8. Double-Stage Training Pipeline

In this section, we elaborate the training procedures of RAIN for multi-agent interacting systems, which consist of two stages: pre-training stage and formal-training stage.

8.1. Pre-Training Stage

8.1.1 GMP Module

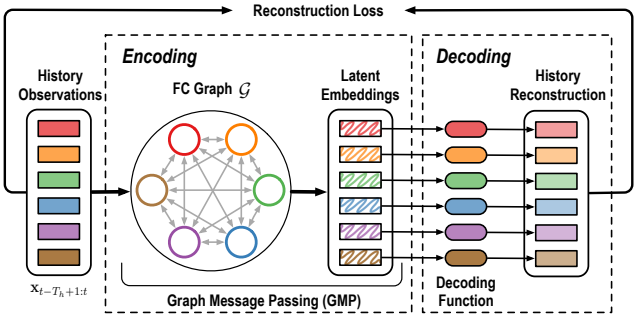

Figure 6. The diagram of the auto-encoder structure for pre-training the GMP module, which consists of an encoding procedure and a decoding procedure.

We employ a standard encoder-decoder structure to pre-train the GMP by unsupervised learning, with the purpose of enabling informative and effective feature extraction in the GMP module. In the auto-encoder structure, the GMP module serves as the encoding process to generate latent embeddings for each node. And an auxiliary decoding function is trained to reconstruct the history information with the latent embeddings. After the model is well-trained, the GMP module is able to extract good representation of the observation information.

The loss function is the standard mean squared error reconstruction loss, which is calculated as

$$L_{\text{GMP}} = \frac{1}{NT_h} \sum_{i=1}^{N} \sum_{t'=t-T_h+1}^{t} ||\mathbf{x}_{t'}^i - \hat{\mathbf{x}}_{t'}^i||^2. \quad (15)$$

After convergence, we save the parameters of the GMP module and discard those of the decoder, since we only use GMP in the following formal-training stage.

8.1.2 SGA-MG Module

In order to enable informative initial reward for the RL-HA module, we pre-train the SGA-MG module with a fully connected topology. The model architecture is the same as in Figure 2 except that the GAT is applied to a fully connected graph. The loss function is a standard mean squared error loss, which is calculated as

$$L_{\text{SGA-MG}} = \frac{1}{NT_f} \sum_{i=1}^{N} \sum_{t'=t+1}^{t+T_f} ||\mathbf{x}_{t'}^i - \hat{\mathbf{x}}_{t'}^i||^2. \quad (16)$$

Algorithm 1: Double-Stage Training Algorithm

Input: history $\mathbf{X}_{t-T_h+1:t}$, true future $\mathbf{X}_{t+1:t+T_f}$, context $\mathbf{C}$, hyperparameters N_{ft}, N_s, E
Initialize the parameters in GMP (ϕ), RL-HA (ψ) and SGA-MG (θ);
/* Pre-training Stage */
Pre-train GMP by unsupervised learning with (15);
Pre-train SGA-MG by supervised learning with (16);
/* Initialize RL */
Initialize the replay buffer $\mathcal{D}$;
Initialize the RL-step index $i \leftarrow 0$;
/* Formal-training Stage */
for *epoch* $\leftarrow$ *1,2,...,E* **do**
  Generate rollout ξ with ϕ, ψ, θ;
  Add rollout ξ into replay buffer $\mathcal{D}$;
  $i \leftarrow i + 1$;
  /* Train RL-HA */
  if $i > N_s$ **then**
    Sample a rollout ξ' from $\mathcal{D}$;
    Update policy and ψ with DDQN;
  end
  /* Finetune SGA-MG */
  for $m \leftarrow 1, 2, ..., N_{ft}$ **do**
    Sample a case of $\mathbf{X}_{t-T_h+1:t}$ and $\mathbf{C}$;
    Use GMP to obtain node attributes on $\mathcal{G}$;
    Use RL-HA to generate $\mathcal{G}'$;
    Use SGA-MG to generate $\hat{\mathbf{X}}_{t+1:t+T_f}$;
    Compute loss by equation (16);
    Update θ by back-propagation;
  end
end

8.2. Formal-Training Stage

In the formal-training stage, we initialize the GMP and SGA-MG with pre-trained parameters. Then we perform an alternating training strategy to train the RL-HA and SGA-MG alternatively until convergence. The detailed pseudo-code of the training pipeline of the whole framework is provided in Algorithm 1.

