LOCALIZED DYNAMICS-AWARE DOMAIN ADAPTATION FOR OFF-DYNAMICS OFFLINE REINFORCEMENT LEARNING

Zhangjie Xia[1] * **Yu Yang**[2] * **Pan Xu**[2]
[1]New York University [2]Duke University
zx1357@nyu.edu {yu.yang,pan.xu}@duke.edu

ABSTRACT

Off-dynamics offline reinforcement learning (RL) aims to learn a policy for a target domain using limited target data and abundant source data collected under different transition dynamics. Existing methods typically address dynamics mismatch either globally over the state space or via pointwise data filtering; these approaches can miss localized cross-domain similarities or incur high computational cost. We propose Localized Dynamics-Aware Domain Adaptation (LoDADA), which exploits localized dynamics mismatch to better reuse source data. LoDADA clusters transitions from source and target datasets and estimates cluster-level dynamics discrepancy via domain discrimination. Source transitions from clusters with small discrepancy are retained, while those from clusters with large discrepancy are filtered out. This yields a fine-grained and scalable data selection strategy that avoids overly coarse global assumptions and expensive per-sample filtering. We provide theoretical insights and extensive experiments across environments with diverse global and local dynamics shifts. Results show that LoDADA consistently outperforms state-of-the-art off-dynamics offline RL methods.

1 INTRODUCTION

In this paper, we study off-dynamics offline RL problem (Liu et al., 2022; Wang et al., 2026), which concerns learning an optimal policy where an agent relies on a limited target dataset collected from a *target domain* together with a larger source dataset collected under different dynamics from a *source domain*. Such mismatch can severely degrade deployment performance in real-world applications where direct interaction with the target environment is costly or unsafe (Kiran et al., 2021).

Prior works include: (i) data augmentation methods that modify rewards in the source data using dynamics-aware regularization (Liu et al., 2022; Wang et al., 2026); (ii) data filtering approaches that select source transitions based on their similarity to target transitions (Wen et al., 2024; Lyu et al., 2025); and (iii) data generation methods that use source data to learn target dynamics and generate synthetic target samples (Guo et al., 2026). However, most methods treat source-target dynamics mismatch globally over the state-action space, which can be overly conservative in regions where the domains align and overly optimistic where they differ substantially.

Our proposed method, **Lo**calized **D**ynamics-**A**ware **D**omain **A**daptation (**LoDADA**), tackles dynamics mismatch by clustering source and target next states to identify locally aligned regions, filtering source samples far from cluster centroids, and estimating local KL divergence between source and target transition dynamics via a source–target classifier trained per cluster. Rather than discarding entire clusters, LoDADA preserves state-space coverage while emphasizing clusters with smaller estimated KL divergence by selectively retaining source transitions. During policy optimization, we regularize the learned policy toward a behavior policy of the target dataset and optimize using Implicit Q-Learning (IQL) (Kostrikov et al., 2021). Our main contributions are summarized as follows:

- We identify a key limitation of existing off-dynamics offline RL methods: they address dynamics mismatch either globally or pointwise, which can lead to data inefficiency and degraded perfor-

*Equal contribution

mance. To address this, we propose LoDADA, which performs cluster-wise filtering by estimating local source-target dynamics discrepancy and retaining the most target-consistent source transitions.
- We provide a theoretical analysis that establishes return lower bounds for the policy learned by LoDADA , offering performance guarantees in the target domain under dynamics mismatch.
- We empirically evaluate our method in MuJoCo environments with different types and levels of off-dynamics shifts and demonstrate the effectiveness of our method with an average of 19.9% improvement over baselines in gravity and friction shifts, 32.7% on morphology shift and 29.3% on local perturbation. We additionally demonstrate the superior performance of LoDADA on challenging navigation tasks in AntMaze and manipulation tasks in Adroit.

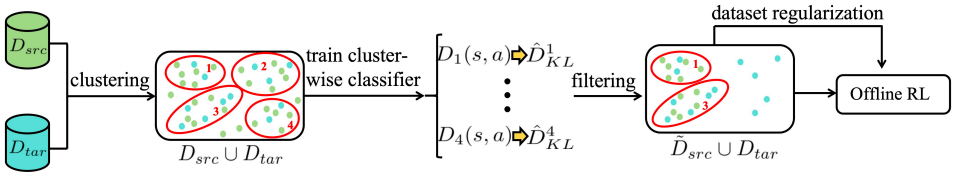


Figure 1: **An overview of our proposed framework.** We first perform K-means clustering on the mixed source and target dataset. Then we estimate the local KL divergence between source and target dynamics in each cluster. We use cluster-wise classifiers to selectively retain source domain transitions for downstream offline RL algorithms. We further introduce a dataset regularization term to ensure policy consistency with the target dataset.

2 METHODOLOGY

Due to space limit, missing theoretical insights and proofs are deferred to Sections C and D.

2.1 THEORETICAL INSIGHTS

The goal of data filtering in the off-dynamics setting is to create a filtered source dataset $\tilde{D}_{src}$ out of the source dataset D_{src} such that the dynamics discrepancy between $\tilde{D}_{src}$ and the target dataset D_{tar} is minimized. Now we present an upper bound for the performance gap in the target domain between the policy learned from the filtered source dataset and the optimal target policy.

Theorem 2.1 (Offline performance bound)**.** *Let $\hat{\pi}^*_{\tilde{D}_{src}}$ be the policy that maximizes the reward in the **filtered** source dataset $\tilde{D}_{src}$, $\pi^*_{D_{tar}}$ be the policy that maximizes the reward in the **unfiltered** target dataset D_{tar}, and π^* be the policy that maximizes the reward in the target domain. Assume that π^* satisfies Theorem C.2 and let C be the offline RL performance bound without dynamics shift, i.e., $\mathbb{E}_{P_{tar},\pi^*}\left[\sum \gamma^t r(s_t,a_t)\right] - \mathbb{E}_{P_{tar},\pi^*_{D_{tar}}}\left[\sum \gamma^t r(s_t,a_t)\right] \le C$. Then $\hat{\pi}^*_{\tilde{D}_{src}}$ receives near-optimal reward on the target domain:*

$$\mathbb{E}_{P_{tar},\hat{\pi}^*_{\tilde{D}_{src}}}\left[\sum_t \gamma^t r(s_t,a_t)\right] \ge \mathbb{E}_{P_{tar},\pi^*}\left[\sum_t \gamma^t r(s_t,a_t)\right] - \underbrace{C}_{\text{offline algorithm error}} - \frac{4R_{\max}}{1-\gamma}\underbrace{\sqrt{1-e^{B-\epsilon}}}_{\text{dynamics deviation}}$$
$$- \frac{2R_{\max}}{1-\gamma}\underbrace{\mathbb{E}_{\substack{\hat{\pi}^*_{\tilde{D}_{src}} \\ \rho_{\mathcal{M}_{src}}, P_{src}}}\left[D_{\mathrm{TV}}\left(\hat{\pi}^*_{\tilde{D}_{src}}(\cdot|s')\|\pi^*_{D_{tar}}(\cdot|s')\right)\right]}_{\text{filtering deviation}}.$$

2.2 CLUSTERING-BASED KL ESTIMATION

Given two datasets $D_{\mathrm{src}} = \{(s_i, a_i, s'_i)\}_{i=1}^{N_{\mathrm{src}}}$ and $D_{\mathrm{tar}} = \{(s_j, a_j, s'_j)\}_{j=1}^{N_{\mathrm{tar}}}$, we first run K-means on the next states s', yielding K disjoint clusters $\{\mathcal{N}^n\}_{n=1}^K$. We then pair each s' in $\mathcal{N}^n$ with its corresponding (s,a), forming $\mathcal{N}^n = \{(z_i, s'_i)\}_{i=1}^{N_0^n+N_1^n}$ with $z_i = s_i \oplus a_i$, where N_0^n and N_1^n are the numbers of source and target samples in $\mathcal{N}^n$. Within each cluster, target samples z_i^{tar} are labeled as class 1 and source samples z_i^{src} as class 0. We aim to train a cluster-specific classifier $D_n(z) = P(y=1|z)$ such that

$$D_n(z) = \frac{P(z|y=1)P(y=1)}{P(z|y=1)P(y=1) + P(z|y=0)P(y=0)}. \tag{1}$$

Within each cluster, s' are similar by construction, so we assume that $P(z|y=1) \approx P(z|s'_{\text{tar}})$ and $P(z|y=0) \approx P(z|s'_{\text{src}})$. Define $w(z) = P(z|s'_{\text{tar}})/P(z|s'_{\text{src}})$ and plug this into Equation (1):

$$D_n(z) = \frac{Aw_n(z)}{Aw_n(z) + (1-A)} \implies \hat{w}_n(z) = \frac{(1-A)D_n(z)}{A(1-D_n(z))},$$

where $A = P(y=1) = N_1^n/(N_1^n + N_0^n)$ and $1 - A = P(y=0) = N_0^n/(N_1^n + N_0^n)$. Since $D_{\text{KL}}(P(z|s'_{\text{src}})\|P(z|s'_{\text{tar}})) = \mathbb{E}_{z\sim P(z|s'_{\text{src}})}[\log 1/w(z)]$, we can estimate it using source samples:

$$\hat{D}^n_{\text{KL}} = \frac{1}{N_0^n}\sum_{i=1}^{N_0^n} \log \frac{1}{\hat{w}_n(z_i^{\text{src}})}, \tag{2}$$

where $\hat{D}^n_{\text{KL}}$ is a *cluster* estimate of the KL divergence and $\log 1/\hat{w}_n(z_i^{\text{src}}) =: d_i$ is a *point* estimate.

2.3 Policy Optimization

In off-dynamics settings, naively imitating the mixed dataset, especially source transitions collected under mismatched dynamics, can degrade target-domain performance. To address this, we (i) filter the source dataset using our KL estimates and (ii) use the (normalized) pointwise KL estimates to adaptively reweight learning. We use the following value-function objective:

$$\mathcal{L}_Q = \mathbb{E}_{D_{tar}}\big[\big(Q_\theta - \mathcal{T}Q_\theta\big)^2\big] + \mathbb{E}_{(s,a,s',d)\sim\tilde{D}_{src}}\big[\exp\big(-\alpha\cdot\hat{d}_i\big)\big(Q_\theta - \mathcal{T}Q_\theta\big)^2\big], \tag{3}$$

where α is a hyperparameter and $\hat{d}_i = \frac{d_i - \max_i d_i}{\max_i d_i - \min_i d_i}$ for $i \in \{1, \ldots, |\tilde{D}_{src}|\}$ is the normalized pointwise KL estimate. Following Lyu et al. (2025), we additionally train a conditional variational autoencoder (CVAE) $\hat{\pi}_b^{\text{tar}}(a|s)$ to model the target behavior policy and encourage the learned policy to stay close to the support of the target dataset. The policy is optimized according to

$$\mathcal{L}_\pi^{\text{reg}} = \mathcal{L}_\pi - \lambda\mathbb{E}_{s\sim\tilde{D}_{\text{src}}\cup D_{\text{tar}}}\big[\log\hat{\pi}_b^{\text{tar}}\big(\pi(\cdot|s)|s\big)\big], \tag{4}$$

where $\mathcal{L}_\pi$ is the policy loss of the underlying offline RL algorithm, and λ is a hyperparameter. Formally, we layout the framework of LoDADA in Figure 1 and display the algorithm in Algorithm 1.

Algorithm 1 Localized Dynamics-Aware Domain Adaptation (LoDADA)

Input: Source data D_{src}, target data D_{tar}
Initialize: Policy π_ϕ, value function Q_θ, batch size B, number of clusters K, diameter threshold δ, critic importance ratio α, regularization weight λ, filtering ratio ξ_1, ξ_2, ξ_3
Localize $D_{tar} \cup D_{src}$ into $\{\mathcal{N}^i\}_{i=1}^K$ based on s'
for i=1, 2..., K **do**
    Train classifiers $D_n(z)$ using cross-entropy loss
    Calculate KL divergence $\hat{D}^n_{KL}$ with Equation (2)
end for
Sort $\{\mathcal{N}^i\}_{i=1}^K$ based on $\hat{D}^n_{KL}$ and filter selectively per cluster to get $\tilde{D}_{src}$
for `each gradient step` **do**
    Sample a mini-batch $b_{\text{src}} := \{(s,a,r,s')\}$ of size $N/2$ from $\tilde{D}_{src}$
    Sample a mini-batch $b_{\text{tar}} := \{(s,a,r,s')\}$ of size $N/2$ from D_{tar}
    Optimize the value function Q_θ with Equation (3)
    Optimize the policy π_ϕ with Equation (4)
end for

The implementation and technical details are deferred to Section G.6.

3 Experiment

In this section, we evaluate the performance of LoDADA on the ODRL benchmark (Lyu et al., 2024b). We begin with MuJoCo locomotion tasks under multiple types and levels of global (Section 3.1) and local (Section H.1) dynamics shifts. In addition, we evaluate on navigation tasks in AntMaze (Section H.2) and manipulation tasks in Adroit (Section H.4). Finally, we conduct ablation studies (Section H.5) to better understand the sensitivity to hyperparameters and experiments (Section H.6) to compare the runtime of LoDADA with other data filtering methods to demonstrate the superiority of our algorithm. Details about experiment setup are deferred to Sections F and G.

Table 1: Performance comparison on MuJoCo tasks (HalfCheetah, Ant, Walker2D, Hopper) under a medium-level offline dataset with dynamic shifts in gravity and friction at levels 0.1, 0.5, 2.0, 5.0. Source domains remain unchanged; target domains are shifted. We report normalized target-domain scores over five seeds. Best and second-best scores are highlighted in green and blue, respectively.

