Supplementary Material for Remember Intentions: Retrospective-Memory-based Trajectory Prediction

Chenxin Xu[1*], Weibo Mao[1*], Wenjun Zhang[1], Siheng Chen[1,2†],
[1]Shanghai Jiao Tong University, [2]Shanghai AI Laboratory
{xcxwakaka,kirino.mao,zhangwenjun,sihengc}@sjtu.edu.cn

1. Network Architecture

In this section, we will show the detailed network architectures in the prediction system including the social encoder $\mathcal{E}_{\text{social}}(\cdot)$, the decoder $\mathcal{D}(\cdot)$ and the fulfillment decoder $\mathcal{D}_{\text{full}}(\cdot)$. The fulfillment encoder $\mathcal{E}_{\text{full}}(\cdot)$ has the same structure as the social encoder.

1.1. Social Encoder

The social encoder $\mathcal{E}_{\text{social}}(\cdot)$ consists of the individual branch and the social branch; see Fig. 1. The individual branch takes the agent's past trajectory $\mathbf{X}$ as the input and outputs the ego information embedding. The social branch takes both the agent's past trajectory $\mathbf{X}$ and its neighbouring agents' past trajectories $\mathbb{X}_{\mathcal{N}}$ as the input and uses the social pooling module to fuse the social information. We adopt the neural motion message passing (NMMP) in [7] as the social pooling module. The output of the two branches is then concatenated as the final output of the social encoder.

In our implementation, the input channel for *Conv1d* is 2 with the output channel equals 16. The input/output size for *GRU* is 16/64. The weights for both *Conv1d* and *GRU* are initialized with Kaiming norm [6] and the biases are initialized to 0. For the social pooling module, we use 2 iterations of neural message passing in NMMP with the bottleneck dimension equals to 64.

1.2. Decoder

The decoder $\mathcal{D}(\cdot)$ aims to reconstruct past trajectory and predict the future intention from input past-intention feature $[\mathbf{k};\mathbf{v}]$. The decoder has a residual structure which consists of two residual blocks; see Fig. 2. Each residual block receives the past trajectory $\mathbf{X}$ along with past-intention feature $[\mathbf{k};\mathbf{v}]$, and produces a part of reconstructed past trajectory $\widehat{\mathbf{X}}_1, \widehat{\mathbf{X}}_2$ and predicted future intention $\widehat{\mathbf{y}}_1^{T_f}, \widehat{\mathbf{y}}_2^{T_f}$. We sum up the two residual blocks' outputs as the final output $\widehat{\mathbf{X}} = \widehat{\mathbf{X}}_1 + \widehat{\mathbf{X}}_2$ and $\widehat{\mathbf{y}}^{T_f} = \widehat{\mathbf{y}}_1^{T_f} + \widehat{\mathbf{y}}_2^{T_f}$. In each residual block, we use a

*Equal contribution.
†Corresponding author.

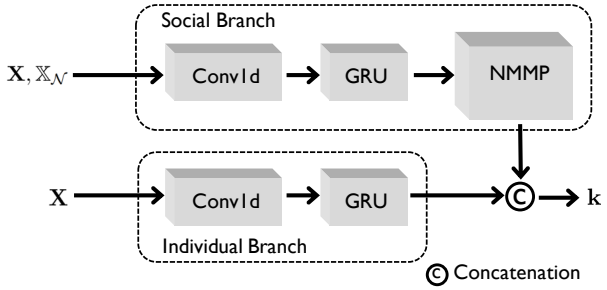

Figure 1. The structure of the social encoder $\mathcal{E}_{\text{social}}(\cdot)$.

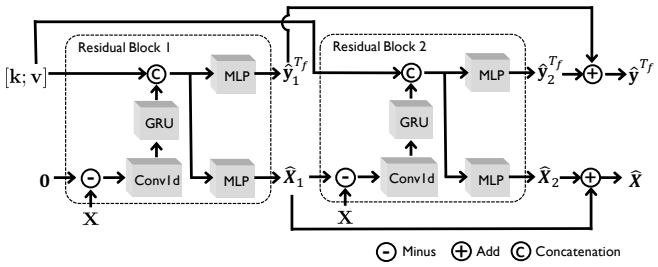

Figure 2. The residual structure of the decoder $\mathcal{D}(\cdot)$.

