Tra-MoE: Learning Trajectory Prediction Model from Multiple Domains for Adaptive Policy Conditioning Supplementary Material

Jiange Yang [1,2] Haoyi Zhu[2,3] Yating Wang[2,4] Gangshan Wu[1] Tong He[2] Limin Wang[1,2*]
[1]State Key Laboratory for Novel Software Technology, Nanjing University, China
[2]Shanghai AI Laboratory, [3]University of Science and Technology of China, [4]Tongji University
jiangeyang.jgy@gmail.com, {gswu,lmwang}@nju.edu.cn,
{zhuhaoyi, wangyating, hetong}@pjlab.org.cn
https://github.com/MCG-NJU/Tra-MoE

A. Appendix

A.1. Limitations and Discussion

In our work, we firstly demonstrate the effectiveness of using sparsely-gated Mixture-of-Experts (MoE) for learning trajectory prediction model from large-scale out-of-domain data, while conducting a thorough experimental study of its training techniques. To the best of our knowledge, this is one of the first attempts of using sparse MoE architecture for learning from multiple domains data to ensure better parameter cooperation and specialization in robotics. We hope it can serve as a strong baseline and facilitate further research in this direction. While we are encouraged by the strong results across a wide range of simulated and real-world experiments, some limitations and future works still remain. On the one hand, the trajectory prediction model learning can further integrate larger-scale human and robot video data. On the other hand, our adaptive policy condition technique can also be extended to other visual prompts.

A.2. The stability of LIBERO experiment results

In our LIBERO simulation experiment results, the training process and results are completely reproducible. Due to the slight randomness in the simulation rendering, the downstream LIBERO evaluation results cannot be fully reproduced. Therefore, we further run the key experiments three times and report the mean and standard deviation in Tab. 1. The results indicate that the improvements brought by our Tra-MoE and adaptive policy conditioning technique are **significant and stable**.

A.3. The comparison of trajectory prediction model

In our work, we find that CoTracker [1] can provide highly accurate labels. On the one hand, we can directly use the ground truth provided by CoTracker to calculate the MSE error with our prediction results to measure the performance of trajectory prediction. On the other hand, considering that trajectory prediction task is multi-modal, we also need directly visualize some samples for prediction performance analysis. *In our experiments, we measure trajectory prediction performance using the MSE loss on validation set of four evaluation suites, as well as qualitatively and quantitatively find that the success rate of downstream manipulation tasks is generally positively correlated with the performance of the trajectory model, as shown in Fig. 1. Therefore, we report the average success rate of downstream manipulation tasks as our metric.*

We also fairly compare the Tra-baseline (the track transformer of ATM) and our Tra-MoE, which are both trained with a mixture of LIBERO and RLbench video data. Their respective MSE error on the LIBERO validation set are **0.0000034773** and **0.0000012449**. Additionally, we further randomly select some samples for visualization, as shown in Fig. 2. The visualization results also indicate that Tra-MoE is significantly more accurate than Tra-baseline. Tra-MoE is generally able to predict accurate trajectories, whereas Tra-baseline occasionally predict stationary or even opposite-direction movements, leading to poorer downstream policy. This is primarily attributed to optimization conflicts arising from multiple domains data joint training. Conversely, our Tra-MoE, with its superior parameter cooperation and specialization, can better handle such situations.

A.4. The Simulation environments details

In this section, we further elaborate on the details of our simulation experiments. The training hyperparameters for the trajectory prediction model and the trajectory-guided policy are shown in Tab. 2 and Tab. 3, respectively. We use the same training hyperparameters to ensure a fair comparison between our Tra-MoE and Tra-baseline. When we train the trajectory model integrating RLbench data, we report the

*Corresponding author.

	Spatial	Goal	Object	Long	Avg.	
Tra-baseline	49.5	67.0	56.5	35.0	52.0	**52.0±0.1**
	51.5	69.0	58.5	29.0	52.0	
	51.5	67.0	58.5	31.5	52.1	
Tra-MoE	62.5	81.0	73.5	28.5	61.4	**61.2±0.4**
	62.5	80.5	72.5	27.5	60.8	
	67.0	79.5	72.5	27.0	61.5	
Tra-MoE + Aaptive Mask	72.5	74.0	86.5	34.5	66.8	**67.7±0.8**
	73.0	78.0	86.5	35.5	68.3	
	72.5	75.5	87.5	36.0	67.9	

Table 1. We rerun the key experiments three times and report the mean and standard deviation.

