

TOP-ERL, SUPPLEMENTARY FOR REBUTTAL

Anonymous authors

Paper under double-blind review

R REBUTTAL: ADDITIONAL DISCUSSION AND ABLATION STUDY

R.1 DISCUSSION OF INITIAL CONDITION ENFORCEMENT

The initial condition enforcement of trajectories generated by ProDMP (Li et al., 2023) is mathematically guaranteed by the definition of initial conditions in dynamic systems, as demonstrated in Figure 2(a) and the upper part of 2(b) in the **original ProDMP paper**. In Figure R1, we illustrate how TOP-ERL leverages this mechanism. The figure is based on the motion trajectory of the first degree of freedom of the robot in the box-pushing task. In the critic update, we use five segments as an example.

ProDMP, as a trajectory generator, models the trajectory as a dynamic system. In TOP-ERL, the RL policy predicts ProDMP parameters, which are used to generate a force signal applied to the dynamic system. The system evolves its state based on this force signal and the given initial conditions, such as the robot’s position and velocity at a specific time. The resulting evolution trajectory, shown as the black curve in Figure R1, can be computed in closed form and used to control the robot.

When the policy is updated and predicts a new force signal for the same task, a new action trajectory is generated, depicted as the red dashed curve, which gradually deviates from the old trajectory. However, by utilizing the dynamic system’s features, we can set the initial condition of each segment of the new trajectory to the corresponding old state. This ensures that the new action sequence used in the target computation in Eq.(7), can start from the old state, as shown across the five segments in the figure. Therefore, we matched the old state and new actions by eliminating the gap between them, as previously discussed in Section 4.3.1.

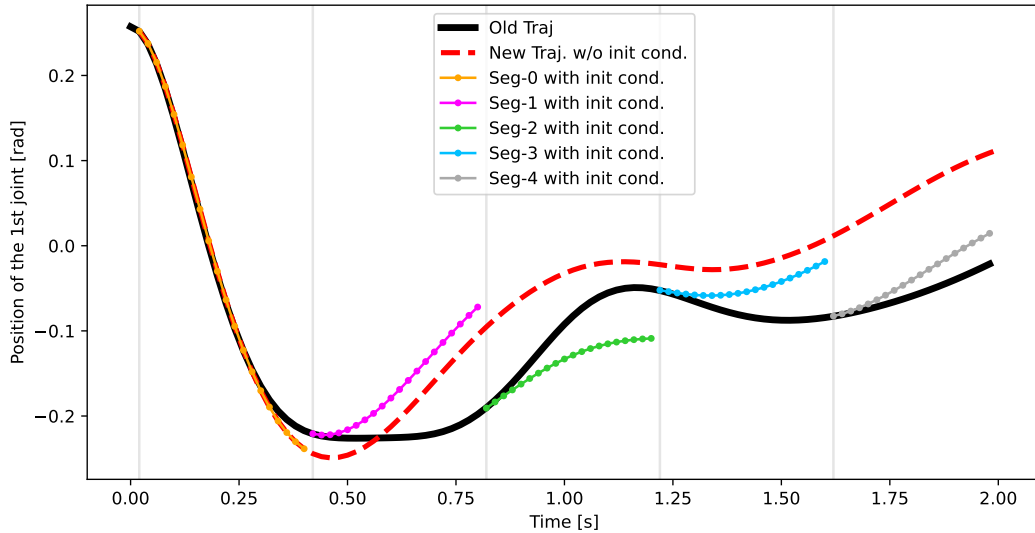


Figure R1: TOP-ERL leverages the initial condition enforcement techniques of a dynamic system to ensure that the new action trajectory starts from the corresponding old state. **These action trajectories are taken from the first DoF of the robot in the box pushing task.**

