Supplemental Materials for "MSN: Multi-Style Network for Trajectory Prediction"

I. Transformer Details

We employ the Transformer [1] as the backbone to encode trajectories and the scene context in each of the two sub-networks. Several trajectory prediction methods (like [2], [3]) have already tried to employ Transformers as their backbones. Experimental results have shown excellent improvements. Transformers can be regarded as a kind of sequence-to-sequence (seq2seq) model [1]. Unlike other recurrent neural networks, the critical idea of Transformers is to use attention layers instead of recurrent cells. With the multi-head self-attention layers [1], the long-distance items in the sequence could directly affect each other without passing through many recurrent steps or convolutional layers. In other words, Transformers can better learn the long-distance dependencies in the sequence while reducing the additional overhead caused by the recurrent calculation. The Transformer we used in the proposed MSN has two main parts, the Transformer Encoder and the Transformer Decoder. Both these two components are made up of several attention layers.

A. Attention Layers

Following definitions in [1], each layer's multi-head dot product attention with H heads is calculated as:

$$\begin{aligned}\text{Attention}(\boldsymbol{q},\boldsymbol{k},\boldsymbol{v}) &= \text{softmax}\left(\frac{\boldsymbol{q}\boldsymbol{k}^T}{\sqrt{d}}\right)\boldsymbol{v},\\ \text{MultiHead}(\boldsymbol{q},\boldsymbol{k},\boldsymbol{v}) &= \text{fc}\left(\text{concat}(\{\text{Attention}_i(\boldsymbol{q},\boldsymbol{k},\boldsymbol{v})\}_{i=1}^{H})\right).\end{aligned} \tag{1}$$

In the above equation, fc() denotes one fully connected layer that concatenates all heads' outputs. Query matrix $\boldsymbol{q}$, key matrix $\boldsymbol{k}$, and value matrix $\boldsymbol{v}$, are the three inputs. Each attention layer also contains an MLP (denoted as MLP_a) to extract the attention features further. It contains two fully connected layers. ReLU activations are applied in the first layer. Formally,

$$\boldsymbol{f}_o = \text{ATT}(\boldsymbol{q},\boldsymbol{k},\boldsymbol{v}) = \text{MLP}_a(\text{MultiHead}(\boldsymbol{q},\boldsymbol{k},\boldsymbol{v})), \tag{2}$$

where $\boldsymbol{q},\boldsymbol{k},\boldsymbol{v}$ are the attention layer's inputs, and $\boldsymbol{f}_o$ represents the layer output.

B. Transformer Encoder

The transformer encoder comprises several encoder layers, and each encoder layer contains an attention layer and an encoder MLP (MLP_e). Residual connections and normalization layers are applied to prevent the network from overfitting. Let $\boldsymbol{h}^{(l+1)}$ denote the output of l-th encoder layer, and $\boldsymbol{h}^{(0)}$ denote the encoder's initial input. For l-th encoder layer, we have

$$\begin{aligned}\boldsymbol{a}^{(l)} &= \text{ATT}(\boldsymbol{h}^{(l)},\boldsymbol{h}^{(l)},\boldsymbol{h}^{(l)}) + \boldsymbol{h}^{(l)},\\ \boldsymbol{a}_n^{(l)} &= \text{Normalization}(\boldsymbol{a}^{(l)}),\\ \boldsymbol{c}^{(l)} &= \text{MLP}_e(\boldsymbol{a}_n^{(l)}) + \boldsymbol{a}_n^{(l)},\\ \boldsymbol{h}^{(l+1)} &= \text{Normalization}(\boldsymbol{c}^{(l)}).\end{aligned} \tag{3}$$

C. Transformer Decoder

Like the Transformer encoder, the Transformer decoder comprises several decoder layers, and each is stacked with two different attention layers. The first attention layer focuses on the essential parts in the encoder's outputs $\boldsymbol{h}_e$ queried by the decoder's input $\boldsymbol{X}$. The second is the same self-attention layer as in the encoder. Similar to Equation 3, we have:

$$\begin{aligned}\boldsymbol{a}^{(l)} &= \text{ATT}(\boldsymbol{h}^{(l)},\boldsymbol{h}^{(l)},\boldsymbol{h}^{(l)}) + \boldsymbol{h}^{(l)},\\ \boldsymbol{a}_n^{(l)} &= \text{Normalization}(\boldsymbol{a}^{(l)}),\\ \boldsymbol{a}_2^{(l)} &= \text{ATT}(\boldsymbol{h}_e,\boldsymbol{h}^{(l)},\boldsymbol{h}^{(l)}) + \boldsymbol{h}^{(l)},\\ \boldsymbol{a}_{2n}^{(l)} &= \text{Normalization}(\boldsymbol{a}_2^{(l)})\\ \boldsymbol{c}^{(l)} &= \text{MLP}_d(\boldsymbol{a}_{2n}^{(l)}) + \boldsymbol{a}_{2n}^{(l)},\\ \boldsymbol{h}^{(l+1)} &= \text{Normalization}(\boldsymbol{c}^{(l)}).\end{aligned} \tag{4}$$