Env	Shift Level	BOSA	IQL	DARA	IGDF	OTDF	Ours
HalfCheetah Gravity	0.1	9.31±1.94	12.90±1.01	9.62±4.27	7.65±6.63	12.17±3.58	**14.09±3.83**
	0.5	39.92 ±5.63	43.68±2.96	40.87±5.46	39.73±4.62	37.12±4.88	**45.00±2.95**
	2.0	28.47 ±1.23	**31.12±0.3**	30.49±0.54	30.16±1.5	25.38±3.12	28.02±1.67
	5.0	11.27±1.77	20.87±21.45	32.42±18.54	30.08±34.02	24.01±17.88	**71.11±1.63**
HalfCheetah Friction	0.1	12.53±3.61	23.69±16.46	**26.39±11.35**	20.56±11.22	10.31±1.68	9.97±0.47
	0.5	63.65±9.14	68.26±1.42	68.41±3.19	66.30±1.79	66.32±0.46	**68.51±0.44**
	2.0	45.53±2.01	42.68±2.64	46.17±1.72	46.22±1.27	45.84±0.34	**48.19±0.75**
	5.0	38.26±10.82	48.97±4.68	45.31±9.83	**54.75±5.6**	38.36±9.82	54.65±1.56
Ant Gravity	0.1	**25.58±2.21**	11.03±1.24	12.53±1.11	13.55±2.13	18.82±5.51	22.38±1.28
	0.5	17.75±2.48	10.53±1.06	9.08±0.88	11.34±1.89	16.77±3.13	**18.61±2.79**
	2.0	37.17±0.96	38.26±3.09	38.54±2.19	41.59±4.63	37.11±0.74	**45.82±1.33**
	5.0	33.48±1.70	31.51±1.97	32.51±1.71	34.44±2.77	34.58±0.25	**48.29±3.68**
Ant Friction	0.1	**58.95±0.71**	55.12±0.24	55.56±0.46	54.88±0.60	53.38±0.42	53.76±1.37
	0.5	**59.72±3.57**	58.92±0.80	59.28±0.80	55.64±5.03	58.20±0.84	55.54±3.80
	2.0	20.18±3.79	17.54±2.47	19.84±3.20	19.63±4.15	32.33±3.45	**62.27±2.23**
	5.0	9.88±1.17	7.79±0.31	7.78±0.26	13.97±7.68	13.17±8.28	**28.62±3.43**
Walker2d Gravity	0.1	18.75±12.02	20.12±5.74	16.04±7.60	24.95±16.19	63.77±6.43	**71.41±1.99**
	0.5	40.09±2.37	29.72±16.02	42.05±10.52	37.36±5.49	32.47±7.04	**44.65±1.49**
	2.0	8.91±2.28	32.20±1.05	25.69±10.67	34.89±6.73	37.23±2.16	**40.27±1.77**
	5.0	5.25±0.50	5.44±0.08	5.42±0.29	5.66±0.32	4.92±0.08	**8.55±3.85**
Walker2d Friction	0.1	7.88±1.88	5.65±0.06	5.72±0.23	7.21±1.80	**30.40±7.54**	20.21±7.90
	0.5	53.42±20.06	67.91±7.03	68.92±11.43	70.83±4.22	69.06±9.44	**81.41±6.47**
	2.0	39.06±17.36	**72.91±0.37**	65.40±7.13	62.55±10.36	59.93±14.53	62.01±4.36
	5.0	5.89±3.42	5.38±0.09	5.28±0.19	5.21±0.36	14.98±9.15	**15.93±3.09**
Hopper Gravity	0.1	27.82±13.41	23.4±11.62	13.10±0.98	13.89±4.70	33.41±1.92	**37.68±6.32**
	0.5	17.74±7.93	17.78±6.85	14.68±5.24	19.81±9.56	35.22±5.94	**43.72±9.72**
	2.0	13.18±2.10	14.71±1.67	14.94±1.08	14.71±0.87	**19.82±1.77**	16.99±3.74
	5.0	7.97±0.61	7.78±0.37	7.81±0.48	8.03±0.50	8.78±0.03	**9.28±0.26**
Hopper Friction	0.1	25.55±2.69	26.13±4.24	24.16±4.50	24.02±5.00	24.76±1.21	**28.06±7.97**
	0.5	25.86±2.17	24.75±3.58	24.84±4.63	29.12±3.01	29.89±6.17	**33.57±1.31**
	2.0	10.32±0.06	10.15±0.03	10.15±0.06	10.35±0.60	**10.52±0.29**	10.44±0.13
	5.0	6.92±1.97	7.92±0.06	7.84±0.04	7.90±0.17	**8.07±0.04**	8.04±0.05
Total		826.06	894.82	886.84	916.98	1007.10	**1207.05**

3.1 Results on MuJoCo Locomotion Tasks

Results on gravity and friction shifts. As shown in Table 1, LoDADA achieves a total score of 1207.05, yielding a 19.9% improvement over the second-best method OTDF (Lyu et al., 2025) and a 34.9% improvement over the backbone algorithm IQL (Kostrikov et al., 2021). Across 32 evaluation tasks (8 environments $\times$ 4 shift levels), LoDADA achieves best (21) or second-best (6) performance in 27 out of 32 tasks and remains competitive on the remaining 5 tasks. **These results indicate that LoDADA improves target-domain performance by effectively leveraging localized filtering.**

Compared to vanilla IQL, we introduce (i) adaptive weighting in critic learning and (ii) an additional CVAE-based behavior-cloning regularizer. As shown in Table 1, these modifications yield significant gains over IQL on 27 out of 32 tasks. **These observations suggest that combining localized filtering with target-aware regularization improves adaptation to the target domain.**

Results on morphology shift. For MuJoCo locomotion tasks with morphology shifts, the results and detailed analysis are deferred to Section H.3. The key takeaway is that LoDADA achieves the best score in 13 out of 16 tasks, yielding a 32.7% improvement over the second-best method.

4 Conclusion

In this work, we studied off-dynamics offline RL through a localized dynamics-aware data filtering method. We provided a theoretical performance bound that motivates us to perform data filtering from a local perspective and introduce a dataset regularization term to encourage the learned policy to align with the target behavior. Building on this insight, we proposed LoDADA and conducted extensive experiments under various types and levels of both global and local dynamics shifts. Experiment results demonstrate that LoDADA outperforms existing baselines across a wide range of tasks. Future works include exploring the effectiveness of LoDADA in real-world environments.

REFERENCES

David Abel, Yuu Jinnai, Sophie Yue Guo, George Konidaris, and Michael Littman. Policy and value transfer in lifelong reinforcement learning. In *International conference on machine learning*, pp. 20–29. PMLR, 2018.

Paul Barde, Julien Roy, Wonseok Jeon, Joelle Pineau, Chris Pal, and Derek Nowrouzezahrai. Adversarial soft advantage fitting: Imitation learning without policy optimization. *Advances in Neural Information Processing Systems*, 33:12334–12344, 2020.

Clément L Canonne. A short note on an inequality between kl and tv. *arXiv preprint arXiv:2202.07198*, 2022.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=SJx63jRqFm.

Benjamin Eysenbach, Shreyas Chaudhari, Swapnil Asawa, Sergey Levine, and Ruslan Salakhutdinov. Off-dynamics reinforcement learning: Training for transfer with domain classifiers. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=eqBwg3AcIAK.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Jingwen Gu, Yiting He, Zhishuai Liu, and Pan Xu. Policy regularized distributionally robust markov decision processes with linear function approximation. *arXiv preprint arXiv:2510.14246*, 2025.

Yihong Guo, Yixuan Wang, Yuanyuan Shi, Pan Xu, and Anqi Liu. Off-dynamics reinforcement learning via domain adaptation and reward augmented imitation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 136326–136360, 2024.

Yihong Guo, Yu Yang, Pan Xu, and Anqi Liu. MOBODY: Model-based off-dynamics offline reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=7c0YS3cuno.

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.

Yiting He, Zhishuai Liu, Weixin Wang, and Pan Xu. Sample complexity of distributionally robust off-dynamics reinforcement learning with online interaction. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=pJdMOKqdSV.

B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE transactions on intelligent transportation systems*, 23(6):4909–4926, 2021.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Jongmin Lee, Wonseok Jeon, Byungjun Lee, Joelle Pineau, and Kee-Eung Kim. Optidice: Offline policy optimization via stationary distribution correction estimation. In *International Conference on Machine Learning*, pp. 6120–6130. PMLR, 2021.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Jinxin Liu, Hongyin Zhang, and Donglin Wang. Dara: Dynamics-aware reward augmentation in offline reinforcement learning. *arXiv preprint arXiv:2203.06662*, 2022.

Jinxin Liu, Ziqi Zhang, Zhenyu Wei, Zifeng Zhuang, Yachen Kang, Sibo Gai, and Donglin Wang. Beyond ood state actions: supported cross-domain offline reinforcement learning. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, pp. 13945–13953, 2024a.

Yao Liu, Adith Swaminathan, Alekh Agarwal, and Emma Brunskill. Provably good batch off-policy reinforcement learning without great exploration. *Advances in neural information processing systems*, 33:1264–1274, 2020.

Zhishuai Liu, Weixin Wang, and Pan Xu. Upper and lower bounds for distributionally robust off-dynamics reinforcement learning. *arXiv preprint arXiv:2409.20521*, 2024b.

Yuping Luo, Huazhe Xu, Yuanzhi Li, Yuandong Tian, Trevor Darrell, and Tengyu Ma. Algorithmic framework for model-based deep reinforcement learning with theoretical guarantees. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BJe1E2R5KX.

Jiafei Lyu, Chenjia Bai, Jing-Wen Yang, Zongqing Lu, and Xiu Li. Cross-domain policy adaptation by capturing representation mismatch. In *International Conference on Machine Learning*, pp. 33638–33663. PMLR, 2024a.

Jiafei Lyu, Kang Xu, Jiacheng Xu, Jing-Wen Yang, Zongzhang Zhang, Chenjia Bai, Zongqing Lu, Xiu Li, et al. Odrl: A benchmark for off-dynamics reinforcement learning. *Advances in Neural Information Processing Systems*, 37:59859–59911, 2024b.

Jiafei Lyu, Mengbei Yan, Zhongjian Qiao, Runze Liu, Xiaoteng Ma, Deheng Ye, Jing-Wen Yang, Zongqing Lu, and Xiu Li. Cross-domain offline policy adaptation with optimal transport and dataset constraint. In *The Thirteenth International Conference on Learning Representations*, 2025.

Marlos C Machado, Marc G Bellemare, and Michael Bowling. A laplacian framework for option discovery in reinforcement learning. In *International Conference on Machine Learning*, pp. 2295–2304. PMLR, 2017.

Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. *Advances in neural information processing systems*, 32, 2019.

Ruhan Wang, Yu Yang, Zhishuai Liu, Dongruo Zhou, and Pan Xu. Return augmented decision transformer for off-dynamics reinforcement learning. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=QDVOr5J9Xp.

Xiaoyu Wen, Chenjia Bai, Kang Xu, Xudong Yu, Yang Zhang, Xuelong Li, and Zhen Wang. Contrastive representation for data filtering in cross-domain offline reinforcement learning. *arXiv preprint arXiv:2405.06192*, 2024.

Haoran Xu, Xianyuan Zhan, Jianxiong Li, and Honglei Yin. Offline reinforcement learning with soft behavior regularization. *arXiv preprint arXiv:2110.07395*, 2021.

Kang Xu, Chenjia Bai, Xiaoteng Ma, Dong Wang, Bin Zhao, Zhen Wang, Xuelong Li, and Wei Li. Cross-domain policy adaptation via value-guided data filtering. *Advances in Neural Information Processing Systems*, 36:73395–73421, 2023.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

A PRELIMINARIES

RL problems are typically formulated as a Markov Decision Process (MDP), specified by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma, \rho_0)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P(s'|s,a)$ is the transition kernel, $r(s,a)$ is the reward function given state-action pair (s,a), $\gamma \in [0,1)$ is the discounting factor, and ρ_0 is the initial state distribution.

In the off-dynamics offline setting, we consider two MDPs: a source domain $\mathcal{M}_{\text{src}} = (\mathcal{S}, \mathcal{A}, P_{\text{src}}, r, \gamma, \rho_0)$ and a target domain $\mathcal{M}_{\text{tar}} = (\mathcal{S}, \mathcal{A}, P_{\text{tar}}, r, \gamma, \rho_0)$. The two domains differ only in their transition dynamics, i.e., $P_{\text{src}}(s'|s,a) \neq P_{\text{tar}}(s'|s,a)$, while sharing the same reward function $r_{\text{src}}(s,a) = r_{\text{tar}}(s,a)$. In the offline setting, the agent cannot interact with the environment and must learn from static datasets (Levine et al., 2020). Suppose we are given a source dataset $D_{\text{src}} = \{(s,a,s',r)_{\text{src}}\}$ and a smaller target dataset $D_{\text{tar}} = \{(s,a,s',r)_{\text{tar}}\}$. The goal is to learn a policy π that maximizes the expected return in the target domain, $\max_\pi \mathbb{E}_{\pi, P_{\text{tar}}}\big[\sum_{t=0}^{\infty} \gamma^t r_{\text{tar}}(s_t, a_t)\big]$, by leveraging the mixed dataset $D_{\text{mix}} = D_{\text{src}} \cup D_{\text{tar}}$. Typically we have $|D_{\text{tar}}| \ll |D_{\text{src}}|$.

Notations: $\mathbb{H}(X)$ is the entropy of random variable X. $z = s \oplus a$ is the vector concatenation of state s and action a, which can be viewed as an identity representation of (s,a).