R.2 EXPLANATION OF POLICY’S TRAINING ADDITIONAL TO SECTION 4.4

We utilize the transformer critic to guide the training of our policy, using the reparameterization trick similar to that introduced by SAC (Haarnoja et al., 2018). Given a task initial state s , the current policy $\pi_\theta(w|s) \sim \mathcal{N}(w|\mu_w, \Sigma_w)$ predicts the Gaussian parameters of the MP’s and samples \tilde{w} as follows:

$$\text{Sample MP parameter vector: } \tilde{w} = \mu_w + L_w \epsilon, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \quad (\text{R.1})$$

Here, L_w is the Cholesky decomposition of the covariance matrix Σ_w , where $L_w L_w^T = \Sigma_w$. This parameterization technique is commonly used for predicting full covariance Gaussian policies. The term ϵ is a Gaussian white noise vector with the same dimensionality as w . Eq. (R.1) represents the full covariance extension of the reparameterization trick typically used in RL, known as $\tilde{a} = \mu_a + \sigma_a \epsilon$, $\epsilon \sim \mathcal{N}(0, 1)$.

The sampled \tilde{w} is then used to compute the new trajectory segments. Using the techniques described in Section 4.3.1 and Section R.1, we ensure each action segment $[\tilde{a}_t^k]_{t=0:N}$ starts from the corresponding old state s_0^k . This is achieved by enforcing the initial condition, such as the robot’s position a and velocity \dot{a} at the old state. The resulting trajectory segments are computed using the linear basis function expression in Eq. (4) from Section 3.2, where the coefficients c_1 and c_2 are determined by the initial conditions and solved using Eq. (22) in Appendix B.3:

$$\text{Compute action segment: } \tilde{a}(t) = \Phi(t)^\top \tilde{w} + c_1 y_1(t) + c_2 y_2(t) \quad (4)$$

Here, $t = 0 : N$ represents the time interval from the beginning to the end of the k -th segment. Thus, the new action sequence $[\tilde{a}_t^k]_{t=0:N}$ predicted by the new policy is conditioned on both the task description and the old state. Intuitively, this can be interpreted as taking new actions for an old situation. We evaluate and maximize the value of these action segments, using their expectation as the policy’s learning objective, as shown in Eq. (9) in Section 4.4:

$$\text{SAC style Objective: } J(\theta) = \mathbb{E}_{s \sim B} \mathbb{E}_{\tilde{w} \sim \pi_\theta(\cdot|s)} \left[\frac{1}{KL} \sum_{k=1}^K \sum_{N=0}^{L-1} Q_\phi(s_0^k, [\tilde{a}_t^k]_{t=0:N}) \right]. \quad (9)$$

Since Eq.(4), (9), (22) and (R.1) are all differentiable, the policy neural network parameters θ can be trained using gradient ascent. Compared to the technique introduced in SAC, TOP-ERL adds only one additional step: computing the action sequence $[\tilde{a}_t^k]_{t=0:N}$ from the sampled MP parameter \tilde{w} .

R.3 TECHNICAL DETAILS OF TRPL

In Episodic Reinforcement Learning (ERL), the parameter space \mathcal{W} generally has a higher dimensionality than the action space \mathcal{A} , creating distinct challenges for achieving stable policy updates. Trust region methods (Schulman et al., 2015; 2017) are widely regarded as reliable techniques for ensuring convergence and stability in policy gradient algorithms.

Although methods like PPO approximate trust regions using surrogate objectives, they lack the ability to enforce trust regions precisely. To address this limitation, Otto et al. (2021) proposed trust region projection layer (TRPL), a mathematically rigorous and scalable approach for exact trust region enforcement in deep RL algorithms. Leveraging differentiable convex optimization layers (Agrawal et al., 2019), trust region projection layer (TRPL) enforces trust regions at the per-state level and has demonstrated robustness and stability in high-dimensional parameter spaces, as evidenced in methods like BBRL (Otto et al., 2022) and TCE (Li et al., 2024).