D. Positional Encoding

Before inputting agents' representations or trajectories into the Transformer, we add the positional coding to inform each timestep's relative position in the sequence. The position coding $\boldsymbol{f}_e^t$ at step t $(1 \le t \le t_h)$ is obtained by:

$$\begin{aligned}\boldsymbol{f}_e^t &= \left(f_{e0}^t, ..., f_{ei}^t, ..., f_{ed-1}^t\right) \in \mathbb{R}^d,\\ \text{where } f_{ei}^t &= \begin{cases}\sin\left(t/10000^{d/i}\right), & i \text{ is even;}\\ \cos\left(t/10000^{d/(i-1)}\right), & i \text{ is odd.}\end{cases}\end{aligned} \tag{5}$$

Then, we have the positional coding matrix f_e that describes t_h steps of sequences:

$$\boldsymbol{f}_e = (\boldsymbol{f}_e^1, \boldsymbol{f}_e^2, ..., \boldsymbol{f}_e^{t_h})^T \in \mathbb{R}^{t_h \times d}. \tag{6}$$

The final Transformer input X_T is the addition of the original input X and the positional coding matrix f_e. Formally,

$$\boldsymbol{X}_T = \boldsymbol{X} + \boldsymbol{f}_e \in \mathbb{R}^{t_h \times d}. \tag{7}$$

E. Transformer Details

We employ $L = 4$ layers of encoder-decoder structure with $H = 8$ attention heads in each Transformer-based sub-networks. The MLP_e and the MLP_d have the same shape.

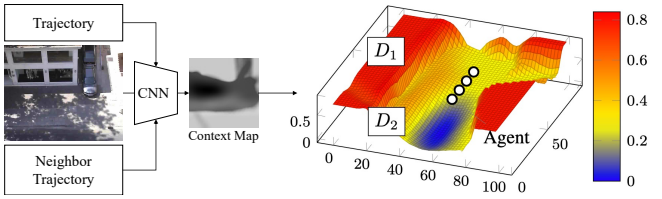


Fig. 1. Visualization of someone's context map. A lower semantic label (colored in blue) means that the place has a higher possibility for that agent to move towards, while a higher label (colored in red) means lower possibility.

Both of them consist of two fully connected layers. The first layer has 512 output units with the ReLU activation, and the second layer has 128 but does not use any activations. The output dimensions of fully connected layers used in multi-head attention layers are set to d = 128.

II. Interaction Representation

In the proposed MSN, we use the context map [4] to describe agents' interaction details through a two-dimensional energy map, which is inferred from scene images and their neighbors' trajectories. Considering both social and scene interactions, it provides potential attraction or repulsion areas for the target agent. Although the focus of this manuscript is not on the modeling of interactive behaviors, we still show the effect of this part, for it is a crucial research part of the trajectory prediction task. We visualize one agent's context map in zara1 dataset in Fig. 1. The target moves from about $(p_{x0}, p_{y0}) = (50, 80)$ to the current $(p_{x1}, p_{y1}) = (50, 50)$ during observation period.

- **Scene constraints**: The scene's physical constraints indicate where they could not move. The context map gives a higher enough value (≈ 1) to show these areas. For example, the higher semantic label in the area $D_1 = \{(p_x, p_y)|p_x \leq 20\}$ reminds pedestrians not to enter the road at the bottom of the zara1 scene. Similarly, another high-value area $\{(p_x, p_y)|p_x \geq 80, p_y \leq 50\}$ reminds pedestrians not to enter the ZARA shop building except the door. It illustrates the ability of the context map to model the scene's physical constraints.
- **Social interaction**: Social interaction refers to the interactive behaviors among agents, such as avoiding and following. The context map does not describe the interaction behavior directly but provides lower semantic labels to areas conducive to agents' passage and higher semantic labels that are not. For example, the high-value area $D_2 = \{(p_x, p_y)|20 \leq p_x \leq 40, p_y \leq 80\}$ shows another group of agents' possible walking place in the future who walk towards the target. The target agent will naturally avoid this area when planning his future activities. Context maps follow the lowest semantic label strategy to describe agents' behaviors. A place with a lower semantic label means that the target agent is more likely to pass through. Thus, it could show agents' social behaviors in the 2D map directly.

III. Classification Strategy

In the proposed MSN, we need to first determine whether different trajectories belong to the same behavioral style mainly based on agents' end-points (or called destinations). We do not explain the rationale for this classification approach due to the space limitation of the manuscript. In this section, we will further explore the rationale for using end-points as their behavioral style classification, as well as further explanations of behavioral style classification strategies.