9. RAIN for Human Skeleton Motions (Cont.)

For human motion forecasting, we employ the state-of-the-art model [59] as the soft attention based motion generator in our framework. We strongly encourage the readers to refer to [59] for better understanding the model details below. Similar to RAIN for the multi-agent interacting systems, we also employ a double-stage training pipeline, including a pre-training stage and a formal-training stage. In the pre-training stage, we pre-train the parameters of a

contextual encoder and the soft attention based motion generator. In the formal-training stage, we train the RL-HA module and finetune the motion generator alternatively.

We denote the complete history motions as $\mathbf{X}_{t-T_h+1:t}$ and the future motions as $\mathbf{X}_{t+1:t+T_f}$. We have the same assumption as [59] that $T_h > T_s+T_f$ where T_s is the length of the motion segments used to compute attention weights.

9.1. Pre-Training Stage

First, we use an auto-encoder structure to train an encoding function that can extract the contextual information from the complete history motion sequence. More formally, the auto-encoder can be written as

$$\mathbf{Z} = \text{Encoding}(\mathbf{X}_{t-T_h+1:t}), \tag{17}$$

$$\hat{\mathbf{X}}_{t-T_h+1:t} = \text{Decoding}(\mathbf{Z}), \tag{18}$$

where Encoding and Decoding functions are neural networks. The loss function of training the auto-encoder is the standard mean squared error reconstruction loss, which is calculated by

$$\text{MSE} = \frac{1}{JT_h}\sum_{j=1}^{J}\sum_{t'=t-T_h+1}^{t} ||\mathbf{x}_{t'}^j - \hat{\mathbf{x}}_{t'}^j||^2, \tag{19}$$

where J is the number of relative angles between joints in a skeleton. For the soft attention based motion generator, since the authors of [59] released their official code and pre-trained models, we directly load their pre-trained parameters in the formal-training stage.

9.2. Formal-Training Stage

In the formal-training stage, we alternatively train the RL-HA module and the motion generator. We define the motion segments in the same way as [59]. More specifically, we first divide the complete motion history $\mathbf{X}_{t-T_h+1:t}$ into $T_h-T_s-T_f+1$ segments $\{\mathbf{X}_{i:i+T_s+T_f-1}\}_{i=t-T_h+1}^{t-T_s-T_f+1}$, each of which contains T_s+T_f consecutive frames of human poses. We use the same setting as [59], where the motion generator exploits the past T_s frames to predict the future T_f frames. The first T_s frames of each segment is used as a key, and the whole segment is then the corresponding value. The query is defined as the latest segment $\mathbf{X}_{t-T_s+1:t}$.

9.2.1 RL-HA Module

In the domain of forecasting human skeleton motions, the RL-HA module is expected to select the key history motion segments for the current prediction based on the latest observation segment. Then the soft attention mechanism in [59] will further rank the relative importance of the selected key segments, which is employed by the motion predictor to generate future motions.

The selection of key segments naturally fits into a reinforcement learning framework. The definition of observations, actions and reward functions of the RL-agent are elaborated in the following.