B RELATED WORK

Domain adaptation in RL. Domain adaptation in RL focuses on adapting policies across domains with differing transition dynamics. Prior work include policy and value function transfer which reuse knowledge from a source domain to accelerate learning in a target domain (Abel et al., 2018; Xu et al., 2023), and robust MDPs which optimize policies against the worst-case dynamics within a predefined uncertainty set (Liu et al., 2024b; He et al., 2025; Gu et al., 2025). However, these approaches often require carefully specified uncertainty sets and may fail to cover the true target dynamics. More importantly, they tend to be overly conservative due to the rigid assumption that there is no access to target data at all. More recent work have explored online RL domain adaptation using reward augmentation from dynamics mismatch (Eysenbach et al., 2021; Guo et al., 2024) and representation deviation (Lyu et al., 2024a). In contrast, our method tackles domain adaptation in a purely offline setting through a clustering-based framework, enabling more flexible and efficient reuse of source data under substantial dynamics shifts.

Off-dynamics offline RL. Off-dynamics offline RL aims to leverage an abundant source dataset together with a limited target dataset to improve policy learning in the target domain where online interactions is infeasible. Prior works approached this problem by aligning source and target distributions via importance weighting (Nachum et al., 2019; Lee et al., 2021), reward augmentation (Liu et al., 2022), using adversarial domain adaptation to reduce the mismatch in state-action dynamics (Barde et al., 2020; Xu et al., 2021), or filtering source data based on similarity to the target (Liu et al., 2020; Wen et al., 2024; Lyu et al., 2025). Another line of work adopts model-based approaches, where a dynamics model is learned in the target domain to enable more principled reuse of source transitions under mismatched dynamics (Yu et al., 2020; Guo et al., 2026). These methods often struggle with severe distribution shifts: importance weighting suffers from high variance under limited coverage, adversarial adaptation can be unstable in practice, heuristic filtering may reduce sample efficiency, and model-based methods are sensitive to model bias. In contrast, our method balances robustness and coverage by retaining useful source experiences while downweighting unreliable clusters.

Clustering-based methods in RL Clustering-based methods in RL exploit structure in the state-action space by grouping similar samples into clusters, thereby reducing variance and improving generalization. Recent approaches use clustering for representation learning, such as learning latent embeddings that capture environment dynamics and then clustering them to discover reusable behaviors or intermediate objectives (Machado et al., 2017; Eysenbach et al., 2019). These approaches often rely on heuristic similarity metrics, which may not accurately reflect the underlying transition dynamics. Our method leverages clustering not only to structure the data, but also integrates distributional alignment within clusters, ensuring that the grouped samples are both coherent and dynamically relevant to the target domain.

C Additional Theoretical Insights

Following Lyu et al. (2024a), we have the following proposition.

Proposition C.1. *For any (s, a), denote its representation as z, and suppose $s'_{src} \sim P_{\mathcal{M}_{\rm src}}(\cdot|z), s'_{\rm tar} \sim P_{\mathcal{M}_{\rm tar}}(\cdot|z)$. Then measuring the representation deviation between the source domain and the target domain is equivalent to measuring the dynamics mismatch between two domains, i.e., $D_{\rm KL}\big(P(z|s'_{\rm src})\|P(z|s'_{\rm tar})\big) = D_{\rm KL}\big(P(s'_{\rm src}|z)\|P(s'_{\rm tar}|z)\big) + \mathbb{H}(s'_{\rm src}) - \mathbb{H}(s'_{\rm tar})$.*

The difference between Theorem C.1 and Theorem 4.3 of Lyu et al. (2024a) lies in the direction of the KL divergence. We derive the KL divergence from the source representation distribution to the target representation distribution (rather than the reverse). This choice matches our goal of characterizing how far the policy learned from filtered source data can be from the target-optimal policy (Theorem 2.1). Because KL divergence is asymmetric, we reproduce the proof in Section D for completeness.

Since the entropy terms are constants for fixed source/target domains, minimizing dynamics deviation is equivalent to minimizing representation deviation. Concretely, let $\mathbb{H}(s'_{\rm src}) - \mathbb{H}(s'_{\rm tar}) = B$ be a constant. If the representation deviation is bounded by ϵ, i.e., $D_{\rm KL}(P(z|s'_{\rm src})\|P(z|s'_{\rm tar})) \le \epsilon$, then the dynamics deviation $D_{\rm KL}(P(s'_{\rm src}|z)\|P(s'_{\rm tar}|z))$ is bounded by $\epsilon - B$.

The following assumption from Eysenbach et al. (2021) quantifies the performance gap that the optimal target policy π^* may suffer when evaluated under source dynamics.

Assumption C.2. Let $\pi^* = \arg\max_\pi \mathbb{E}_p\big[\sum \gamma^t r(s_t, a_t)\big]$ be the reward-maximizing policy in the target domain. Then the expected reward in source and target domains differs by at most $2R_{\max}\sqrt{1 - e^{B-\epsilon}}$:

$$\left|\mathbb{E}_{\pi^*, P_{\rm src}}\Big[\sum_t \gamma^t r(s_t, a_t)\Big] - \mathbb{E}_{\pi^*, P_{\rm tar}}\Big[\sum_t \gamma^t r(s_t, a_t)\Big]\right| \le 2R_{\max}\sqrt{1 - e^{B-\epsilon}},$$

where $B = \mathbb{H}(s'_{\rm src}) - \mathbb{H}(s'_{\rm tar})$.

D Proof of the Main Theoretical Results

D.1 Proof of Theorem C.1

Proof of Theorem C.1. This proof is similar to that of Theorem 4.3 in Lyu et al. (2024a). However, we derive this result by calculating the KL divergence between the source representation distribution and the target representation distribution, rather than vice versa. This is because our proposed method LoDADA requires characterizing the distance between the policy learned from the filtered source data and the target optimal policy, as demonstrated in Theorem 2.1. Due to the asymmetry of KL divergence, we reproduce the proof here for the completeness of our work.

We would like to establish a connection between the representation deviations in the two domains and the dynamics discrepancies between the two domains. To achieve this, we introduce the concept of mutual information:

$$\begin{aligned}
h(z; s'_{\rm src}, s'_{\rm tar}) &= I(z; s'_{\rm src}) - I(z; s'_{\rm tar}) \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z, s'_{\rm src}) \log \frac{P(z, s'_{\rm src})}{P(z)P(s'_{\rm src})} dz ds'_{\rm src} - \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z, s'_{\rm tar}) \log \frac{P(z, s'_{\rm tar})}{P(z)P(s'_{\rm tar})} dz ds'_{\rm tar} \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z, s'_{\rm src}) \log \frac{P(z|s'_{\rm src})}{P(z)} dz ds'_{\rm src} - \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z, s'_{\rm tar}) \log \frac{P(z|s'_{\rm tar})}{P(z)} dz ds'_{\rm tar} \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}}\int_{\mathcal{S}} P(z, s'_{\rm src}, s'_{\rm tar}) \log \frac{P(z|s'_{\rm src})}{P(z)}\, dz\, ds'_{\rm src}\, ds'_{\rm tar} \\
&\quad - \int_{\mathcal{Z}}\int_{\mathcal{S}}\int_{\mathcal{S}} P(z, s'_{\rm tar}, s'_{\rm src}) \log \frac{P(z|s'_{\rm tar})}{P(z)}\, dz\, ds'_{\rm tar}\, ds'_{\rm src} \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}}\int_{\mathcal{S}} P(z, s'_{\rm src}, s'_{\rm tar}) \log \frac{P(z|s'_{\rm src})}{P(z|s'_{\rm tar})} dz ds'_{\rm src} ds'_{\rm tar}
\end{aligned}$$

$$= D_{\mathrm{KL}}(P(z|s'_{\mathrm{src}})\|P(z|s'_{\mathrm{tar}})).$$

Note that the definition of the KL-divergence already involves expectations over s'_{src} and s'_{tar}. While one can also write $\mathbb{E}_{s'_{\mathrm{src}},s'_{\mathrm{tar}}}[D_{\mathrm{KL}}(P(z|s'_{\mathrm{src}})\|P(z|s'_{\mathrm{tar}}))]$ and it should not affect the result. At the same time, we also have

$$\begin{aligned}
h(z;s'_{\mathrm{src}},s'_{\mathrm{tar}}) &= I(z;s'_{\mathrm{src}}) - I(z;s'_{\mathrm{tar}}) \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z,s'_{\mathrm{src}}) \log \frac{P(z,s'_{\mathrm{src}})}{P(z)P(s'_{\mathrm{src}})}\, dz\, ds'_{\mathrm{src}} - \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z,s'_{\mathrm{tar}}) \log \frac{P(z,s'_{\mathrm{tar}})}{P(z)P(s'_{\mathrm{tar}})}\, dz\, ds'_{\mathrm{tar}} \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z,s'_{\mathrm{src}}) \log \frac{P(s'_{\mathrm{src}}|z)}{P(s'_{\mathrm{src}})}\, dz\, ds'_{\mathrm{src}} - \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z,s'_{\mathrm{tar}}) \log \frac{P(s'_{\mathrm{tar}}|z)}{P(s'_{\mathrm{tar}})}\, dz\, ds'_{\mathrm{tar}} \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}}\int_{\mathcal{S}} P(z,s'_{\mathrm{src}},s'_{\mathrm{tar}}) \log \frac{P(s'_{\mathrm{src}}|z)}{P(s'_{\mathrm{src}})}\, dz\, ds'_{\mathrm{src}}\, ds'_{\mathrm{tar}} \\
&\quad - \int_{\mathcal{Z}}\int_{\mathcal{S}}\int_{\mathcal{S}} P(z,s'_{\mathrm{tar}},s'_{\mathrm{src}}) \log \frac{P(s'_{\mathrm{tar}}|z)}{P(s'_{\mathrm{tar}})}\, dz\, ds'_{\mathrm{tar}}\, ds'_{\mathrm{src}} \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}}\int_{\mathcal{S}} P(z,s'_{\mathrm{src}},s'_{\mathrm{tar}}) \log \frac{P(s'_{\mathrm{src}}|z)}{P(s'_{\mathrm{tar}}|z)}\, dz\, ds'_{\mathrm{src}}\, ds'_{\mathrm{tar}} \\
&\quad - \int_{\mathcal{S}} P(s'_{\mathrm{src}}) \log P(s'_{\mathrm{src}})\, ds'_{\mathrm{src}} + \int_{\mathcal{S}} P(s'_{\mathrm{tar}}) \log P(s'_{\mathrm{tar}})\, ds'_{\mathrm{tar}} \\
&= D_{\mathrm{KL}}(P(s'_{\mathrm{src}}|z)\|P(s'_{\mathrm{tar}}|z)) + \mathbb{H}(s'_{\mathrm{src}}) - \mathbb{H}(s'_{\mathrm{tar}}).
\end{aligned}$$

One can see that the defined function is also connected to the dynamics discrepancy term $D_{\mathrm{KL}}(P(s'_{\mathrm{src}}|z)\|P(s'_{\mathrm{tar}}|z))$ and two entropy terms. Nevertheless, we observe that the source domain and the target domain are specified and fixed, and their state distributions are also fixed, indicating that the entropy terms are constants. So in the end we have

$$\underbrace{D_{\mathrm{KL}}(P(z|s'_{\mathrm{src}})\|P(z|s'_{\mathrm{tar}}))}_{\text{representation deviation}} = \underbrace{D_{\mathrm{KL}}(P(s'_{\mathrm{src}}|z)\|P(s'_{\mathrm{tar}}|z))}_{\text{dynamics deviation}} + \underbrace{\mathbb{H}(s'_{\mathrm{src}}) - \mathbb{H}(s'_{\mathrm{tar}})}_{\text{constants}}.$$

This completes the proof. □

D.2 Proof of Theorem 2.1

Proof of Theorem 2.1. For simplicity, denote $\tilde{r} = \sum \gamma^t r(s_t, a_t)$. Since It is infeasible to directly interact an online policy π^* with an offline policy $\hat{\pi}^*_{\hat{D}_{src}}$ in different domains, we introduce $\pi^*_{\mathrm{src}} \in \Pi_{\text{no exploit}}$ that maximizes the reward in the filtered source domain. Then we have

$$\begin{aligned}
\mathbb{E}_{P_{\mathrm{tar}},\pi^*}(\tilde{r}) - \mathbb{E}_{P_{\mathrm{tar}},\hat{\pi}^*_{\hat{D}_{src}}}(\tilde{r}) &= \underbrace{\mathbb{E}_{P_{\mathrm{tar}},\pi^*}(\tilde{r}) - \mathbb{E}_{P_{\mathrm{src}},\pi^*}(\tilde{r})}_{(a)} + \underbrace{\mathbb{E}_{P_{\mathrm{src}},\pi^*}(\tilde{r}) - \mathbb{E}_{P_{\mathrm{src}},\pi^*_{\mathrm{src}}}(\tilde{r})}_{(b)} \\
&\quad + \underbrace{\mathbb{E}_{P_{\mathrm{src}},\pi^*_{\mathrm{src}}}(\tilde{r}) - \mathbb{E}_{P_{\mathrm{tar}},\pi^*_{\hat{D}_{tar}}}(\tilde{r})}_{(c)} + \underbrace{\mathbb{E}_{P_{\mathrm{tar}},\pi^*_{\hat{D}_{tar}}}(\tilde{r}) - \mathbb{E}_{P_{\mathrm{tar}},\hat{\pi}^*_{\hat{D}_{src}}}(\tilde{r})}_{(d)}.
\end{aligned}$$

