Supplementary Material of DySeT: a Dynamic Masked Self-distillation Approach for Robust Trajectory Prediction

Mozhgan Pourkeshavarz, Junrui Zhang, and Amir Rasouli

Noah's Ark Lab, Huawei, Canada
firstname.lastname@huawei.com

Algorithm 1: Masked Self-distillation

Input : Embedding of visible tokens $\mathcal{I}_v$, Embedding of all tokens $\mathcal{I}$, Smoothing parameter α, Original MAE as the student $S(.;\theta)$, num of tokens N, num of visible tokens n_v

Output : Masked self-distillation loss $\mathcal{L}_{\text{MSD}}$

a. Initiate EMA teacher $T(.;\bar{\theta})$

$\bar{\theta}_t = \alpha\bar{\theta}_{t-1} + (1-\alpha)\theta_t, \qquad \bar{\theta}_0 = \theta$

b. Obtain *distillation target* for all tokens

$\sim \bar{R}_i|_{i=1}^{N} \leftarrow T(\mathcal{I};\bar{\theta}_t)$

c. Obtain *distillation prediction*

`// obtain feature vector for visible tokens`

$\sim R'_i|_{i=1}^{n_v} \leftarrow S(\mathcal{I}_v;\theta_t)$

`// merge with a shared learnable feature vector`

$\sim R'_i|_{i=1}^{N} \leftarrow \sim R'_i|_{i=1}^{n_v} + F'_i|_{i=1}^{N-n_v}$

d. Pass through an *online quantizer*

`// Distillation target`

$\bar{R}_i|_{i=1}^{N} \leftarrow h(\sim \bar{R}_i|_{i=1}^{N})$

`// Distillation prediction`

$R'_i|_{i=1}^{N} \leftarrow h(\sim R'_i|_{i=1}^{N})$

e. Calculate masked-self distillation loss

$\mathcal{L}_{\text{MSD}} = \frac{1}{N}\sum_{i\in\mathcal{I}} -\bar{R}_i^T \log R'_i$

return *Self-distillation loss* $\mathcal{L}_{MSD}$

1 Masked Self-distillation

As mentioned in the paper, the proposed masked self-distillation method is designed to distill representation from a fully visible scene to its masked counterpart's predicted representation. Here, we summarize the steps of the proposed method as described in Algorithm 1.

In this method, we first initiate the EMA teacher that acts as the model's own temporal ensemble (a). We then proceed to obtain the distillation target using the entire tokens and the EMA teacher (b). To obtain distillation prediction, we start by feeding the sampled visible tokens (from the sampler) into the student model, subsequently integrating these encodings with shared, learnable

Table 1: Analyzing robustness against missing data and road perturbation attacks on Argoverse2 val set. Org/Pert. stand for Original/Perturbed.

Attack	**Model**	**Org/Pert.**	**abs**(Δ)(↓)	**PSR**(↑)
miss.-data	FMAE	1.408/1.584	0.148±0.03	89%
	Ours	1.278/1.381	0.084±0.01	93%
road-pert.	FMAE	1.408/1.643	0.231±0.01	84%
	Ours	1.278/1.414	0.126±0.02	90%

feature vectors representative of the masked tokens. (c). To distill the intermediate representations, the output features are passed through an online quantizer, establishing a soft codewords distribution (d). Finally, we calculate the masked self-distillation loss by minimizing the cross entropy between the target and predicted representations (e).

2 Impact of Masking Ratio

With *increasing masking ratio*, the network has *fewer visible tokens* available during the reconstruction phase. Hence, the importance of *visible* token selection becomes more apparent at *high masking ratios*, where each token is crucial. As shown in Fig. 3 in the paper: 1) Our model's performance changes smoothly and remains stable when increasing the masking ratio to a more restrictive value (from 0.5 to 0.8). Whereas, random sampling causes more performance fluctuation over the same range due to its inherent unpredictability; 2) The larger gap at higher masking ratios, where visible tokens are crucial, underscores the superiority of our proposed dynamic masking compared to random selection; 3) At each ratio, the proposed dynamic sampling surpasses random masking on both metrics. Note that the only difference between the two models in the figure is the token selection method and both use the self-distillation (SD) objective.

