RL-Studio: A System for Multi-Phase Reinforcement Learning Experimentation

Whiyoung Jung[1], Sunghoon Hong[1], Deunsol Yoon[1], Jeonghye Kim[2*], Yongjae Shin[2*], Suhyun Jung[1], Hyundam Yoo[1], Youngjin Kim[1], Chanwoo Moon[1], Woohyung Lim[1], Soonyoung Lee[1], Kanghoon Lee[1]

[1] LG AI Research
[2] KAIST
{whiyoung.jung, sunghoon.hong, dsyoon}@lgresearch.ai, {jeonghye.kim, yongjae.shin}@kaist.ac.kr
{sh.jung, hyundamyoo, youngjin.kim, chanwoo.moon, w.lim, soonyoung.lee, kanghoon.lee}@lgresearch.ai

Abstract

Reinforcement learning (RL) has evolved beyond monolithic training, yet existing frameworks remain limited to single algorithms or simple offline-to-online transitions. We present multi-phase RL, a framework that orchestrates multiple learning phases for continual policy improvement. It enables efficient fine-tuning of pretrained policies with new data and smooth adaptation from simulation to real-world environments. To support this paradigm, we introduce RL-Studio, a platform that addresses key implementation barriers, including neural architecture mismatches, parameter transfer complexities, and experiment management overhead. It provides phase orchestration, transition-point monitoring, and full experiment lineage tracking. We demonstrate the effectiveness of multi-phase RL through representative scenarios and highlight RL-Studio's capabilities.

Introduction

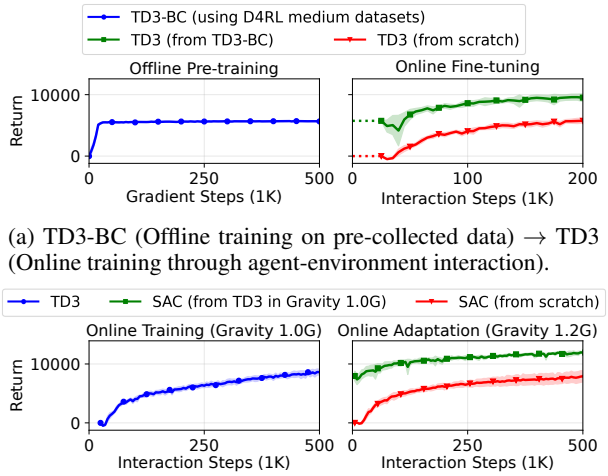
Reinforcement Learning (RL) has evolved beyond single-paradigm approaches to encompass diverse learning strategies. Traditional methods include online learning, which gathers data through direct environmental interaction, and offline learning, which relies exclusively on pre-collected datasets (Levine et al. 2020). Recently, hybrid approaches have emerged to combine different learning paradigms. Notably, offline-to-online methods leverage existing datasets for initial training, then transition to environmental interaction for policy refinement (Lee et al. 2022).

While offline-to-online RL represents a significant advance, we pose a fundamental question:

> "What lies beyond offline-to-online RL?"

We argue that this is merely one instance of a broader paradigm. Building upon prior ideas such as transfer learning and sim-to-real adaptation, we introduce multi-phase RL, a framework that enables flexible transitions across arbitrary sequences of learning paradigms. Such transitions may involve: (1) shifting learning modes (offline ↔ online), (2) switching algorithms (e.g., TD3 → SAC), or (3) transitioning between deployment contexts (sim-to-real scenario).

Unlike existing frameworks with predefined transitions, multi-phase RL enables arbitrary phase sequences based on task requirements. We present RL-Studio, a platform for designing, implementing, and evaluating multi-phase algorithms, providing infrastructure to explore optimal phase transitions for improved efficiency and performance.



(a) TD3-BC (Offline training on pre-collected data) → TD3 (Online training through agent-environment interaction).

(b) TD3 (Online training in simulated-environments) → SAC (Online training in real-world settings).

Figure 1: Performance comparison across two phase transitions in HalfCheetah-v2. The x-axis represents gradient updates for offline and environment steps for online phases.

Multi-Phase RL

Multi-phase RL sequences multiple learning phases, each potentially using different algorithms and data sources. Each phase represents a complete experimental configuration. Upon completion, learned networks initialize the next phase, enabling continuous adaptation across learning modes.