TRPL operates on the standard outputs of a Gaussian policy—the mean vector μ and covariance matrix Σ —and enforces trust regions through a state-specific projection operation. The adjusted Gaussian policy, represented by $\tilde{\mu}$ and $\tilde{\Sigma}$, serves as the foundation for subsequent computations. The dissimilarity measures for the mean and covariance, denoted as d_{mean} and d_{cov} (e.g., KL-divergence), are bounded by thresholds ϵ_μ and ϵ_Σ , respectively. The optimization problem for each state s is

expressed as:

$$\begin{aligned} \arg \min_{\tilde{\mu}_s} d_{\text{mean}}(\tilde{\mu}_s, \mu(s)), \quad \text{s.t.} \quad d_{\text{mean}}(\tilde{\mu}_s, \mu_{\text{old}}(s)) \leq \epsilon_{\mu}, \text{ and} \\ \arg \min_{\tilde{\Sigma}_s} d_{\text{cov}}(\tilde{\Sigma}_s, \Sigma(s)), \quad \text{s.t.} \quad d_{\text{cov}}(\tilde{\Sigma}_s, \Sigma_{\text{old}}(s)) \leq \epsilon_{\Sigma}. \end{aligned} \quad (\text{R.2})$$

If the unconstrained, newly predicted per-state Gaussian parameters $\mu(s)$ and $\Sigma(s)$ exceed the trust region bounds defined by ϵ_{μ} and ϵ_{Σ} , respectively, TRPL projects them back to the trust region boundary, ensuring stable update steps. In TOP-ERL, the old Gaussian parameters, $\mu_{\text{old}}(s)$ and $\Sigma_{\text{old}}(s)$, can be derived either from the behavior policy that interacted with the environment or from an exponentially moving averaged (EMA) policy, which serves as a delayed version of the current policy. This approach is analogous to the concept employed in the target critic network.

R.4 LONGER TRAINING FOR BOX PUSHING TASKS

In the original manuscript, we concluded the training of TOP-ERL at 14M data points in the box-pushing tasks because it clearly outperformed the baseline methods. We have now extended the training to 40M data points, as illustrated in Figure R2.

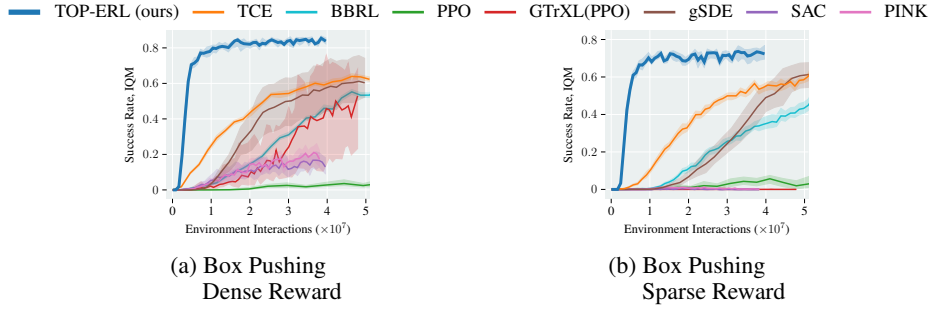


Figure R2: Continue training TOP-ERL till 40M.

R.5 ABLATION: REMOVING THE TARGET NETWORK

We conducted an ablation study by removing the target critic network in the box-pushing task under the dense reward setting and report the result in Fig. R3. We observed poorer performance in the ablated model, highlighting the importance of the target network in our current approach. However, a recent method, Cross-Q (Bhatt et al., 2024), proposed several techniques leveraging batch normalization in critic updates, effectively removing the need for a target network in SAC (Haarnoja et al., 2018). Incorporating these techniques into our transformer critic is left as future work.

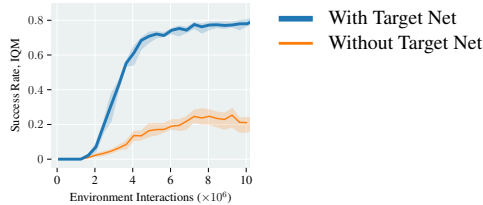


Figure R3: With target net or not

R.6 ABLATION: EFFECTIVENESS OF THE CRITIC TRANSFORMER

We conducted two additional ablation studies to prove the effectiveness of our transformer critic in the box pushing dense reward setting.