3 Robust Trajectory Prediction

Despite advancement of trajectory prediction models, recent evidence [1–3, 5, 6] shows they are susceptible to slight distribution shifts, a result of overfitting. This vulnerability is commonly referred to as *lack of robustness* against adversarial attacks [2, 6] or unseen scenes (e.g., cities not [1]. Here, robustness can also serve as a measure of generalizability. We employ self-supervised learning, specifically MAE with an improved masking strategy, to enhance representation learning during the pre-training phase. This is motivated by studies in the ML community [4], indicating that unsupervised pre-training, before fine-tuning on downstream tasks, enhances generalization by learning more meaningful representations. Based on this definition, we use perturbation resistance score (PRS) to measure the robustness of the models in some specific scenarios. As requested, we conducted two additional attack studies (see Table 1), including 1) Missing data, where agents' observations are randomly masked with a probability of 0.5

and 2) Road perturbation [1], which involves altering road layouts while remaining realistic using ripple-road transformation (see [1]). We report on minFDE to showing the attack's effect on agents' final goal.

4 Baseline Details

We established our baseline model similar to Forecast-MAE [3]. In this section, we provide additional implementation details.

Embeddings. As mentioned in the paper, we use vectorized representations and employ a tokenizer that models agents' trajectories and lane segments as polylines, serving as *tokens* in our framework. Specifically, the embedding step can be formulated as follows.

$$\begin{aligned} &E_h = \mathrm{FPN}\left(X\right), E_f = \mathrm{FPN}\left(Y\right), E_{h,f} \in \mathbb{R}^{n \times c} \\ &E_l = \mathrm{PointNet}(L), E_l \in \mathbb{R}^{m \times c}, \end{aligned} \tag{1}$$

where E_h, E_f and E_l are history (X), future (Y) and lane (L) token embeddings. n and m represent the number of agents and the number of lane segments within a certain radius of the target agent. c is the embedding dimension. Additionally, semantic attributes such as agent categories (e.g., vehicle, pedestrian, cyclist) and lane types are initialized as learnable embeddings and incorporated into the embedded tokens. Since the coordinates of agents and lane features are normalized, it is essential to include global position information in the tokens. The position embedding (PE) is implemented using a simple two-layer MLP, as detailed below.

$$\mathrm{PE} = \mathrm{MLP}\left([x, y, \cos(\theta), \sin(\theta]\right), \mathrm{PE} \in \mathbb{R}^c \tag{2}$$

where (x, y, θ) represents the latest observed pose of agents or the geometric center pose for lane polylines. The PE is added to the tokens before being processed by the autoencoder.

Reconstruction Loss. The prediction heads generate the normalized 2-dimensional coordinates for the historical and future trajectories $P_{h/f}$, as well as for the lane polylines, P_l,

$$\begin{aligned} P_h &= \mathrm{PredictionHead}\left(R'_h\right), P_h \in \mathbb{R}^{\alpha N \times O \times 2}, \\ P_f &= \mathrm{PredictionHead}\left(R'_f\right), P_f \in \mathbb{R}^{\alpha N \times T \times 2}, \\ P_l &= \mathrm{PredictionHead}\left(R'_l\right), P_l \in \mathbb{R}^{\alpha M \times P \times 2}, \end{aligned} \tag{3}$$

where α is the masking ratio. O and T are trajectory observation length and future horizon, and P stands for the number of points of each polyline. We utilize loss for trajectory reconstruction $\mathcal{L}_t$ and mean squared error (MSE) loss for lane polyline reconstruction $\mathcal{L}_l$. The final loss is formulated as bellow.

$$\mathcal{L}_R(.;\theta) = w_t \mathcal{L}_t(.;\theta) + w_l \mathcal{L}_l(.;\theta) \tag{4}$$

where w_t and w_l represent the corresponding loss weights. We practically set both weights to 0.5.

References

1. Bahari, M., Saadatnejad, S., Rahimi, A., Shaverdikondori, M., Shahidzadeh, A.H., Moosavi-Dezfooli, S.M., Alahi, A.: Vehicle trajectory prediction works, but not everywhere. In: CVPR (2022) 2, 3
2. Cao, Y., Xiao, C., Anandkumar, A., Xu, D., Pavone, M.: Advdo: Realistic adversarial attacks for trajectory prediction. In: ECCV (2022) 2
3. Cheng, J., Mei, X., Liu, M.: Forecast-mae: Self-supervised pre-training for motion forecasting with masked autoencoders. In: ICCV (2023) 2, 3
4. Hendrycks, D., Mazeika, M., Kadavath, S., Song, D.: Using self-supervised learning can improve model robustness and uncertainty. In: NeurIPS (2019) 2
5. Pourkeshavarz, M., Sabokrou, M., Rasouli, A.: Adversarial backdoor attack by naturalistic data poisoning on trajectory prediction in autonomous driving. In: CVPR (2024) 2
6. Zhang, Q., Hu, S., Sun, J., Chen, Q.A., Mao, Z.M.: On adversarial robustness of trajectory prediction for autonomous vehicles. In: CVPR (2022) 2