Supplementary Materials for *Multi-Agent Trajectory Prediction by Combining Egocentric and Allocentric Views*

Xiaosong Jia[1], Liting Sun[2], Hang Zhao[3], Masayoshi Tomizuka[2], and Wei Zhan[2]

[1]Shanghai Jiao Tong University - jiaxiaosong1997@gmail.com
[2]University of California, Berkeley - {litingsun,wzhan}@berkeley.edu, tomizuka@me.berkeley.edu
[3]Tsinghua University - zhaohang0124@gmail.com

A Qualitative Results

In this section, we give visualizations of the prediction results of the proposed model against baseline methods in Fig. 1 and Fig.2. From the visualizations, we could find that the proposed model could better capture the complex interactions among agents and make accurate predictions compared to baseline methods.

B Discussion About Normalization of Trajectory Data

The widely used representation format for the motions of multiple agents is Cartesian coordinates. However, the origin and orientation of the coordinate system could be arbitrary to describe the same pattern. As a result, to make the neural network based model generalize well instead of overfitting the usually limited data, the input space and output space should be regularized cautiously.

Generally speaking, there are three kinds of design philosophy:

- P1: Input all possible representations of the same pattern to train the neural network
- P2: Preprocess the pattern into a canonical form. As a result, different forms of the same pattern would be the same input for the neural network.
- P3: Represent in a relative form to avoid using the absolute origin and orientation.

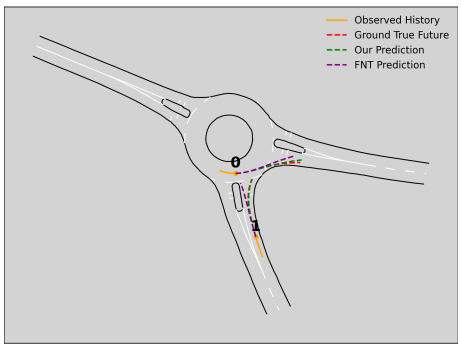
Only adopt P1 is infeasible since the range of origins is infinite. Many prediction models adopt P2 by 1. using the ego agent's coordinate at a certain time-step as the origin. 2. making the direction of the x-axis align with the ego agent's yaw angle at a certain time-step. In this paper, we adopt this kind of normalization as default and use t_0 since the choice of that certain time-step is arbitrary. We explore the effects of other normalization ways including relative features and random rotation of axis in Sec. C.

C Exploration of Different Normalization Strategies

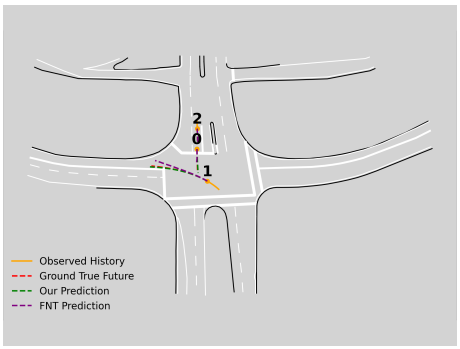
Besides the most widely adopted normalization method, we further conducted experiments with our method under other normalization strategies including:

- Temporally Relative Movements (Rel): in the original setting, we select the states of ego agent at a certain time step to normalize. In Rel, others' features are normalized at each time-step by ego agent's states at the corresponding time-step similar to [13]. Take the coordinate as an example, the input edge is the relative coordinates of other agents under the coordinate system where the ego agent is at the origin all the time. Note that the node input is processed as the original normalization so that the absolute motion information is

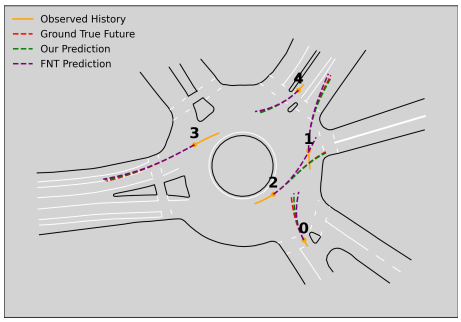
5th Conference on Robot Learning (CoRL 2021), London, UK.



