Appendix

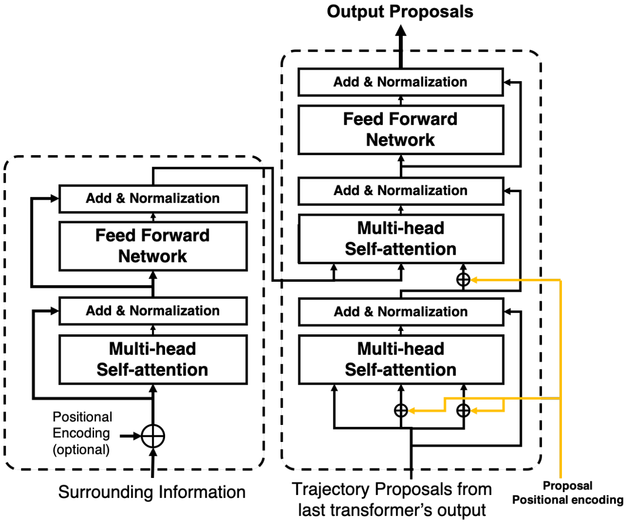
Output Proposals
Add & Normalization
Feed Forward Network
Add & Normalization
Multi-head Self-attention
Add & Normalization
Multi-head Self-attention
Add & Normalization
Feed Forward Network
Add & Normalization
Multi-head Self-attention
Positional Encoding (optional)
Surrounding Information
Trajectory Proposals from last transformer's output
Proposal Positional encoding

Figure 1. The detailed structure of the transformer module.

A. Implementation Details

Following previous transformer-based approaches [1, 4], we utilize AdamW[3] as the optimizer, with the initial learning rate, weight decay and gradient max norm set to 1×10^{-3}, 1×10^{-4} and 0.1 respectively. All parameters in mmTransformer are initialized using Xavier initialization [2]. All transformer modules in mmTransformer contain 128 hidden units. Each transformer module has two encoder layers and two decoder layers, except for the decoder of social constructor, which contains four layers. The map information in our implementation covers the $65m \times 65m$ local region, centered at the position of target agent at the last observation point. Besides, the heading direction of the target agent at last observation point is aligned to $+y$ axis, as mentioned before. To enhance the robustness of the model, we further conduct data augmentation by flipping the trajectories horizontally and randomly masking the trajectories at the first ten time steps.

B. Detailed Architecture

The detailed architecture of the transformer used in mmTransformer is visualized in Fig 1. Surrounding information is passed to the encoder to derive memory of the contextual features. Besides, spatial positional encodings are added to the queries and keys at each multi-head self-attention layer. Then, the decoder receives proposals (randomly initialized), positional encoding of proposals, as well as encoder memory, and produces the refined trajectory proposals through multiple multi-head self-attention and decoder-encoder attention layers. It is noted that the first self-attention layer in the first decoder layer of motion extractor can be skipped.

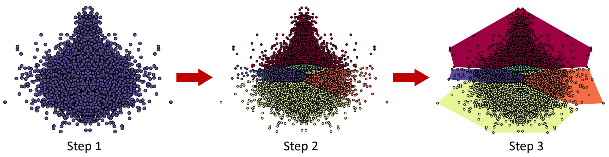

Figure 2. Visualization of the partition procedure.

C. Classification Loss

The idea is to encourage the trajectory proposals which are assigned to the ground truth region to have higher scores. Specifically, each logit in loss term is the sum of the scores belonging to the corresponding region

$$P = \{p_i \in \mathbb{R} : p_i = \sum_{j=i}^{i+N} c_j\}, \tag{1}$$

where $i \in \{1, N+1, 2N+1, \ldots, (M-1)N+1\}$, then we apply the cross entropy loss to calculate the penalty as

$$\mathcal{L}_{\mathrm{cls}} = -\sum_{\mathrm{i}=1}^{\mathrm{M}} \delta(\mathrm{i} - \mathrm{gt}) \log \mathrm{p}_{\mathrm{i}}, \tag{2}$$

where the gt is the ground truth region index and δ is a indicator function. The auxiliary loss benefits to the generalization and convergence of our model.

D. The Procedure of Partition Algorithm

Let's take K-means as a example to described partition procedure. Step1: Extract all normalized GT trajectory endpoints, using normalization described in line 435 of our paper. Step2: Apply constrained K-means [5] to divide these samples into M clusters equally. Step3: Find the vertices of each region with convex hull algorithm; gather these vertices to form the regions. The procedure is visualized in Fig 2

E. Inference

During the inference stage, we utilize NMS algorithm to filter duplicated trajectories. The detail of NMS algorithm goes as follow: we first sort the predicted trajectories according to their confidence scores in descending order, and then pick them greedily. Specifically, we set a threshold and exclude trajectories that are close to any of the selected trajectories. We keep repeating above two steps until collecting sufficient predicted trajectories.

References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *arXiv preprint arXiv:2005.12872*, 2020. 1

[2] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010. 1

[3] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 1

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 1

[5] Kiri Wagstaff, Claire Cardie, Seth Rogers, Stefan Schrödl, et al. Constrained k-means clustering with background knowledge. In *Icml*, volume 1, pages 577–584, 2001. 1