Observations: The observation O of RL-agent at RL-step η ($\leq T_{\text{RL}}$) includes a tuple of key, query, contextual information $\mathbf{Z}$ as well as the current segment selection status s_i (0: "retained" or 1: "discarded"). T_{RL} is the upper bound of RL-steps. The observation O_η is obtained by

$$O_\eta = [f_k(\mathbf{X}_{i:i+T_s-1}),\ f_q(\mathbf{X}_{t-T_s+1:t}),\ \mathbf{Z},\ s_{i,\eta}], \tag{20}$$

where f_k and f_q are mapping functions modeled by neural networks. Note that the dimension of O_η only depends on the dimensions of key, query and contextual information, which enables the applicability to the scenarios with varying numbers of history motion segments. The policy network of RL-agent takes the observation O_η as input and decides the action at each RL-step.

Actions: There are two possible actions for the RL-agent: "staying the same" (action 0) and "changing to the opposite" (action 1). At each RL-step, the RL-agent makes decision for each history motion segment. The policy can be written as $a = \pi(O)$. We do not enforce any constraints on the selection of motion segments, i.e., there is no lower / upper bound on the number of selected segments. The actions of RL-agent may change the key motion segments after each RL-step, which further influences the soft attention based motion generator.

Rewards: The reward consists of two parts: regular reward R_{reg} and improvement reward R_{imp}. More specifically, the *regular reward* is the negative mean squared error of future predictions calculated by

$$R_{\text{reg},\eta} = -\frac{1}{J}\sum_{j=1}^{J}\sum_{t'=t+1}^{t+T_f} ||\mathbf{x}_{t'}^j - \hat{\mathbf{x}}_{t',\eta}^j||^2. \tag{21}$$

The *improvement reward* encourages the decrease of prediction error via applying a sign function to the error change between consecutive RL-steps, which is obtained by

$$R_{\text{imp},\eta} = \text{sign}(R_{\text{reg},\eta} - R_{\text{reg},\eta-1}). \tag{22}$$

The whole reward is obtained by $R_\eta = R_{\text{reg},\eta} + \beta_{\text{imp}} R_{\text{imp},\eta}$, where β_{imp} is a hyperparameter.

9.2.2 Alternating Training Strategy

The contextual encoding function is initialized with well-trained parameters in the pre-training stage and fixed in the formal-training stage. The soft attention based motion generator is initialized with the pre-trained model in [59]. We perform alternating optimization of RL-HA and motion generator (MG) modules: (a) train the RL-HA module