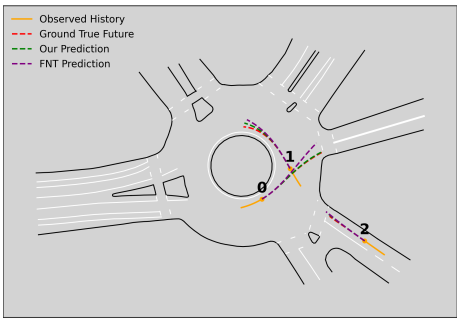
(a) The proposed model captured the interactive relations of v0-v1 and successfully predicted that v0 would like to go first and made a curve to occupy the road and v1 had to go after v0 while the baseline FNT model simply predicted lane-following behaviors.



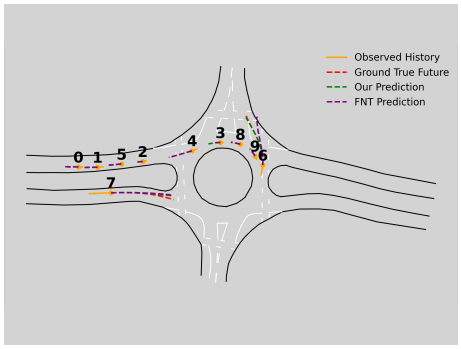
(b) The proposed model captured the interactions between v0 and v1 and successfully predicted that v1 had to make turn with less cure considering v0 made aggressive forward moving while the FNT model failed to predict the situation.



(c) The baseline FNT model failed to predicted the goal of v2 and thus its predictions about v0 and v2 deviated from the ground truth. On the other side, the proposed model correctly predicted v2's future motion and correctly predicted that v0 could not directly move toward its goal and had to follow v2.



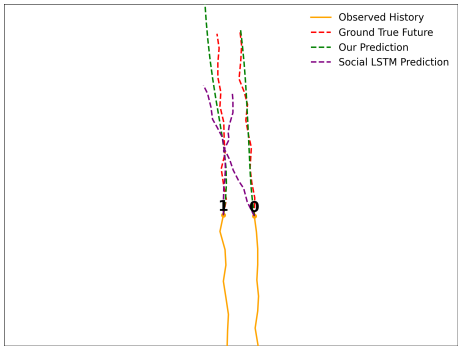
(d) The proposed model found the influence from v1 to v0 and correctly predicted v0's avoiding tendency while the FNT model failed to do so.



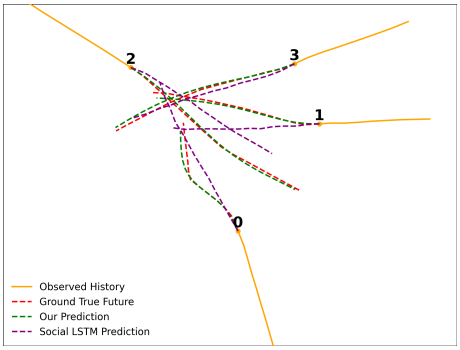
(e) v6 and v9 were close to each other which means that v6 should move cautiously toward its goal to avoid collision. The proposed correctly gave the path v6 had taken to cross while the FNT model gave a path which has collision with curbstone.

(f) The FNT model failed to predict the slow movement of v1 and thus wrongly predicted v2 as making aggressive and dangerous movement while the proposed model made the correct prediction for both v1 and v2.

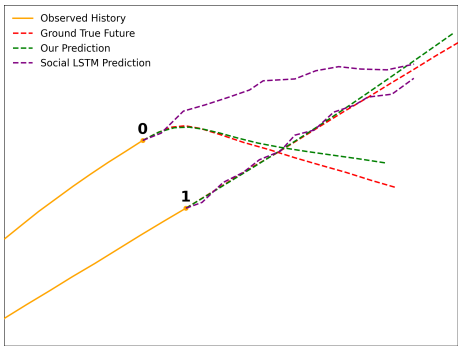
Figure 1: Qualitative results on validation set of INTERACTION Dataset. The observed history trejctories of agents are in yellow, ground truth of future trajectories are red, the proposed model's predictions are are in green, and the FNT model's predictions are in purple. The number are the ID of agents and we will refer the vehicle with ID i as vi.



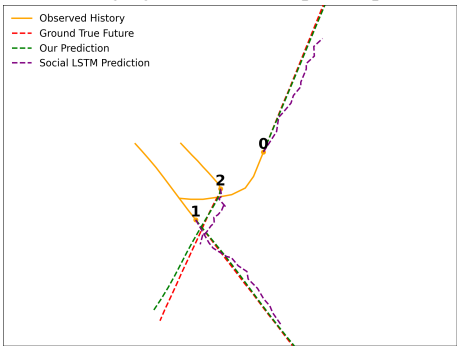
(a) This is the group moving behavior of pedestrians. The proposed model successfully predicted how two pedestrians walk together in a group while the Social LSTM's prediction caused a collision.