First note that term (b) is 0 because π^*_{src} is the optimal policy in the filtered source domain, which should be identical to the target domain, meaning π^*_{src} is almost the same as π^* and they should achieve the same reward in the same domain. Term (a) is bounded above by $2R_{\max}\sqrt{1-e^{B-\epsilon}}$ by Theorem C.2. To bound term (c), we first need the following result by applying Theorem E.1:

$$\mathbb{E}_{P_{\mathrm{src}},\pi^*_{\mathrm{src}}}(\tilde{r}) - \mathbb{E}_{P_{\mathrm{tar}},\pi^*}(\tilde{r}) \le \mathbb{E}_{P_{\mathrm{src}},\pi^*_{\mathrm{src}}}(\tilde{r}) - \mathbb{E}_{P_{\mathrm{tar}},\pi^*_{\mathrm{src}}}(\tilde{r}) \le 2R_{\max}\sqrt{1-e^{B-\epsilon}}.$$

Next we can use this and the performance bound of usual offline RL algorithm to find a bound for term (c):

$$\mathbb{E}_{P_{\mathrm{src}},\pi^*_{\mathrm{src}}}(\tilde{r}) \le 2R_{\max}\sqrt{1-e^{B-\epsilon}} + \mathbb{E}_{P_{\mathrm{tar}},\pi^*}(\tilde{r}) \le 2R_{\max}\sqrt{1-e^{B-\epsilon}} + C + \mathbb{E}_{P_{\mathrm{tar}},\pi^*_{\hat{D}_{tar}}}(\tilde{r}).$$

Rearrange the terms can we find the upper bound for term (c). Finally we will deal with term (d):

$$(d) = \underbrace{\mathbb{E}_{P_{\mathrm{tar}},\pi^*_{\hat{D}_{tar}}}(\tilde{r}) - \mathbb{E}_{P_{\mathrm{src}},\pi^*_{\hat{D}_{tar}}}(\tilde{r})}_{I_1} + \underbrace{\mathbb{E}_{P_{\mathrm{src}},\pi^*_{\hat{D}_{tar}}}(\tilde{r}) - \mathbb{E}_{P_{\mathrm{src}},\hat{\pi}^*_{\hat{D}_{src}}}(\tilde{r})}_{I_2} + \underbrace{\mathbb{E}_{P_{\mathrm{src}},\hat{\pi}^*_{\hat{D}_{src}}}(\tilde{r}) - \mathbb{E}_{P_{\mathrm{tar}},\hat{\pi}^*_{\hat{D}_{src}}}(\tilde{r})}_{I_3}.$$

We bound I_2 by applying Theorem E.3:

$$
\begin{aligned}
I_2 &\leq \left| \mathbb{E}_{P_{\text{src}}, \hat{\pi}^*_{\tilde{D}_{src}}}(\tilde{r}) - \mathbb{E}_{P_{\text{src}}, \pi^*_{D_{tar}}}(\tilde{r}) \right| \\
&= \left| \frac{1}{1-\gamma} \mathbb{E}_{\rho^{\hat{\pi}^*_{\tilde{D}_{src}}}_{\mathcal{M}_{\text{src}}}(s,a), s' \sim P_{\text{src}}} \left[\mathbb{E}_{a' \sim \hat{\pi}^*_{\tilde{D}_{src}}} \left[Q^{\pi^*_{D_{tar}}}_{\mathcal{M}_{\text{src}}}(s', a') \right] - \mathbb{E}_{a' \sim \pi^*_{D_{tar}}} \left[Q^{\pi^*_{D_{tar}}}_{\mathcal{M}_{\text{src}}}(s', a') \right] \right] \right| \\
&= \frac{1}{1-\gamma} \mathbb{E}_{\rho^{\hat{\pi}^*_{\tilde{D}_{src}}}_{\mathcal{M}_{\text{src}}}(s,a), s' \sim P_{\text{src}}} \left| \sum_{a' \in \mathcal{A}} \left(\hat{\pi}^*_{\tilde{D}_{src}}(a'|s') - \pi^*_{D_{tar}}(a'|s') \right) Q^{\pi^*_{D_{tar}}}_{\mathcal{M}_{\text{src}}}(s', a') \right| \\
&\leq \frac{R_{\max}}{1-\gamma} \mathbb{E}_{\rho^{\hat{\pi}^*_{\tilde{D}_{src}}}_{\mathcal{M}_{\text{src}}}(s,a), s' \sim P_{\text{src}}} \left| \sum_{a' \in \mathcal{A}} \left(\hat{\pi}^*_{\tilde{D}_{src}}(a'|s') - \pi^*_{D_{tar}}(a'|s') \right) \right| \\
&= \frac{2R_{\max}}{1-\gamma} \mathbb{E}_{\rho^{\hat{\pi}^*_{\tilde{D}_{src}}}_{\mathcal{M}_{\text{src}}}(s,a), s' \sim P_{\text{src}}} \left[D_{\text{TV}} \left(\hat{\pi}^*_{\tilde{D}_{src}}(\cdot|s') \,\|\, \pi^*_{D_{tar}}(\cdot|s') \right) \right].
\end{aligned}
$$

We bound I_1 by applying Theorem E.2:

$$
\begin{aligned}
I_1 &\leq \left| \mathbb{E}_{P_{\text{src}}, \pi^*_{D_{tar}}}(\tilde{r}) - \mathbb{E}_{P_{\text{tar}}, \pi^*_{D_{tar}}}(\tilde{r}) \right| \\
&= \left| \frac{\gamma}{1-\gamma} \mathbb{E}_{\rho^{\pi^*_{D_{tar}}}_{\mathcal{M}_{\text{src}}}(s,a)} \left[\mathbb{E}_{s' \sim P_{\text{src}}} \left[V^{\pi^*_{D_{tar}}}_{\mathcal{M}_{\text{tar}}}(s') \right] - \mathbb{E}_{s' \sim P_{\text{tar}}} \left[V^{\pi^*_{D_{tar}}}_{\mathcal{M}_{\text{tar}}}(s') \right] \right] \right| \\
&= \left| \frac{\gamma}{1-\gamma} \mathbb{E}_{\rho^{\pi^*_{D_{tar}}}_{\mathcal{M}_{\text{src}}}(s,a)} \left[\int_{\mathcal{S}} (P_{\mathcal{M}_{\text{src}}}(s' \mid s, a) - P_{\mathcal{M}_{\text{tar}}}(s' \mid s, a)) V^{\pi^*_{D_{tar}}}_{\mathcal{M}_{\text{tar}}}(s') ds' \right] \right| \\
&\leq \frac{\gamma}{1-\gamma} \mathbb{E}_{\rho^{\pi^*_{D_{tar}}}_{\mathcal{M}_{\text{src}}}(s,a)} \left[\left| \int_{\mathcal{S}} (P_{\mathcal{M}_{\text{src}}}(s' \mid s, a) - P_{\mathcal{M}_{\text{tar}}}(s' \mid s, a)) \right| \times \left| V^{\pi^*_{D_{tar}}}_{\mathcal{M}_{\text{tar}}}(s') \right| ds' \right] \\
&\leq \frac{2\gamma}{1-\gamma} R_{\max} [D_{\text{TV}}(P_{\mathcal{M}_{\text{src}}}(\cdot \mid s, a) \| P_{\mathcal{M}_{\text{tar}}}(\cdot \mid s, a))] \\
&\leq \frac{2\gamma}{1-\gamma} R_{\max} \sqrt{1 - e^{B-\epsilon}}.
\end{aligned}
$$

Similarly for I_3, we get the same upper bound. Combining all bounds together, we get the desired result. □

E USEFUL LEMMAS

Lemma E.1. *Let R_{max} be the maximum (entropy-regularized) return of any trajectory. Then the following inequality holds:*

$$
\left| \mathbb{E}_{\pi, P_{src}} \left[\sum \gamma^t r(s_t, a_t) \right] - \mathbb{E}_{\pi, P_{tar}} \left[\sum \gamma^t r(s_t, a_t) \right] \right| \leq 2R_{\max} \sqrt{1 - e^{B-\epsilon}}.
$$

Note that the original result from Eysenbach et al. (2021) was derived within a policy class, $\Pi_{\text{no exploit}}$, which is defined to be set of policies that have constrained dynamics discrepancy. Since in our case we assume the KL divergence between the source and target dynamics is bounded by $\epsilon - B$, the above result is not restricted to the policy class $\Pi_{\text{no exploit}}$ anymore.

Proof. This proof is similar to that of Lemma B.2 in Eysenbach et al. (2021), and the only difference is that we use B-H bound for TV distance instead of Pinsker's inequality in the last step. We repeat the proof here for readers' reference. We know $z = f(s, a)$ is injective, and apply Holder's inequality and Bretagnolle–Huber bound, which is never worse than Pinsker's inequality (Canonne, 2022), to obtain the desired result:

$$
\left| \mathbb{E}_{P_{\text{src}}} \left[\sum \gamma^t r(s_t, a_t) \right] - \mathbb{E}_{P_{\text{tar}}} \left[\sum \gamma^t r(s_t, a_t) \right] \right| = \left| \sum_{\tau} (P_{\text{src}}(\tau) - P_{\text{tar}}(\tau)) \Big(\sum \gamma^t r(s_t, a_t) \Big) \right|
$$

$$
\begin{aligned}
&\leq \Big\| \sum \gamma^t r(s_t, a_t) \Big\|_\infty \cdot \|P_{\text{src}}(\tau) - P_{\text{tar}}(\tau)\|_1 \\
&\leq \Big(\max_\tau \sum \gamma^t r(s_t, a_t) \Big) \cdot 2\sqrt{1 - e^{-D_{\text{KL}}(P_{\text{src}}(\tau) \| P_{\text{tar}}(\tau))}} \\
&\leq 2R_{\max}\sqrt{1 - e^{B-\epsilon}}.
\end{aligned}
$$

This completes the proof. □

Lemma E.2 (Telescoping lemma, Lemma 4.3 from Luo et al. (2019))**.** . *Denote* $\mathcal{M}_1 = (\mathcal{S}, \mathcal{A}, P_1, r, \gamma)$ *and* $\mathcal{M}_2 = (\mathcal{S}, \mathcal{A}, P_2, r, \gamma)$ *as two MDPs that only differ in their transition dynamics. Then for any policy* π*, we have*

$$
J_{\mathcal{M}_1}(\pi) - J_{\mathcal{M}_2}(\pi) = \frac{\gamma}{1-\gamma} \mathbb{E}_{\rho^\pi_{\mathcal{M}_1}(s,a)} \big[\mathbb{E}_{s' \sim P_1} \big[V^\pi_{\mathcal{M}_2}(s') \big] - \mathbb{E}_{s' \sim P_2} \big[V^\pi_{\mathcal{M}_2}(s') \big] \big].
$$

Lemma E.3. *Denote* $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$ *as the underlying MDP. Suppose we have two policies* π_1, π_2*, then the performance difference of these policies in the MDP gives:*

$$
J_{\mathcal{M}}(\pi_1) - J_{\mathcal{M}}(\pi_2) = \frac{1}{1-\gamma} \mathbb{E}_{\rho^{\pi_1}_{\mathcal{M}}(s,a), s' \sim P} \big[\mathbb{E}_{a' \sim \pi_1} \big[Q^{\pi_2}_{\mathcal{M}}(s', a') \big] - \mathbb{E}_{a' \sim \pi_2} \big[Q^{\pi_2}_{\mathcal{M}}(s', a') \big] \big].
$$

Proof. This is Lemma B.3 in Lyu et al. (2024a), please check the proof there. □

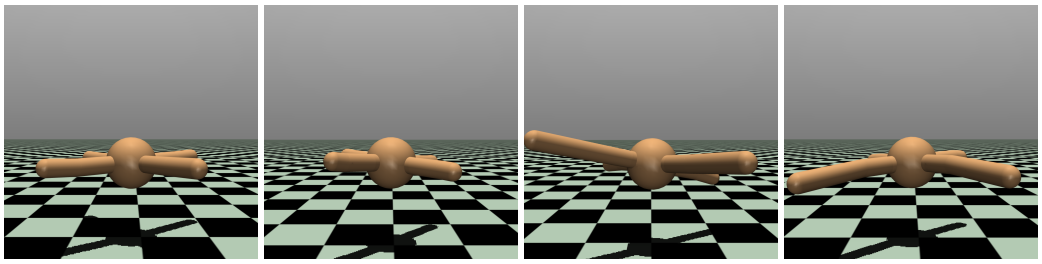
F Experimental Setup

Task and Environments. We evaluate LoDADA on the ODRL benchmark (Lyu et al., 2024b) across MuJoCo environments with different dynamics shifts. Each environment includes three types of dynamics shifts: gravity, friction and morphology shifts. The source dataset is collected in the original MuJoCo environment from D4RL (Fu et al., 2020), and the target dataset is collected by the behavior policy in the corresponding environment with dynamics shifts. All datasets are collected using SAC (Haarnoja et al., 2018) trained to a medium performance level in either the original or dynamics-shifted environment. We use the datasets provided by ODRL, where the source dataset contains 1 million transitions while the target one has 5000 transitions. To construct dynamics-shifted target environments, we follow ODRL and modify MuJoCo simulator parameters for gravity, friction and morphology. For gravity, we rescale the global gravity magnitude while keeping its direction unchanged. For friction, we rescale the friction parameters. In both cases, we consider four shift levels $\{0.1, 0.5, 2.0, 5.0\}$, where the original gravity or friction is multiplied by the corresponding factor. Morphology shifts are created by changing the size of specific limbs or the torso of the robot.

In addition to the locomotion tasks, we evaluate on AntMaze problems with different map layouts following ODRL. We focus on the medium-sized AntMaze setting and use 6 different map layouts as target environments, denoted as $\{1, 2, 3, 4, 5, 6\}$. We use the *play* and *diverse* datasets as source datasets, both drawn from D4RL (Fu et al., 2020), while the target dataset is collected by an agent trained directly in the target environment and provided by ODRL. The diverse dataset consists of trajectories that start from random locations and reach random goals, whereas the play dataset contains trajectories that start from hand-picked locations and reach a fixed hand-picked goal. Consequently, these two source datasets exhibit different state coverage.

