Efficiently Robust In-Context Reinforcement Learning with Adversarial Generalization and Adaptation

Juncheng Dong* **Hao-Lun Hsu*** **Miroslav Pajic†** **Vahid Tarokh†**
Duke University
{juncheng.dong, hao-lun.hsu, miroslav.pajic, vahid.tarokh}@duke.edu

Abstract

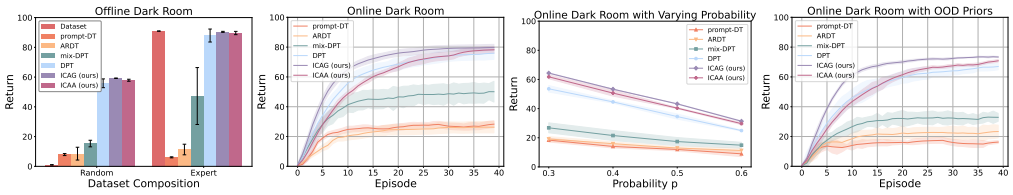
Transformer models (TMs) pretrained on diverse data exhibit strong in-context learning (ICL), enabling adaptation to new tasks without parameter updates. In reinforcement learning (RL), this motivates in-context RL (ICRL), where TMs are pretrained on diverse RL tasks to adapt to unseen ones. However, pretrained TMs are vulnerable to disturbances at deployment. We propose a pretraining framework that augments tasks with adversarial variations, improving robustness and generalization without the cost or pessimism of worst-case optimization. We further introduce an adaptive variant that uses the ICRL model itself to generate high-quality training data via online rollouts. Extensive experiments show our methods significantly improve robustness and generalization.

1 Introduction

Transformer models (TMs) pretrained on a massive amount of data have achieved remarkable successes spanning a wide range of application areas [1, 24, 37] with impressive in-context learning (ICL) capabilities, enabling them to solve unseen tasks using only a few examples *without parameter updates* [28]. Recent progress extends ICL to reinforcement learning (RL), where **in-context reinforcement learning** (ICRL) enables TMs to infer policies for new tasks by conditioning on a small set of trajectories. A common ICRL approach uses supervised pretraining over tasks $\{\tau^i\}_{i=1}^m$, drawn from a task distribution p_τ, with each task providing a context dataset D^i of trajectory data and an optimal action label $a_i^* \sim \pi_{\tau^i}^\star(s_{\text{query}}^i)$, where $\pi_{\tau^i}^\star$ is the optimal policy for τ^i and s_{query}^i is a query state. The pretraining dataset is $\mathcal{D}_{pre} = \{D^i, s_{\text{query}}^i, a_i^\star\}_{i=1}^m$ [16]. While effective, these models are brittle to deployment-time disturbances due to unseen distributional shifts. We address this with **In-context Adversarial Generalization** (ICAG), a novel ICRL framework built on the insight that *worst-case optimization is costly but not necessary* if TMs can generalize in-context to diverse disturbances. ICAG augments pretraining tasks with adversarial variants, promoting generalization across both tasks and perturbations. To reduce reliance on optimal labels, we introduce **In-context Adversarial Adaptation** (ICAA), which uses a pretrained model to generate high-quality (but not necessarily optimal) action labels via online rollouts, enabling efficient, iterative robustness improvement. Theoretically (Appendix G), we show ICAG supports implicit posterior sampling over tasks and adversaries, while ICAA approximates sequential supervised pretraining with no performance degradation under mild assumption. Empirically, ICAG and ICAA consistently improve robustness and generalization across multiple RL benchmarks.

*Equal contribution
†Equal advising

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: What Can('t) Transformers Do?.



(a) (b) (c) (d)

Figure 1: Performance on held-out Dark Room goals with average return over 10 random seeds. The error bar and the shaded area represent the standard error. (a) Offline evaluation given random and expert datasets. (b) Online evaluation without disturbances. (c) Online evaluation under disturbances with higher probability. (d) Online evaluation under disturbances with unseen priors.

2 In-Context Adversarial Generalization and Adaptation

We present ICAG and ICAA in a self-contained manner, with full details in the Appendices. Results on three benchmarks are provided, with only Dark Room [15] shown here due to space. In Figure 1(a), both methods outperform baselines, generalizing well without expert supervision. In Figure 1(b)–1(d), ICAG achieves the highest returns, particularly under unseen priors and stronger disturbances.

Our goal is to pretrain TMs to generalize over an *adversary-augmented* task space $\mathcal{M}_v := \{(\tau, \phi) : \tau \in \mathcal{M}; \phi \in \Phi(\tau)\}$, where $\Phi(\tau)$ represents a pre-specified set of adversaries for τ. Each pair $(\tau, \phi) \in \mathcal{M}_v$ defines a **variation environment**, an environment modified by an adversarial disturbance policy π_ϕ, which is treated as an integral part of the environment τ. Thus, τ and ϕ jointly create an environment. The resulting MDP can be defined as $(\mathcal{S}, \mathcal{A}, P_{\tau,\phi}, R_{\tau,\phi}, \gamma, \rho_\tau)$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $\gamma \in (0, 1)$ is the discount factor, and $\rho_\tau \in \Delta(\mathcal{S})$ is the initial state distribution. $P_{\tau,\phi}(s, a) = \mathbb{E}_{a' \sim \pi_\phi(s)}[P^G_\tau(s, a, a')]$ and $R_{\tau,\phi}(s, a) = \mathbb{E}_{a' \sim \pi_\phi(s)}[R^G_\tau(s, a, a')]$ with P^G_τ and R^G_τ defined as in the Markov Game formulation in Section C.1. To pretrain TMs to generalize across $\mathcal{M}_v$, we sample K adversarial variants $\phi^{i,k} \in \Phi(\tau^i)$ for each task τ^i, forming variation environments $\{(\tau^i, \phi^{i,k})\}_{i\in[m],k\in[K]}$. For every variation environment, we sample an in-context dataset D, a query state s_{query}, and an optimal action from the optimal policy $\pi^\star_{\tau,\phi}(s_{\text{query}})$, where $\pi^\star_{\tau,\phi} \in \operatorname{argmax}_\pi \mathcal{R}_\tau(\pi, \phi)$ avoids max-min optimization and supports parallel policy learning.

ICAG's reliance on optimal labels can be costly, as each (τ, ϕ) requires training *from scratch*. **In-Context Adversarial Adaptation (ICAA)** addresses this by using a pretrained model to generate high-quality (but not optimal) labels through interaction. ICAA starts with model T^0_θ trained on a dataset $\mathcal{D}^0$ (e.g., trajectories from original environments), then deploying it in each variation environment $(\tau^i, \phi^{i,k})$. The model interacts with the environment, producing a sequence of trajectories $\{\xi_n\}_{n=0}^N$ (see Section A for details). We construct a new pretraining dataset by: (1) using the first $\underline{N} < N$ trajectories as context datasets D_n, and (2) sampling query–action pairs $\{s^n_{\text{query}}, a_n\}_{n=0}^{\underline{N}}$ from later, higher-return trajectories to ensure label quality. Each variation yields data of the form $\{D_n = \xi_n, s^n_{\text{query}}, a_n\}_{n=0}^{\underline{N}}$, which are aggregated across all tasks and variations, combined with $\mathcal{D}^0$ into an updated dataset $\mathcal{D}^1$, and used to retrain the model to T^1_θ. Repeating this for J rounds produces a robust final model T^J_θ, without ever training task-specific policies from scratch.

3 Discussion and Conclusion

We propose ICAG and ICAA—two complementary frameworks that improve robustness and generalization in ICRL. ICAG augments pretraining with adversarial variants, while ICAA boosts data efficiency by letting a pretrained model adapt via self-generated rollouts. Together, they offer scalable, flexible tools for building robust ICRL agents ready for real-world deployment.

References

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[2] L. Busoniu, R. Babuska, and B. De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, 2008.

[3] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

[4] J. Dong, W. Mo, Z. Qi, C. Shi, E. X. Fang, and V. Tarokh. Pasta: pessimistic assortment optimization. In *International Conference on Machine Learning*, pages 8276–8295. PMLR, 2023.

[5] J. Dong, M. Guo, E. X. Fang, Z. Yang, and V. Tarokh. In-context reinforcement learning from suboptimal historical data. In *International conference on machine learning*. PMLR, 2025.

[6] J. Dong, H.-L. Hsu, Q. Gao, V. Tarokh, and M. Pajic. Variational adversarial training towards policies with improved robustness. In *Artificial intelligence and statistics*. PMLR, 2025.

[7] S. Fujimoto and S. S. Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

[8] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[9] H.-L. Hsu, H. Meng, S. Luo, J. Dong, V. Tarokh, and M. Pajic. Reforma: Robust reinforcement learning via adaptive adversary for drones flying under disturbances. In *2024 International Conference on Robotics and Automation (ICRA)*, 2024.

[10] G. N. Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 30(2): 257–280, 2005.

[11] Y. Jin, Z. Yang, and Z. Wang. Is pessimism provably efficient for offline rl? In *International Conference on Machine Learning*, pages 5084–5096. PMLR, 2021.

[12] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[13] R. Kidambi, A. Rajeswaran, P. Netrapalli, and T. Joachims. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33:21810–21823, 2020.

[14] A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

[15] M. Laskin, L. Wang, J. Oh, E. Parisotto, S. Spencer, R. Steigerwald, D. Strouse, S. Hansen, A. Filos, E. Brooks, et al. In-context reinforcement learning with algorithm distillation. *International Conference on Learning Representations*, 2023.

[16] J. Lee, A. Xie, A. Pacchiano, Y. Chandak, C. Finn, O. Nachum, and E. Brunskill. Supervised pretraining can learn in-context reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[17] S. Levine, A. Kumar, G. Tucker, and J. Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[18] W. Li, H. Luo, Z. Lin, C. Zhang, Z. Lu, and D. Ye. A survey on transformers in reinforcement learning. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL `https://openreview.net/forum?id=r30yuDPvf2`. Survey Certification.

[19] L. Lin, Y. Bai, and S. Mei. Transformers as decision makers: Provable in-context reinforcement learning via supervised pretraining. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=yN4Wv17ss3.

[20] Z. Liu, Z. Guo, Y. Yao, Z. Cen, W. Yu, T. Zhang, and D. Zhao. Constrained decision transformer for offline safe reinforcement learning. In *International Conference on Machine Learning*, pages 21611–21630. PMLR, 2023.

[21] T. Matsushima, H. Furuta, Y. Matsuo, O. Nachum, and S. Gu. Deployment-efficient reinforcement learning via model-based offline optimization. *arXiv preprint arXiv:2006.03647*, 2020.

[22] J. Moos, K. Hansel, H. Abdulsamad, S. Stark, D. Clever, and J. Peters. Robust reinforcement learning: A review of foundations and recent advances. *Machine Learning and Knowledge Extraction*, 4(1):276–315, 2022.

[23] A. Nilim and L. E. Ghaoui. Robust control of markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

[24] OpenAI, J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, R. Avila, I. Babuschkin, S. Balaji, V. Balcom, P. Baltescu, H. Bao, M. Bavarian, J. Belgum, I. Bello, J. Berdine, G. Bernadett-Shapiro, C. Berner, L. Bogdonoff, O. Boiko, M. Boyd, A.-L. Brakman, G. Brockman, T. Brooks, M. Brundage, K. Button, T. Cai, R. Campbell, A. Cann, B. Carey, C. Carlson, R. Carmichael, B. Chan, C. Chang, F. Chantzis, D. Chen, S. Chen, R. Chen, J. Chen, M. Chen, B. Chess, C. Cho, C. Chu, H. W. Chung, D. Cummings, J. Currier, Y. Dai, C. Decareaux, T. Degry, N. Deutsch, D. Deville, A. Dhar, D. Dohan, S. Dowling, S. Dunning, A. Ecoffet, A. Eleti, T. Eloundou, D. Farhi, L. Fedus, N. Felix, S. P. Fishman, J. Forte, I. Fulford, L. Gao, E. Georges, C. Gibson, V. Goel, T. Gogineni, G. Goh, R. Gontijo-Lopes, J. Gordon, M. Grafstein, S. Gray, R. Greene, J. Gross, S. S. Gu, Y. Guo, C. Hallacy, J. Han, J. Harris, Y. He, M. Heaton, J. Heidecke, C. Hesse, A. Hickey, W. Hickey, P. Hoeschele, B. Houghton, K. Hsu, S. Hu, X. Hu, J. Huizinga, S. Jain, S. Jain, J. Jang, A. Jiang, R. Jiang, H. Jin, D. Jin, S. Jomoto, B. Jonn, H. Jun, T. Kaftan, Łukasz Kaiser, A. Kamali, I. Kanitscheider, N. S. Keskar, T. Khan, L. Kilpatrick, J. W. Kim, C. Kim, Y. Kim, J. H. Kirchner, J. Kiros, M. Knight, D. Kokotajlo, Łukasz Kondraciuk, A. Kondrich, A. Konstantinidis, K. Kosic, G. Krueger, V. Kuo, M. Lampe, I. Lan, T. Lee, J. Leike, J. Leung, D. Levy, C. M. Li, R. Lim, M. Lin, S. Lin, M. Litwin, T. Lopez, R. Lowe, P. Lue, A. Makanju, K. Malfacini, S. Manning, T. Markov, Y. Markovski, B. Martin, K. Mayer, A. Mayne, B. McGrew, S. M. McKinney, C. McLeavey, P. McMillan, J. McNeil, D. Medina, A. Mehta, J. Menick, L. Metz, A. Mishchenko, P. Mishkin, V. Monaco, E. Morikawa, D. Mossing, T. Mu, M. Murati, O. Murk, D. Mély, A. Nair, R. Nakano, R. Nayak, A. Neelakantan, R. Ngo, H. Noh, L. Ouyang, C. O'Keefe, J. Pachocki, A. Paino, J. Palermo, A. Pantuliano, G. Parascandolo, J. Parish, E. Parparita, A. Passos, M. Pavlov, A. Peng, A. Perelman, F. de Avila Belbute Peres, M. Petrov, H. P. de Oliveira Pinto, Michael, Pokorny, M. Pokrass, V. H. Pong, T. Powell, A. Power, B. Power, E. Proehl, R. Puri, A. Radford, J. Rae, A. Ramesh, C. Raymond, F. Real, K. Rimbach, C. Ross, B. Rotsted, H. Roussez, N. Ryder, M. Saltarelli, T. Sanders, S. Santurkar, G. Sastry, H. Schmidt, D. Schnurr, J. Schulman, D. Selsam, K. Sheppard, T. Sherbakov, J. Shieh, S. Shoker, P. Shyam, S. Sidor, E. Sigler, M. Simens, J. Sitkin, K. Slama, I. Sohl, B. Sokolowsky, Y. Song, N. Staudacher, F. P. Such, N. Summers, I. Sutskever, J. Tang, N. Tezak, M. B. Thompson, P. Tillet, A. Tootoonchian, E. Tseng, P. Tuggle, N. Turley, J. Tworek, J. F. C. Uribe, A. Vallone, A. Vijayvergiya, C. Voss, C. Wainwright, J. J. Wang, A. Wang, B. Wang, J. Ward, J. Wei, C. Weinmann, A. Welihinda, P. Welinder, J. Weng, L. Weng, M. Wiethoff, D. Willner, C. Winter, S. Wolrich, H. Wong, L. Workman, S. Wu, J. Wu, M. Wu, K. Xiao, T. Xu, S. Yoo, K. Yu, Q. Yuan, W. Zaremba, R. Zellers, C. Zhang, M. Zhang, S. Zhao, T. Zheng, J. Zhuang, W. Zhuk, and B. Zoph. Gpt-4 technical report, 2024.