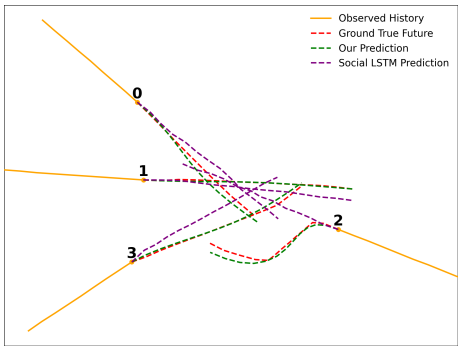
(b) p1, p2, p3, and p4 all have their own goal. Social LSTM made simple predictions of moving forward and caused very closed distance at some moments in the pairs p0-p3 and p1-p2 which is what pedestrians try to avoid in reality. The proposed model successfully predicted the changing of direction of p0 and p1.



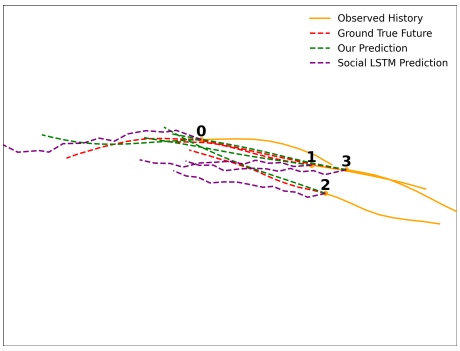
(c) In this case, p0 needed to make a right turn in front of p1. Social LSTM predicted v0's goal wrongly while the proposed model correctly predicted the path of v0 and v1.



(d) In this case, Social LSTM made too conservative predictions for p0 and p2. The proposed model correctly predicted the fast and efficient motion of the pedestrians.



(e) This is also a Collision-Avoidance case in which p0, p1, p2 and p3 had different goals and overlapped paths. The proposed captured the complex interactions and correctly predicted the bypassing of all agents while Social LSTM made a more straightforward predictions which may not often happen in reality.



(f) This is a Leader-Follower case where p0 was the leader and p1, p2, and p3 are the followers. The proposed correctly predicted their behaviors and intentions while the Social LSTM failed to predict the closely following pattern of p1, p2, and p3.

Figure 2: Qualitative results on validation set of TrajNet++ Dataset. The observed history trejctories of agents are in yellow, ground truth of future trajectories are red, the proposed model's predictions are are in green, and the Social LSTM's predictions are in purple. The number are the ID of agents and we will refer the pedestrian with ID i as pi.

preserved. In this setting, we use different W_k, W_v in the multi-head attention for self-loop edge and other edges so that the absolute and relative information could be processed respectively.

- Random Rotation (Rot): some SOTA methods [5, 7, 13, 20] use random rotation to augment the data instead of aligning the positive x-axis. This is feasible since the range of orientation is limited ($0 \sim 2\pi$) and it is possible to let the neural network learn to handle all possible orientations. Its benefit is that it can avoid the sensitivity to the yaw angle which might be inaccurate for the noisy data with lots of fluctuations and thus make the model much robust. However, its downside is that it needs much longer training epochs considering the input and output space are larger.

Table 1: Comparison of different normalization strategies.

	INTERACTION			TrajNet++
	Val ADE/FDE	Test ADE/FDE	Generalization ADE/FDE	Val ADE/FDE
origianl	0.1723/0.5988	0.1903/0.6563	0.3394/1.1983	0.2079/0.4270
Rel	0.1773/0.6091	0.1925/0.6573	0.3412/1.1772	0.2207/0.4509
original+Rel	0.1768/0.6072	0.2115/0.6884	0.3395/1.1123	0.2203/0.4535
Rot	0.1732/0.6047	0.1906/0.6592	0.3297/1.1589	0.2573/0.5145
original+Rel+Rot	0.1857/0.6273	0.2031/0.6798	0.3283/1.1489	0.2200/0.4323

With the two aforementioned normalization strategies, we compare the following 5 settings: original, Rel, original+Rel (concatenation of the two kinds of input features), Rot, original+Rel+Rot. Considering normalization methods have a large influence on the generalization ability of the model, we conduct extra experiments on the Test Track (similar scenes with train/val) and Generalization Track (different scenes of train/val) of INTERACTION dataset whose labels are both held by the organizer. From the results in Tab. 1, we can draw the following conclusion: 1. It seems that there is no gain to use the relative feature of each time-step (Rel). We conjecture that it is because this mathematical transformation is trivial for the neural network to learn. Additionally, the extra parameters W_K, W_V make it easier to overfit. 2. Random rotation (Rot) also does not have advantages in these two datasets, which makes the longer training not worth it.

