Context Value Informed In-Context Reinforcement Learning

Wenhao Zhang
Shanghai Jiao Tong University
wenhao_zhang@sjtu.edu.cn

Shao Zhang
Shanghai Artificial Intelligence Laboratory
zhangshao@pjlab.org.cn

Xihuai Wang
Shanghai Jiao Tong University
leoxhwang@sjtu.edu.cn

Yang Li[†]
Shanghai Jiao Tong University
yang.li.cs@sjtu.edu.cn

Ying Wen[†]
Shanghai Jiao Tong University
Shanghai Innovation Institute
ying.wen@sjtu.edu.cn

Abstract

In-context reinforcement learning (ICRL) has emerged as an effective paradigm for test time adaptation to unseen tasks without parameter updates. However, existing ICRL methods can exhibit brittle and unstable adaptions, and the mechanisms underlying such adaptions remain poorly understood. We provide a policy-gradient view of ICRL and argue that relying on trajectory-level return feedback can lead to high-variance updates when test time rollout is limited. Motivated by the actor–critic principle, we propose **Context Value Informed ICRL (CV-ICRL)**, which equips the in-context policy with an explicit *context value*: the expected discounted return conditioned on the current state and accumulated interaction context. CV-ICRL trains a value head and writes its prediction back into the context as a value token, enabling TD-style return targets for lower-variance policy updates. Experiments on the Dark Room, Minigrid, and Procgen testbeds show that CV-ICRL substantially improves the stability of test time adaptation and achieves higher returns across tasks and environments. Code and data are available at https://github.com/Bluixe/CV-ICRL.

1 Introduction

In-context reinforcement learning (ICRL) aims to endow a single pretrained sequence model with the ability to adapt to previously unseen tasks at test time solely by conditioning on its interaction history, without parameter updates [2, 3]. This paradigm is particularly appealing for cross-task generalization, where conventional reinforcement learning agents often overfit to their training environments and exhibit brittle transfer under distribution shift [11, 6]. A common approach operationalizes this idea by pretraining causal Transformers on trajectory corpora that emulate online RL, interleaving observations, actions, rewards, and episode boundaries, so as to enable context-dependent policy improvement and cross-episode learning at inference time [16, 32].

Despite the appeal of this paradigm, the adaptation process in ICRL remains largely implicit and exhibits unstable test-time behavior, sometimes degrading with additional interaction [20, 13]. Build-

[†]Corresponding authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

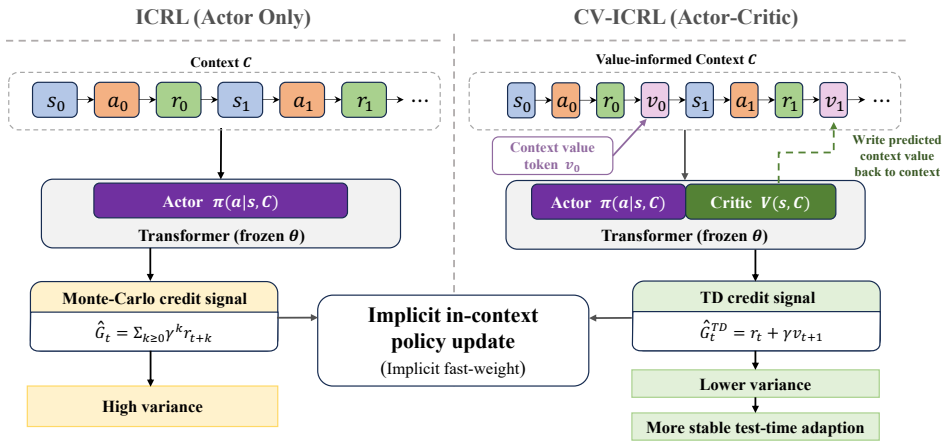


Figure 1: **From actor-only ICRL to context value informed (actor–critic-style) ICRL.** (*Left*) The PG-like view models return-based credit assignment using the Monte-Carlo return $\widehat{G}_t = \sum_{k\geq 0}\gamma^k r_{t+k}$. (*Right*) CV-ICRL augments the same Transformer with a context critic that predicts a context value v_{t+1} and writes it back into the context, enabling a TD-style one-step target $\widehat{G}_t^{\mathrm{TD}} = r_t + \gamma v_{t+1}$ for lower-variance test-time adaptation.

ing on recent theory that interprets in-context learning as implicit gradient-based optimization over context-induced variables [27, 9], we view ICRL test-time adaptation as a policy-gradient-like (PG-like) [25] credit assignment process, in which actions leading to higher realized returns are reinforced in subsequent predictions.