SAC + MP The approach involves directly using SAC (Haarnoja et al., 2018) to predict the MP’s parameters while retaining the trajectory generation and environment rollout procedure described in Section 4.1. We sum up the rewards collected during trajectory execution and use this as the return for the entire trajectory. Here, the trajectory is treated as a single action, and the learning objective does not incorporate any temporal information within the trajectory.

Degenerate Critic Transformer We degenerate our critic transformer from taking sequence of actions to a single action. The approach retains the main architecture of TOP-ERL but reduces the segment length to one in critic updates. In this case, the transformer critic degenerates to a standard critic network, as the input consists of only one state and one action rather than a sequence of actions.

The empirical results in Figure R4 illustrate the poor performance of both ablate approaches in the box pushing task with dense reward. For the **SAC + MP** approach, we infer that the main issue lies in its inability to leverage the temporal structure of the trajectory sequence to efficiently update the policy, making it difficult to assign credit to critical actions. This method fails to distinguish the distinct contributions of each action to task success, instead tending to average them. Additionally, the high dimensionality of the MP parameters compared to low-level actions poses challenges for Q-function learning.

For the **Degenerate Critic Transformer** method, we infer that although the action trajectory is temporally correlated, the critic fails to capture the sequence’s value when only single actions are provided as input. In contrast, the default TOP-ERL model, which takes action sequences as input, effectively captures their values, leading to more efficient learning outcomes. Furthermore, we infer that using varying lengths of action sequences can implicitly regularize the training of the value function.

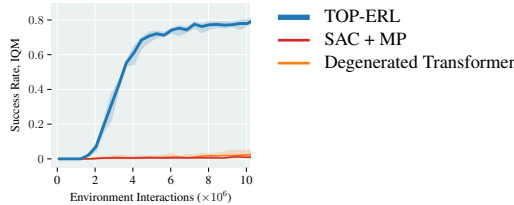


Figure R4: Demonstrate the effectiveness of critic transformer

Impact of the Transformer-based Critic on Value Prediction. To assess the impact of incorporating a transformer-based critic on value prediction, we adopt the analysis framework from Clipped Double Q-learning (Fujimoto et al., 2018). Specifically, we compare the average values predicted by the critics with the true values, computed using Monte Carlo returns. These findings are presented in Fig. R5. Additionally, we analyze the critic bias, defined as the difference between the values predicted by the critics and the Monte Carlo returns. The results, shown in Fig. R6, indicate that TOP-ERL consistently achieves less-biased value predictions compared to its ablated version. In contrast, a standard critic network, which only takes a single action as input, suffers from overestimation bias and often relies on additional techniques, such as Clipped Double Q-learning (Fujimoto et al., 2018) and critic ensembles (Chen et al., 2021), to mitigate these issues.

The advantages of using a transformer-based critic arise from several key factors. First, it processes action sequences as input, which implicitly regularizes the training of the value function. For instance, in Eq. (7), the expectation of the Q-value over L actions is used as the target for the V-function prediction. By varying L across different update iterations, the V-function is trained on Q-values derived from diverse action sequences, providing an implicit regularization effect on the critic network.

Furthermore, as off-policy RL methods depend on the critic to guide policy updates, the transformer-based critic enables decision-making over action sequences rather than individual per-step actions. This capability significantly enhances training efficiency and improves task performance.