Motion	Directions					Greeting					Phoning					Posing				
millisecond	80	160	320	400	1k	80	160	320	400	1k	80	160	320	400	1k	80	160	320	400	1k
Res-sup [40]	0.41	0.64	0.80	0.92	–	0.57	0.83	1.45	1.60	–	0.59	1.06	1.45	1.60	–	0.45	0.85	1.34	1.56	–
CSM [27]	0.39	0.60	0.80	0.91	1.45	0.51	0.82	1.21	1.38	1.72	0.59	1.13	1.51	1.65	1.81	0.29	0.60	1.12	1.37	2.65
Traj-GCN [34]	0.26	0.45	0.70	0.79	–	**0.35**	0.61	0.96	1.13	–	0.53	1.02	1.32	1.45	–	0.23	0.54	1.26	1.38	–
DMGNN [31]	**0.25**	0.44	0.65	0.71	–	0.36	0.61	**0.94**	**1.12**	–	0.52	0.97	1.29	1.43	–	0.20	0.46	1.06	1.34	–
LTD-10-10 [39]	0.26	0.45	0.71	0.79	1.35	0.36	**0.60**	0.95	1.13	1.59	0.53	1.02	1.35	1.48	1.74	**0.19**	0.44	**1.01**	1.24	2.55
HisRepItself [59]	**0.25**	**0.43**	**0.60**	**0.69**	**1.27**	**0.35**	**0.60**	0.95	1.14	**1.57**	0.53	1.01	1.31	1.43	1.68	**0.19**	0.46	1.09	1.35	**2.32**
Ours (hybrid)	**0.25**	0.46	0.65	0.75	1.34	**0.35**	0.62	0.99	1.21	1.64	**0.49**	**0.95**	**1.23**	**1.35**	**1.59**	**0.19**	**0.43**	1.02	**1.23**	2.46
Motion	Purchases					Sitting					Sitting Down					Taking Photo				
millisecond	80	160	320	400	1k	80	160	320	400	1k	80	160	320	400	1k	80	160	320	400	1k
Res-sup [40]	0.58	0.79	1.08	1.15	–	0.41	0.68	1.12	1.33	–	0.47	0.88	1.37	1.54	–	0.28	0.57	0.90	1.02	–
CSM [27]	0.63	0.91	1.19	1.29	2.52	0.39	0.61	1.02	1.18	1.67	0.41	0.78	1.16	1.31	2.06	0.23	0.49	0.88	1.06	1.40
Traj-GCN [34]	0.42	0.66	1.04	1.12	–	0.29	0.45	0.82	0.97	–	0.30	0.63	0.89	1.01	–	0.15	0.36	0.59	0.72	–
DMGNN [31]	0.41	**0.61**	1.05	1.14	–	0.26	0.42	0.76	0.97	–	0.32	0.65	0.93	1.05	–	0.15	0.34	0.58	0.71	–
LTD-10-10 [39]	0.43	0.65	1.05	1.13	2.27	0.29	0.45	0.80	0.97	1.52	0.30	0.61	0.90	1.00	1.67	0.14	0.34	0.58	0.70	1.05
HisRepItself [59]	0.42	0.65	**1.00**	**1.07**	**2.22**	0.29	0.47	0.83	1.01	1.55	0.30	0.63	0.92	1.04	1.70	0.16	0.36	0.58	0.70	1.08
Ours (hybrid)	0.41	0.63	1.07	1.14	2.25	**0.24**	**0.40**	**0.75**	**0.96**	**1.47**	**0.27**	**0.58**	**0.85**	**0.97**	**1.58**	**0.14**	**0.32**	**0.53**	**0.64**	**0.91**
Motion	Waiting					Walking Dog					Walking Together					Average				
millisecond	80	160	320	400	1k	80	160	320	400	1k	80	160	320	400	1k	80	160	320	400	1k
Res-sup [40]	0.32	0.63	1.07	1.26	–	0.52	0.89	1.25	1.40	–	0.27	0.53	0.74	0.79	–	0.40	0.69	1.04	1.18	–
CSM [27]	0.30	0.62	1.09	1.30	2.50	0.59	1.00	1.32	1.44	1.92	0.27	0.52	0.71	0.74	1.28	0.38	0.68	1.01	1.13	1.77
Traj-GCN [34]	0.23	0.50	0.92	1.15	–	0.46	0.80	1.12	1.30	–	0.15	0.35	0.52	0.57	–	0.27	0.53	0.85	0.96	–
DMGNN [31]	0.22	0.49	0.88	1.10	–	0.42	0.72	1.16	1.34	–	0.15	0.33	**0.50**	0.57	–	0.27	0.52	0.83	0.95	–
LTD-10-10 [39]	0.23	0.50	0.91	1.14	2.37	0.46	0.79	1.12	1.29	1.86	0.15	0.34	0.52	0.57	1.16	0.27	0.52	0.83	0.95	1.62
HisRepItself [59]	0.22	0.49	0.92	1.44	2.30	0.46	0.78	1.05	1.23	1.82	**0.14**	0.32	**0.50**	**0.55**	1.16	0.27	0.52	0.82	0.94	1.57
Ours (hybrid)	**0.21**	**0.46**	**0.83**	**1.03**	**2.18**	**0.41**	**0.71**	**1.01**	**1.15**	**1.72**	**0.14**	**0.31**	**0.50**	0.56	**1.15**	**0.25**	**0.48**	**0.79**	**0.91**	**1.51**

Table 5. Comparison of mean angle errors (MAE) of different methods for both short-term ($\leq$ 400 milliseconds) and long-term prediction (1k milliseconds) on the other 11 actions in the Human3.6M dataset. "–" indicates that the original paper did not report the result.

	full+soft	hybrid (w/o GMP)	hybrid (w GMP)
Vehicle	0.63/1.28	0.59/1.23	**0.54/1.12**
Pedestrian	0.32/0.56	0.31/0.54	**0.26/0.51**

Table 6. 4.0s minADE_{20}/minFDE_{20} in different ablation settings (nuScenes).