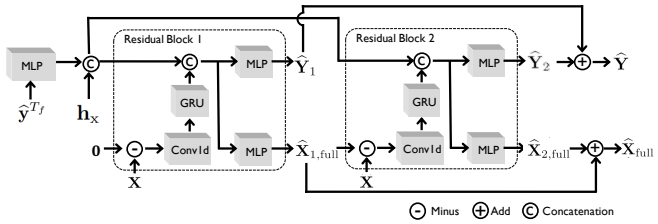

Figure 3. The residual structure of the fulfillment decoder $\mathcal{D}_{\text{full}}(\cdot)$.

Conv1d and a *GRU* to extract past trajectory's feature and two MLPs as output heads to produce reconstructed past trajectory and predicted future intention.

In our implementation, MLPs in the decoder have the hidden size of $64 \times 3 \to 1024 \to 512 \to 1024$. *Conv1d* and *GRU* modules have the same implementation as in the social encoder $\mathcal{E}_{\text{social}}(\cdot)$.

1.3. Fulfillment Decoder

The fulfillment decoder $\mathcal{D}_{\text{full}}(\cdot)$ has the similar structure with the decoder $\mathcal{D}(\cdot)$ except that the input hidden representation now contains the intention information, i.e., $\mathbf{h}'_x = [\mathbf{h}_x; \mathcal{F}_d(\widehat{\mathbf{y}}^{T_f})]$; see Fig. 3.

In our implementation, the intention encoding function $\mathcal{F}_d$ is a MLP with the hidden size $\rightarrow 8 \rightarrow 16 \rightarrow 16$. The hidden size of MLPs in the fulfillment decoder is $64 \times 3 + 16 \rightarrow 1024 \rightarrow 512 \rightarrow 1024$. *Conv1d* and *GRU* modules have the same implementation as in the social encoder $\mathcal{E}_{\text{social}}(\cdot)$.

2. Experiment Details

2.1. Baselines

We compare our MemoNet against several state-of-the-art prediction methods, which are now briefly described below:

- Social-LSTM [1]: It proposes a LSTM based model which shares the social information between layers to predict group behaviors.

- Social-GAN [5]: It adopts a GAN based model with a pooling mechanism to learn social interactions. This paper also uses a variety loss to encourage prediction diversity.

- STGAT [8]: It proposes a sequence-to-sequence model using LSTM and graph attention network to better capture the spatial-temporal interactions for pedestrians involved in a scene.

- Social-STGCNN [12]: This paper models social interactions as a graph and apply a kernel function in the weighted adjacency matrix for data efficiency.

- Tranformer-TF [3]: It tries the use of a transformer network for trajectory prediction. This paper only uses attention based mechanisms to model social interactions and achieve a better long-term prediction.

- STAR [15]: It proposes a transformer based graph convolution mechanism to model intra-graph social interaction for predicting trajectories.

- Trajectron++ [14]: It incorporates the heterogeneous data and produces future-conditional predictions that respect dynamics constraints tightly integrated with downstream robotic modules.

- SOPHIE [13]: It proposes an interpretable framework based on GAN jointly modeling path history and scene context information to provide plausible predictions.

- NRI [4]: It proposes dynamic neural relational inference based on sequential latent variable models to model dynamic entity relations for every time-step.

- NMMP [7]: It proposes a neural message passing module to explicitly model the interactions between actors. This work uses an individual branch for a single actor and an interactive branch for interactions between actors.

- MANTRA [11]: This paper proposes the use of a memory module for trajectory prediction. A learnable controller is introduced to control the memory size which only takes the prediction error as the input.

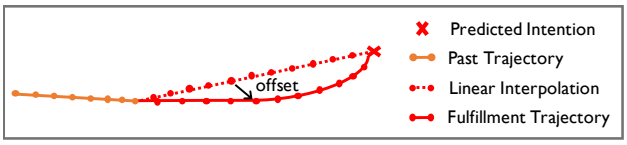

Figure 4. The fulfillment decoder predicts offsets based on the linear interpolation.