[25] I. Osband and B. Van Roy. Why is posterior sampling better than optimism for reinforcement learning? In *International conference on machine learning*, pages 2701–2710. PMLR, 2017.

[26] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta. Robust adversarial reinforcement learning. In *International conference on machine learning*, pages 2817–2826. PMLR, 2017.

[27] R. F. Prudencio, M. R. Maximo, and E. L. Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[28] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[29] A. Rajeswaran, S. Ghotra, B. Ravindran, and S. Levine. Epopt: Learning robust neural network policies using model ensembles. *International Conference on Learning Representations*, 2017.

[30] P. Rashidinejad, B. Zhu, C. Ma, J. Jiao, and S. Russell. Bridging offline reinforcement learning and imitation learning: A tale of pessimism. *Advances in Neural Information Processing Systems*, 34:11702–11716, 2021.

[31] A. Reddi, M. Tölle, J. Peters, G. Chalvatzaki, and C. D'Eramo. Robust adversarial reinforcement learning via bounded rationality curricula. *International Conference on Learning Representations*, 2024.

[32] F. Scarselli and A. C. Tsoi. Universal approximation using feedforward neural networks: A survey of some existing methods, and some new results. *Neural networks*, 11(1):15–37, 1998.

[33] T. Tanabe, R. Sato, K. Fukuchi, J. Sakuma, and Y. Akimoto. Max-min off-policy actor-critic method focusing on worst-case robustness to model misspecification. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=rcMG-hzYtR`.

[34] X. Tang, A. Marques, P. Kamalaruban, and I. Bogunovic. Adversarially robust decision transformer. *Advances in neural information processing systems*, 2024.

[35] C. Tessler, Y. Efroni, and S. Mannor. Action robust reinforcement learning and applications in continuous control. In *International Conference on Machine Learning*, pages 6215–6224. PMLR, 2019.

[36] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[37] H. Touvron, L. Martin, K. R. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. M. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. S. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. M. Kloumann, A. V. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288, 2023. URL `https://api.semanticscholar.org/CorpusID:259950998`.

[38] S. Tunyasuvunakool, A. Muldal, Y. Doron, S. Liu, S. Bohez, J. Merel, T. Erez, T. Lillicrap, N. Heess, and Y. Tassa. dm control: software and tasks for continuous control. *Software Impacts*, 6, 2020.

[39] H. Wang, Y. Pan, F. Sun, S. Liu, K. Talluri, G. Chen, and X. Li. Understanding the training and generalization of pretrained transformer for sequential decision making. *arXiv preprint arXiv:2405.14219*, 2024.

[40] T. Wu, S. He, J. Liu, S. Sun, K. Liu, Q.-L. Han, and Y. Tang. A brief overview of chatgpt: The history, status quo and potential future development. *IEEE/CAA Journal of Automatica Sinica*, 10(5):1122–1136, 2023.

[41] Y. Wu, G. Tucker, and O. Nachum. Behavior regularized offline reinforcement learning, 2019.

[42] S. M. Xie, A. Raghunathan, P. Liang, and T. Ma. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=RdJVFCHjUMI.

[43] M. Xu, Y. Shen, S. Zhang, Y. Lu, D. Zhao, J. Tenenbaum, and C. Gan. Prompting decision transformer for few-shot policy generalization. In *international conference on machine learning*, pages 24631–24645. PMLR, 2022.

[44] T. Yamagata, A. Khalil, and R. Santos-Rodriguez. Q-learning decision transformer: Leveraging dynamic programming for conditional sequence modelling in offline rl. In *International Conference on Machine Learning*, pages 38989–39007. PMLR, 2023.

[45] M. Yin and Y.-X. Wang. Towards instance-optimal offline reinforcement learning with pessimism. *Advances in neural information processing systems*, 34:4065–4078, 2021.

[46] W. Yuan, J. Chen, S. Chen, L. Lu, Z. Hu, P. Li, D. Feng, F. Liu, and J. Chen. Transformer in reinforcement learning for decision-making: A survey. *Authorea Preprints*, 2023.

[47] H. Zhang, H. Chen, C. Xiao, B. Li, M. Liu, D. Boning, and C.-J. Hsieh. Robust deep reinforcement learning against adversarial perturbations on state observations. *Advances in Neural Information Processing Systems*, 33:21024–21037, 2020.

[48] Q. Zheng, A. Zhang, and A. Grover. Online decision transformer. In *international conference on machine learning*, pages 27042–27059. PMLR, 2022.

[49] I. Zisman, V. Kurenkov, A. Nikulin, V. Sinii, and S. Kolesnikov. Emergence of in-context reinforcement learning from noise distillation. In *International conference on machine learning*. PMLR, 2024.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We provide both theoretical results and extensive empirical results to support our claims.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We talk about the limitation of our method in both Section 3 and Appendix K.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We provide full results in Appendix G and H.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Implementation details—including algorithms, model architectures, and hyperparameters—are provided in Appendix D. The experimental environment settings and the procedure for generating the pretraining dataset are described in Appendix E.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Implementation details—including algorithms, model architectures, and hyperparameters—are provided in Appendix D. The experimental environment settings and the procedure for generating the pretraining dataset are described in Appendix E.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We have multiple runs of each experiment. We report the mean value and standard error in the plots.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the GPU as well as the RAM and the computation time for each experiment.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our data is standard benchmarks in the RL research field.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: This work investigates general in-context reinforcement learning problems, and there is no direct societal impact.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We run the experiment on the simulated RL benchmarks; thus, no such issue exists.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We provide citations to all the data and related work in our paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We include the code in our paper. Also, details about the implementation are included in the appendix.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our experiments are conducted on the RL benchmarks and thus do not involve any crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our research and experiment don't require IRB as we conducted experiments on simulated RL benchmarks.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

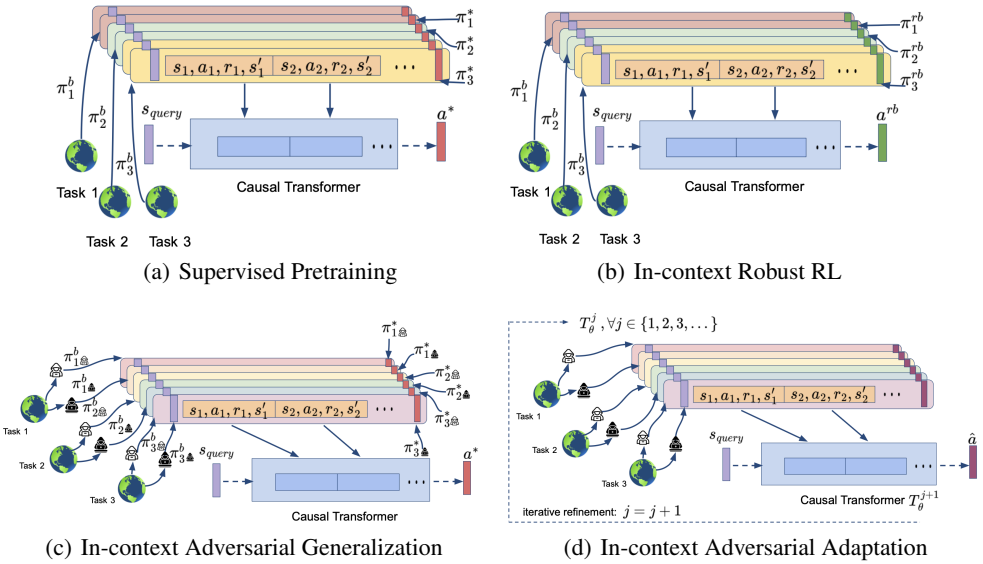
    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development and results do not involve LLMs. We only use LLM for editing (e.g., grammar, spelling, word choice)

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.



(a) Supervised Pretraining (b) In-context Robust RL

(c) In-context Adversarial Generalization (d) In-context Adversarial Adaptation

Figure 2: Overview of the supervised pretraining framework for ICRL across multiple tasks, where each task corresponds to an MDP instance τ^i, with $\tau^i \in \{1, 2, 3\}$ as illustrated. **(a)** A TM is pretrained to predict the optimal action across RL tasks [16], using optimal action labels $a^\star$ from expert policies $\pi^\star_{\tau^i}$, given query states s_{query} and offline trajectories collected from different RL tasks by distinct behavioral policies $\pi^b_{\tau^i}$. **(b)** To enhance policy robustness, a robust action label a^{rb} is sampled from the robust policy $\pi^{rb}_{\tau^i}$ rather than $\pi^*_{\tau^i}$. **(c)** To approximate robust learning without explicit max-min optimization, each task τ^i is augmented with K adversarial variations $\{\phi^{i,k}\}_{k=1}^K$ (e.g., black and white hackers), and the optimal action labels are derived from $\pi^\star_{\tau^i,\phi}$, representing the optimal policy under disturbance ϕ. **(d)** Alternatively, high-quality action labels $\hat{a}$ and trajectories are generated for K task variants via recursive online adaptation using pretrained models T^j_θ, for $j \in \{1, 2, 3, \ldots\}$.

Acknowledgments

This work is sponsored in part by the AFOSR under the award number FA9550-19-1-0169, and by the NSF under NAIAD Award 2332744 as well as the National AI Institute for Edge Computing Leveraging Next Generation Wireless Networks, Grant CNS-2112562.

A Preliminary

Markov Decision Process (MDP). Sequential decision-making tasks are modeled as MDPs [12, 2]. An MDP τ is defined by the tuple $(\mathcal{S}, \mathcal{A}, P_\tau, R_\tau, \gamma, \rho_\tau)$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $P_\tau : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the state transition function, $R_\tau : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $\gamma \in (0, 1)$ is the discount factor, and $\rho_\tau \in \Delta(\mathcal{S})$ is the initial state distribution. At each time step h, an agent selects an action $a_h \in \mathcal{A}$, receives a reward $r_h = R_\tau(s_h, a_h)$, and transitions to the next state s_{h+1} according to $P_\tau(s_h, a_h)$. A policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ maps the current state to an action distribution. The agent's goal is to learn the optimal policy $\pi^\star_\tau$ that maximizes the expected cumulative reward $G_\tau(\pi) = \mathbb{E}[\sum_{h=1}^\infty \gamma^{h-1} r_h \mid \pi, \tau]$ in τ.