Finally, we evaluate LoDADA on manipulation tasks from the Adroit benchmark. We consider the *pen* and *door* environments. Similar to the MuJoCo experiments, we construct target environments by introducing kinematic and morphological shifts. For kinematic shifts, we restrict the rotation ranges of hand joints, referred to as *broken-joint*. For morphological shifts, we shrink the sizes of fingers, referred to as *shrink-finger*. We consider shift levels of medium and hard in our experiments. The source dataset is drawn from D4RL at the human level, while the target dataset is obtained from ODRL. Additional details about all these environments can be found in Section G.

Baselines. We compare LoDADA against several strong baseline methods including DARA (Eysenbach et al., 2021), BOSA (Liu et al., 2024a), IQL (Kostrikov et al., 2021), IGDF (Wen et al., 2024), and OTDF (Lyu et al., 2025). DARA applies reward augmentation learned from the discrepancy



(a) alllegs (M) (b) alllegs (H) (c) halflegs (M) (d) halflegs (H)

Figure 2: Visualization of morphology shift environments for Ant.

between source and target dynamics to the source dataset to mimic the trajectories in the target environment. BOSA enforces support constraints with two objectives to mitigate OOD state–action pairs and OOD dynamics while filtering unreliable transitions. IQL is an offline RL algorithm that learns a value function and an advantage-weighted policy to achieve higher performance, which demonstrates strong performance in offline RL tasks trained on mixed datasets without explicit domain adaptation (Wen et al., 2024). IGDF learns a contrastive representation to estimate a mutual information-based domain gap for filtering source transitions. OTDF is the most recent state-of-the-art method which uses optimal transport to match source–target transitions and performs non-parametric filtering based on the optimal transport cost. Implementation details for all methods, including the architecture and hyperparameters, are presented in Section G.

G Experiment Details

In this section, we provide more details for the experiment settings and hyperparameters in Section 3.

G.1 Locomotion tasks

In this section, we demonstrate the detailed modification in the MuJoCo `xml` file to construct the target environment. In the locomotion tasks, we follow the ODRL, providing three dynamics shifts including gravity, friction and morphology shift. In the gravity and friction shift, we consider the shift level with $\{0.1, 0.5, 2.0, 5.0\}$. In the morphology shift, we consider the *medium* and *hard* level shift.

Gravity Shift. Following the ODRL benchmark (Lyu et al., 2024b), we modify the gravity of the environment by editing the gravity attribute. For example, the gravity of the HalfCheetah in the target is modified to 0.5 times the gravity in the source domain with the following code.

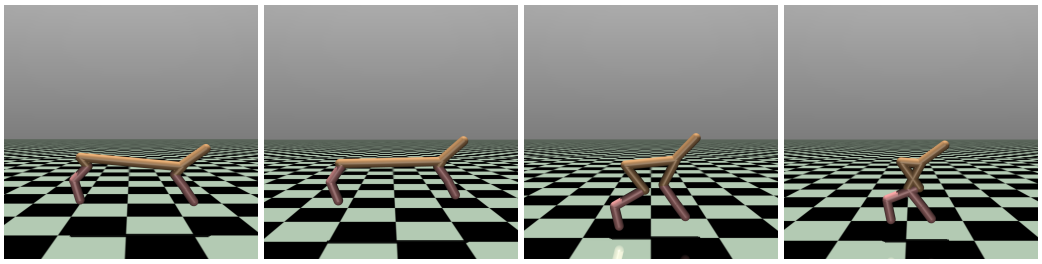
```
# gravity
<option gravity="0 0 -4.905" timestep="0.01"/>
```

Friction Shift. The friction shift is generated by modifying the friction attribute in the geom elements. The frictional components are adjusted to 0.1, 0.5, 2.0, 5.0 times the frictional components in the source domain, respectively.

Morphology Shift. The morphology shift is achieved by modifying the size of specific limbs or torsos of the simulated robot in MuJoCo without altering the state space and action space. Visualizations can be found in Figure 2, Figure 3, Figure 4 and Figure 5.

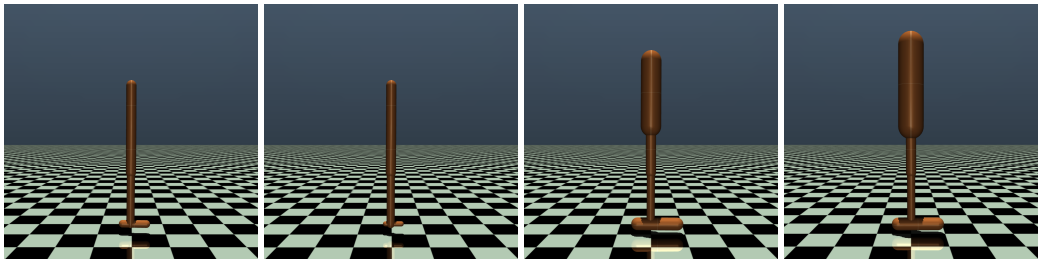
G.2 Navigation tasks

In this section, we present the detailed modification in the AntMaze map to construct the target environment. In this setting, we follow the ODRL benchmark and consider the medium size with six different target maps. Visualization of the map can be found in Figure 6.



(a) thigh (M) (b) thigh (H) (c) torso (M) (d) torso (H)

Figure 3: Visualization of morphology shift environments for HalfCheetah.



(a) foot (M) (b) foot (H) (c) torso (M) (d) torso (H)

Figure 4: Visualization of morphology shift environments for Hopper.

G.3 MANIPULATION TASKS

In this section, we discuss the detailed modifications in the Adroit environment. In this setting, we follow the ODRL benchmark using the *pen* and *door* as the tasks. The *pen* task is to control the 24-DoF shadow hand robot to twirl a pen, and the *door* task is to open a door. We consider the two types of dynamics shifts under this environment, denoted as the kinematic shift and the morphology task. We only consider the *medium* and *hard* level shift. Visualizations can be found in Figure 7 and Figure 8.

Kinematic Shift. The kinematic shift in the Adroit robot hand occurs at all finger joints of the index finger and the thumb. We make constraints on the range of torsos. We use two types of shift levels, denoted as medium and hard level for the kinematic shift.

Morphology Shift. The morphology shift in the Adroit robot hand occurs on the fingers. We shrink the sizes of these fingers with the medium and hard levels, which multiply by 0.25 and 0.125 for the size of the fingers, respectively.

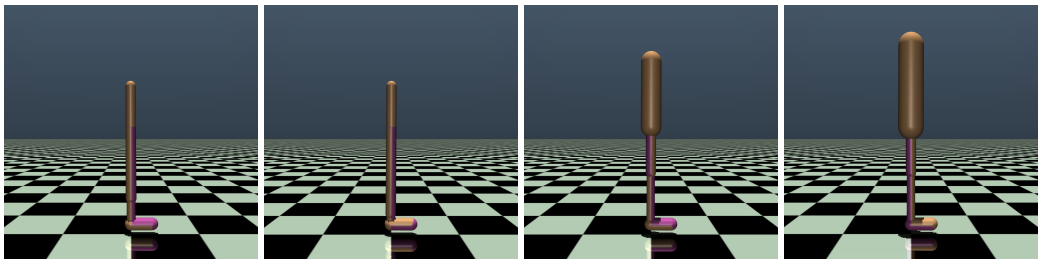
G.4 DATASETS

Note that we follow D4RL and adopt the *normalized score* metric to better characterize the agent's performance across different tasks. The normalized score in the target domain is defined as:

$$\text{NS} = \frac{J - J_r}{J_e - J_r} \times 100,$$

where J denotes the return achieved by the agent in the target domain, and J_r, J_e represent the returns obtained by the random and expert policies in the same domain, respectively.

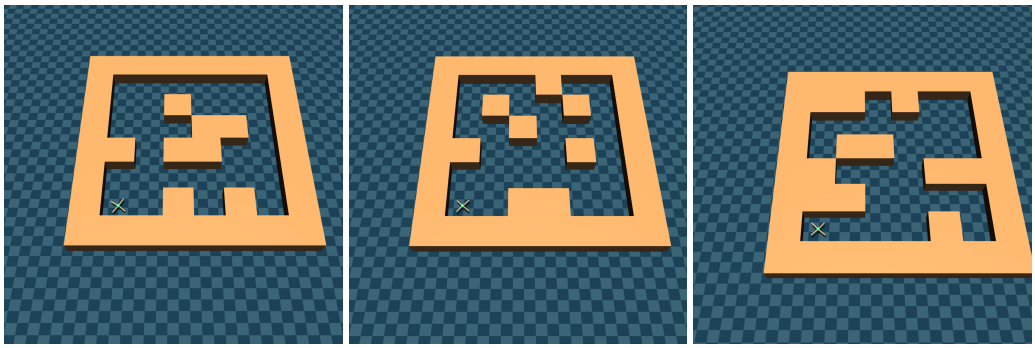
We summarize the reference scores of J_r and J_e under different dynamics shift scenarios in Table 2, Table 3 and Table 4.



(a) leg (M) (b) leg (H) (c) torso (M) (d) torso (H)

Figure 5: Visualization of morphology shift environments for Walker2d.

(a) antmaze-medium-1 (b) antmaze-medium-2 (c) antmaze-medium-3



(d) antmaze-medium-4 (e) antmaze-medium-5 (f) antmaze-medium-6

Figure 6: Visualization of the AntMaze map structure. We consider medium size map with 6 different map layouts.

G.5 Technical Details about Baseline Algorithms

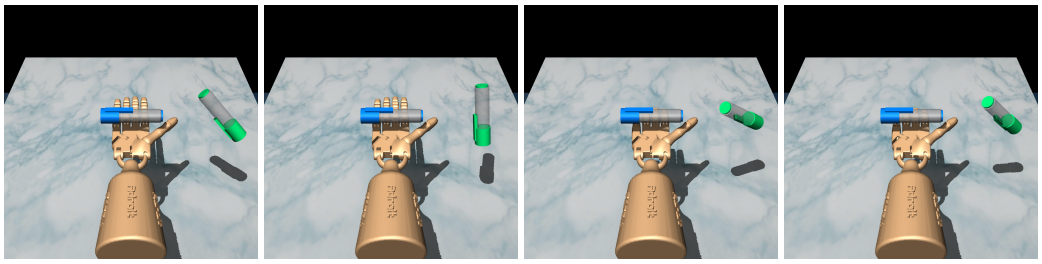
In this section, we describe the implementation details of the baseline methods, following the ODRL benchmark (Lyu et al., 2024b). Furthermore, we list the hyperparameter setup used for all methods in Table 5.

BOSA (Liu et al., 2024a). BOSA deals with the OOD state–action pairs through a supported policy optimization and addresses the OOD dynamics issue through a supported value optimization by data filtering. Specifically, the policy is updated by maximizing the following objective function:

$$\mathcal{L}_{\text{actor}} = \mathbb{E}_{s\sim D_{\text{src}}\cup D_{\text{trg}},\, a\sim\pi_\phi(s)}\big[Q(s,a)\big], \quad \text{s.t.} \quad \mathbb{E}_{s\sim D_{\text{src}}\cup D_{\text{trg}}}\big[\hat{\pi}_{\theta_{\text{offline}}}(\pi_\phi(s)\mid s)\big] > \epsilon.$$

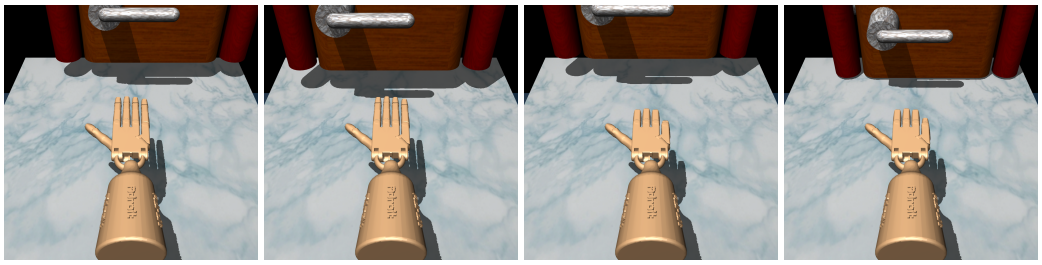
Here, ϵ is the threshold, $\hat{\pi}_{\theta_{\text{offline}}}$ is the learned policy for the combined offline dataset. The value critic is updated with

$$\mathcal{L}_{\text{critic}} = \mathbb{E}_{(s,a)\sim D_{\text{src}}}\big[Q(s,a)\big]$$



(a) broken-joint (M) (b) broken-joint (H) (c) shrink-finger (M) (d) shrink-finger (H)

Figure 7: Visualization of morphology and kinematic shift environments for Adroit Pen.



(a) broken-joint (M) (b) broken-joint (H) (c) shrink-finger (M) (d) shrink-finger (H)

Figure 8: Visualization of morphology and kinematic shift environments for Adroit Door.

$$+\,\mathbb{E}_{(s,a,r,s')\sim D_{\text{src}}\cup D_{\text{trg}},\, a'\sim\pi_\phi(\cdot|s')}\Big[I(\hat{p}_{\text{trg}}(s'\mid s,a)>\epsilon')\big(Q_{\theta_i}(s,a)-y\big)^2\Big],$$

where $I(\cdot)$ is the indicator function, $\hat{p}_{\text{trg}}(s'\mid s,a)=\arg\max\,\mathbb{E}_{(s,a,s')\sim D_{\text{trg}}}\big[\log\hat{p}_{\text{trg}}(s'\mid s,a)\big]$ is the estimated target domain dynamics and ϵ' is the threshold. The above objective is transformed to a relaxed Lagrangian form for optimization. Both the behavior policy and the dynamics models are modeled by the CVAE. We implement BOSA by following the instructions in the original paper. BOSA is trained for 500K gradient steps in practice by drawing samples from both the source domain dataset and the target domain dataset.