D Experimental Details

D.1 Datasets

INTERACTION Datset [12] consists of various highly interactive driving situations, including highway ramps, roundabouts, and intersections, recorded using drones or fixed cameras worldwide. For sample generation and train/val split, we use the official code to generate samples. The only difference is that we divide the snippets by time while the official code further chooses each agent in the scene as the ego agent and generates multiple samples. As a result, we have 43668 training snippets and 10751 validation snippets. The average number of agents in each sample is 10.7. Same as the default setting, the frames are sampled at 10Hz and we use 10 frames as input to predict 30 frames in the future. Results reported in the existing papers are all based on the single agent prediction. Considering the only difference is that we treat a scene with N examples as one sample while the default setting treats as N samples, the val ADE/FDE could be directly compared since it is a metric calculated by average over agents. In fact, all of their reported results are equivalent to being obtained from forward-N-times (FNT) of their model. Additionally, INTERACTION has an online leader board for two tracks whose labels are held: the test track which is in similar scenes with train/val data, and the generalization track which is in very different scenes. Since the competition is for single agent prediction, we only upload the target agent's prediction from our model.

TrajNet++ Dataset [13] comprises several popular public pedestrian datasets including ETH/UCY/WildTrack/L-CAS/CFF and data generated by ORCA simulators. Their authors selected those highly interactive samples from the original data and made sure different kinds of interactions have a proper ratio. Here, we do not use the CFF data in it since this scene has too many agents in one sample (100+) which makes it impossible to run the simplest GNN for FNT setting under batch-size 1 on RTX 2080 Ti. Note that existing results reported in this dataset are based on the results of the primary pedestrian of each scene where the primary pedestrian is defined by the author

as 'the interesting agent' in the scene. Thus, we re-divide the scene by time and predict all agents' future trajectories in the same snippet. We split the train/val in 8:2 and make sure there are no overlaps in time between train and validation set. As a result, we have obtained 60860 training samples and 15219 validation samples. The average number of agents in each sample is 5.3. Same as the default setting, the frames are sampled at 2.5Hz and we use 9 frames as input to predict 12 frames in the future. Since prediction for agents other than primary pedestrians are not considered in the ADE/FDE calculating of the original dataset, we re-implement two top-rank methods on it under our multi-agent prediction setting.

D.2 Loss

For all models, we set an MLP prediction head to output all the future coordinates of each agent based on their corresponding output node feature from GNN. We use the widely adopted smooth L1 loss as in Equ. 1 where B is the batch size and T is the number of steps predicted.

$$\begin{aligned} \mathcal{L} &= \frac{1}{2BT}\sum_{b=1}^{B}\sum_{t=1}^{T} l(x_{bt}-\hat{x}_{bt}) + l(y_{bt}-\hat{y}_{bt}) \\ \text{where} \quad l(x) &= \begin{cases} 0.5x^2 & ||x|| < 1 \\ ||x|| - 0.5 & ||x|| \geq 1 \end{cases} \end{aligned} \tag{1}$$

D.3 Implementation Details

For simplicity, we use the fully-connected graph as an example in the paper. Note that similar conclusions could be drawn when it comes to the non-fully-connected version. To make a fair comparison on the GNN setting, for other modules of the framework, we use the same configuration for all models in the ablation study. Specifically, we use Resnet-18 1D version+MLP as the temporal encoder for the input sequences of both node and edge. In the aggregation layers, we use Transformer (multi-head-attetnion+feed-forward-network). In the prediction layer, for each agent, we use a shared MLP to output their future trajectories based on each one's output node feature from GNN.