Supervised Pretraining. Our approach builds on the DPT architecture [16], a supervised pretraining method designed to equip transformer models with ICRL capabilities. In DPT, a set of tasks $\{\tau^i\}_{i=1}^m$ is drawn from a task distribution p_τ. Each task $\tau^i \in \mathcal{M}$ represents an instance of an MDP where $\mathcal{M}$ is the space of all tasks of interest, and for each task, a context dataset D^i is generated from interactions between a behavioral policy and τ^i, i.e., $D^i = \{(s^i_h, a^i_h, s^i_{h+1}, r^i_h)\}_h$, where a^i_h is selected by the behavioral policy. For each task τ^i, a query state $s^i_{\text{query}} \in \mathcal{S}$ is chosen, and the optimal action $a^\star_i$ is sampled from $\pi^\star_{\tau^i}(s^i_{\text{query}})$, where $\pi^\star_{\tau^i}$ is the optimal policy for τ^i. The complete pretraining dataset

is then denoted as $\mathcal{D}_{pre} = \{D^i, s^i_{\text{query}}, a^\star_i\}_{i=1}^m$. Let T_θ represent a causal GPT-2 transformer with parameters θ [28]. The pretraining objective of DPT is formulated as follows:

$$\min_\theta \frac{1}{m} \sum_{i=1}^{m} -\log T_\theta \left(a^\star_i | s^i_{\text{query}}, D^i\right). \tag{1}$$

ICRL Deployment. The pretrained autoregressive TM T_θ can be deployed as an agent in both **offline** and **online** settings. During deployment, an unseen task τ is sampled from the task distribution p_τ. In offline deployment, a dataset D_{off} is first collected from τ, typically using trajectories generated by a random policy. Once available, DPT selects actions based on the policy $T_\theta(\cdot|s_h, D_{\text{off}})$ after observing the state s_h at time step h. For online deployment, DPT begins with an empty dataset D_{on}. At each episode, DPT follows $T_\theta(\cdot|s_h, D_{\text{on}})$ to collect a trajectory $\xi = \{s_1, a_1, r_1, \ldots, s_H, a_H, r_H\}$, which is appended to D_{on}. This process repeats for a predefined number of episodes. The pseudocode for this process is provided in Algorithm 1 in Appendix J.

B Related Work

Offline RL. Our work is situated within the broader field of offline RL [17, 21, 27]. Offline RL methods often employ approaches such as value pessimism or policy regularization to address the distributional shift between behavioral and optimal policies [41, 13, 14, 30, 45, 11, 4, 7]. While offline RL aims to solve the *same* tasks from which the offline datasets are collected, ICRL aims to efficiently generalize and adapt to *unseen* tasks.

Transformers for Decision Making. Autoregressive models [28, 1, 40, 37, 24] have achieved remarkable successes across various domains. Their application to sequential decision-making tasks, such as bandit and MDP problems, has shown transformers outperforming traditional methods [18, 46]. Decision Transformer (DT) [3, 48, 20, 44] formulates offline RL as return-conditioned supervised learning and scales well across multi-task settings. Methods like prompt-DT [43], Algorithm Distillation (AD) [15], and DPT [16] aim to enhance in-context generalization to new goals and tasks. Recent efforts [5, 49] further relax the assumptions on pretraining datasets by leveraging suboptimal trajectories or policies. However, challenges remain in addressing out-of-distribution (OOD) contexts with different environment dynamics and robustness to disturbances. To tackle these issues, we propose a transformer-based approach that leverages in-context learning to address the robustness problem and further utilize online adaptation to learn high-quality action without training separate policies for each adversarial task from scratch.

Robust RL and Adversarial Training. Robust reinforcement learning (RL) focuses on generalizing to out-of-distribution (OOD) environments by optimizing worst-case performance across various transition models [23, 10]. Deep RL methods commonly achieve robustness during training through parametric uncertainty, which considers a range of simulation parameters to optimize for worst-case performance [33, 29], or through adversarial training, which introduces perturbations to the environment (e.g., actions, observations, or transitions) to simulate potential deployment-time disturbances [26, 35, 47, 31, 9]. These methods enhance generalization to dynamics unseen during training but often sacrifice in-distribution (ID) performance.

C Extending Robust RL for Robust ICRL

To improve the robustness of ICRL methods, a natural starting point is to adapt existing robust RL techniques developed for standard RL settings to the ICRL context. In Section C.1, we briefly review **adversarial training**, a widely used technique for enhancing policy robustness in standard RL. Then, in Section C.2, we explore how adversarial training can be directly extended to ICRL within the supervised pretraining framework and highlight its **inherent limitations** in this setting. *These limitations, in turn, motivate our proposed frameworks* in Section 2, which leverage the in-context learning capabilities of transformer models to overcome these challenges more effectively.

C.1 Robust RL From Adversarial Training

While the objective of standard RL is to find a policy π_τ that performs well for a specific target MDP τ, robust RL aims to ensure performance even when the deployment MDP τ' differs from the pre-specified target τ [22].

Adversarial training is one of the most effective robust RL methods, which proposes to learn a policy robust to adversarial attacks and disturbances [26]. Adversarial training has shown great success in improving the policy robustness to both **(i)** environment mis-specification and **(ii)** external disturbances. It can be formulated as a **Markov Game**, defined by a tuple of 6 elements $\tau^{\mathrm{G}} = (\mathcal{S}, \mathcal{A}, \mathcal{A}^a, P_\tau^G, R_\tau^G, \gamma, \rho_\tau)$; here, $\mathcal{S}$, γ, and ρ_τ are defined as in the MDP formulation, representing the set of states, the discount factor, and the initial state distribution, respectively; $\mathcal{A}$ and $\mathcal{A}^a$ are respectively the sets of actions that the agent (protagonist) and the adversary can take; $P_\tau^G : \mathcal{S} \times \mathcal{A} \times \mathcal{A}^a \to \Delta(\mathcal{S})$ is the transition function that describes the distribution of the next state given the current state and actions taken by the agent and the adversary; $R_\tau^G : \mathcal{S} \times \mathcal{A} \times \mathcal{A}^a \to \mathbb{R}$ is the reward function for the agent. Consider a parameterized adversary $\pi_\phi^a : \mathcal{S} \to \Delta(\mathcal{A}^a)$ where ϕ is its parameter. We use $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ to denote the agent policy to learn. Let $s_h \in \mathcal{S}$ be the state of the environment, and let $a_h \in \mathcal{A}$ (respectively $a_h^a \in \mathcal{A}^a$) denote the action of the agent (respectively the adversary) at time step h. We use

$$\mathcal{R}_\tau(\pi, \phi) := \mathbb{E}\big[\sum\nolimits_{t=0}^{\infty} \gamma^t R_\tau^G(s_h, a_h, a_h^a) | s_0 \sim \rho_\tau\big] \tag{2}$$

where $a_h \sim \pi(s_h), a_h^a \sim \pi_\phi^a(s_h)$ to represent the cumulative discounted reward that the agent π can receive under the disturbance of the adversary following policy π_ϕ^a. For a specific target MDP τ, the **objective of adversarial training** for robustness is defined as

$$\pi_\tau^{rb} \in \operatorname*{argmax}_{\pi} \min_{\phi \in \Phi(\tau)} \mathcal{R}_\tau(\pi, \phi), \tag{3}$$

where $\Phi(\tau)$ is a pre-defined adversary space for τ. In this approach, the RL agent π optimizes the worst-case performance among all the adversarial disturbances from Φ.

C.2 In-Context Robust Reinforcement Learning

Although primarily designed for standard RL scenarios, the adversarial training approach can be directly adapted to ICRL within the supervised pretraining framework. We term this approach *In-context Robust RL* (IC2RL), which we illustrate in Figure 2(b) and elaborate next.

Specifically, given the pretraining tasks $\{\tau^i\}_{i=1}^m$, we can construct a dataset for supervised pretraining following two steps. In the first step, we solve for a robust policy $\pi_{\tau^i}^{rb}$ for each task τ^i, defined as

$$\pi_{\tau^i}^{rb} \in \operatorname*{argmax}_{\pi} \min_{\phi \in \Phi(\tau^i)} \mathcal{R}_{\tau^i}(\pi, \phi), \tag{4}$$

where $\Phi(\tau^i)$ is the adversary set for task τ^i. The second step mirrors the data collection process of supervised pretraining in Section A, except that the sampled query state $s_{\text{query}}^i \in \mathcal{S}$ is annotated with a **robust action label** a_i^{rb} drawn from the robust policy $\pi_{\tau^i}^{rb}(s_{\text{query}}^i)$ rather than the optimal policy $\pi_{\tau^i}^*$ used in standard supervised pretraining. Thus, the complete pretraining dataset for IC2RL is $\mathcal{D}_{pre}^{rb} = \{D^i, s_{\text{query}}^i, a_i^{rb}\}_{i=1}^m$, and we can pretrain a causal transformer with the same pretraining objective as in (1). In this approach, the TMs are pretrained to infer an action which would be taken by a robust RL policy, conditioning on a given context (thus the name In-Context Robust RL). By definition in (4), the robust policies optimize their worst-case performance under disturbances. However, as a straightforward extension of the regular robust RL approaches, this approach inherits a couple of *long-standing limitations* within robust RL and induces *considerable computation cost*.

First, the max-min problem in (4) poses a challenging optimization problem, especially in the current era of deep RL, where such problems are often non-convex and non-concave. Solving this problem requires substantial computational effort to approximate robust RL for a single task. While this computational burden may be manageable in standard robust RL settings with only one task of interest, the (often vast) number of RL tasks involved in ICRL pretraining renders the required computational cost prohibitively high. **Second**, the success of the max-min approach in robust RL relies on effective construction of the adversary parameter set. If not properly constructed, an adversary set which is too broad leads to over-pessimistic policies due to the worst-case optimization, and, on the other hand, an adversary set which is too limited leads to policies with insufficient robustness to disturbances and environment mis-specification [6].

D Implementation Details

D.1 Algorithm.

Soft Actor-Critic (SAC) [8]. Soft Actor-Critic (SAC) is an off-policy reinforcement learning algorithm designed to balance exploration and exploitation by optimizing the trade-off between expected rewards and action entropy. SAC aims to learn a policy that not only maximizes long-term rewards but also encourages exploration by maximizing the entropy of the policy's action distribution. The algorithm utilizes an actor network to select actions, along with two critic networks that estimate the Q-values of state-action pairs. The learning objective is to maximize a soft Bellman equation: $J(\pi) = \sum_h \mathbb{E}_{(s_h,a_h)\sim D}[Q(s_h, a_h) - \alpha \log \pi(a_h|s_h)]$, where $Q(s_h, a_h)$ represents the value of the state-action pair as estimated by the critics, α is a temperature parameter controlling the exploration-exploitation balance, and $\pi(a_h|s_h)$ is the probability of selecting action a_h in state s_h. SAC is trained by sampling mini-batches from a replay buffer to update both the policy and the Q-value estimates. We use the implementation without adversarial training from Reddi et al. [31].

Decision-Pretrained Transformer (DPT) [16]. The Decision-Pretrained Transformer (DPT) is designed for in-context learning in reinforcement learning (RL) tasks by leveraging supervised pretraining. The key idea behind DPT is to train a transformer model to predict optimal actions for a given query state, using an in-context dataset that includes interactions from a variety of tasks. These interactions are represented as state-action-reward tuples, which provide the necessary context for decision-making. During pretraining, DPT samples a distribution of tasks. For each task τ_i, an in-context dataset D^i, is constructed, consisting of sequences of state-action-reward transitions that reflect prior experience with that task. A query state s^i_{query} is then sampled from the MDP's state space, and the model is trained to predict the optimal action based on both the query state and the task-specific context D^i. The training objective is to minimize the expected loss over the sampled task distribution, where the model learns to predict a distribution of actions given the query state and context. DPT shares the same set of finite environments as ICAG and ICAA, but its pretraining dataset does not consider robustness.

Mixed Decision-Pretrained Transformer (mix-DPT) [39]. The mix-DPT framework extends DPT by splitting the learning process into two phases: the early training phase and the mixed training phase, addressing the out-of-distribution (OOD) issue between training and testing. During the early training phase, data are generated using a pre-specified decision function f, such as a random policy for the Dark Room task or SAC-trained policies for Meta-World and MuJoCo control. In the mixed training phase, data are generated using both the function f and the current DPT model, with the proportion controlled by a hyper-parameter κ. In the experiments conducted by Wang et al. [39], historical trajectories include the optimal actions for each time step. To ensure a fair comparison, we follow their setup but use only a query state and the corresponding optimal action, as done in DPT and ICAG. For our experiments, we follow the parameter choice $\kappa = 1/3$ from Wang et al. [39], with the total number of training iterations set to 40% of the overall training iterations.

Prompt-based Decision Transformer (prompt-DT) [43]. Prompt-DT builds upon Decision Transformer [3] and organizes its data to enable few-shot policy generalization through the use of trajectory prompts. For each task T_i, a prompt τ_i^* of length K^* is constructed from a set of few-shot demonstrations P_i, which consist of state-action-reward-to-go tuples $(s^*, a^*, \hat{G}^*)$. This prompt captures the task-specific context required for policy adaptation. To further enrich the context, the most recent trajectory history τ_i, sampled from an offline dataset D_i, is appended to the task-specific prompt. This forms the complete input sequence τ_{input} input. Specifically, the input sequence is represented as: $\tau_{input} = (\tau_i^*, \tau_i)$. This sequence consists of $3(K^* + K)$ tockens, following the state-action-reward format. The full sequence τ_{input} is then processed by a Transformer model, which autoregressively predicts the next actions corresponding to each state token. We follow the original prompt-DT setup and set $k = 20$. Prompt-DT uses the same pretraining dataset as DPT, but lacks query state-action pairs, highlighting the architecture's effectiveness in DPT-based methods (e.g., DPT, ICAG, ICAA).

Adversarially Robust Decision Transformer (ARDT) [34]. ARDT enhances the Decision Transformer by associating worst-case returns-to-go via minimax expectile regression with trajectories to improve robustness against adversarial perturbations. Specifically, the estimated Q values from expectile regression replaces the returns-to-go in vanilla DT during training.