	$\tau = 2$	$\tau = 5$	$\boldsymbol{\tau = 10}$	$\tau = 20$
Vehicle	0.50/1.08	0.52/1.09	0.54/1.12	0.58/1.17
Pedestrian	0.24/0.50	0.25/0.51	0.26/0.51	0.29/0.54

Table 7. 4.0s minADE_{20}/minFDE_{20} with different τ values (nuScenes).

with fixed parameters of MG with Double Deep Q-Learning (DDQN) method [18]; (b) finetune MG with fixed parameters of RL-HA using back-propagation methods.

10. Additional Experimental Results

In this section, we provide complementary experimental results on the nuScenes and Human3.6M datasets.

10.1. nuScenes Dataset: Traffic Scenarios

Besides the ablation studies (Table 2, 3) in the main paper, we show the results of an additional ablation setting hybrid (w/o GMP) in Table 6 to illustrate the effectiveness of GMP module. In this setting, the RL agent can only observe the self features of each node (agent) without knowing the relational/social features extracted by GMP, which leads to larger prediction errors compared to hybrid (w/ GMP). Note that hybrid (w/o GMP) outperforms full+soft, which implies the RL based hybrid attention can still help with motion prediction even without relational features. These results demonstrate that all modules are indispensable with improvement.

In our experiments, we used $\tau = 10$ (10 frames in 2.0s). We show additional results with different τ values in Table 7. Generally, the minADE_{20}/minFDE_{20} reduces as τ becomes smaller (i.e., the frequency of graph topology inference increases), which implies the advantage of dynamic prediction mechanism. However, the running time will increase as τ becomes smaller.

10.2. Human3.6M Dataset: Human Motions

We provide the comparison of prediction results on the other 11 actions for skeleton based human motion forecasting in Table 5. It is shown that *Ours (hybrid)* achieves the smallest MAE in most actions as well as in average compared with baselines. The action "Directions" is an interesting exception where HisRepItself outperforms our method. A potential reason is that in the "Directions" action, there is no clear pattern of the temporal dependency between the current observation and previous motions, which makes it hard for the RL-agent to discriminate and select the appropriate history motion segments to pay attention to.

We also visualize the prediction of human skeletons and the learned hybrid attention weights in typical testing cases in Figure 7. It shows that our method can accurately forecast the human motions. More specifically, we visualize the