IQL (Kostrikov et al., 2021). IQL learns the state value function and state–action value function simultaneously by minimizing the following expectile regression loss function

$$\mathcal{L}_V=\mathbb{E}_{(s,a)\sim D_{\text{src}}\cup D_{\text{trg}}}\big[L_2^\tau\big(Q_\theta(s,a)-V_\psi(s)\big)\big],$$

where $L_2^\tau(u)=|\tau-I(u<0)|\|u\|^2$, $I(\cdot)$ is the indicator function, and θ is the target network parameter. The state–action value function is then updated by

$$\mathcal{L}_Q=\mathbb{E}_{(s,a,r,s')\sim D_{\text{src}}\cup D_{\text{trg}}}\Big[\big(r(s,a)+\gamma V_\psi(s')-Q_\theta(s,a)\big)^2\Big].$$

The advantage function is $A(s,a)=Q(s,a)-V(s)$. The policy is optimized by the advantage-weighted behavior cloning

$$\mathcal{L}_{\text{actor}}=\mathbb{E}_{(s,a)\sim D_{\text{src}}\cup D_{\text{trg}}}\big[\exp\big(\beta\cdot A(s,a)\big)\log\pi_\phi(a\mid s)\big],$$

where β is the inverse temperature coefficient. We implement IQL by following its official codebase and also train the IQL agent on offline datasets from both domains for 500K gradient steps.

DARA (Eysenbach et al., 2021). DARA is the offline version of DARC (Eysenbach et al., 2021). It trains two domain classifiers $q_{\theta_{\text{SAS}}}(\text{target}\mid s_t,a_t,s_{t+1})$, $q_{\theta_{\text{SA}}}(\text{target}\mid s_t,a_t)$ with the following objectives

$$\mathcal{L}(\theta_{\text{SAS}})=\mathbb{E}_{D_{\text{tar}}}\big[\log q_{\theta_{\text{SAS}}}(\text{target}\mid s_t,a_t,s_{t+1})\big]+\mathbb{E}_{D_{\text{src}}}\big[\log\big(1-q_{\theta_{\text{SAS}}}(\text{target}\mid s_t,a_t,s_{t+1})\big)\big],$$
$$\mathcal{L}(\theta_{\text{SA}})=\mathbb{E}_{D_{\text{tar}}}\big[\log q_{\theta_{\text{SA}}}(\text{target}\mid s_t,a_t)\big]+\mathbb{E}_{D_{\text{src}}}\big[\log\big(1-q_{\theta_{\text{SA}}}(\text{target}\mid s_t,a_t)\big)\big],$$

Table 2: The referenced minimum and maximum scores for MuJoCo datasets under various dynamics shift scenarios. These reference scores are used to compute normalized performance in the target domain.

Task Name	Dynamics Shift Type	Reference Min Score (J_r)	Reference Max Score (J_e)
HalfCheetah	Gravity - 0.1	-280.18	2466.85
	Gravity - 0.5	-280.18	9509.15
	Gravity - 2.0	-280.18	9509.15
	Gravity - 5.0	-280.18	3756.24
	Friction - 0.1	-280.18	41696.55
	Friction - 0.5	-280.18	7357.07
	Friction - 2.0	-280.18	11255.97
	Friction - 5.0	-280.18	10199.33
	Local	-280.18	12135.0
Hopper	Gravity - 0.1	-26.34	3234.3
	Gravity - 0.5	-26.34	3234.3
	Gravity - 2.0	-26.34	3234.3
	Gravity - 5.0	-26.34	3234.3
	Friction - 0.1	-26.34	3234.3
	Friction - 0.5	-26.34	3234.3
	Friction - 2.0	-26.34	3234.3
	Friction - 5.0	-26.34	3234.3
	Local	-20.272305	3234.3
Walker2d	Gravity - 0.1	10.08	2074.90
	Gravity - 0.5	10.08	5194.71
	Gravity - 2.0	10.08	5056.45
	Gravity - 5.0	10.08	3665.39
	Friction - 0.1	10.08	3360.18
	Friction - 0.5	10.08	4229.35
	Friction - 2.0	10.08	5180.04
	Friction - 5.0	10.08	4988.84
	Local	1.629008	4592.3
Ant	Gravity - 0.1	-325.6	2782.09
	Gravity - 0.5	-325.6	4317.07
	Gravity - 2.0	-325.6	6705.12
	Gravity - 5.0	-325.6	6226.89
	Friction - 0.1	-325.6	7938.96
	Friction - 0.5	-325.6	8301.34
	Friction - 2.0	-325.6	5167.38
	Friction - 5.0	-325.6	4545.02
	Local	-325.6	3879.7

to estimate the dynamics gap

$$\log \frac{P_{\mathcal{M}_{\text{tar}}}(s_{t+1} \mid s_t, a_t)}{P_{\mathcal{M}_{\text{src}}}(s_{t+1} \mid s_t, a_t)}$$

between the source domain and the target domain. DARA approximates this term by leveraging the trained classifiers and proposes to modify the source domain rewards as follows

$$\hat{r}_{\text{DARA}} = r - \lambda \times \delta r,$$

Table 3: The referenced minimum and maximum scores for AntMaze medium datasets under various dynamics shift scenarios. These reference scores are used to compute normalized performance in the target domain.

Task Name	Dynamics Shift Type	Reference Min Score (J_r)	Reference Max Score (J_e)
AntMaze-medium	1	0.0	1.0
	2	0.0	1.0
	3	0.0	1.0
	4	0.0	1.0
	5	0.0	1.0
	6	0.0	1.0

Table 4: The referenced minimum and maximum scores for Adroit datasets under various dynamics shift scenarios. These reference scores are used to compute normalized performance in the target domain.

Task Name	Dynamics Shift Type	Reference Min Score (J_r)	Reference Max Score (J_e)
Pen	broken-joint-medium	-12.17	6408.38
	broken-joint-hard	-12.17	6408.38
	shrink-finger-medium	-12.17	6408.38
	shrink-finger-hard	-12.17	6408.38
Door	broken-joint-medium	-52.34	2880.57
	broken-joint-hard	-52.34	2880.57
	shrink-finger-medium	-52.34	2880.57
	shrink-finger-hard	-52.34	2880.57

$$\delta r(s_t, a_t) = -\log \frac{q_{\theta_{\text{SAS}}}(\text{target} \mid s_t, a_t, s_{t+1})\, q_{\theta_{\text{SA}}}(\text{source} \mid s_t, a_t)}{q_{\theta_{\text{SAS}}}(\text{source} \mid s_t, a_t, s_{t+1})\, q_{\theta_{\text{SA}}}(\text{target} \mid s_t, a_t)},$$

where λ is an important hyperparameter that controls the strength of the reward penalty, which is set to be 0.1 by default. After that, DARA is trained on data from both domains for 500K gradient steps. We implement DARA by following the original paper and clip the reward penalty term to lie in $[-10, 10]$. We use IQL as the base algorithm for DARA.

IGDF (Wen et al., 2024). IGDF captures the dynamics gap between the source domain and the target domain through contrastive learning. It trains a score function $h(\cdot)$ using $(s, a, s'_{\text{tar}}) \sim D_{\text{tar}}$ from the target domain as positive samples, and mixed transition (s, a, s'_{src}) as negative samples, where $(s, a) \sim D_{\text{tar}}$, $s'_{\text{src}} \sim D_{\text{src}}$. The score function is optimized via the following contrastive learning objective:

$$\mathcal{L}_{\text{contrastive}} = -\mathbb{E}_{(s,a,s'_{\text{tar}})}\mathbb{E}_{S'^-}\left[\log \frac{h(s, a, s'_{\text{tar}})}{\sum_{s' \sim S'^- \cup s'_{\text{tar}}} h(s, a, s')}\right],$$

where S'^- is a collection of the next states in negative samples. Practically, IGDF adopts two neural networks $\phi(s, a), \psi(s')$ to learn representations of state-action pairs and states, and approximate the score function with a linear parameterization of them:

$$h(s, a, s') = \exp\left(\phi(s, a)^\top \psi(s')\right).$$

Based on the measured score function, IGDF proposes to filter out source domain data when training value functions:

$$\mathcal{L}_{\text{critic}} = \tfrac{1}{2}\,\mathbb{E}_{D_{\text{tar}}}\left[(Q_\theta - \mathcal{T}Q_\theta)^2\right] + \tfrac{1}{2}\,\alpha \cdot h(s, a, s')\,\mathbb{E}_{(s,a,s') \sim D_{\text{src}}}\left[\mathbf{1}(h(s, a, s') > h_{\xi\%})(Q_\theta - \mathcal{T}Q_\theta)^2\right],$$

where α is the importance coefficient for weighting the TD error of the source domain data, and ξ is the data selection ratio akin to that in OTDF. We run IGDF by using its official codebase and use IQL as its backbone.

OTDF (Lyu et al., 2025). OTDF performs off-dynamics offline policy adaptation by aligning the transition dynamics of the source and target domains using optimal transport. It first solves the optimal transport problem using

$$W(u, u') = \min_{\mu \in \mathcal{M}} \sum_{t=1}^{|D_{\text{src}}|} \sum_{t'=1}^{|D_{\text{tar}}|} C(u_t, u'_{t'})\, \mu_{t,t'},$$

given cost function C. Suppose the optimal coupling obtained is μ^*, then it defines the deviation between a source domain data point and the target domain dataset as:

$$d(u_t) = -\sum_{t'=1}^{|D_{\text{tar}}|} C(u_t, u'_{t'})\, \mu^*_{t,t'}, u_t = (s_t^{\text{src}}, a_t^{\text{src}}, (s'_{\text{src}})_t) \sim D_{\text{src}}.$$

After computing and normalizing d, it trains the value function via

$$\mathcal{L}_Q = \mathbb{E}_{(s,a,s')\sim D_{\text{tar}}}\left[(Q_\theta - \mathcal{T}Q_\theta)^2\right] + \mathbb{E}_{(s,a,s')\sim D_{\text{src}}}\left[exp(\hat{d}) \cdot \mathbf{1}(d > d_{\xi\%})\,(Q_\theta - \mathcal{T}Q_\theta)^2\right],$$

where $\hat{d}$ is normalized d and $\mathbf{1}(\cdot)$ is the indicator function. It then trains a Conditional Variational Autoencoder (CVAE) using loss function

$$\mathcal{L}_{\text{CVAE}} = \mathbb{E}_{(s,a)\sim D_{\text{tar}},\, z\sim E_\nu(s,a)}\Big[\big(a - D_\varsigma(s,z)\big)^2\Big] + D_{\text{KL}}\big(E_\nu(s,a)\,\|\,\mathcal{N}(0,I)\big),$$

where $E_\nu(s,a)$ is the encoder and $D_\varsigma(s,z)$ is the decoder, to model the target dynamics and generates multiple latent transitions. Finally, they train the policy via

$$\hat{\mathcal{L}}_\pi = \mathcal{L}_\pi - \beta\, \mathbb{E}_{s\sim D_{\text{src}}\cup D_{\text{tar}}}\left[\log\left(\sum_{i=1}^{M} \exp\left(\log \hat{\pi}^i_{\text{tar}}(\pi(\cdot|s) \mid s)\right)\right)\right],$$

where $\hat{\pi}^i_{\text{tar}}$ denotes the i-th constructed Gaussian behavior policy in the target domain. A data selection mechanism filters source samples whose generated transitions closely match the target. We run OTDF using its official codebase.

G.6 Technical Details about LoDADA

The LoDADA algorithm (Localized Dynamics-Aware Domain Adaptation), summarized in Algorithm 2, leverages clustering and divergence-based filtering to improve off-dynamics policy learning. First, target domain data are clustered by next-state variables to form clusters, and source samples are assigned to clusters based on proximity to centroids within a diameter threshold. For each cluster, a classifier estimates KL divergence between source and target distributions, and clusters are ranked accordingly. Source samples are then filtered adaptively based on the divergence ranking of their assigned clusters, which are divided into three classes according to their KL divergence: low, medium, and high. Clusters in the low-divergence class, which are most similar to the target dataset, admit the largest proportion ξ_1 of their source samples; clusters in the medium-divergence class admit a moderate proportion ξ_2; and clusters in the high-divergence class—indicating greater dissimilarity—admit only a small fraction ξ_3.

After filtering, each retained source sample is assigned an importance weight derived from its normalized KL estimate, so that samples from more similar clusters have higher influence during training, while samples from less similar clusters are down-weighted. During training, mini-batches from both domains are used to update the value function, compute target values, and optimize the state-action value function with divergence-weighted losses. Finally, the target network is updated, and the policy is optimized on filtered source samples using CVAE, enabling effective adaptation across domains. Although LoDADA is sensitive to the policy regularization weight λ, we could have a λ that generally receives a relatively good performance per setting. Specifically, we find that $\lambda = 0.1$ can be used for friction experiments, $\lambda = 0.5$ can be used for gravity experiments, $\lambda = 1.0$ is adopted in morphology experiments, and $\lambda \in \{0.5, 1.0\}$ is used in all local perturbation experiments. In the AntMaze setting, we sweep the number of clusters over $\{10, 20, 30, 50\}$ and observe that smaller cluster counts consistently yield better performance. We hypothesize that, unlike locomotion tasks in MuJoCo environments, the aligned transitions in both the source and target AntMaze environments

are concentrated within a small number of clusters (i.e., map position with overlap) induced by the map layout. As a result, using a smaller number of clusters is sufficient and yields better performance. For the policy regularization weight λ, we sweep $\lambda = \{0.1, 0.5, 1.0\}$ to achieve the best performance. In the Adroit settings, we follow similar hyperparameters in MuJoCo tasks. We sweep the number of clusters in $\{30, 50\}$ and the policy regularization weight in $\lambda = \{0.1, 1.0\}$.