Robust Adversarial Reinforcement Learning (RARL) [26]. RARL trains a protagonist to compete against destabilizing forces introduced by an adversary in a zero-sum Markov game, where the optimal strategy (i.e., the rational strategy) corresponds to a Nash equilibrium. In this setup, the protagonist selects actions to maximize performance, while the adversary is trained to take actions that minimize the same performance metric. By training under these destabilizing perturbations, the protagonist learns to develop robust skills that help it counter distribution shifts and adversarial attacks when deployed in real-world scenarios. We implement RARL using the framework from Reddi et al. [31], where both the protagonist and adversary are represented as agents with SAC policies.

Quantal Adversarial Reinforcement Learning (QARL) [31]. QARL formulates a robust adversarial reinforcement learning objective with entropy regularization, designed to model Markov games under bounded rationality. It introduces two temperature coefficients for the Shannon entropy of both the protagonist's and adversary's action distributions, allowing the optimization problem between the two players to be framed as a Quantal Response Equilibrium (QRE). QRE is a generalization of the Nash equilibrium, extending it to scenarios where agents may not act with complete rationality. We implement QARL using the framework from Reddi et al. [31], where both the protagonist and adversary are represented by SAC-based agents.

D.2 Model Architecture and Hyper-parameters

Transformer. Our models, DPT, MDPT, and prompt-DT are all based on a causal GPT-2 architecture [28]. It consists of 4 attention layers, each with a single attention head. In DarkRoom, the embedding size is 32, while Meta-World and MuJoCo environments' embedding size are 256. Prompt-DT and ARDT are built on Decision Transformer, separating the individual (s, a, s', r) into their own embeddings to be made into one long sequence. The remaining transformer-based baselines and our models view the transition tuples in the dataset as their own singletons, to be related with other singletons in the dataset through the attention mechanism. We use the AdamW optimizer with a weight decay of $1e-4$, a learning rate of $1e-3$, and a batch size of 128.

Multilayer perceptron (MLP). For all non-transformer agents, e.g., SAC, RARL, QARL, we directly list their shared hyper-parameters and architecture in Table 1.

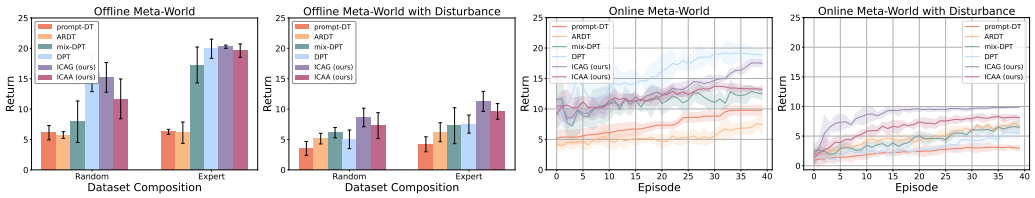
Table 1: Non-transformer agent hyper-parameters (e.g, SAC, RARL, QARL)

Hyper-parameters	Values
No of hidden layers	3
No of hidden units per layer	256
activation function	ReLU
optimizer (actor and critic)	Adam
actor learning rate	$1 \cdot 10^{-4}$
critic learning rate	$3 \cdot 10^{-4}$
initial replay memory size	$3 \cdot 10^{3}$
max replay memory size	$1 \cdot 10^{6}$
warmup transitions	$5 \cdot 10^{3}$
batch size	256
target smoothing coefficient	$5 \cdot 10^{-3}$
initial temperature	$5 \cdot 10^{3}$
temperature learning rate	$3 \cdot 10^{-4}$

E Environment Settings and Pretraining Dataset

E.1 Darkroom.

Dark Room is a sparse-reward navigation task set in a discrete 10×10 grid. At the beginning of each episode, the agent is randomly placed in one of the grid cells, while the goal location is hidden and



(a) (b) (c) (d)

Figure 3: Performance on held-out Meta-World goals with average return over 10 random seeds. The error bar and the shaded area represent the standard error. (a) Offline evaluation given random and expert datasets. (b) Offline evaluation given random and expert datasets under disturbances. (c) Online evaluation without disturbances. (d) Online evaluation under disturbances.

fixed at a random cell. The agent receives an observation of its current (x, y) position and selects from five discrete actions: move left, right, up, down, or stay in place. The episode horizon is $H = 100$ steps. The agent receives a reward of $r = 1$ only when it reaches the goal, and $r = 0$ otherwise. At test time, the agent always starts from the origin $(0, 0)$.

To construct the pretraining dataset, we generate $100,000$ trajectories using a uniform-random policy, evenly distributed across 100 different goal locations. For each, we sample query states uniformly and compute optimal actions using a heuristic that first aligns the agent's y-coordinate with the goal and then the x-coordinate. The dataset is partitioned into $80,000$ training examples (corresponding to 80 goal locations) and $20,000$ validation examples (corresponding to the remaining 20 goals).

To improve robustness, we augment the environment with adversarial perturbations. Specifically, for each of the m training tasks $\{\tau^i\}_{i=1}^m$, we construct a set of perturbed environments $\mathcal{M}_v = \{(\tau^i, \phi^{i,k})\}_{i\in[m],k\in[K]}$ by allowing an adversarial action to override the agent's intended action with probability $p \sim \mathcal{U}(0.0, 0.2)$. The adversarial action is sampled from a Dirichlet distribution by priors $\boldsymbol{\alpha}$, with the most probable action selected. With probability $1 - p$, the agent executes its original action. This construction models each perturbed task as a probabilistic MDP [35], capturing structured uncertainty in action dynamics.

The Dirichlet distribution used satisfies $\sum_{i=1}^I x_i = 1$ and $x_i \in [0, 1]$ for all i. The probability density function is given by:

$$f(x_1, ..., x_I; \alpha_1, ..., \alpha_I) = \frac{1}{\beta(\boldsymbol{\alpha})} \prod_{i=1}^I x_i^{\alpha_{i-1}}.$$

, where $\beta(\boldsymbol{\alpha})$ is the multivariate beta function, which can be expressed in terms of the gamma function

$$\beta(\boldsymbol{\alpha}) = \frac{\prod_{i=1}^I \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^I \alpha_i)}, \qquad \boldsymbol{\alpha} = (\alpha_1, ..., \alpha_I).$$

These perturbed environments are used to generate additional trajectories in the style of DPT, increasing the diversity and robustness of the pretraining dataset.

E.2 Meta-World.

In the Meta-World benchmark, the agent is tasked with controlling a robotic hand to reach target positions in 3D space. Meta-World includes 20 tasks in total. To assess the generalization capability of our approach to unseen reinforcement learning tasks, we train on 15 tasks and evaluate on the remaining 5. Offline performance is evaluated using both random and expert trajectories, as shown in Figure 3(a) and Figure 3(b), while online performance is reported in Figure 3(c) and Figure 3(d). Across both offline and online settings, we observe that all methods exhibit performance degradation under state disturbances. Notably, while DPT suffers a significant drop in performance, ICAG and ICAA maintain higher robustness, consistently outperforming other methods under perturbation.

We focus on the ML1 pick-place benchmark, where the agent must grasp an object and place it at a designated target location. Each task is defined by a 39-dimensional state space that includes the gripper's position and binary open/close state, the 3D pose of the object, and the coordinates of the target. The agent operates in a continuous action space, allowing it to adjust its end-effector position in three dimensions and control the gripper's open/close state to facilitate object manipulation.

The environment provides shaped rewards to guide learning, including incentives for approaching the object, establishing a grasp, transporting the object, and releasing it at the target. The goal is to learn a policy that effectively sequences these skills to solve the overall manipulation task. For generalization evaluation, we train on 15 task configurations and test on 5 held-out tasks with novel object and target positions.

To construct the pretraining dataset, we collect historical trajectories from agents trained using Soft Actor-Critic (SAC). For each task, SAC is trained until convergence, and we sample from the resulting trajectories to form offline datasets. The built-in deterministic policy is treated as the optimal expert policy.

To simulate real-world deployment challenges, we introduce robustness through adversarial perturbations to the observation space. Specifically, for each task τ^i, we construct K perturbed variations $\{(\tau^i, \phi^{i,k})\}_{i\in[m],k\in[K]}$, where each $\phi^{i,k}$ is implemented as a fixed-weight neural network with bounded magnitude ($|\phi^{i,k}| \leq 1$). These perturbations target only the perceptually estimated components of the observation—namely, object pose, end-effector position, and goal location—while avoiding internal robot states such as joint angles or velocities, which are typically less affected by real-world noise. This design mimics sensor uncertainty (e.g., from depth cameras or visual drift) without destabilizing control signals that are precise on physical hardware.

E.3 MuJoCo.

We evaluate our methods on a diverse set of continuous control tasks from the DeepMind Control Suite [38], covering 6 environments and 11 tasks (see Appendix E). Pretraining datasets are built from historical trajectories of SAC agents. To improve robustness during ICAG supervised pretraining, we introduce K fixed adversaries $\{\phi^{i,k}\}_{k=1}^{K}$ for each task τ^i, each modeled by a separate neural network. SAC is trained to convergence under adversarial disturbances $\phi^{i,k}$, and the resulting trajectories form variation environments $\{(\tau^i, \phi^{i,k})\}_{i\in[m],k\in[K]}$ for m pretraining tasks. Additional dataset details are in Appendix E.

The MuJoCo environments [36] used in our experiments are standard implementations from the DeepMind Control Suite [38]. Specifically, we consider modified versions of these environments that incorporate adversarial settings (i.e., RARL, QARL, and pretraining for ICAG). In each environment, adversarial actions and force magnitudes are carefully selected to challenge the agent and promote robust behavior. The adversarial action spaces are intentionally designed to differ from those of the protagonist agent to exploit domain knowledge. The adversary forces are calibrated to be sufficiently large to foster agent robustness and generalization, while still posing a significant challenge to the protagonist.

Table 2: MuJoCo Environment-specific parameters for adversarial training and (protagonist) agent's standard observation/action space

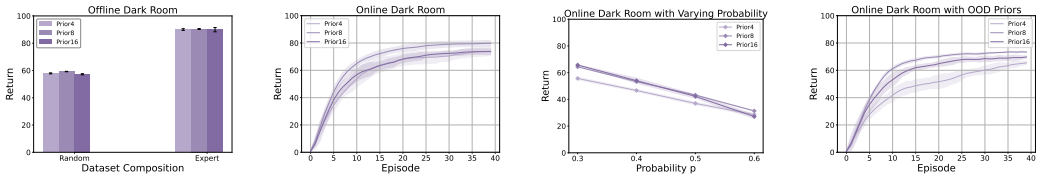
Environment	Adversary max force	Adversary action space	Observation space	Action space
Cartpole	0.005	2D forces on pole (1)	5	1
Cheetah	1.0	2D force on feet & torso (6)	17	6
Hopper	1.0	2D force on feet & torso (4)	15	4
Quadruped	10	3D force on torso (1)	78	12
Reacher	0.1	2D force on arm (2)	6	2
Walker	1.0	2D force on feet (4)	24	6

We conduct experiments in 6 environments shown in Figure 4. Then environment-specific parameters, along with the corresponding observation and action spaces for the standard environments, are detailed in Table 2. For all environments, the discount factor is set to 0.9 and the horizon is set to 200.

Each environment is associated with one or more problems, defined as instances of the model with specific Markov Decision Process (MDP) structures. For example, in the CartPole environment,

(a) Cartpole (b) Cheetah (c) Hopper (d) Quadruped (e) Reacher (f) Walker

Figure 4: Illustrations of the MuJoCo environments.



(a) (b) (c) (d)

Figure 5: Ablation study for ICAG with different number of training priors on held-out Dark Room goals from test tasks with average return over 10 random seeds. The error bar and the shaded area represent the standard error. (a) Offline evaluation given random and expert datasets. (b) Online evaluation without disturbances. (c) Online evaluation under disturbances with higher probability. (d) Online evaluation under disturbances with unseen priors. Note that prior $K \in \{4, 8, 16\}$ in each subfigure implies K adversaries involved in pretraining dataset.

we define two problems: **swingup**, where the pole starts pointing downward, and **balance**, where the pole begins near the upright position. The goal in both problems is to manipulate the forces applied to a cart at the base in order to either swing up or balance an unactuated pole. In the Reacher environment, a two-link planar arm must reach a randomized target location, with a reward of 1 when the end effector reaches the target sphere. Two problems are defined here: in the **easy** problem, the target sphere is larger than in the **hard** problem.

In the Hopper environment, a planar one-legged hopper is initialized in a random configuration and is rewarded based on torso height and forward velocity. The remaining three environments: Walker, Cheetah, and Quadruped focus on maximizing forward velocity. In the Cheetah environment, the reward is linearly proportional to the forward velocity, capped at a maximum of $10m/s$. The Walker environment includes two tasks: **walk** and **run**, which differ in their velocity requirements and include components to encourage upright posture and minimal lateral movement. For standard in-context reinforcement learning (ICRL) methods, such as DPT [16], We use m pretraining tasks $\{\tau^i\}_{i=1}^m \subset \mathcal{M}$ with varying internal conditions. For each task τ^i, we construct a pretraining dataset using 6000 historical trajectories collected from SAC agents trained to convergence.