Under this PG-like view, a key limitation of prior ICRL training emerges. Existing approaches [16, 32, 13] mainly distill actions without explicitly supervising a value signal for test-time credit assignment. In the data-scarce test-time regime, where adaptation must happen from only a few rollouts, such return-based signals can be high-variance, contributing to brittle improvement. Inspired by how reinforcement learning addresses this via a learned critic [15], we seek to equip ICRL with an analogous value-based signal for guiding inference-time adaptation.

Based on the actor-critic principle, we propose **Context Value Informed ICRL (CV-ICRL)**, which augments an in-context policy with an explicit value signal, the **context value**, defined as the expected discounted return of the in-context policy conditioned on the current state and accumulated interaction context. As shown in Figure 1, predicting the context value and writing it back into the context enables a bootstrapped one-step TD return [24] in place of the high-variance Monte-Carlo return. We show that, when using the true conditional value, this TD target is a Rao–Blackwellization of the Monte-Carlo return under mild measurability assumptions, providing a principled variance reduction for PG-style in-context updates. In practice, we implement CV-ICRL by distilling an online RL training run into a single Transformer following prior work [16, 13], training the model to predict both each checkpoint's action distribution and a stable checkpoint-level proxy for the context value derived from the checkpoint's evaluated return.

In our experiments, we evaluate CV-ICRL on Dark Room, a more challenging Minigrid suite [5], and the visually complex Procgen benchmark [6]. We empirically observe that prior ICRL method AD [16] exhibits sharp drops in episode return at test time, consistent with the high-variance hypothesis. In contrast, CV-ICRL substantially improves adaptation stability and achieves higher average returns across all three testbeds. CV-ICRL also maintains robust adaptation under out-of-distribution shifts in dynamics or observations, and ablations confirm that both value prediction and value conditioning contribute to the gains.

In summary, our main contributions are given below:

- We provide a policy-gradient-like (PG-like) viewpoint for in-context adaption of ICRL, which helps explain why prior actor-only distillation can be unstable at test time with only a few rollouts and provides an improvement direction for ICRL methods.

- We introduce **Context Value Informed ICRL (CV-ICRL)**, an actor–critic-style ICRL framework that predicts a long-horizon context value from the interaction history and appends it to the context to guide inference-time policy improvement with a lower-variance signal.
- We conduct extensive experiments on Dark Room, Minigrid, and Procgen, showing that CV-ICRL improves test time adaptation stability and average returns, with additional gains in out-of-distribution generalization.

2 Related Work

In-context Reinforcement Learning. In-context reinforcement learning (ICRL) uses causal Transformers to adapt from trajectory context, enabling few-shot cross-task generalization without gradient updates [1, 20]. A canonical example is the Decision Transformer (DT), which casts RL as conditional sequence modeling and predicts actions autoregressively from sequences of states, actions, and return-to-go (RTG) tokens [4]. Most existing ICRL approaches follow action distillation: Algorithm Distillation (AD) trains a Transformer on trajectories generated along an online RL run and supervises it primarily at the action level [16]; later variants largely preserve this recipe [32, 18, 13]. Complementarily, Decision-Pretrained Transformer (DPT) studies supervised pretraining for ICRL from a posterior-sampling perspective [17]. These AD-like methods primarily supervise actions without explicitly supervising a value signal for test-time credit assignment. RTG conditioning makes goals more explicit [7, 18, 13], but test-time RTG updates still depend on sparse rewards. Some methods explore value heads [19, 14]; our focus is on writing predicted context values back into the cross-episode interaction history. Reinformer also predicts a scalar return signal and conditions action selection on it, using expectile regression to estimate the maximum return-to-go in offline RL [31]. CV-ICRL instead retains predicted context values across episodes to support adaptation to unseen tasks. Our method also distills an online RL learning process into a sequence model, but introduces a learned context value that is inferred from and written into the context, enabling actor–critic-style TD credit assignment during inference for more stable test time adaptation.