Algorithm 2 LoDADA : Localized Dynamics-Aware Domain Adaptation

Input: Source domain data D_{src}, target domain data D_{tar}
Initialize: Policy π_ϕ, value networks V_ψ, Q_θ, target Q function $Q_{\theta'}$, batch size B, number of clusters K, diameter threshold δ, critic importance ratio α, regularization weight λ, filtering ratio ξ_1, ξ_2, ξ_3, target update rate η, CVAE with encoder $E_\nu(s,a)$ and decoder $D_\varsigma(s,z)$, number of sampled latent variables M
Train CVAE policy to model the behavior policy in the target domain dataset using

$$\mathcal{L}_{\text{CVAE}} = \mathbb{E}_{(s,a)\sim\mathcal{D}_{\text{tar}},\, z\sim E_\nu(s,a)}\Big[\|a - D_\varsigma(s,z)\|_2^2 + D_{\text{KL}}(E_\nu(s,a)\|\mathcal{N}(0,I))\Big].$$

Cluster D_{tar} into $\{\mathcal{N}^i\}_{i=1}^K$ based on s', with centroid c^i and diameter d^i
for `each source sample` $(s,a,s') \in D_{src}$ **do**
  Find nearest centroid c^i among $\{\mathcal{N}^i\}_{i=1}^K$
  if dist$(s', c^i) \le \delta \cdot \frac{1}{K}\sum_{i=1}^K d^i$ **then**
    Assign (s,a,s') to neighborhood $\mathcal{N}^i$
  end if
end for
for i=1, 2..., K **do**
  Train classifiers $D_n(z)$ using

$$L = -\mathbb{E}_{z\sim P(z|s'_{\text{tar}})}[\log D_n(z)] - \mathbb{E}_{z\sim P(z|s'_{\text{src}})}[\log(1 - D_n(z))].$$

  Calculate cluster estimate of KL divergence $\hat{D}^n_{KL}$ based on Equation (2)
end for
Sort $\{\mathcal{N}^i\}_{i=1}^K$ based on $\hat{D}^n_{KL}$ from smallest to largest and get $\{\mathcal{N}^{(j)}\}_{j=1}^K$
for each neighborhood $\mathcal{N}^{(j)}$, $j = 1, \ldots, K$ **do**
  Rank point estimates $\{d_i^j\}_{i=1}^{|\mathcal{N}_0^{(j)}|}$ of sampled source data
  Admit the top $\xi_{g(j)}\%$ samples, where $g(j) = \begin{cases} 1, & j \le \lceil K/3 \rceil, \\ 2, & \lceil K/3 \rceil < j \le \lceil 2K/3 \rceil, \\ 3, & j > \lceil 2K/3 \rceil. \end{cases}$
end for
Normalize point estimate of KL divergence via $\hat{d}_i = \frac{d_i - \max_i d_i}{\max_i d_i - \min_i d_i}, i \in \{1, ..., |\tilde{D}_{src}|\}$
for `each gradient step` **do**
  Sample a mini-batch b_{src} of size $B/2$ from $\tilde{D}_{src}$ and b_{tar} of size $B/2$ from D_{tar}
  Update the state value function V_ψ via: $\mathcal{L}_V = \mathbb{E}_{(s,a)\sim\tilde{D}_{src}\cup D_{tar}}\Big[L_2^\tau\big(Q_{\theta'}(s,a) - V_\psi(s)\big)\Big]$
  Compute the target value via: $y = r + \gamma V_\psi(s')$
  Optimize the state-action value function Q_θ on $b_{\text{src}} \cup b_{\text{tar}}$ via:

$$\mathcal{L}_Q = \mathbb{E}_{D_{tar}}[(Q_\theta - y)^2] + \mathbb{E}_{(s,a,s',d)\sim\tilde{D}_{src}}[\exp(-\alpha \cdot \hat{d}_i)(Q_\theta - y)^2].$$

  Update the target network via: $\theta' \leftarrow \eta\theta + (1-\eta)\theta'$
  Decode M actions from the CVAE and construct Gaussian distributions $\{\hat{\pi}_i^{\text{tar}}(\cdot \mid s)\}_{i=1}^M$.
  Compute the advantage A and optimize the policy π_ϕ on $b_{\text{src}} \cup b_{\text{tar}}$ using

$$\mathcal{L}_\pi = \mathbb{E}_{(s,a)\sim\tilde{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}}}[\exp(A)\log\pi_\phi(a \mid s)] - \lambda\mathbb{E}_{s\sim\tilde{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}}}\left[\log\left(\sum_{i=1}^M \hat{\pi}_{\text{tar}}^i(\pi_\phi(\cdot \mid s) \mid s)\right)\right].$$

end for

G.7 Compute Infrastructure

We run the experiment on a single GPU: NVIDIA RTX A5000-24564MiB with 8-CPUs: AMD Ryzen Threadripper 3960X 24-Core. Each experiment requires 12GB RAM and require 20GB available disk space for storage of the data.

H More Experimental Results

In this section, we provide more experimental results that are omitted from the main text due to space limitations.

H.1 Experiments on Localized Perturbations

In this experiment, we evaluate our localized filtering method under *locally perturbed* dynamics. Unlike the previous setting with *global* shifts (e.g., gravity or friction changes that affect transitions uniformly), here the dynamics vary across regions of the state space. This setting is more challenging because the distribution mismatch is heterogeneous and cannot be well addressed by methods that rely on global assumptions, such as DARA (Eysenbach et al., 2021) and IGDF (Wen et al., 2024).

We keep the target domain unchanged and perturb only the source dataset to simulate localized variations. Policies are trained on the perturbed source dataset and evaluated in the unchanged target domain. For the target dataset, we randomly sample 5,000 transitions from D4RL (Fu et al., 2020) without introducing any dynamics shift. For the source dataset, we start from the original D4RL dataset and partition the next-state space into 15 regions using K-means. We then divide the regions into M groups and inject Gaussian noise into state variables, with all samples in the same group sharing the same noise variance. We evaluate performance using the normalized score defined above. We consider all four MuJoCo environments (Ant, Hopper, HalfCheetah, Walker2d) and set $M \in \{3, 5\}$. When $M = 3$, the noise variances are $\{0.1, 0.5, 2.0\}$, $\{0.2, 1.0, 4.0\}$, and $\{0.3, 1.5, 6.0\}$. When $M = 5$, the variances are $\{0.1, 0.2, 0.3, 0.4, 0.5\}$, $\{0.2, 0.4, 0.6, 0.8, 1.0\}$, and $\{0.3, 0.6, 0.9, 1.2, 1.5\}$.

As shown in Table 6, LoDADA achieves a total score of 943.09, corresponding to a 29.3% improvement over the second-best approach OTDF and a 55.1% improvement over IQL. Across the 24 evaluation tasks (4 environments $\times$ 6 shift levels), LoDADA achieves best (19) or second-best (3) performance in 22 out of 24 tasks. In contrast, DARA and IGDF often underperform in this setting due to their reliance on global mismatch assumptions. **These results demonstrate that LoDADA is particularly effective under localized perturbations, where cluster-aware filtering becomes more beneficial than under global shifts.**

H.2 Results on Antmaze

In this section, we present results on the navigation tasks described in Section F. We compare LoDADA with baseline methods on the medium-sized AntMaze setting using the *diverse* source dataset across six target environments. The results are summarized in Table 7. As shown in Table 7, LoDADA consistently outperforms the baselines on the AntMaze-medium benchmark with the diverse dataset. Overall, LoDADA achieves the highest total score of 416.4, corresponding to an 8.4% improvement over the second-best method. LoDADA also achieves the best performance on 5 out of 6 maps and the second-best score on the remaining map. **These results demonstrate that LoDADA generalizes beyond locomotion tasks and achieves strong performance on navigation problems.**

In structured map environments, our localized filtering strategy effectively retains source transitions that are closer to the target transitions. Specifically, LoDADA only keep the source transitions that have similar map positions in state space with the transition in the target dataset. In contrast, prior methods such as DARA and IGDF often fail to preserve these informative transitions, limiting their effectiveness.

Additionally, we present experiment results on the AntMaze benchmark using the play source dataset. We follow the same evaluation procedure as in the diverse dataset setting and report normalized target environment performance across 6 AntMaze medium environments.The results are summarized in Table 8. As shown in Table 8, our proposed LoDADA achieves the best overall performance under

the play dataset, with a total score of 398.8, outperforming all baseline methods. Compared to the second-best method, LoDADA improves the total score by 5.3%, demonstrating its robustness under limited state coverage in the source dataset. Our method achieves the best performance on 3 out of 6 target maps and ranks second on 2 additional maps. Specifically, LoDADA performs best on maps $\{2, 3, 4\}$ and achieves second-best performance on maps $\{5, 6\}$.

H.3 Results under morphology shifts

As shown in Table 9, our method improves the total score to 597.88, yielding a 32.7% improvement over the second best approach OTDF, and a 63% improvement over backbone algorithm IQL. Across the 16 evaluation tasks (8 environments × 2 shift levels), LoDADA achieves best performance in 13 out of 16 tasks. These results indicate that our proposed approach consistently enhances robustness under morphology shifts. Relative to the baseline IQL, our method delivers consistent and significant performance gains across 13 out of 16 tasks.

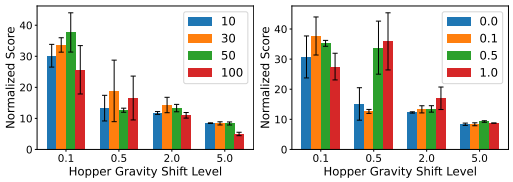
H.4 Results on Adroit tasks

We report the performance of our method and baseline approaches on the Adroit manipulation benchmarks in Table 10. As shown in Table 10, our proposed LoDADA consistently outperforms all baseline methods across both the Pen and Door tasks. Overall, LoDADA achieves the highest total score of 402.24, yielding a substantial improvement over the second-best method (OTDF) with a total score of 330.41. This demonstrates the strong robustness of our approach under severe morphology and kinematic mismatches in real-world application. These results indicate that LoDADA generalizes well beyond navigation tasks and achieves strong performance on challenging manipulation benchmarks. By leveraging localized dynamics-aware filtering, our method is able to retain source transitions that align well with the target dynamics, while prior methods relying on global assumption struggle under morphology and kinematic shifts.

H.5 Ablation study

In this section, we conduct several ablation studies to gain deeper insight into LoDADA and the effect of key hyperparameters in locomotion tasks. For demonstration, we study sensitivity in the Hopper environment in gravity shift with four shift levels (0.1, 0.5, 2.0, 5.0), focusing on the number of clusters K and the policy-regularization weight λ.

Number of clusters K. K controls the level of locality of the state space partition used in our cluster-aware filtering mechanism. A smaller K induces coarser clusters, which may merge heterogeneous transition dynamics, while a large K can lead to overly fine partitions with insufficient samples per cluster. To study this trade-off, we sweep $K \in \{10, 30, 50, 100\}$ and run LoDADA in the hopper-gravity environment with medium-level source and target datasets. As shown in Figure 9a, performance degrades when K is either too small (*e.g.*, 10) or too large (*e.g.*, 100), while intermediate values achieve consistently better results. **These results indicate that a moderate level of locality is critical for effective cluster-aware filtering, balancing dynamics homogeneity and data sufficiency.** As a result, we set $K = 30$ or 50 in all of our experiments.



(a) Ablation study on K. (b) Ablation study on λ.

Figure 9: Parameter sensitivity study.

Policy regularization weight λ. λ controls the strength of the regularization term in policy optimization, which encourages the learned policy to stay close to the target behavior policy. A larger λ enforces stronger regularization, potentially restricting policy improvement, while a smaller λ allows more aggressive policy updates but may incur instability or overfitting to imperfect filtered

data. To study this trade-off, we sweep $\lambda \in \{0, 0.1, 0.5, 1.0\}$ and run LoDADA in the hopper-gravity environment with medium-level source and target datasets. Results shown in Figure 9b demonstrate that diminishing the role of dataset regularization (*i.e.*, $\lambda = 0$) leads to poor performance. LoDADA is sensitive to the choice of λ and the best λ can vary for different tasks (*e.g.*, 0.1 for hopper-gravity-0.1 and 1.0 for hopper-gravity-0.5). However, as shown in Figure 9b, **choosing $\lambda = 0.5$ serves as a generally effective default policy regularization weight across different settings in MuJoCo.**

H.6 TIME COMPARISON OF DATA FILTERING METHODS

To understand what is the merit of LoDADA compared with other filtering methods, we study the runtime for the data filtering process between our method and IGDF (Wen et al., 2024) and OTDF (Lyu et al., 2025). We only compare the runtime of the filtering process instead of the whole training process as all these methods can share the same policy optimization backbone, so the main difference in runtime lies before the training begins. Specifically, we choose ant-gravity-0.5 environment and modify the replay buffer so that it samples without replacement, meaning it will go over the entire dataset per epoch. We fix the batch size to be 32 and number of epochs to be 7000 for fairness. As shown in Table 11, given that the target dataset is fixed, for OTDF, since it computes optimal transport pointwise between source and target datasets, its runtime significantly scales with data size, which makes it data-inefficient especially in real-world scenarios. For IGDF, while its runtime is the least in all three cases, based on results in Table 1 and Table 6, we argue that its performance is the poorest in all three filtering methods, indicating that the efficiency gains come at the cost of substantially degraded policy quality and its filtering mechanism is insufficient to capture reliable transitions under dynamics shifts. This further demonstrates the superiority of our method compared with other filtering methods by balancing model performance and data efficiency.