We train all models with AdamW, lr 5e-4, weight-decay 1e-2, dropout 0.05, hidden dim 128, head number 4, GNN layer 2 if not specified. We train each model for 300 epochs and report the best results on the val set. The raw feature used in INTERACTION Dataset includes the coordinate, velocity, yaw angle, width, and length of vehicles. As for the TrajNet++ dataset, we used coordinates and calculated the velocity and yaw angle and take the three features as raw input. For the relative feature, we use relative coordinate, relative velocity, and relative yaw angle. As for all the angle data, we further use its corresponding cos and sin value as the input for ease of learning. Fig. 3 gives the detailed structure of the proposed model with an example of 3 agents.

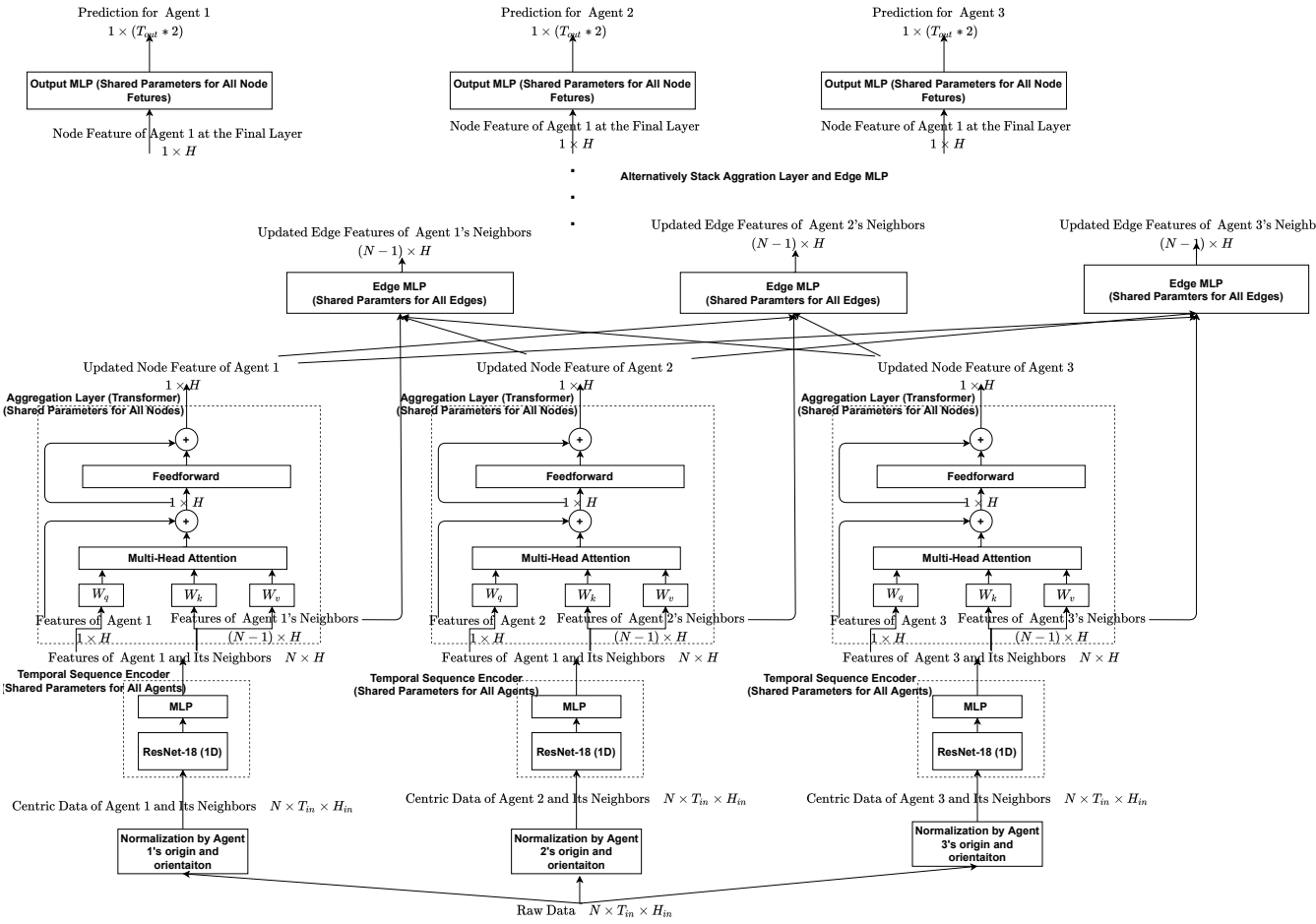


Figure 3: Implementation Details of the proposed method.