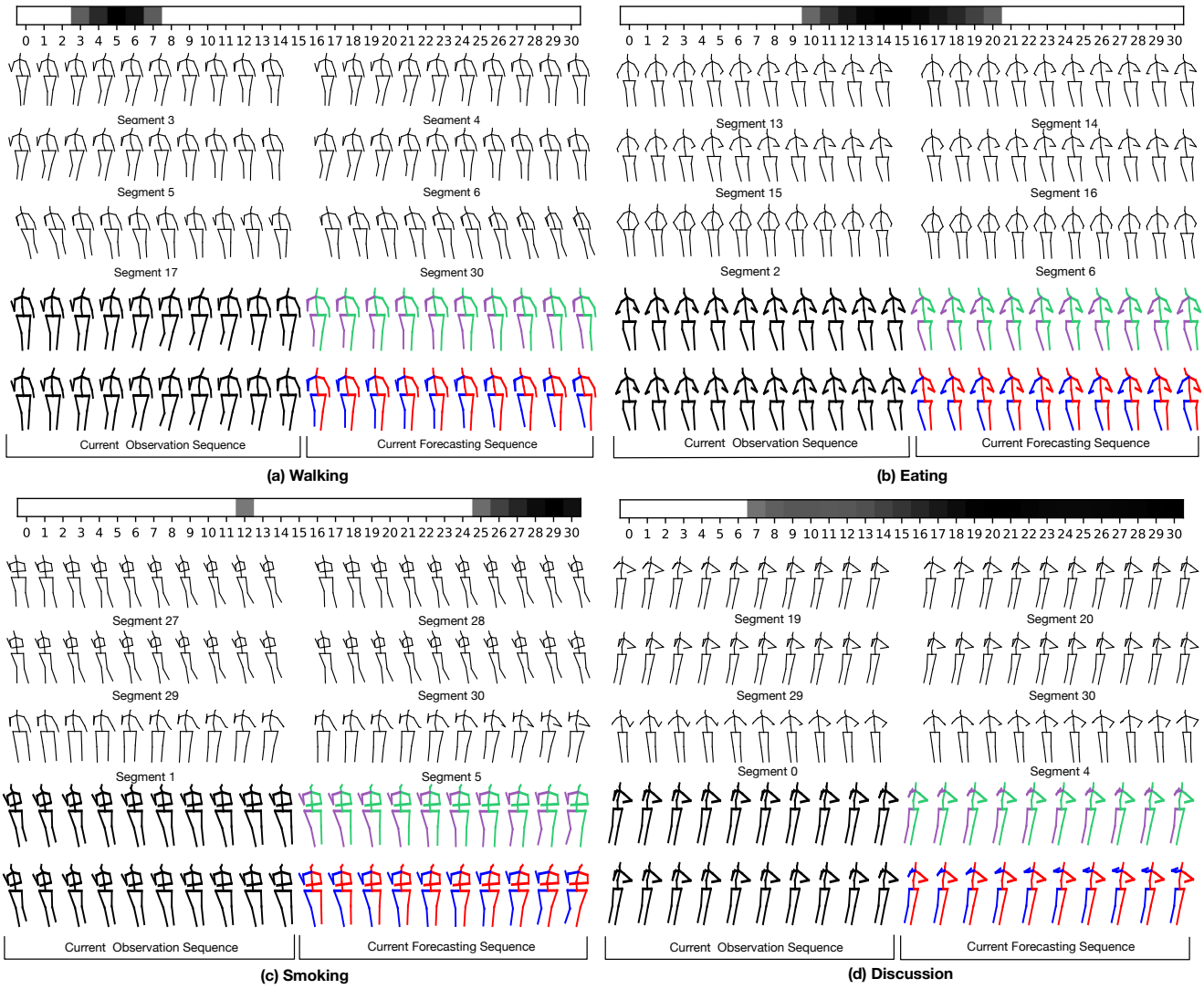

Figure 7. The visualization of human skeleton motion forecasting of four typical actions with hybrid attention maps. The black skeletons at the bottom are the latest observation sequences which are used to calculate current attention weights. The purple-green skeletons are the prediction hypotheses of our method. The blue-red skeletons are the ground truth. In our experimental setting, for each case there are 31 available history motion segments with a length of 10 frames for the RL hard attention module to select and the soft attention is only applied to the selected segments. In the hybrid attention maps, darker colors indicate larger attention weights. White color means the corresponding motion segment is not selected as key information. Best viewed in color.

top four motion segments with the largest four hybrid attention weights in each case (i.e., the motion segments in the first two rows). It shows that these segments have similar patterns with the current observation sequence and thus are selected as key information, which implies that the learned hybrid attention is reasonable and interpretable. We also visualize some irrelevant segments that are discarded by the model (i.e., the motion segments in the third row). It can be easily found that these segments are dissimilar to the latest observation sequence, thus are unimportant for the current prediction.

In addition, in case (a), (b) and (c), the learned hybrid attention is sparse, which implies that the model is able to effectively discriminate and only focus on the key information. An interesting exception is case (d), where most history segments are selected. A potential reason is that most history segments in this case are very similar to the current observation sequence, which leads the model to take them all into account for current prediction.