- EvolveGraph [9]: It proposes a generic forecasting framework for a multi-agent interacting system. It uses latent interaction graphs among multiple interactive agents to recognize and predict explicit relational structure.

- CF-VAE [2]: It uses the conditional normalizing flow based prior for better capture complex multi-modal conditional distribution predicting trajectories.

- PECNet [10]: This work uses variational autoencoder structure to infer the destination first. With the assistant of distant trajectory endpoints, it then will interpolate trajectories.

- AgentFormer [16]: It proposes an agent-aware transformer-based prediction structure, which simultaneously models the time and social dimensions.

2.2. Training Details

In paper submission, we show the overall training pipeline. Here we will further demonstrate details in our experiments including warming up the learnable addresser and predicting offsets for the fulfillment decoder.

Warm up the learnable addresser. To search the addresses of similar past memory instances in the memory bank for an input past trajectory feature, we propose a trainable addresser with a novel similarity loss. However, we find that it is hard to train this addresser from scratch due to the coupling between $\mathcal{F}_q$ and $\mathcal{F}_k$. Notice that if both functions are identity functions, then the learnable addresser will degenerate into a cosine similarity function. Inspired by this, we first will warm up the addresser with the loss function $\mathcal{L}_q = ||\mathcal{F}_q(q) - q||_2$ and $\mathcal{L}_k = ||\mathcal{F}_k(k_i) - k_i||_2$. After warming up, we then will use the similarity loss to finetune the learnable addresser.

Offsets prediction for the fulfillment decoder. To prevent a sudden change between the intentions and fulfilled trajectory, we first apply a linear interpolation between the location on timestamp 0 and the intention then fulfill the trajectory by predict the offset with the interpolated trajectory. Fig. 4 gives an example. Let $\widehat{\mathbf{y}}_i^{T_f}$ be the position of ith intention after clustering and $\mathbf{x}^0$ be the last observed frame. Then the final predicted frame can be expressed as $\widehat{\mathbf{y}}_i^j = \frac{j}{T_f}(\widehat{\mathbf{y}}_i^{T_f} - \mathbf{x}^0) + \mathbf{x}^0 + \mathbf{y}_{i,\text{full}}^j$, where $j = 1, 2, ..., T_f - 1$ and $\mathbf{y}_{i,\text{full}}^j$ is the MLP's output for the jth frame.

Table 1. Hyperparameters for different datasets.

Dataset		Memory Size	Destination Clustering
SDD		14653(81.5%)	120
ETH-UCY	ETH	27506(90.8%)	250
	Hotel	26489(89.3%)	260
	Univ	8168(82.7%)	370
	Zara1	24336(85.2%)	240
	Zara2	23391(89.7%)	280
NBA		341K(95.5%)	120

Table 2. Influence of d_T for training the learnable addresser on SDD.

d_T	20	80	140	200	500
minADE$_{20}$	8.79	8.56	8.70	8.73	8.77
minFDE$_{20}$	12.94	12.66	12.78	12.82	12.93

2.3. Hyperparameters for Different Datasets

Here we report the hyperparameters for different datasets including memory size and destination clustering; see Table 1. As verified in the ablation experiments, both memory size and destination clustering have significant influences on the model performance. Intuitively, it is hard for the memory bank that only has seen walking (east turning) cases to accurately predict running (west turning) cases. Therefore, we adopt data normalization for some datasets. Here we mainly consider two types of data normalization: rotation θ and scale s, both of which only use the first and the last frames in the past trajectory. Mathematically, $\theta = \arctan[\frac{(\mathbf{x}^{-T_p+1} - \mathbf{x}^0)[1]}{(\mathbf{x}^{-T_p+1} - \mathbf{x}^0)[0]}]$ and $s = ||\mathbf{x}^{-T_p+1} - \mathbf{x}^0||$. We then construct a rotation matrix with $M(\theta) = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$ and normalize the past trajectory $\mathbf{X}$ using $\mathbf{X}_{\text{norm}} = \frac{\mathbf{X}M(\theta)^T}{s}$.We use the rotation normalization for ETH, Hotel, and Univ subsets and the scale normalization for all the ETH-UCY subsets.