Hyperparameters	In-domain data	Out-of-domain data
Number of videos	400	2200 / 2660
Number of tasks	40	130 / 222
Epoch	1000	300
Batch size	2048	
Optimizer	AdamW	
Learning rate	1e-4	
Weight decay	1e-4	
LR scheduler	Cosine	
LR warm-up	5	
Clip grad	10	
Point sampling	Variance filtering	
Number of points	32	
Track length	16	
Augmentation	ColorJitter, RandomShift	
dropout	0.2	
depth	8	
dimension	384	

Table 2. Hyperparameters of our trajectory model training.

specific tasks used in Tab. 4. For the majority of the hyperparameters, we inherit the settings from ATM [3]. Additionally, when we extend Tra-baseline in depth, the depth is increased from 8 to 14; when we extend Tra-baseline in width, the dimension is increased from 384 to 512. Finally, following the original LIBERO [2] setup, we perform 20 trials for each task evaluation, ensuring a total of 800 (20×40) trials for each model evaluation.

A.5. The Real-World environments details

In this section, we further elaborate on the details of our real-world experiments. Specifically, we use two leader arms to perform teleoperation for follower arms data collection, where 50 demonstrations are collected for each task for trajectory model and policy training. For our real-world evaluations, we conduct 20 trials for each task, while ensuring, to the extent possible, that the object poses in the training set differ from those in the test set. For the relevant training hyperparameters, we maintain consistency with the simulation experiments.

References

[1] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. *arXiv preprint arXiv:2307.07635*, 2023. 1, 4

[2] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[3] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023. 2

Hyperparameters	Policy
Number of demonstrations	100
epoch	120
batch size	384
optimizer	AdamW
learning rate	5e-4
weight decay	1e-4
lr scheduler	Cosine
lr warm up	0
clip grad	100
point sampling	grid
number of points	32
track length	16
frame stack	10
augmentation	ColorJitter,RandomShift
dropout	0.1

Table 3. Hyperparameters of our trajectory-guided policy training.

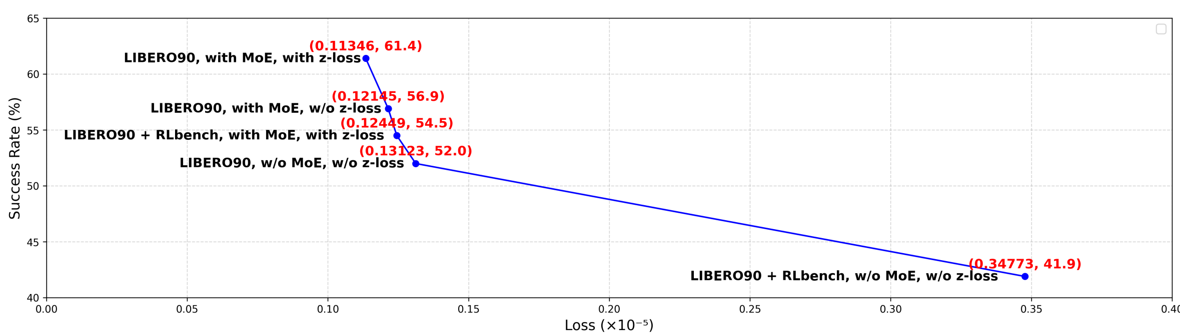

Figure 1. The quantitative relationship between downstream policy success rate and trajectory model performance.