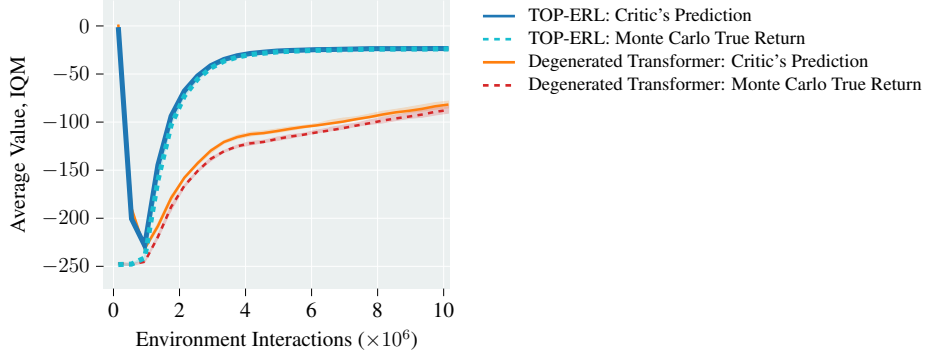


Figure R5: Comparison of average critic predictions and average Monte Carlo True returns across TOP-ERL and its ablated version. The analysis methodology is consistent with the approach discussed in Clipped Double Q-learning (Fujimoto et al., 2018). The steps before 0.8M represent the warm-up phase of the critic network, where the policy net was not trained.

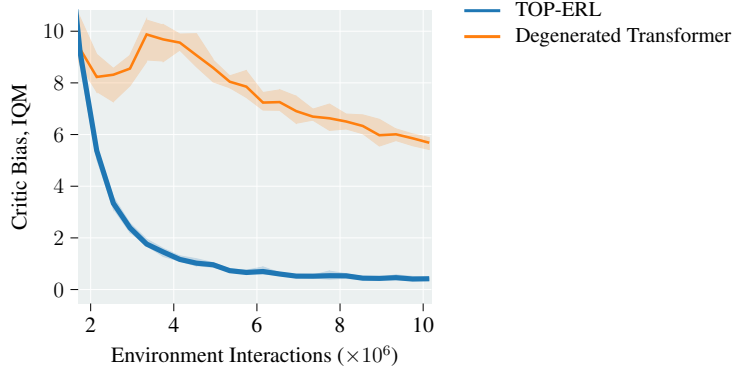


Figure R6: Comparison of critic bias between TOP-ERL and its ablated version. The results are reported starting from 2M environment interactions (corresponding to 1.2M steps since the policy training begins).

R.7 INDIVIDUAL PERFORMANCE OF METAWORLD 50 TASKS

We reported each individual Metaworld(Yu et al., 2020) task in Fig. R7 and Fig. R8. These tasks cover a wide range of types and complexities.