We generate K variations of each pretraining task τ^i, incorporating adversaries $\phi^{i,k}$ to form a set of variation environments $\{(\tau^i, \phi^{i,k})\}_{i\in[m],k\in[K]}$. In our experiments, $m = 16$ tasks (from a 4×4 grid) and $K = 10$ variations, with hyper-parameter tuning in Appendix F.3.

We follow the SAC training approach in Reddi et al. [31], training $K = 10$ SAC policies for each τ^i under different variations, which are treated as distinct tasks. Each variation is initialized as a fixed-weight neural network ($|\phi^{i,k}| \leq 1$), unlike RARL or QARL, where adversarial policies are updated during training. These variations act as adversaries, with SAC policies trained under disturbance.

F Additional Experimental Results

F.1 Ablation on number of priors for adversaries in Dark Room

We conduct ablation studies to assess the impact of varying the number of adversaries $\{\phi^{i,k}\}_{k=1}^K$ with different Dirichlet priors in the pretraining dataset to ICAG. The total number of pretraining trajectories remains consistent across different priors. Specifically, we explore $K \in \{4, 8, 16\}$ with varying $\boldsymbol{\alpha}$ values (see Appendix E.1), and present the results in Figure 5.

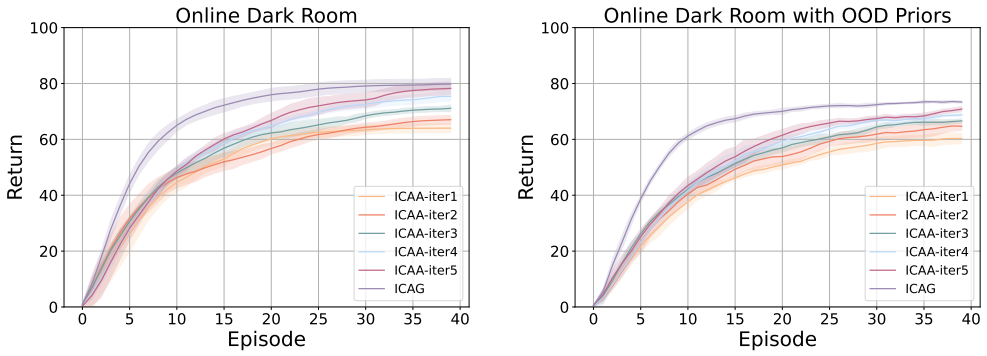


Figure 6: Performance on held-out Dark Room goals with average return over 10 random seeds among our methods. Note that ICAA-iter1 denotes the online evaluation with T_θ^1. The error bar and the shaded area represent the standard error. (a) Online evaluation without disturbances. (b) Online evaluation under disturbances with unseen priors.

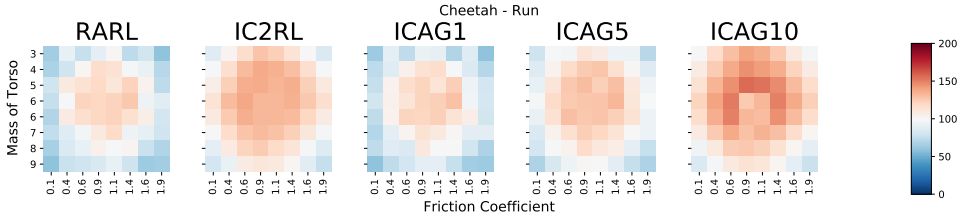


Figure 7: Generalization capability mainly among IC2RL and ICAG with varying K variations, where ICAG K, with $K \in \{1, 5, 10\}$.

In offline evaluation (shown in Figure 5(a)), both random and expert trajectories yield similar performance across different K values. For online evaluation, Figure 5(b) shows results without disturbances, while Figure 5(d) introduces disturbances with a probability of $p = 0.2$, which were absent during pretraining. When fewer adversaries are used (e.g., $K = 4$), the agent requires more episodes to adapt to environments with unseen priors, as indicated in Figure 5(d).

Finally, we evaluate all methods with a single adversary using a uniform distribution for disturbances, varying the disturbance probability in Figure 5(c). Based on these results, we select $K = 8$ for a better balance between online and offline performance, and use it for comparisons with other baselines in Figure 1.

F.2 Ablation on the Number of ICAA Rounds in Dark Room

In Algorithm 3, ICAA employs an online approach to generate high-quality action labels and iteratively refine the model T_θ^J using the dataset $\mathcal{D}^J$, thereby reducing dependence on optimal action labels. We present the online evaluation results both with and without disturbances in Figure 6, and compare them against ICAG. Here, ICAA-iter1 represents the online evaluation using the model T_θ^1. The results demonstrate that increasing the number of refinement iterations consistently improves performance under both without and with disturbances. However, the performance of ICAA remains bounded above by ICAG, as ICAG directly leverages optimal action labels.

F.3 Variant of ICAG and IC2RL

As discussed in Appendix C.2, extending traditional robust adversarial RL approaches to in-context settings presents challenges. To explore this, we conduct experiments in the Cheetah environment, where for each task τ^i with 16 pairs of internal conditions, we learn a robust policy via RARL, as shown in Figure 2(b). In Figure 7, we demonstrate that IC2RL improves RARL performance, but it is relatively impractical, solving max-min optimizations for all tasks. Additionally, we investigate the

effect of varying the number of adversaries K in $\{(\tau^i, \phi^{i,k})\}_{i\in[m],k\in[K]}$. We show that increasing K acts as a data augmentation technique, supporting our statement in Appendix F.3 that the augmented task space $\mathcal{M}_v$ covers the original task space $\mathcal{M}$ ($\mathcal{M} \subset \mathcal{M}_v$), leading to better generalization. Notably, ICAG10 with $K = 10$, as shown in Figure 7, outperforms IC2RL.

ICAG Augments Tasks. On top of improving the robustness of TMs for ICRL, ICAG can also be viewed as a data augmentation method. In particular, consider the adversary set $\Phi(\tau)$, which includes the null disturbance ϕ_0 such that the MDP of the variation environment (τ, ϕ_0) remains identical to the original MDP τ. This can be trivially achieved by simply omitting an adversary when augmenting the pretraining tasks. In this case, the augmented task space $\mathcal{M}_v$ covers the original task space $\mathcal{M}$, i.e., $\mathcal{M} \subset \mathcal{M}_v$. Thus, ICAG effectively *increase the diversity and number of pretraining tasks*, leading to improved ICRL performance.

F.4 MuJoCo Performance and Robustness

Notably, we use SAC agents in an adversarial training framework in both RARL and QARL, making SAC a natural baseline. Our adversarial setup follows [31], with environment-specific parameters detailed in Table 2. We investigate two types of robustness:

Robustness to Agent Disturbance. We measure this by evaluating learned policies under a worst-case adversary, which minimizes the agent's return without modifying the policy parameters.

Robustness to Environmental Change. We evaluate how well policies generalize to environments with variations in physical parameters such as mass and friction. Generalization is assessed by deploying policies in environments with shifted dynamics coefficients.

To analyze performance, we group MuJoCo tasks into three domains: cartpole, reacher, and locomotion (e.g., quadruped, cheetah, walker, hopper). For all experiments, both standard RL baselines (SAC and QARL) and ICRL methods are deployed in the same training environments used by SAC, favoring the former. *This gives SAC and QARL significant advantages* and explains why all ICRL methods other than ours are outperformed by QARL while *ICAG consistently outperforms QARL and other ICRL methods in all evaluations*.

Performance metrics are visualized through box plots and heatmaps in Figure 8, Figure 9, Figure 10, Figure 11, Figure 12, and Figure 13. Note: PDT denotes prompt-DT [43], and MDPT refers to mix-DPT [39]. Specifically, ICAG improves robustness against adversaries compared to DPT. Consistent with findings in standard robust RL [26, 6], we observe that training for robustness can also improve performance in disturbance-free settings. ICAG particularly excels in high-dimensional control problems, such as the quadruped task (Figure 10).

Furthermore, ICAG demonstrates strong generalization under out-of-distribution (OOD) environmental changes. In Figure 11, Figure 12, and Figure 13, we evaluate how methods adapt when tested on parameter grids not seen during training. Each 8×8 heatmap varies two internal environment parameters, while the pretraining dataset only covers the central 4×4 grid. ICAG consistently achieves the best performance on these OOD environments, outperforming all baselines in tests such as Figure 13(b) and Figure 13(d).

$$\pi^\star_{\tau,\phi} \in \operatorname*{argmax}_{\pi} \mathcal{R}_\tau(\pi, \phi). \tag{5}$$

G Theoretical Results

Here we provide theoretical guarantees for ICAG to gain further insights into their efficacy. Complete proofs of results in this section can be found in Appendix H.

We present two results. Our **first** result show that ICAG addresses the adversaries (disturbances) encountered during deployment in a manner similar to Posterior Sampling (PS) [25], which is widely recognized as the most sample-efficient algorithm for many sequential decision-making problems. Our **second** result shows that ICAA continuously improve the quality of its action labels so that we can improve the performance of ICRL models in each ICAA iteration.

To facilitate analysis, we consider a slightly modified supervised pretraining framework similar to Lee et al. [16] and Lin et al. [19] for ICAG where, for any variation environment (τ, ϕ), the

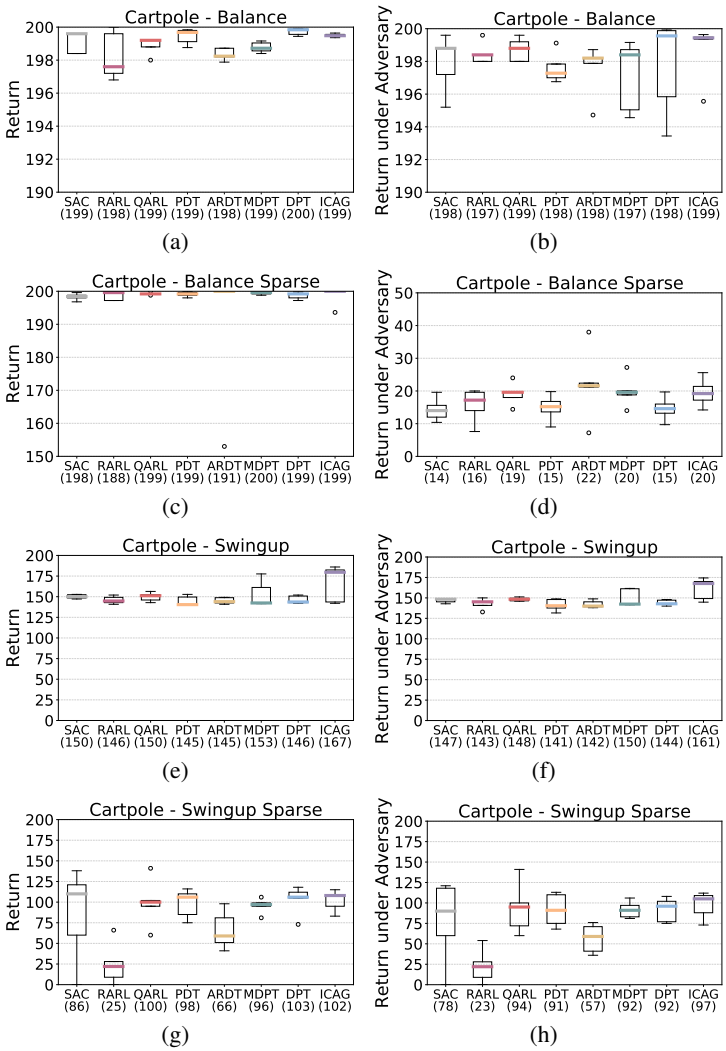


Figure 8: Performance and robustness on Cartpole problems (see the title of each boxplot), which are evaluated at the end of training without an adversary (left column) and against an adversary (right column). The number next to the name of each algorithm is the average performance across 10 seeds..

TMs also condition on a sequence $\zeta_h = (s_1, a_1^\star, s_2, \ldots, s_h, a_h^\star)$ where $s_{1:h}$ follows the distribution $p_s \in \Delta(\mathcal{S}^h)$ and $a_j^\star \sim \pi_{\tau,\phi}^\star(s_h)$ where $\pi_{\tau,\phi}^\star$ is the optimal policy for (τ, ϕ) as defined in (5). Thus, the joint distribution over $(\tau, \phi, D, s_{\text{query}}, \zeta_h)$ for ICAG pretraining is

$$P(\tau, \phi, D, s_{\text{query}}, \zeta_h) = p_\tau(\tau) p_{\Phi(\tau)}(\phi) p_D(D; \tau, \phi) \\ p_{\text{query}}(s_{\text{query}}) p_s(s_{1:h}) \prod_{j=1}^{h} \pi_{\tau,\phi}^\star(a_j^\star | s_h), \tag{6}$$

where p_τ and $p_{\Phi(\tau)}$ are the sampling distributions of the environment and disturbance for ICAG, respectively; $p_D(D; \tau, \phi)$ is the distribution of context dataset D given the variation environment (τ, ϕ); p_{query} is the distribution for sampling query states. Given the joint distribution in (6), posterior distributions such as $P(\tau, \phi | D)$ are well-defined.