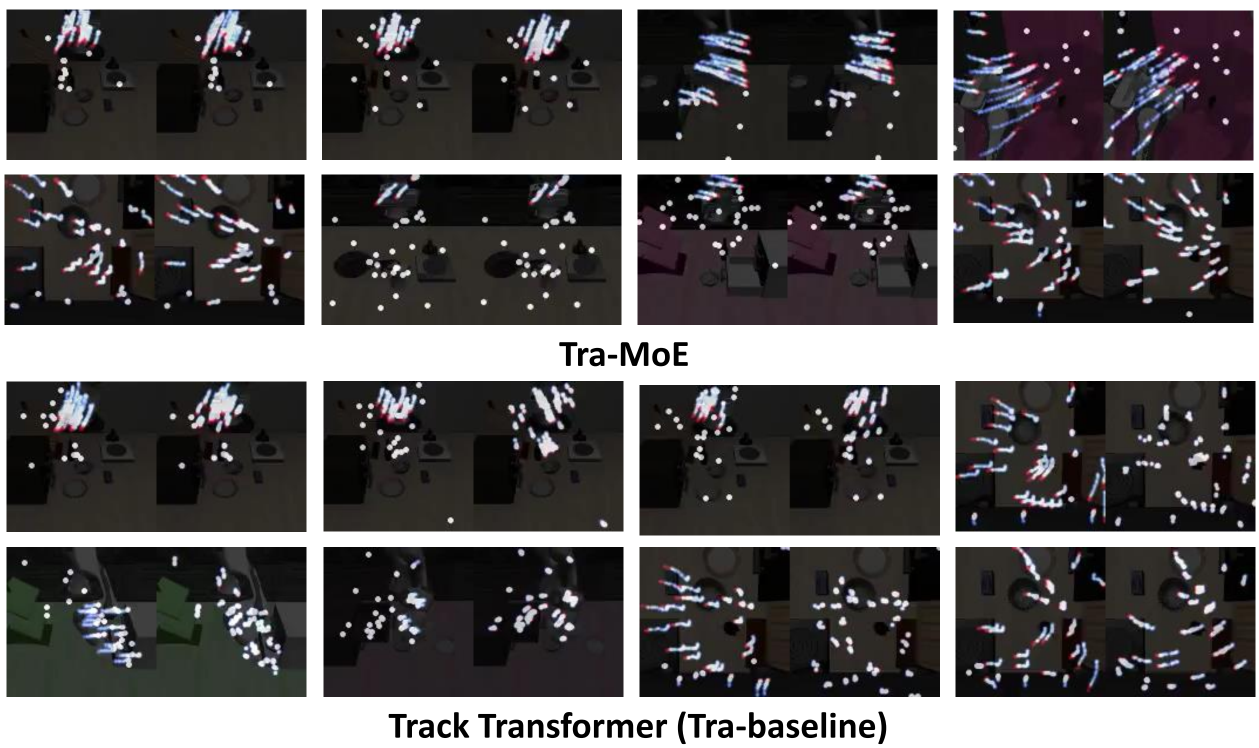

Figure 2. The trajectory prediction results visualization. **Left:** the ground truth by CoTracker [1]. **Right:** the prediction result.

beat the buzz	lamp off
block pyramid	place hanger on rack
put umbrella in umbrella stand	take money out safe
place shape in shape sorter	take umbrella out of umbrella stand
reach and drag	take tray out of oven
lamp on	push button
change channel	take toilet roll off stand
light bulb in	setup checkers
play jenga	close door
reach target	open door
take plate off colored dish rack	meat on grill
change clock	close drawer
light bulb out	stack cups
plug charger in power supply	take usb out of computer
remove cups	slide cabinet open and place cups
take shoes out of box	slide block to target
close box	put bottle in fridge
meat off grill	toilet seat down
pour from cup to cup	put groceries in cupboard
scoop with spatula	toilet seat up
press switch	put item in drawer
screw nail	stack blocks
move hanger	close grill
close fridge	open microwave
open box	put books on bookshelf
setup chess	put knife in knife block
close jar	empty container
open drawer	turn tap
close laptop lid	open grill
open fridge	close microwave
solve puzzle	turn oven on
tv on	wipe desk
put knife on chopping board	hockey
stack wine	take cup out from cabinet
unplug charger	put rubbish in bin
get ice from fridge	open wine bottle
open oven	hit ball with queue
put money in safe	weighing scales
straighten rope	sweep to dustpan
water plants	put plate in colored dish rack
hang frame on hanger	open window
phone on base	put tray in oven
put shoes in box	place cups
take frame off hanger	insert usb in computer
insert onto square peg	take item out of drawer
pick and lift put toilet roll on stand	take lid off saucepan

Table 4. The language annotations of 92 RLbench tasks.