3. Supplementary Experiments

3.1. Influence of d_T

In our designed pseudo label for training the learnable addresser: $\mathcal{L}_{\text{Addr}} = \sum_{i=1}^{M}(s_i - \max(0, \frac{d_T - d_i}{d_T}))^2$. There exists a distance threshold hyperparameter whose influence will be discussed here. Table 2 shows the results for different d_T. We see that i) the best performance is achieved when $d_T = 80$ on SDD; and ii) too large or too small threshold will lead to the performance degradation because of the generated unreasonable pseudo label.

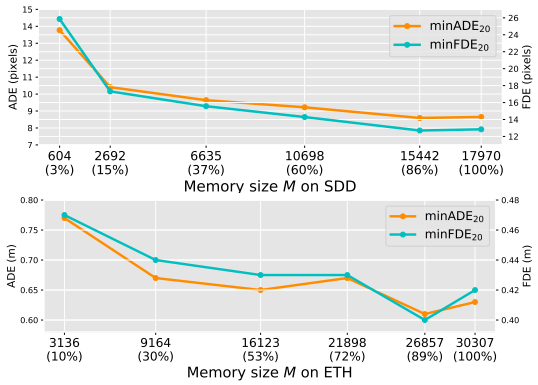

Figure 5. ADE/FDE as a function of the memory size M.

3.2. Influence of memory size

We report the relation between the memory size and the performance in Table 5 in the paper. Here we also draw curves on both SDD and ETH dataset to have a more clear representation; see Figure 5. We see that i) when memory size is too small, the memory bank preserves insufficient information and is hard to provide relevant instances, deteriorating the performance. ii) when memory size is too large, the memory bank preserves redundant information and decreases intention diversity, affecting the performance.

References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 961–971, 2016. 2

[2] Apratim Bhattacharyya, Michael Hanselmann, Mario Fritz, Bernt Schiele, and Christoph-Nikolas Straehle. Conditional flow variational autoencoders for structured sequence prediction. *arXiv preprint arXiv:1908.09008*, 2019. 2

[3] Francesco Giuliari, Irtiza Hasan, Marco Cristani, and Fabio Galasso. Transformer networks for trajectory forecasting. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 10335–10342. IEEE, 2021. 2

[4] Colin Graber and Alexander G Schwing. Dynamic neural relational inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8513–8522, 2020. 2

[5] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2255–2264, 2018. 2

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1026–1034. IEEE Computer Society, 2015. 1

[7] Yue Hu, Siheng Chen, Ya Zhang, and Xiao Gu. Collaborative motion prediction via neural motion message passing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6319–6328, 2020. 1, 2

[8] Yingfan Huang, Huikun Bi, Zhaoxin Li, Tianlu Mao, and Zhaoqi Wang. Stgat: Modeling spatial-temporal interactions for human trajectory prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6272–6281, 2019. 2

[9] Jiachen Li, Fan Yang, Masayoshi Tomizuka, and Chiho Choi. Evolvegraph: Multi-agent trajectory prediction with dynamic relational reasoning. *Proceedings of the Neural Information Processing Systems (NeurIPS)*, 2020. 2

[10] Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon. It is not the journey but the destination: Endpoint conditioned trajectory prediction. In *European Conference on Computer Vision*, pages 759–776. Springer, 2020. 2

[11] Francesco Marchetti, Federico Becattini, Lorenzo Seidenari, and Alberto Del Bimbo. Mantra: Memory augmented networks for multiple trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7143–7152, 2020. 2

[12] Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. 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*, pages 14424–14432, 2020. 2

[13] Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese. 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*, pages 1349–1358, 2019. 2

[14] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Multi-agent generative trajectory forecasting with heterogeneous data for control. 2020. 2

[15] Cunjun Yu, Xiao Ma, Jiawei Ren, Haiyu Zhao, and Shuai Yi. Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In *European Conference on Computer Vision*, pages 507–523. Springer, 2020. 2

[16] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. *arXiv preprint arXiv:2103.14023*, 2021. 2