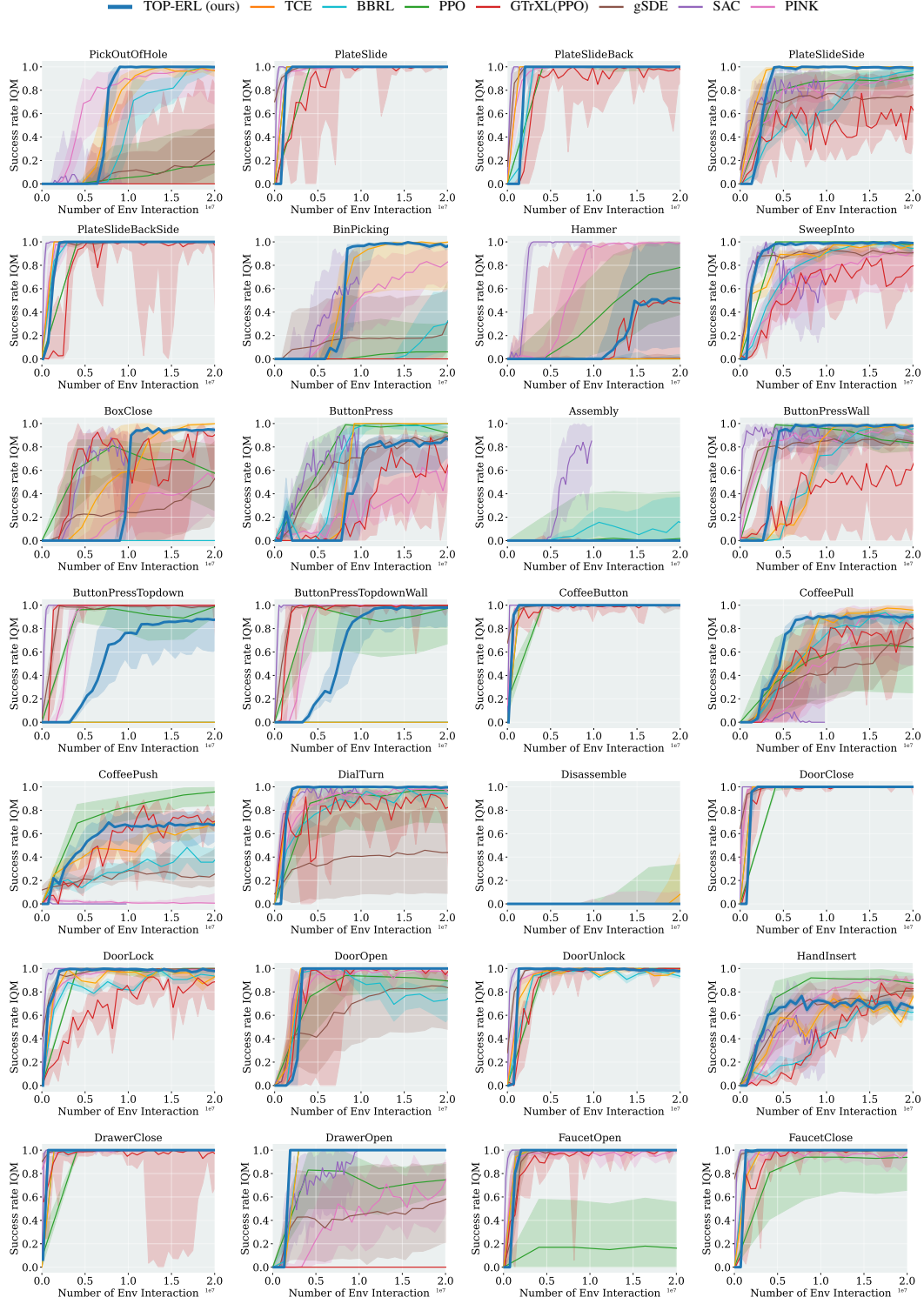


Figure R7: Success Rate IQM of each individual Metaworld tasks.