The classification method can be written together as:

$$\begin{aligned} &\text{Category}(\boldsymbol{d}|\mathcal{D}) = c_s = s, \\ &\text{where} \quad s = \underset{i=1,2,...,K_c}{\arg\min} \|\boldsymbol{D}_i - \boldsymbol{d}\|. \end{aligned} \tag{8}$$

The end-points are used as a very important basis for the classification of behavioral styles. However, in this case, there may be situations where different trajectories have completely different directions although they have the same end-points.

As we mentioned in this manuscript, agents' behavioral preferences tend to be continuously distributed, and it is also more difficult to directly classify these preferences. The main concern that the manuscript wishes to explore is the multi-behavioral style property of the agent. With this one constraint, we should fully ensure that the model preserves the different behavioral style features in the trajectory. Moreover, often more constraints mean fewer possibilities. Therefore, we want to determine each category by a minimal classification criterion so that each category can "cover" more trajectories.

In contrast, if a more strict category judgment approach is used, then each category will cover fewer trajectories, thus requiring a larger number of categories (*i.e.*, a larger K_c) to be set to capture more differentiated behavioral preferences. On the one hand, this has higher data requirements and may make the network difficult to train. On the other hand, applying the prediction model to more specific scenarios is difficult because the trajectory preferences and interaction behaviors vary significantly in different scenarios. The strict categorization restriction will also lead to a decrease in the generalization ability of the model to more prediction scenarios.

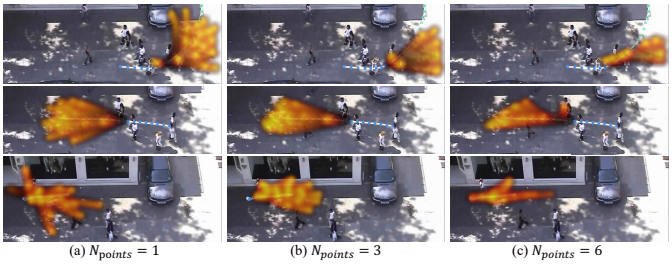
In addition, we have further investigated and explored the problem of classification strategies. Specifically, we investigate the trajectory prediction style of the network through a more rigorous classification strategy. In detail, we expand the end-point $\boldsymbol{d}$ into the trajectory $\boldsymbol{j}$, and the destinations $\{\boldsymbol{D}_i\}_i$ in the set of 2-tuples $\mathcal{D} = \{(\boldsymbol{D}_i, c_i)\}_i$ into trajectory keypoints' proposals $\{\boldsymbol{D}_{key_i}\}_i$ in the new set of 2-tuples $\mathcal{D}_{key} = \{(\boldsymbol{D}_{key_i}, c_i)\}_i$. Given a set of indexes for the temporal keypoints

$$\mathcal{T}_{key} = \left\{t^1_{key}, t^2_{key}, ..., t^{N_{key}}_{key}\right\}, \tag{9}$$

we have the expanded classification function:

$$\begin{aligned} &\text{Category}(\boldsymbol{j}|\mathcal{D}_{key}) = c_s = s, \\ &\text{where} \quad s = \underset{i=1,2,...,K_c}{\arg\min} \sum_{t \in \mathcal{T}_{key}} \|\boldsymbol{D}_{key_{it}} - \boldsymbol{j}_t\|. \end{aligned} \tag{10}$$

When the set of temporal keypoints contains only the last moment of the prediction period (*i.e.*, $\mathcal{T}_{key} = \{t_h + t_f\}$), the above Equation 10 will degenerate to Equation 8.



(a) $N_{points} = 1$ (b) $N_{points} = 3$ (c) $N_{points} = 6$

Fig. 2. The number of trajectory points and prediction styles. We show the visualized predictions with different N_{points} configurations.

As shown in Fig. 2, we select 1, 3, and 6 keypoints (including the end-points) instead of the classification strategy used like Equation 8, the network exhibits a completely different prediction style. The prediction results in the figure show that when there are more constraints on the classification (*i.e.*, 6 keypoints), the predicted results will appear more cautious and lack possibilities and multi-style properties. Considering that the primary concern of this manuscript is still the multi-style prediction, we choose only one end-point (destination) as the reference for classification.

REFERENCES

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[2] C. Yu, X. Ma, J. Ren, H. Zhao, and S. Yi, "Spatio-temporal graph transformer networks for pedestrian trajectory prediction," in *European Conference on Computer Vision*. Springer, 2020, pp. 507–523.

[3] F. Giuliari, I. Hasan, M. Cristani, and F. Galasso, "Transformer networks for trajectory forecasting," pp. 10 335–10 342, 2021.

[4] B. Xia, C. Wong, Q. Peng, W. Yuan, and X. You, "Cscnet: Contextual semantic consistency network for trajectory prediction in crowded spaces," *Pattern Recognition*, p. 108552, 2022.