Table 5: Hyperparameters of LoDADA and baselines.

Hyperparameter	Value
Shared	
Actor network	(256, 256)
Critic network	(256, 256)
Learning rate	3×10^{-4}
Optimizer	Adam
Discount factor	0.99
Replay buffer size	10^6
Nonlinearity	ReLU
Target update rate	5×10^{-3}
Source domain batch size	128
Target domain batch size	128
DARA	
Temperature coefficient	0.2
Maximum log std	2
Minimum log std	-20
Classifier network	(256, 256)
Reward penalty coefficient λ	0.1
BOSA	
Temperature coefficient	0.2
Maximum log std	2
Minimum log std	-20
Policy regularization coefficient λ_{policy}	0.1
Transition coefficient $\lambda_{\text{transition}}$	0.1
Threshold parameter ϵ, ϵ'	$\log(0.01)$
Value weight ω	0.1
CVAE ensemble size	1 (behavior), 5 (dynamics)
IGDF	
Representation dimension	{16, 64}
Contrastive encoder network	(256, 256)
Encoder pretraining steps	7000
Importance coefficient	1.0
Data selection ratio ξ	75%
OTDF	
CVAE training steps	10000
CVAE learning rate	0.001
Number of sampled latents M	10
Gaussian std	$\sqrt{0.1}$
Cost function	cosine
Policy coefficient β	0.5
Data selection ratio ξ	80%
IQL	
Temperature coefficient	0.2
Maximum log std	2
Minimum log std	-20
Inverse temperature β	3.0
Expectile τ	0.7
LoDADA	
Number of clusters K	{30, 50}
Diameter threshold	{1.5, 5.0}
Filtering ratio (ξ_1, ξ_2, ξ_3)	(90%, 80%, 70%)
Input noise std σ	1.0
Importance weight α	1.0

Table 6: Performance comparison on MuJoCo tasks (Halfcheetah, Ant, Walker2d, Hopper) under a medium-level offline dataset. Target domains remain unchanged; source datasets are shifted by adding local perturbations with varying shift levels across regions. We report normalized target-domain scores (mean ± std over five seeds). Best and second-best scores are highlighted in green and blue, respectively.

Env	# Groups	Shift Level	BOSA	DARA	IQL	IGDF	OTDF	Ours
HalfCheetah	3	0.1-0.5-2.0	27.36±2.22	19.86±4.42	17.57±3.41	14.59±3.13	29.26±3.10	**34.97±0.85**
		0.2-1.0-4.0	25.28±3.10	16.94±3.31	16.07±3.45	13.16±4.83	24.21±2.82	**28.36±2.76**
		0.3-1.5-6.0	20.75±2.66	6.59±0.45	10.73±2.88	8.73±2.31	22.84±0.54	**27.24±4.37**
	5	0.1-0.2-0.3-0.4-0.5	19.30±2.38	12.43±1.95	10.83±1.05	11.27±2.35	28.99±1.85	**34.28±2.91**
		0.2-0.4-0.6-0.8-1.0	23.59±5.81	13.18±2.09	14.77±2.41	11.43±3.96	21.41±3.95	**25.73±5.38**
		0.3-0.6-0.9-1.2-1.5	**23.95±2.23**	7.10±0.97	8.98±2.61	7.37±3.03	11.53±2.04	18.74±1.55
Walker2d	3	0.1-0.5-2.0	5.28±0.54	17.85±10.67	36.44±15.19	14.02±10.20	34.52±11.33	**37.45±18.71**
		0.2-1.0-4.0	6.42±1.10	19.79±3.58	15.29±9.44	14.62±8.16	20.66±7.06	**31.10±18.00**
		0.3-1.5-6.0	7.16±2.91	9.86±4.44	12.28±8.14	11.14±3.69	16.78±9.39	**18.69±10.51**
	5	0.1-0.2-0.3-0.4-0.5	7.17±1.48	8.10±3.46	6.96±2.01	6.99±2.77	33.87±16.17	**35.84±16.01**
		0.2-0.4-0.6-0.8-1.0	6.51±4.60	13.90±9.23	12.07±7.55	9.37±2.01	13.87±6.95	**24.92±5.96**
		0.3-0.6-0.9-1.2-1.5	6.43±2.53	8.65±3.18	7.91±3.80	8.99±5.44	**18.52±15.46**	12.98±3.88
Hopper	3	0.1-0.5-2.0	7.84±2.84	8.70±6.67	8.14±6.31	13.08±10.44	33.96±4.57	**40.77±5.04**
		0.2-1.0-4.0	11.91±2.60	9.61±5.56	11.97±5.99	8.35±3.56	32.58±6.89	**37.51±9.16**
		0.3-1.5-6.0	11.85±3.19	8.11±3.54	10.17±5.50	6.59±2.59	31.77±3.04	**33.41±2.72**
	5	0.1-0.2-0.3-0.4-0.5	10.42±3.55	6.37±1.67	6.92±1.85	15.81±9.78	22.25±8.74	**40.30±4.61**
		0.2-0.4-0.6-0.8-1.0	11.63±1.87	14.07±5.65	9.82±4.29	11.24±7.86	21.42±9.11	**36.06±4.59**
		0.3-0.6-0.9-1.2-1.5	13.97±5.15	11.14±4.01	12.01±2.06	15.63±10.74	29.95±5.01	**35.02±5.70**
Ant	3	0.1-0.5-2.0	55.51±10.37	69.67±8.59	69.65±10.93	63.79±9.55	70.30±5.24	**85.28±6.52**
		0.2-1.0-4.0	31.75±10.39	54.02±5.97	59.71±7.84	55.91±7.68	54.55±6.35	**61.51±6.89**
		0.3-1.5-6.0	14.41±4.68	67.13±5.07	**69.03±9.18**	50.24±7.41	34.47±4.88	53.54±7.07
	5	0.1-0.2-0.3-0.4-0.5	44.36±8.50	75.03±7.08	66.50±8.95	63.40±10.26	58.51±10.29	**78.96±7.34**
		0.2-0.4-0.6-0.8-1.0	30.26±11.22	56.84±5.10	**67.07±4.40**	55.65±13.74	47.99±8.48	61.92±10.42
		0.3-0.6-0.9-1.2-1.5	13.14±5.58	**51.81±6.79**	46.98±9.14	51.36±8.39	15.23±4.79	48.51±9.77
Total			436.25	586.77	607.87	542.73	729.44	**943.09**

Table 7: Performance comparison on AntMaze tasks under the diverse source dataset with dynamic shifts in the map. Source domains remain unchanged; target domains are shifted. We report normalized target-domain scores (mean ± std over five seeds) with the mean of 50 episodes. Best and second-best scores are highlighted in green and blue, respectively.

Env	Shift Level	BOSA	DARA	IQL	IGDF	OTDF	Ours
Antmaze Medium	1	43.6 ± 11.2	53.6 ± 9.9	**76.5±8.4**	60.4 ± 9.1	51.6 ± 10.53	66.8 ± 4.6
	2	22.4 ± 5.5	33.6 ± 3.3	52.8 ± 7.9	45.2 ± 7.8	56.4 ± 4.34	**69.2±8.9**
	3	39.6 ± 11.5	50.8 ± 3.0	57.6 ± 7.7	65.2 ± 6.6	71.6±5.90	**73.2±7.3**
	4	40.0 ± 3.2	48.8 ± 8.3	58.4 ± 4.8	60.4 ± 6.2	52.4 ± 8.53	**65.2±5.9**
	5	21.2 ± 7.6	41.6 ± 12.7	66.4±8.3	56.8 ± 7.6	57.2 ± 7.16	**66.8±4.8**
	6	24.8 ± 14.0	59.2 ± 12.4	72.4 ± 10.7	73.6 ± 1.7	65.2 ± 9.86	**75.2±6.7**
Total		191.6	287.6	384.1	361.6	354.4	**416.4**

Table 8: Performance comparison on Antmaze tasks under the play offline dataset with dynamic shifts in the map. Source domains remain unchanged; target domains are shifted. We report normalized target-domain scores (mean ± std over five seeds). Best and second-best scores are highlighted in green and blue, respectively.

Env	Shift Level	BOSA	DARA	IQL	IGDF	OTDF	Ours
Antmaze Medium	1	33.6 ± 8.0	52.4 ± 7.8	**76.4±10.1**	58.0 ± 12.4	56.8 ± 6.87	52.8 ± 9.4
	2	25.2 ± 4.6	29.6 ± 6.2	46.8 ± 4.1	42.4 ± 8.2	63.2 ± 7.95	**67.6±6.7**
	3	30.0 ± 8.1	56.4 ± 11.1	66.4 ± 6.7	64.8 ± 7.0	69.6 ± 5.90	**71.6±3.8**
	4	34.0 ± 8.9	52.8 ± 6.3	48.8 ± 6.9	58.0 ± 1.4	44.8 ± 3.90	**71.6±6.8**
	5	22.8 ± 13.9	34.4 ± 9.3	**73.6±8.2**	54.4 ± 6.2	55.2 ± 3.03	60.8 ± 7.2
	6	12.8 ± 4.8	62.8 ± 3.0	66.8 ± 8.1	**74.8±4.1**	70.8 ± 5.93	74.4 ± 3.0
Total		158.4	288.4	378.8	352.4	360.4	**398.8**

Table 9: Performance comparison on HalfCheetah, Ant, Walker2d, and Hopper environments under **morphology shifts** across 5 seeds. We use **M** and **H** to denote the medium and hard levels of the dynamics shift.

Env	Type	Level	BOSA	IQL	DARA	IGDF	OTDF	Ours
HalfCheetah	morph-thigh	M	25.12±4.33	22.22±1.13	20.68±2.49	19.31±4.35	21.94±6.30	**26.42±3.44**
		H	20.74±3.22	15.90±1.47	16.35±0.97	15.57±2.06	22.75±2.76	**25.81±5.29**
	morph-torso	M	0.66±0.90	1.95±1.69	1.40±1.66	1.84±0.37	3.38±1.30	**9.96±4.07**
		H	16.44±12.82	**30.87±10.00**	29.89±5.88	28.85±3.89	26.06±6.00	23.44±4.82
Ant	morph-halflegs	M	64.52±7.46	67.15±4.33	65.82±5.31	69.07±5.76	73.86±0.79	**77.48±0.77**
		H	37.21±6.15	54.27±5.60	61.31±5.71	60.08±2.09	59.76±9.27	**64.47±4.62**
	morph-alllegs	M	31.09±7.09	30.05±7.41	26.49±4.52	37.87±17.24	32.98±4.91	**77.55±3.44**
		H	13.89±1.46	5.65±3.30	3.21±3.26	7.24±1.40	12.84±2.06	**24.25±0.02**
Walker2d	morph-torso	M	8.63±3.49	11.44±0.91	14.80±1.69	14.16±2.19	11.47±4.93	**29.67±11.80**
		H	1.93±0.53	4.78±1.13	6.86±2.38	5.38±1.34	7.05±0.57	**7.76±2.42**
	morph-leg	M	28.10±8.64	40.61±4.78	43.87±6.22	41.71±10.14	45.07±10.09	**52.86±8.82**
		H	10.17±1.60	20.54±5.17	20.33±3.49	18.78±3.84	15.13±3.25	**59.01±1.76**
Hopper	morph-foot	M	12.87±0.09	**25.62±18.89**	20.05±7.32	14.91±4.52	22.56±8.65	13.11±0.05
		H	9.84±0.17	10.72±2.84	25.74±10.70	18.35±8.08	66.78±9.79	**67.53±14.03**
	morph-torso	M	13.43±0.37	13.66±0.98	14.70±4.48	13.06±4.33	16.84±1.44	**27.81±9.08**
		H	11.28±0.24	11.30±0.44	10.82±0.19	10.66±1.19	**12.23±0.15**	10.75±0.89
Total			305.92	366.73	382.32	376.84	450.70	**597.88**

Table 10: Performance comparison on the Door and Pen environments under **morphology shifts** and **kinematic shifts** across 5 seeds. We use **M** and **H** to denote the medium and hard levels of the dynamics shift.

Env	Type	Level	BOSA	IQL	DARA	IGDF	OTDF	Ours
Pen	kin-broken-joint	M	42.56±11.31	45.46±8.54	46.17±6.26	50.74±11.20	52.14±6.51	**64.31±11.33**
		H	19.36±5.42	24.02±5.16	22.41±4.73	18.62±10.47	20.95±7.23	**28.16±18.04**
	morph-shrink-finger	M	10.33±2.79	15.64±2.97	15.33±2.39	13.19±4.41	18.02±5.64	**19.88±8.02**
		H	19.47±5.01	19.40±5.05	18.24±2.59	19.05±3.87	20.34±2.85	**33.06±11.48**
Door	kin-broken-joint	M	20.65±10.72	40.17±7.10	36.82±4.63	37.76±7.02	39.82±4.75	**55.12±12.50**
		H	19.80±15.61	52.37±5.52	49.31±5.53	52.61±4.83	58.24±6.37	**59.81±7.23**
	morph-shrink-finger	M	41.05±15.36	54.85±7.79	54.49±5.88	54.73±5.28	59.15±4.89	**71.48±8.80**
		H	38.92±18.85	61.84±4.16	58.91±3.05	61.07±6.89	61.74±3.24	**70.42±4.26**
Total			212.15	313.75	301.67	307.78	330.41	**402.24**

Table 11: Runtime comparison in Ant-Gravity-0.5 with different numbers of source samples.

# Sources	IGDF	OTDF	Ours
1000000	∼41.49s	∼4597.82s	∼134.62s
100000	∼38.71s	∼628.16s	∼80.18s
10000	∼55.64s	∼126.39s	∼72.92s