Consider the following general PS process for a fixed task τ' and disturbance ϕ': initialize the posterior distribution as the ICRL pretraining distribution $p_{\tau,\phi}^{(1)} = p_\tau p_{\Phi(\tau)}$, and initialize an empty dataset D to collect transitions; for $h \in \{1, \ldots, H\}$: ***(i)*** sample a variation environment $(\tau^{(h)}, \phi^{(h)}) \sim p_{\tau,\phi}^{(h)}$; ***(ii)*** solve for $\pi_{\tau^{(h)},\phi^{(h)}}^\star$; ***(iii)*** given the current state $s^{(h)}$, take an action following $a^{(h)} \sim \pi_{\tau^{(h)},\phi^{(h)}}^\star(s^{(h)})$,

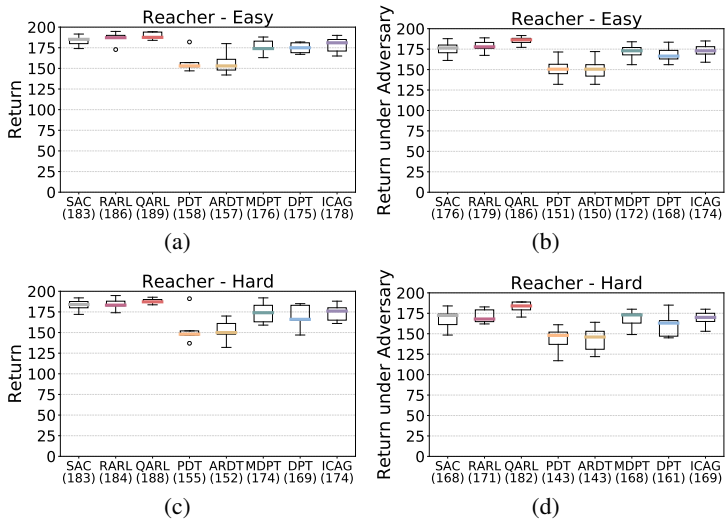


Figure 9: Performance and robustness on Reacher problems (see the title of each boxplot), which are evaluated at the end of training without an adversary (left column) and against an adversary (right column). The number next to the name of each algorithm is the average performance across 10 seeds.

and observe the reward $r^{(h)}$ and next state $s^{(h+1)}$; ***(iv)*** add the transition $(s^{(h)}, a^{(h)}, r^{(h)}, s^{(h+1)})$ into D, and update the posterior $p_{\tau,\phi}^{(h+1)} = P(\tau, \phi | D)$.

Note that although PS is provably sample-efficient, computing the posterior $P(\tau, \phi|D)$ is often intractable in practice. Next, *we prove that ICAG is an implicit PS*: during deployment, ICAG takes actions like the PS process above, *inferring the underlying environment and adversary without explicitly computing the posterior*. We first make some common mild assumptions for analysis [16].

Assumption G.1. Consider the context dataset $D = \{s_h, a_h, r_h, s_{h+1}\}_h$. The actions a_h are conditionally independent of the variation environment (τ, ϕ) given the history, i.e., $p_D(a_h|s_h, D_{h-1}) = p_D(a_h|s_h, D_{h-1}, \tau, \phi)$ where $D_h = \{s_{h'}, a_{h'}, r_{h'}, s_{h'+1}\}_{h' \le h}$.

This assumption in essence assumes that the behavioral policies for collecting context datasets are functions of the history only. This holds, for example, when the context dataset is collected by random policies not depending on current states or any RL algorithms only using the history.

Assumption G.2. Consider a sufficiently expressive and pretrained TM T_θ. For all $(s_{\text{query}}, D, \zeta_h)$, $T_\theta(a|s_{\text{query}}, D, \zeta_h) = P(a|s_{\text{query}}, D, \zeta_h)$ for all a.

The purpose of Assumption G.2, which states that the pretrained TM T_θ matches the pretraining distribution P, is to focus the analysis on ICRL deployment rather than the quality of pretraining. This assumption is a common assumption for ICRL [16] and in-context learning analysis [42]. To see why this assumption is valid, it is well-established that deep learning models, such as T_θ, are universal approximators [32]. Moreover, the maximum likelihood (ML)-based pretraining loss for T_θ is equivalent to find a minimizer of the following expected Kullback–Leibler (KL) divergence

$$\mathbb{E}_{(s_{\text{query}}, D, \zeta_h) \sim P} \left[\text{KL} \left(P(a|s_{\text{query}}, D, \zeta_h) \| T_\theta(a|s_{\text{query}}, D, \zeta_h) \right) \right],$$

where $\text{KL}(p, q) = \mathbb{E}_p[\log(p/q)]$ is the KL divergence between two distributions. Assuming sufficient expressiveness, the above divergence can be minimized at $T_\theta = P$. Thus, with extra coverage assumptions regarding P and sufficient amount of pretraining data, Assumption G.2 can hold with T_θ and P arbitrarily close to each other. However, we omit this proof as this would distract the focus of the analysis. Next, we present our main theoretical results.

Theorem G.3. *Fix an environment τ' with a disturbance ϕ' for deployment and a context dataset $D \sim p_D(D; \tau', \phi')$ for the pretrained TM T_θ to condition on for ICRL. Consider the random sequence $\Upsilon_h = (S^{(1)}, A^{(1)}, S^{(2)}, A^{(2)}, \ldots, S^{(h)}, A^{(h)})$. It holds that*

$$P_{PS}(\Upsilon_h | \tau', \phi', D) = P_\theta(\Upsilon_h | \tau', \phi', D),$$

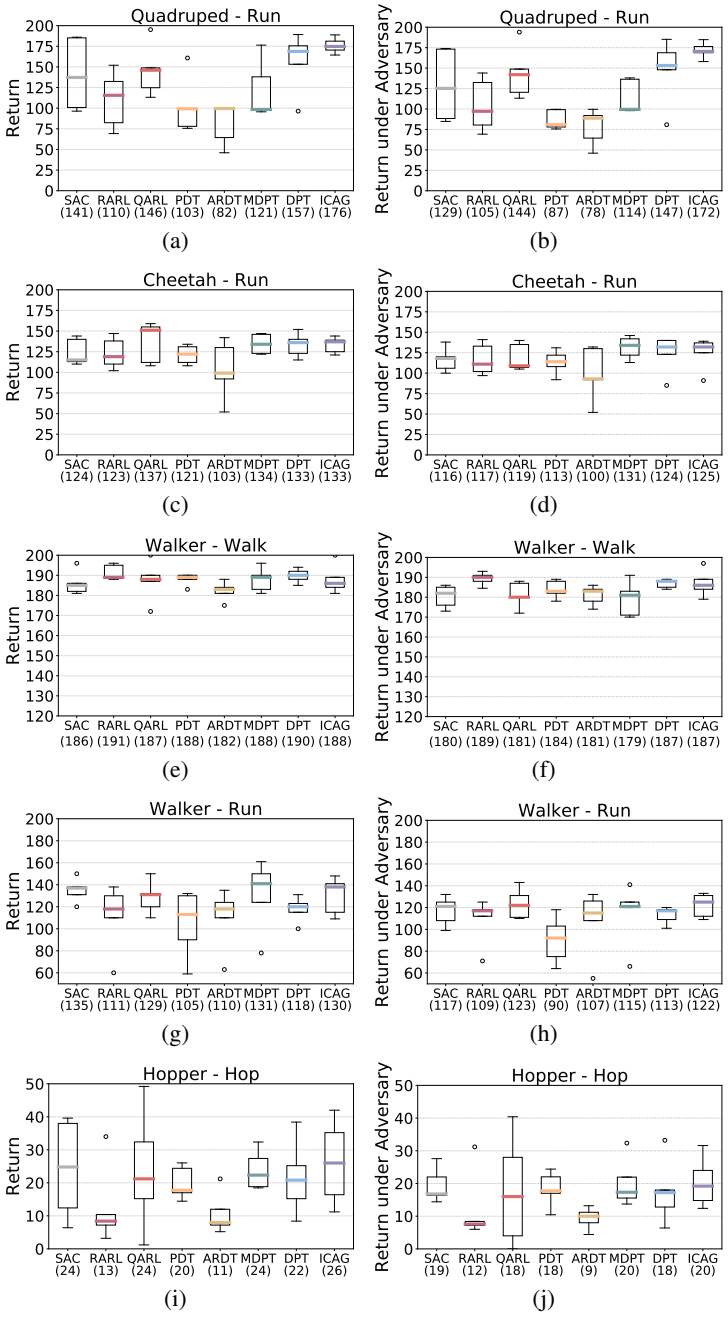


Figure 10: Performance and robustness on locomotion problems, i.e., Quadruped, Cheetah, Walker, and Hopper (see the title of each boxplot), which are evaluated at the end of training without an adversary (left column) and against an adversary (right column). The number next to the name of each algorithm is the average performance across 10 seeds.

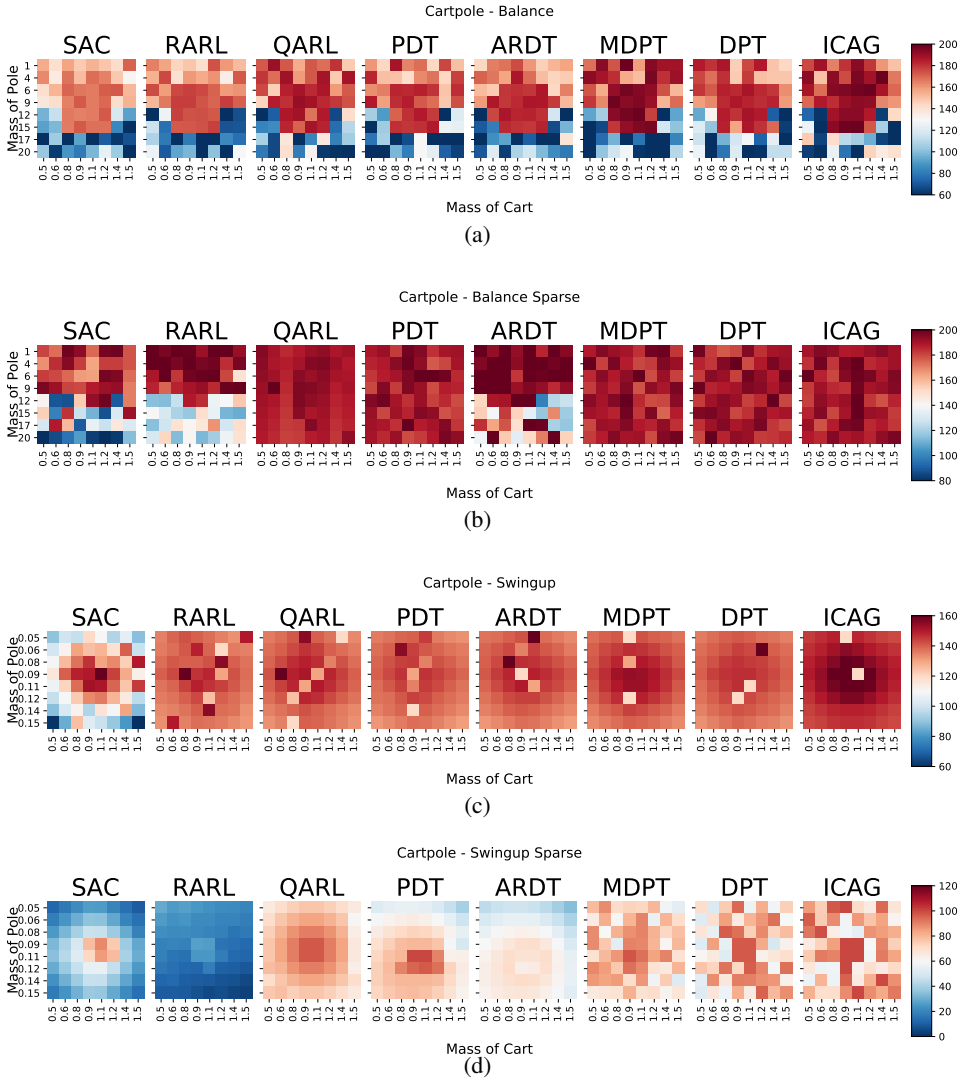


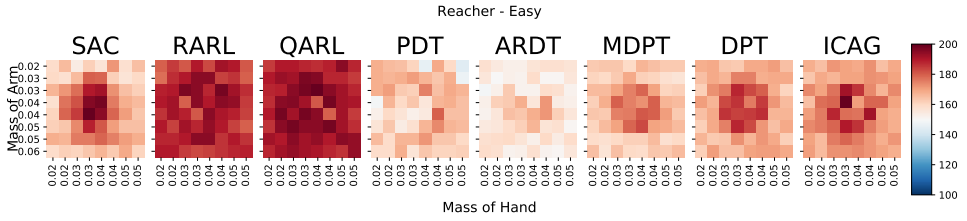
Figure 11: Robustness analysis on Cartpole (see the title of each heatmap set). Heatmaps show the performance obtained after training for varying properties of the environment, described in the $x - y$ axes. The number next to the name of each algorithm is the average performance across 10 seeds.

where (i) $P_{PS}(\Upsilon_h|\tau', \phi', D)$ *is the distribution of* Υ_h *following the PS algorithm:* $(\tau_{ps}, \phi_{ps}) \sim P(\tau, \phi|D)$, $S^{(1)} \sim \rho_{\tau',\phi'}$, *for all* $h' \leq h$, $A^{(h')} \sim \pi^{\star}_{\tau_{ps},\phi_{ps}}(S^{(h')})$, $S^{h'+1} \sim P_{\tau',\phi'}(S^{(h)}, A^{(h)})$; *(ii)* $P_\theta(\Upsilon_h|\tau', \phi', D)$ *is the distribution of* Υ_h *following ICRL with* T_θ*:* $S^{(1)} \sim \rho_{\tau',\phi'}$, *for all* $h' \leq h$, $A^{(h')} \sim T_\theta(a|S^{(h')}, D, \Upsilon_{h'-1})$, *and* $S^{h'+1} \sim P_{\tau',\phi'}(S^{(h')}, A^{(h')})$.