This framework excels in two key scenarios:

Fine-Tuning. Figure 1(a) illustrates the conventional offline-to-online scenario, where offline pretraining is fol-

*Work done during an internship at LG AI Research.

lowed by online fine-tuning. The pretrained policy enables sample-efficient online learning, surpassing both pure offline and from-scratch online approaches.

Adaptation. Figure 1(b) illustrates online-to-online transition, valuable for sim-to-real transfer. After training in simulation, agents adapt using limited real-world datasets, achieving faster convergence than training from scratch.

These scenarios raise important research questions: *When should phases transition? Which sequences optimize specific tasks? How can we predict transition outcomes?* RL-Studio enables systematic exploration of these questions.

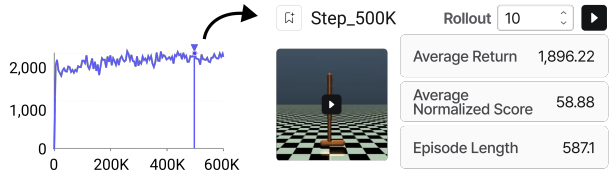
The RL-Studio System

RL-Studio addresses technical challenges of implementing multi-phase RL—resolving neural architecture mismatches between different algorithmic implementations, managing parameter transfer complexities when transitioning between training phases, and reducing experiment management overhead through systematic automation—enabling researchers to focus on the scientific questions of multi-phase learning. The system comprises three core components as follows.

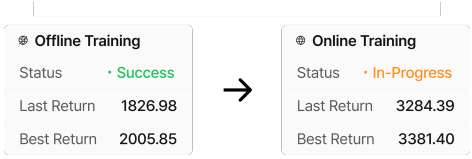
Pre-Built Assets. Each phase integrates algorithm, environment, dataset, and pre-trained model options: (1) **Algorithms**: We support (i) *online RL*: PPO(Schulman et al. 2017), TD3 (Fujimoto, Hoof, and Meger 2018), SAC (Haarnoja et al. 2018), AWAC (Nair et al. 2020), IQL (Kostrikov, Nair, and Levine 2022), PARS (Kim et al. 2025) [1]; (ii) *offline RL*: BC, TD3-BC(Fujimoto and Gu 2021), CQL (Kumar et al. 2020), AWAC, IQL, PARS; and (iii) *offline-to-online RL*: AWAC, Off2On(Lee et al. 2022), IQL, Cal-QL (Nakamoto et al. 2023), SPOT (Wu et al. 2022), RLPD (Ball et al. 2023), PARS, OPT (Shin et al. 2025). (2) **Environments and datasets**: OpenAI Gym (Brockman et al. 2016)-compatible environments (MuJoCo (Todorov, Erez, and Tassa 2012), AntMaze (Nachum et al. 2018), Adroit (Kumar 2016)) and D4RL benchmarks (Fu et al. 2020). (3) **Pre-trained models**: Default checkpoints from the above algorithms and user-provided models. Users can extend these with custom algorithms, environments, and datasets.

Phase Orchestrator. The phase orchestrator enables continuous learning across any training phases through two mechanisms: (1) **Compatibility checker** verifies whether neural network components from different algorithms can be integrated; (2) **Model loader** transfers compatible parameters regardless of naming while randomly initializing non-matching components. These mechanisms enable complex pipelines with continuous knowledge transfer.

Web Platform Features. Our platform addresses gaps in general ML platforms (Amazon Web Services 2017; Biewald 2020) with three RL-focused features: (1) **Behavioral visualization** (Figure 2 (a)) lets users select checkpoints from learning curves and instantly view corresponding behavior videos, helping assess whether training is sufficient or phase transitions are needed; (2) **Training lineage tracking** (Figure 2 (b)) visualizes the complete learning progression—phase sequences, algorithms used, and performance evolution through transitions—helping identify optimal combinations; (3) **Paradigm-aware performance tracking** automatically segregates learning curves by paradigm (agent-environment interactions for online; *sample complexity*, and gradient update steps for offline; *training iterations*), enabling fair cross-paradigm comparisons. These features streamline multi-phase RL workflows, letting researchers focus on algorithmic innovation.