11. Further Experimental Details

In this section, we provide further details of the experiments, which includes dataset generation, baseline meth-

ods, as well as implementation details.

11.1. Datasets and Evaluation Metrics

11.1.1 Mixed Particle Simulation

In the mixed particle system, there are two types of particles: charged particles and uncharged particles. The charged particles are uniformly sampled to carry positive or negative charges $q_i \in \{\pm q\}$, which interact with each other via Coulomb forces, which is given by

$$F_{ij} = C \cdot (q_i \cdot q_j) \cdot \frac{r_i - r_j}{||r_i - r_j||^3}, \tag{23}$$

where C is a constant. These charged particles may either attract or repel each other, although the forces may be weak when the distance in between is large. However, the motions of uncharged particles are totally independent since there is no force applied to them. They move straight with a constant velocity the same as the random initialization. In this paper, we have 3 charged particles and 3 non-charged ones in each case. We generated 8k samples for training, and 4k samples for validation and testing, respectively.

The simulation process of charged particles is mainly adopted from NRI [23]. In order to prevent the force divergence issue when two particles move with a very small distance, we adopt the same strategy as suggested in [23] to avoid numerical issues, which is to clip the forces to some maximum threshold. Despite that this is not exactly physically precise, the generated trajectories are not distinguishable to human observers and do not affect the conclusion of the paper.

The evaluation metric for trajectory prediction in this experiment is the mean squared error, which is calculated by

$$\text{MSE} = \frac{1}{NT_f} \sum_{i=1}^{N} \sum_{t'=t+1}^{t+T_f} ||\mathbf{x}_{t'}^i - \hat{\mathbf{x}}_{t'}^i||^2. \tag{24}$$

11.1.2 nuScenes Dataset [4]

The nuScenes dataset is a widely used large-scale driving dataset with a full set of sensor suite, which was collected in Boston and Singapore. It provides the point cloud information, trajectory annotations of heterogeneous traffic participants (e.g., cars, pedestrians and cyclists), as well as the map information. We processed the original data into segments with a length of 6 seconds to construct our dataset (2 seconds as history and 4 seconds as future). We generated about 8k samples for training, and 2k samples for validation and testing, respectively.

We adopt the standard evaluation metrics in trajectory prediction, which include minADE_K, minFDE_K and miss rate (MR). In this paper, we use the same $K = 20$ as most baselines. The MR(@dm) is calculated by

$$\text{MR}(@d\text{m}) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\left(\min_k ||\mathbf{x}_{t+T_f}^{i,k} - \hat{\mathbf{x}}_{t+T_f}^{i,k}||_2 > d\right), \tag{25}$$

where $\mathbb{I}(\cdot)$ denotes an indicator function to indicate whether the current case is a failure case, and d is a manually defined threshold.

11.1.3 Human3.6M Dataset [21]

The Human3.6M dataset is a widely used skeleton-based human motion dataset for pose estimation and motion forecasting, which includes 15 different activities performed by 7 professional actors. The human skeleton information is provided in two representations: 3D joint positions and joint angles. The skeleton has 32 joints, the 3D coordinates of which can be computed by applying the forward kinematics. As in [59], we also down-sample the motion sequences to 25 frames per second, and remove the global rotation, translation and constant angles. In this paper, we chose relative angles between joint to represent the skeleton state.

11.2. Baseline Methods

11.2.1 Ablative Baselines

- Ours (true+soft): This is the model that only applies soft graph attention to the true relation graph. Note that this model is only used for the experiments on mixed particle simulation since the true relation graph is not accessible in real-world scenarios and dataset.

- Ours (full+soft): This is the model that only applies soft graph attention to a fully connected relation graph.

- Ours (ELBO+soft): This is the model where only the RL-HA module is replaced by an ELBO based module with other modules not changed, which is trained end-to-end. The purpose of this ablation setting is to provide a baseline based on an alternative way to obtain hard attention.