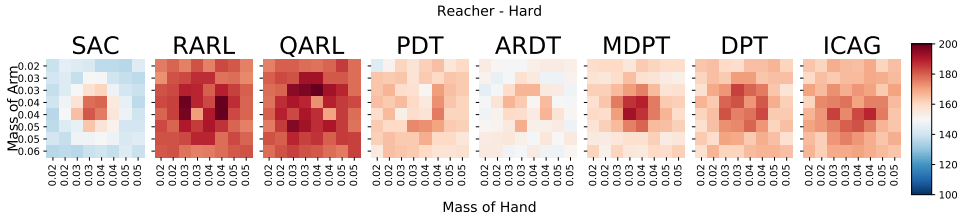
Theorem G.3 states that the trajectory distribution under ICRL with the pretrained TM T_θ is the same as the trajectory distribution under PS, establishing that ICAG pretrains TMs for implicit PS. In particular, ICAG implicitly estimates the posterior distribution of the environment τ and the adversary ϕ so that *ICAG can act optimally if there exists an adversary* ϕ *that can perturb the environment* τ. Moreover, this process of estimating and adapting to a potential adversary is provably sample-efficient given the optimal sample efficiency of PS.

Next, we show that ICAA can *refine its action labels in an iterative manner*.

Assumption G.4 (Posterior Consistency)**.** For any pair of underlying task and adversary $\tau^\star, \phi^\star$, as the context dataset D contains more transitions, the posterior $P(\tau, \phi|D)$ concentrates toward the true task and adversary, i.e., for any neighbor U of $(\tau^\star, \phi^\star)$, it holds that $P(U|D) \to 1 \quad \text{as} \quad |D| \to +\infty$.



(a)



(b)

Figure 12: Robustness analysis on Reacher (see the title of each heatmap set). Each heatmap shows the performance obtained after training for varying properties of the environment, described in the $x - y$ axes. The number next to the name of each algorithm is the average performance across 10 seeds.

Assumption G.4 is a standard assumption and, in general, a fact for any Bayesian method. Following Theorem G.3 that ICRL models are performing Posterior Sampling, this implies that the transformer policy T_θ has increasing performance when the context size $|D|$ increases.

Theorem G.5. *Under Assumptions G.1, G.2 and G.4, in every iteration of ICAA (Algorithm 3), and for every variation environment (τ, ϕ) with sufficient exploration, the action label generation policy achieves performance no worse than the transformer policy T_θ from the previous iteration within the same environment (τ, ϕ).*

In particular, by matching the action labels generated by a stronger policy for the variation environment (τ, ϕ) in the finetuning stage of each ICAA iterations, the performance of transformer policy T_θ increases and can generalize better to new tasks and adversaries. Theorem G.5 implies that *ICAA continues to improve the quality of its action labels until it saturates*.

H Proofs of Theoretical Results

H.1 Proof of Theorem G.3

Fix an environment τ' with a disturbance ϕ' for deployment and a context dataset $D \sim p_D(D; \tau', \phi')$ for the pretrained TM T_θ to condition on for ICRL. Consider the random sequence $\Upsilon_h = (S^{(1)}, A^{(1)}, S^{(2)}, A^{(2)}, \ldots, S^{(h)}, A^{(h)})$. It holds that

$$P_{PS}(\Upsilon_h | \tau', \phi', D) = P_\theta(\Upsilon_h | \tau', \phi', D),$$

where (i) $P_{PS}(\Upsilon_h | \tau', \phi', D)$ is the distribution of Υ_h following the PS algorithm: $(\tau_{ps}, \phi_{ps}) \sim P(\tau, \phi | D)$, $S^{(1)} \sim \rho_{\tau', \phi'}$, for all $h' \leq h$, $A^{(h')} \sim \pi^\star_{\tau_{ps}, \phi_{ps}}(S^{(h')})$, $S^{h'+1} \sim P_{\tau', \phi'}(S^{(h)}, A^{(h)})$; (ii) $P_\theta(\Upsilon_h | \tau', \phi', D)$ is the distribution of Υ_h following ICRL with T_θ: $S^{(1)} \sim \rho_{\tau', \phi'}$, for all $h' \leq h$, $A^{(h')} \sim T_\theta(a | S^{(h')}, D, \Upsilon_{h'-1})$, and $S^{h'+1} \sim P_{\tau', \phi'}(S^{(h')}, A^{(h')})$.

Proof of Theorem G.3. The proof is based on induction on h. Given that the results are for any fixed τ', ϕ', D, we omit the conditional dependence on them to improve clarity when there is no confusion. Recall that P denotes the ICAG pretraining dataset distribution as defined in (6). We first prove a result to be used in the proof. Under Assumption G.1, we have

$$P_{PS}(\tau_{ps} = ps, \phi_{ps} = \phi | D) = P(\tau, \phi | D), \tag{7}$$

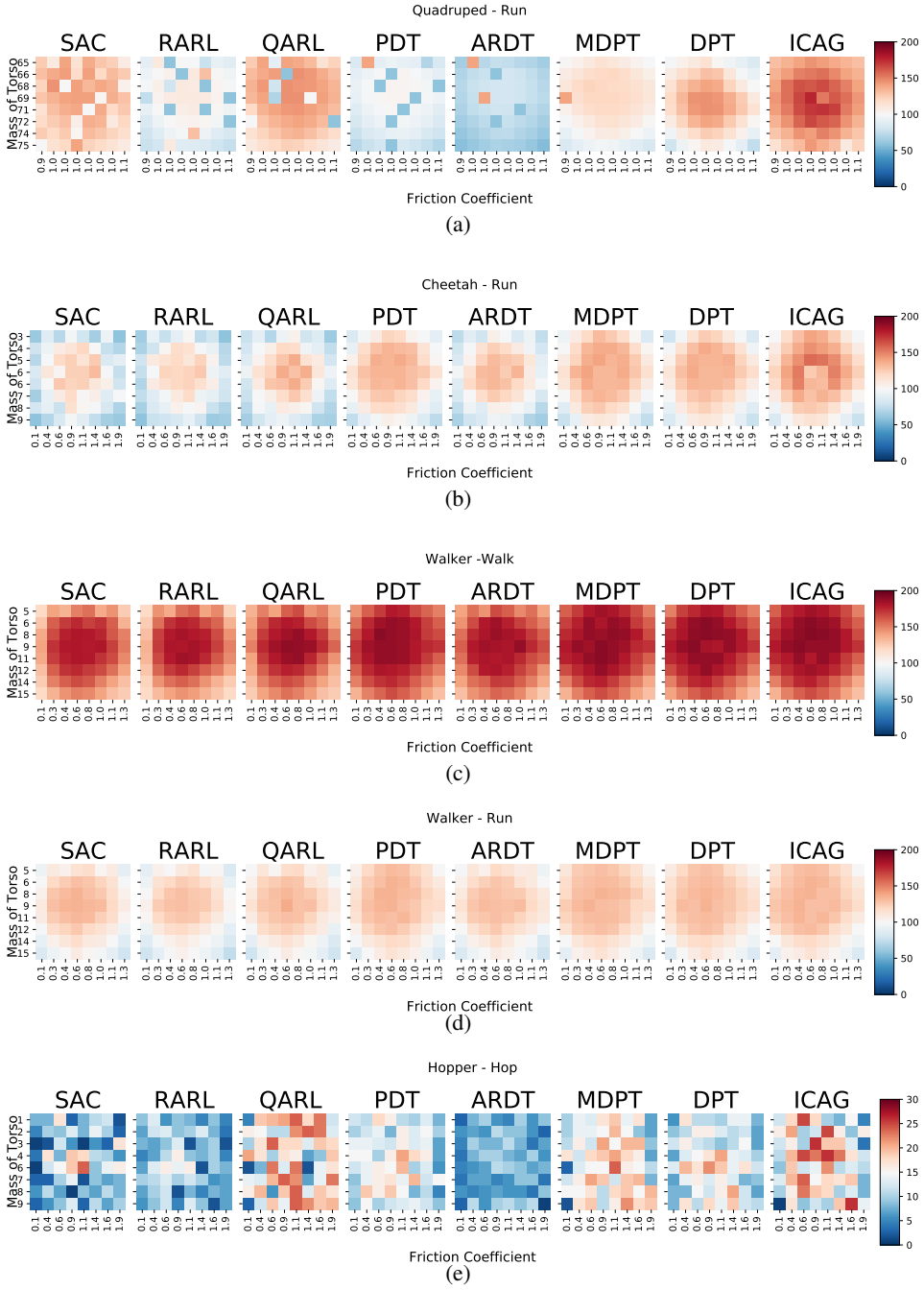


Figure 13: Robustness analysis on locomotion problems, i.e., Quadruped, Cheetah, Walker, and Hopper (see the title of each heatmap set). Each heatmap shows the performance obtained after training for varying properties of the environment, described in the $x - y$ axes. The number next to the name of each algorithm is the average performance across 10 seeds.

where $P(\tau|D)$ is the posterior distribution of the ICAG pretraining distribution P. This follows from

$$\begin{aligned}
P_{PS}(\tau_{ps}=\tau,\phi_{ps}=\phi|D) &\propto P_{PS}(\tau_{ps},\phi_{ps}=\phi,D) = P(\tau,\phi)P_{PS}(D|\tau,\phi)\\
&\propto P(\tau,\phi)\rho_{\tau',\phi'}(s_1)\prod_{(s_h,a_h,r_h,s_{h+1})\in D} P_{\tau',\phi'}(s_{h+1}|s_h,a_h)p_D(a_h|D_{h-1})\\
&\propto P(\tau,\phi)P(D;\tau;\phi) = P(\tau,\phi,D) \propto P(\tau,\phi|D),
\end{aligned}$$

where the second $\propto$ is due to Assumption G.1 so that we can plug in $p_D(a_h|D_{h-1})$. Now we begin the proof of induction.

Case $h=1$. We have

$$\begin{aligned}
P_{PS}(S^{(1)},A^{(1)}) &= P_{PS}(S^{(1)})P_{PS}(A^{(1)}|S^{(1)}) = p_{\tau',\phi'}(S^{(1)})\int_{\tau,\phi} P_{PS}(A^{(1)},\tau_{ps}=\tau,\phi_{ps}=\phi|S^{(1)})d\tau d\phi\\
&= p_{\tau',\phi'}(S^{(1)})\int_{\tau} P_{PS}(A^{(1)}|S^{(1)},\tau_{ps}=\tau,\phi_{ps}=\phi)P_{PS}(\tau_{ps}=\tau,\phi_{ps}=\phi|S^{(1)})d\tau d\phi\\
&= p_{\tau',\phi'}(S^{(1)})\int_{\tau} \pi^{\star}_{\tau,\phi}(A^{(1)}|S^{(1)})P_{PS}(\tau_{ps}=\tau,\phi_{ps}=\phi|S^{(1)})d\tau d\phi.
\end{aligned}$$

To continue, note that $P_{PS}(\tau_{ps}=\tau,\phi_{ps}=\phi|S^{(1)}) = P_{PS}(\tau_{ps}=\tau,\phi_{ps}=\phi|D)$ given that the posterior sampling does not depend on $S^{(1)}$. With (7), we further have

$$P_{PS}(\tau_{ps}=\tau,\phi_{ps}=\phi|D) = P(\tau,\phi|D).$$

Thus,

$$\begin{aligned}
P_{PS}(S^{(1)},A^{(1)}) &= p_{\tau',\phi'}(S^{(1)})\int_{\tau,\phi} \pi^{\star}_{\tau,\phi}(A^{(1)}|S^{(1)})P(\tau,\phi|D)d\tau d\phi\\
&= p_{\tau',\phi'}(S^{(1)})\int_{\tau,\phi} P(A^{(1)}|\tau,\phi,S^{(1)})P(\tau,\phi|D)d\tau d\phi\\
&= p_{\tau',\phi'}(S^{(1)})P(A^{(1)}|S^{(1)}) = p_{\tau',\phi'}(S^{(1)})P_{\theta}(A^{(1)}|S^{(1)}) = P_{\theta}(S^{(1)},A^{(1)}),
\end{aligned}$$

where the last line is due to and $T_{\theta}(A^{(1)}|S^{(1)},D) = P(A^{(1)}|S^{(1)},D)$ under Assumption G.2.