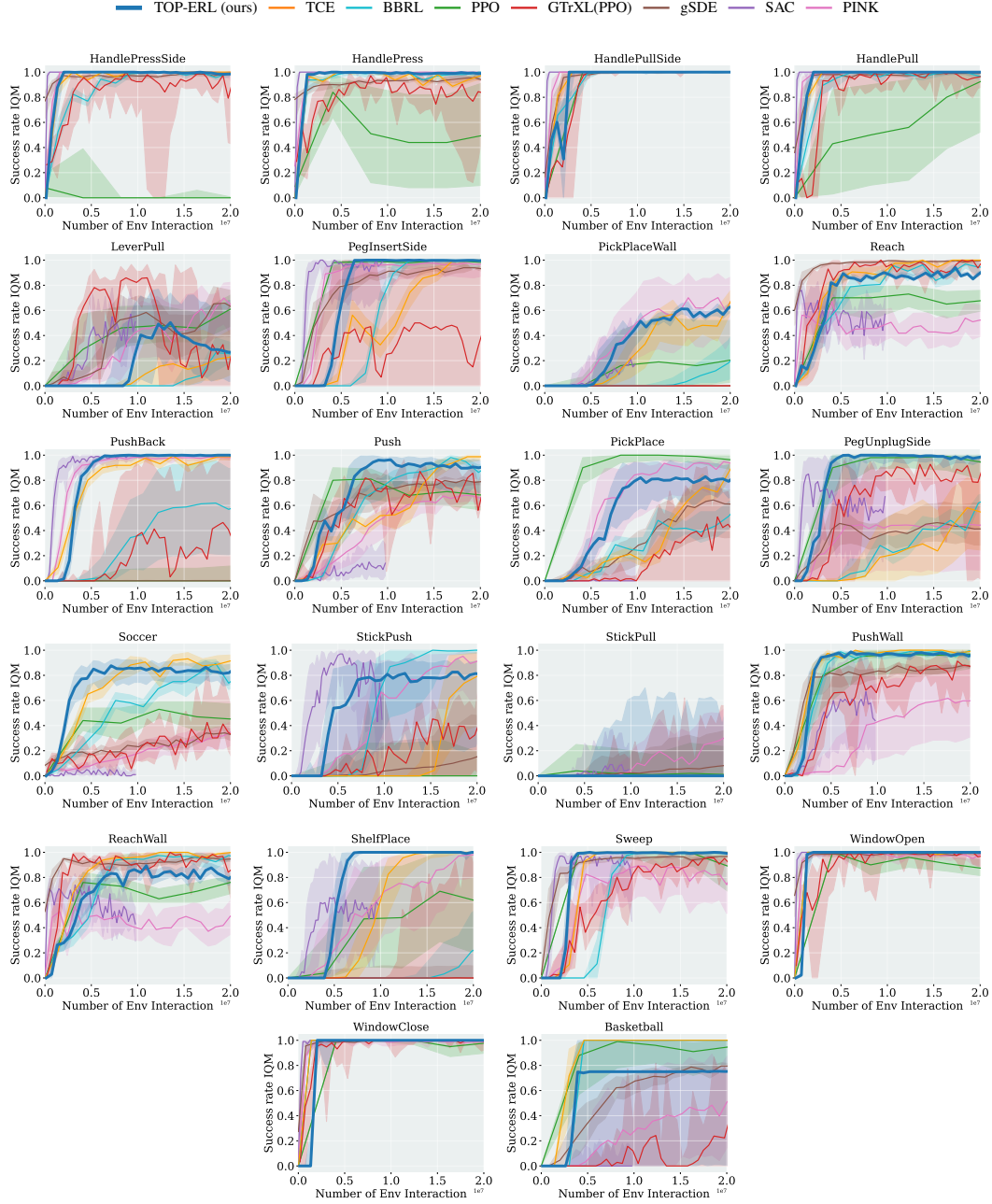


Figure R8: Success Rate IQM of each individual Metaworld tasks.

REFERENCES

- Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J Zico Kolter. Differentiable convex optimization layers. *Advances in neural information processing systems*, 32, 2019.
- Aditya Bhatt, Daniel Palenicek, Boris Belousov, Max Argus, Artemij Amiranashvili, Thomas Brox, and Jan Peters. Crossq: Batch normalization in deep reinforcement learning for greater sample efficiency and simplicity. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=PczQtTsTIX>.
- Xinyue Chen, Che Wang, Zijian Zhou, and Keith W Ross. Randomized ensembled double q-learning: Learning fast without a model. In *International Conference on Learning Representations*, 2021.
- Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pp. 1587–1596. PMLR, 2018.
- Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.
- Ge Li, Zeqi Jin, Michael Volpp, Fabian Otto, Rudolf Lioutikov, and Gerhard Neumann. Prodmpp: A unified perspective on dynamic and probabilistic movement primitives. *IEEE Robotics and Automation Letters*, 8(4):2325–2332, 2023.
- Ge Li, Hongyi Zhou, Dominik Roth, Serge Thilges, Fabian Otto, Rudolf Lioutikov, and Gerhard Neumann. Open the black box: Step-based policy updates for temporally-correlated episodic reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=mnipav175N>.
- Fabian Otto, Philipp Becker, Ngo Anh Vien, Hanna Carolin Ziesche, and Gerhard Neumann. Differentiable trust region layers for deep reinforcement learning. *International Conference on Learning Representations*, 2021.
- Fabian Otto, Onur Celik, Hongyi Zhou, Hanna Ziesche, Vien Anh Ngo, and Gerhard Neumann. Deep black-box reinforcement learning with movement primitives. In *Conference on Robot Learning*, pp. 1244–1265. PMLR, 2022.
- John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pp. 1094–1100. PMLR, 2020.