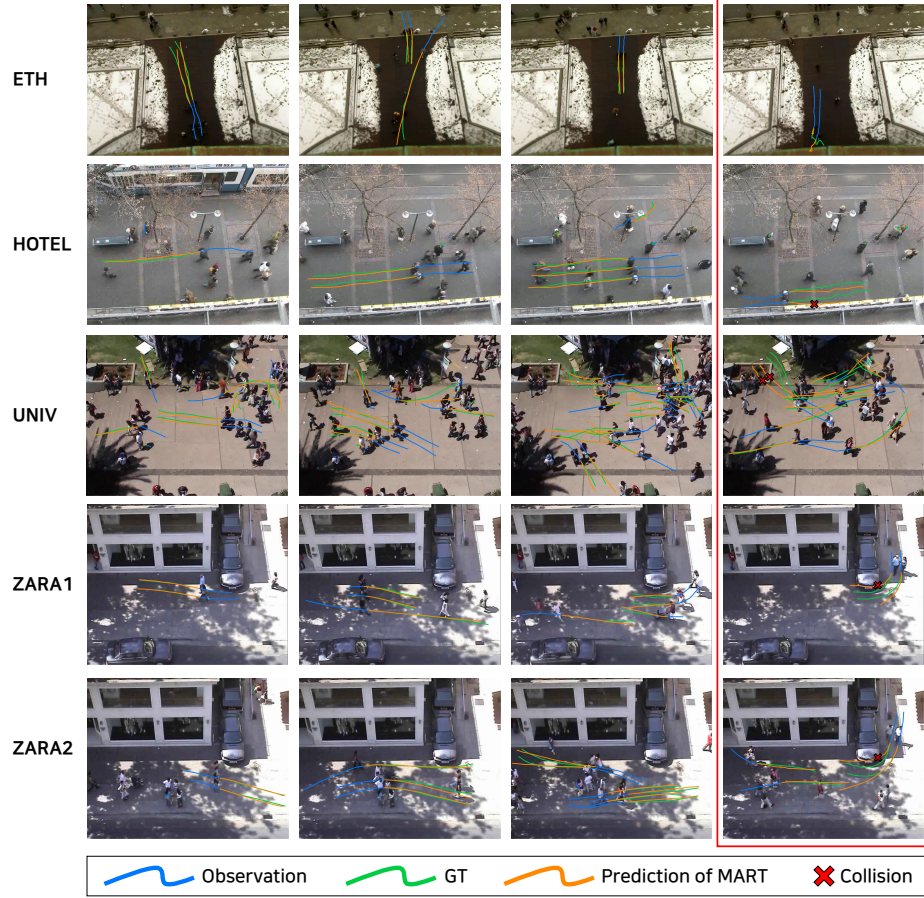
Fig. 10: Examples of trajectory prediction results on the ETH-UCY dataset. The figure displays trajectory prediction results of MART with the best of 20 predictions. The column boxed in red (□) indicates bad examples, where the predicted path collides with obstacles such as walls, flower beds, and cars.
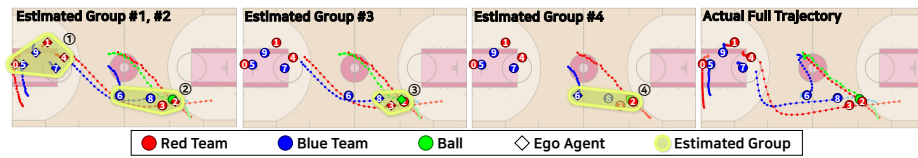

Fig. 11: Visualization of group estimation result. The left side of the court represents the blue team's region, while the right side of the court represents the red team's region. Group IDs are denoted as ①, ②, ③, and ④. Player IDs are provided in the figure on the far right (ID 0-4: red team, ID 5-9: blue team).
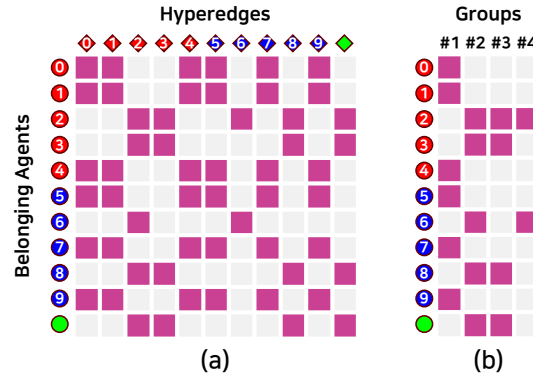
Fig. 12: Group estimation result. (a) Visualization of the group incidence matrix **G** (b) Visualization of unique groups (i.e., the group incidence matrix that replicated groups excluded). Both resulted from the identical test data sample used in Figure 11.