- Ours (hybrid, static): This is the model that applies both RL hard attention to obtain the inferred relation graph and soft attention to figure out relative importance. The inferred relation graph remains static during the whole prediction horizon and the model performs one-shot prediction.

- Ours (hybrid, dynamic): This model setup is very similar to Ours (hybrid, static). The difference is that the model performs iterative prediction with a fixed horizon of sliding window. The inferred relation graph is dynamically evolving over time.

- Ours (hybrid): This model setup is only used for human motion prediction. The RL hybrid attention and soft attention work together to extract informative history features for the motion generator in [59].

11.2.2 For Mixed Particle Simulation

- Corr. (LSTM): The baseline method for edge recognition used in [23].
- LSTM (single) / LSTM (joint): The baseline methods for state sequence prediction in [23].
- NRI: The NRI model with static latent interaction graph [23].
- DNRI: A model for neural relational inference with dynamic interaction graphs [17].
- Supervised: Since the true relation graph is accessible in the simulation data, we can use supervised learning to train a binary classifier to infer the existence of the edges in the graph. The ground truth labels include 0 (w/o edge) and 1 (w/ edge).

11.3. Implementation Details

In this section, we introduce the details of model architecture, hyperparameters and specific experimental settings for each dataset.

11.3.1 For Multi-Agent Interacting Systems

We trained the models for 100 epochs for both particle simulation and nuScenes dataset. They shared the same model architecture and specific details of model components are introduced below:

- GMP: The state embedding functions f_s^m, f_n^m, node attribute update function f_v, and the encoding function f_{enc} are all three-layer MLPs with hidden size = 64. During the pre-training stage, the decoding function is also a three-layer MLP with hidden size = 64. The context embedding function f_c is a four-layer convolutional block with kernel size = 5. The layer structure is [[Conv, ReLU, Conv, ReLU, Pool], [Conv, ReLU, Conv, ReLU, Pool]]. The context embedding is only applied to the "traffic scenario", where the context information is the projected point cloud images.
- RL-HA: The maximum RL-step η is set to 10. In the total reward, the hyperparameters are $\beta_{\text{imp}} = \beta_{\text{sti}} = \beta_{\text{pun}} = 0.01$. We also set $\Omega_s = \Omega_p = 1.0$. In the "traffic scenario", we define a "success case" where the end-point error is less than the miss rate threshold and a "failure case" as the opposite. The stimulation reward is applied when the current case changes from "failure case" to "success case", and the punishment reward is applied for the opposite situation.

  All the coefficients and thresholds in reward function were decided empirically. $R_{\text{reg}}/R_{\text{imp}}$ reward the overall improvement of prediction while $R_{\text{sti}}/R_{\text{pun}}$ are mainly related to endpoint prediction. We found that increasing the weights of $R_{\text{sti}}/R_{\text{pun}}$ could improve both minADE and minFDE in a certain range while overly large weights could have negative effects on minADE. The miss rate thresholds should be specifically decided for various types of agents.
- SGA-MG: The Embedding LSTM (E-LSTM) and Generation LSTM (G-LSTM) both have a hidden size of 128. For the particle simulation, we performed one-shot prediction with a static inferred relation graph; for the nuScenes dataset, we performed progressive forecasting with a sliding window of 2 seconds (10 frames) with dynamic relation graphs.

11.3.2 For Human Skeleton Motions

We trained the models for 50 epochs on the Human3.6M dataset. We adopted exactly the same experimental settings as [59]. More specifically, during training, we trained the model to predict the future 10 frames based on the history 50 frames and the attention weights are calculated based on the latest observation sequence with 10 frames. During testing, we enabled progressive long-term prediction with a sliding window to generate future 25 frames.

Specific details of model components are introduced in the following:

- Encoding / Decoding (pre-training stage): They are three-layer MLPs with hidden size = 128.
- RL-HA: The maximum RL-step η is set to 10. In the total reward, the hyperparameter is $\beta_{\text{imp}} = 0.01$.
- Motion Generator: We adopted the same model architecture and hyperparameters as in [59].