Case h. Assume that Case $h-1$ holds that

$$P_{PS}(\Upsilon_{h-1}) = P_{\theta}(\Upsilon_{h-1}),$$

and we aims to prove $P_{PS}(\Upsilon_h) = P_{\theta}(\Upsilon_h)$, which is equivalent to prove

$$P_{PS}(S^{(h)},A^{(h)}|\Upsilon_{h-1}) = P_{\theta}(S^{(h)},A^{(h)}|\Upsilon_{h-1}),$$

because of the factorization

$$P_{PS}(S^{(h)},A^{(h)}|\Upsilon_{h-1})P_{PS}(\Upsilon_{h-1}) = P_{\theta}(S^{(h)},A^{(h)}|\Upsilon_{h-1})P_{\theta}(\Upsilon_{h-1}).$$

To this end, we have

$$\begin{aligned}
&P_{PS}(S^{(h)},A^{(h)}|\Upsilon_{h-1}) = P_{PS}(S^{(h)}|\Upsilon_{h-1})P_{PS}(A^{(h)}|S^{(h)},\Upsilon_{h-1}) = P_{\tau',\phi'}(S^{(h)}|\Upsilon_{h-1})P_{PS}(A^{(h)}|S^{(h)},\Upsilon_{h-1})\\
&= P_{\tau',\phi'}(S^{(h)}|\Upsilon_{h-1})\int_{\tau,\phi} P_{PS}(A^{(h)},\tau_{ps}=\tau,\phi_{ps}=\phi|S^{(h)},\Upsilon_{h-1})d\tau d\phi\\
&= P_{\tau',\phi'}(S^{(h)}|\Upsilon_{h-1})\int_{\tau,\phi} P_{PS}(A^{(h)},\tau_{ps}=\tau,\phi_{ps}=\phi|S^{(h)},\Upsilon_{h-1})d\tau d\phi\\
&= P_{\tau',\phi'}(S^{(h)}|\Upsilon_{h-1})\int_{\tau,\phi} P_{PS}(A^{(h)}|\tau_{ps}=\tau,\phi_{ps}=\phi,S^{(h)},\Upsilon_{h-1})P_{PS}(\tau_{ps}=\tau,\phi_{ps}=\phi|S^{(h)},\Upsilon_{h-1})d\tau d\phi\\
&= P_{\tau',\phi'}(S^{(h)}|\Upsilon_{h-1})\int_{\tau,\phi} \pi^{\star}_{\tau,\phi}(A^{(1)}|S^{(1)})P_{PS}(\tau_{ps}=\tau,\phi_{ps}=\phi|S^{(h)},\Upsilon_{h-1})d\tau d\phi.
\end{aligned}$$

To continue, we prove that $P_{PS}(\tau_{ps}=\tau,\phi_{ps}=\phi|S^{(h)},\Upsilon_{h-1})=P(\tau,\phi|S^{(h)},\Upsilon_{h-1})$. Indeed,

$$\begin{aligned}
P_{PS}(\tau,\phi|S^{(h)},\Upsilon_{h-1}) &= P_{PS}(\tau,\phi,S^{(h)},\Upsilon_{h-1})/P_{PS}(S^{(h)},\Upsilon_{h-1}) \\
&\propto P_{PS}(\tau,\phi,S^{(h)},\Upsilon_{h-1}) \\
&= P_{PS}(\Upsilon_{h-1}|\tau,\phi)P_{PS}(S^{(h)}|\Upsilon_{h-1})P_{PS}(\tau,\phi|D) \\
&\propto P(\tau,\phi|D)p_s(S^{(1:h)})\prod_{h'\le h}\pi^\star_{\tau,\phi}(A^{(h')}|S^{(h')}) \\
&= P(\tau,\phi,S^{(h)},\Upsilon_{h-1}) \propto P(\tau,\phi|S^{(h)},\Upsilon_{h-1}).
\end{aligned}$$

Given that $P_{PS}(\tau,\phi|S^{(h)},\Upsilon_{h-1})$ and $P(\tau,\phi|S^{(h)},\Upsilon_{h-1})$ are distributions, $P_{PS}(\tau,\phi|S^{(h),\Upsilon_{h-1}},\Upsilon_{h-1}) \propto P(\tau,\phi|S^{(h)},\Upsilon_{h-1})$ implies that $P_{PS}(\tau,\phi|S^{(h)},\Upsilon_{h-1}) = P(\tau,\phi|S^{(h)},\Upsilon_{h-1})$. Thus,

$$\begin{aligned}
P_{PS}(S^{(h)},A^{(h)}|\Upsilon_{h-1}) &= P_{\tau',\phi'}(S^{(h)}|\Upsilon_{h-1})\int_{\tau,\phi}\pi^\star_{\tau,\phi}(A^{(h)}|S^{(h)})P(\tau,\phi|S^{(h)},\Upsilon_{h-1})d\tau d\phi \\
&= P_{\tau',\phi'}(S^{(h)}|\Upsilon_{h-1})\int_{\tau,\phi}P(A^{(h)}|\tau,\phi,S^{(h)},\Upsilon_{h-1})P(\tau,\phi|S^{(h)},\Upsilon_{h-1})d\tau d\phi \\
&= P_{\tau',\phi'}(S^{(h)}|\Upsilon_{h-1})P(A^{(h)}|S^{(h)},\Upsilon_{h-1}) \\
&= P_{\tau',\phi'}(S^{(h)}|\Upsilon_{h-1})P_\theta(A^{(h)}|S^{(h)},\Upsilon_{h-1}) = P_\theta(S^{(h)},A^{(h)}|\Upsilon_{h-1}),
\end{aligned}$$

where the last line is due to Assumption G.2. Hence, the induction is complete and this concludes the proof. □

H.2 Proof of Theorem G.5

Under Assumptions G.1, G.2 and G.4, in every iteration of ICAA (Algorithm 3), and for every variation environment (τ,ϕ) with sufficient exploration $|D|$, the action label generation policy achieves performance no worse than the transformer policy T_θ from the previous iteration within the same environment (τ,ϕ).

Proof of Theorem G.5. We first prove a useful lemma.

Lemma H.1. *Under Assumptions G.4, when deploying a pretrained T_θ to a task $\tau^\star$ with an adversary $\phi^\star$, the performance of T_θ converges to that of the optimal policy, i.e.,*

$$\mathbb{E}\left[\mathcal{R}_{\tau^\star}\left(T_\theta(\cdot|\cdot,D),\phi^\star\right)\right] \to \max_\pi \mathcal{R}_{\tau^\star}(\pi,\phi^\star) \quad as \quad |D|\to+\infty.$$

Proof of Lemma H.1. As $|D|\to+\infty$, from Assumption G.4, we have the posterior $P(\tau,\phi|D)$ concentrates toward the truth $(\tau^\star,\phi^\star)$. As proved by Theorem G.3, a pretrained T_θ is performance Posterior Sampling during deployment. This leads to, for any neighbor $U(\epsilon)$ of $(\tau^\star,\phi^\star)$ with radius $\epsilon>0$,

$$\mathbb{E}\left[\mathcal{R}_{\tau^\star}\left(T_\theta(\cdot|\cdot,D),\phi^\star\right)\right] = \int P(\tau,\phi|D)\mathcal{R}_{\tau^\star}(\pi^\star_{\tau,\phi},\phi^\star) \tag{8}$$

$$\ge P(U|D)\inf_{\tau',\phi'\in U(\epsilon)}\mathcal{R}_{\tau^\star}(\pi^\star_{\tau',\phi'},\phi') \overset{|D|\to+\infty,\epsilon\to0}{\longrightarrow} \mathcal{R}_{\tau^\star}(\pi^\star_{\tau^\star,\phi^\star},\phi^\star) = \max_\tau \mathcal{R}_{\tau^\star}(\pi,\phi^\star). \tag{9}$$

In addtion, by definition of $\mathcal{R}$. we have $\mathbb{E}\left[\mathcal{R}_{\tau^\star}\left(T_\theta(\cdot|\cdot,D),\phi^\star\right)\right] \le \max_\tau \mathcal{R}_{\tau^\star}(\pi,\phi^\star)$ almost surely. This proves that $\mathbb{E}\left[\mathcal{R}_{\tau^\star}\left(T_\theta(\cdot|\cdot,D),\phi^\star\right)\right] \to \max_\tau \mathcal{R}_{\tau^\star}(\pi,\phi^\star)$. □

For any D of finite transitions, with Lemma H.1, we can outperform $T_\theta(\cdot|\cdot,D)$ by extending the exploration to have D' where $|D'|>|D|$ such that the expected performance of $T_\theta(\cdot|\cdot,D')$ is arbitrarily close to the optimal one. Here, by definition of ICAA algorithm, $T_\theta(\cdot|\cdot,D')$ is the action label generation policy, thus concluding the proof. □

I Computation Requirements

We conduct each experiment on a single GPU: Nvidia RTX A5000 with 24GB RAM. In DarkRoom, all transformer models converge within 150 epochs within an hour. In Meta-World, we run each experiment up to 500 episodes with early termination mechanism. In MuJoCo, all models can converge within 200 epochs with less than two hours.

J Pseudocode

Algorithm 1 Deployment of ICRL Models

Input: Pretrained transformer Model T_θ; Horizon of episodes H; Number of episodes N for online testing; Offline dataset $D_{\text{off}} = \{(s_h, a_h, s_{h+1}, r_h)\}_h$, consisting of transitions collected by a behavioral policy.
// Offline Testing
for every time step $h \in \{1, \dots, H\}$ **do**
  Observe state s_h
  Sample action with T_θ:

$$a_h \sim T_\theta\left(\cdot | s_h, D_{\text{off}}\right)$$

  Collect reward r_h
end for
// Online Testing
Initialize an empty online data buffer $D_{\text{on}} = \{\}$
for every online trial $n \in \{1, \dots, N\}$ **do**
  for every time step $h \in \{1, \dots, H\}$ **do**
    Observe state s_h
    Sample action with T_θ:

$$a_h \sim T_\theta\left(\cdot | s_h, D_{\text{on}}\right)$$

    Collect reward r_h
  end for
  Append the collected transitions $\{(s_h, a_h, s_{h+1}, r_h)\}_h$ into D_{on}
end for

K Limitations

While ICAG involves generating expert policies across perturbed task variants, this step is performed efficiently in parallel and amortized over pretraining, making it feasible for a modest number of variants. ICAA uses self-generated labels derived from a robust pretrained model, which, despite not being optimal, have been shown empirically to improve performance with minimal data. These design choices strike a practical balance between robustness and scalability. While the proposed methods improve robustness under structured perturbations, their generalization is currently focused on tasks with in-distribution environment variations. Extending this framework to handle broader generalization—such as out-of-distribution task structures and disturbances—remains a valuable direction for future work.

Algorithm 2 In-Context Adversarial Generalization

Input: Causal TM T_θ; number of pretraining tasks m; number of variation environments per task K
// Construction of Pretraining Dataset
Initialize an empty pretraining dataset $\mathcal{D}_{pre}$
for $i \in \{1, \ldots, m\}$ **do**
  Sample a task $\tau^i \in \mathcal{M}$;
  for $k \in \{1, \ldots, K\}$ **do**
    Sample a disturbance $\phi^{i,k} \in \Phi(\tau^i)$;
    Collect a context dataset $D^{i,k}$ from the variation environment $(\tau^i, \phi^{i,k})$;
    Sample a query state $s^{i,k}_{\text{query}}$;
    Train an optimal policy $\pi^\star_{\tau^i, \phi^{i,k}}$ following (5);
    Sample an action label $a^\star_{i,k} \sim \pi^\star_{\tau^i, \phi^{i,k}}(s^{i,k}_{\text{query}})$;
    Append $(D^{i,k}, s^{i,k}_{\text{query}}, a^\star_{i,k})$ into $\mathcal{D}_{pre}$;
  end for
end for
// Supervised Pretraining
Pretrain the TM T_θ by

$$\min_\theta \frac{1}{mK} \sum_{i=1}^{m} \sum_{k=1}^{K} -\log T_\theta(a^\star_{i,k} | s^{i,k}_{\text{query}}, D^{i,k}).$$

Algorithm 3 In-Context Adversarial Adaptation

Input: number of pretraining tasks m; number of variation environments per task K; initial pretraining dataset $\mathcal{D}^0$
// Initial Supervised Pretraining
Pretrain a causal transformer with $\mathcal{D}^0$ to have the pretrained TM T^0_θ.
Set $\mathcal{D}_{pre} = \mathcal{D}^0$.
for $j \in \{0, \ldots, J\}$ **do**
  // Collecting New Pretraining Data
  Initialize an empty dataset $\mathcal{D}^{j+1}$ for new data.
  for $i \in \{1, \ldots, m\}$ **do**
    Sample a task $\tau^i \in \mathcal{M}$;
    for $k \in \{1, \ldots, K\}$ **do**
      Sample a disturbance $\phi^{i,k} \in \Phi(\tau^i)$;
      Deploy T^j_θ for each variation environment $(\tau^i, \phi^{i,k})$ (following Algorithm 1) with $N+1$ trials to have trajectories $\xi_n = \{(s_h, a_h, s_{h+1}, r_h)\}_h$, $n \in \{0, \cdots, N\}$.
      Use the first $\underline{N}+1$ trajectories as context datasets $D_n = \xi_n, n \in \{0, \cdots, \underline{N}\}$.
      Sample query state-action label pairs from the remaining trajectories: $\{(s^n_{\text{query}}, a^n)\}_{n=0}^{\underline{N}} \sim \bigcup_{n=\underline{N}}^{N} \{(s_h, a_h) \in \xi_n\}_{h=0}^{H-1}$
      Append $\{D_n, s^n_{\text{query}}, a^n\}_{n=0}^{\underline{N}}$ into $\mathcal{D}^{j+1}$;
    end for
  end for
  // Supervised Fine-Tuning
  Update the pretraining dataset with the new data $\mathcal{D}_{pre} = \mathcal{D}_{pre} \cup \mathcal{D}^{j+1}$.
  Pretrain the latest TM T^j_θ with the updated $\mathcal{D}_{pre}$ to have T^{j+1}_θ.
end for