(a) Behavior Visualization.



(b) Training Lineage.

Figure 2: RL-Studio platform features for multi-phase RL.

Demonstration

Multi-Phase Experiment Workflow. RL-Studio enables comprehensive multi-phase RL through a streamlined workflow. Users (1) create a project for their target environment and configure initial experiments by selecting datasets, pre-trained models, and algorithms; (2) monitor training progress through interactive learning curves with behavior videos automatically generated at checkpoints; (3) transition between phases by initializing new experiments from selected checkpoints, enabling algorithm or paradigm switches; and (4) track complete training lineages and explore historical results via unified visualization tools.

Performance Comparison Workflow. The platform streamlines algorithm comparison through integrated analysis tools. Learning curves are automatically organized by training paradigm (online vs. offline) when users select experiments, enabling meaningful comparisons.

Limitations and Future Work

A key challenge lies in handling phase transitions across heterogeneous models—spanning different network architectures (e.g., MLP $\leftrightarrow$ Transformer) and learning paradigms (e.g., actor-critic $\leftrightarrow$ value-based)—which are currently limited to partial transfer. Overcoming this limitation could enhance multi-phase adaptation across diverse representations. Extending RL-Studio toward real-world deployment and continual improvement would further evolve it into a closed-loop system for efficient refinement.

[1]While AWAC, IQL, and PARS are offline-to-online methods, their offline and online phases can be used independently.

References

Amazon Web Services. 2017. Amazon SageMaker. https://aws.amazon.com/sagemaker/. Accessed: 2025-08-22.

Ball, P. J.; Smith, L.; Kostrikov, I.; and Levine, S. 2023. Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning*, 1577–1594.

Biewald, L. 2020. Experiment Tracking with Weights and Biases. Software available from wandb.com.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym. *arXiv preprint arXiv:1606.01540*.

Fu, J.; Kumar, A.; Nachum, O.; Tucker, G.; and Levine, S. 2020. D4RL: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*.

Fujimoto, S.; and Gu, S. S. 2021. A minimalist approach to offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 34, 20132–20145.

Fujimoto, S.; Hoof, H.; and Meger, D. 2018. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, 1587–1596.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, 1861–1870.

Kim, J.; Shin, Y.; Jung, W.; Hong, S.; Yoon, D.; Sung, Y.; Lee, K.; and Lim, W. 2025. Penalizing infeasible actions and reward scaling in reinforcement learning with offline data. In *International Conference on Machine Learning*, 30769–30790.

Kostrikov, I.; Nair, A.; and Levine, S. 2022. Offline reinforcement learning with implicit Q-learning. *International Conference on Learning Representations*.

Kumar, A.; Zhou, A.; Tucker, G.; and Levine, S. 2020. Conservative Q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, 1179–1191.

Kumar, V. 2016. *Manipulators and manipulation in high dimensional spaces*. Ph.D. thesis, University of Washington, Seattle.

Lee, S.; Seo, Y.; Lee, K.; Abbeel, P.; and Shin, J. 2022. Offline-to-online reinforcement learning via balanced replay and pessimistic Q-ensemble. In *Conference on Robot Learning*, 1702–1712.

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

Nachum, O.; Gu, S. S.; Lee, H.; and Levine, S. 2018. Data-efficient hierarchical reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 31, 3303–3313.

Nair, A.; Gupta, A.; Dalal, M.; and Levine, S. 2020. AWAC: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*.

Nakamoto, M.; Zhai, S.; Singh, A.; Sobol Mark, M.; Ma, Y.; Finn, C.; Kumar, A.; and Levine, S. 2023. Cal-QL: Calibrated offline RL pre-training for efficient online fine-tuning. In *Advances in Neural Information Processing Systems*, volume 36, 62244–62269.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shin, Y.; Kim, J.; Jung, W.; Hong, S.; Yoon, D.; Jang, Y.; Kim, G.-H.; Chae, J.; Sung, Y.; Lee, K.; and Lim, W. 2025. Online pre-training for offline-to-online reinforcement learning. In *International Conference on Machine Learning*, 55122–55144.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 5026–5033.

Wu, J.; Wu, H.; Qiu, Z.; Wang, J.; and Long, M. 2022. Supported policy optimization for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 35, 31278–31291.