ICRL as a PG-like Credit Assignment Process. Prior work shows that Transformers can adapt to prompt context at inference time with frozen parameters [2, 10], and recent theory explains this as implicit optimization realized by the forward computation [29, 9]. Wang et al. [28] show that Transformers can learn TD methods for in-context policy evaluation, while Demircan et al. [8] identify TD-error-like representations in LLMs solving RL tasks. These works study implicit TD-related computation; CV-ICRL makes a predicted context-value signal explicitly available through write-back. Inspired by the policy-gradient view in reinforcement learning [30, 25], we interpret ICRL as a PG-like credit assignment process within the context window: the in-context model induces a policy over actions, and the realized return from the interaction context plays the role of the credit signal. This perspective highlights a potential variance bottleneck under limited test-time rollouts. In traditional RL, actor–critic methods mitigate this issue by learning a value baseline and using bootstrapping for lower-variance credit assignment [15]. These insights motivate our use of value-informed, actor–critic-style credit assignment.

3 Preliminaries and Formulations

3.1 In-context reinforcement learning

ICRL aims to produce a single policy that can adapt at test time to previously unseen tasks from only a small amount of online interaction, without updating model parameters. Accordingly, an ICRL agent conditions its action not only on the current state but also on a context that summarizes its interaction history. We formalize the testing environment as an MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma, \rho_0)$, whose transition and reward functions are initially unknown to the agent.

At time t, the agent observes the current state s_t together with a context that records past transitions and rewards, $C_t \triangleq \big((s_0, a_0, r_0), \ldots, (s_{t-1}, a_{t-1}, r_{t-1})\big)$ with $C_0 = \emptyset$. The agent then selects an action according to an in-context policy $\pi_\theta(a_t \mid s_t, C_t)$, where adaptation is realized through the evolution of C_t while θ remains fixed. After executing a_t and observing r_t, the context is updated by

concatenation:

$$C_{t+1} = U(C_t, (s_t, a_t, r_t)). \tag{1}$$

Given the unknown MDP $\mathcal{M}$, the objective of an in-context policy is to maximize the expected discounted return

$$J(\pi; \mathcal{M}) \triangleq \mathbb{E}_{\pi_\theta, \mathcal{M}} \left[\sum_{t=0}^{\infty} \gamma^t r_t \right]. \tag{2}$$

3.2 A Policy-Gradient View of ICRL

At test time, the ICRL agent interacts with the unknown task MDP $\mathcal{M}$ to maximize the objective in (2); the policy $\pi_\theta(a_t \mid s_t, C_t)$ is history-dependent through the growing context C_t. We adopt the augmented-state view $x_t \triangleq (s_t, C_t)$, under which $\pi_\theta(a_t \mid s_t, C_t) \equiv \pi_\theta(a_t \mid x_t)$ is a stationary Markov policy and the objective is preserved. This restores the Markov property required for the policy-gradient analysis below.

Fast weights induced by context. In many ICRL instantiations, π_θ is a pretrained causal Transformer with frozen weights. Although θ is fixed, the model computes a context-dependent internal representation that controls its action distribution. We make this explicit by introducing *fast weights* $\phi_t \triangleq \Phi_\theta(C_t)$ such that $\pi_\theta(a_t \mid s_t, C_t) \equiv \pi(a_t \mid s_t, \phi_t)$, where ϕ_t can be understood as the Transformer's hidden state / cache summarizing the interaction history. As the context grows, ϕ_t changes according to an implicit update rule induced by the model computation:

$$\phi_{t+1} = U_\theta(\phi_t, s_t, a_t, r_t). \tag{3}$$

Dherin et al. [9] relate contextual-block computation to a query-dependent implicit rank-one update ΔW of an MLP weight W. This motivates our context-dependent fast-weight abstraction $\phi_t = \Phi_\theta(C_t)$, whose evolution is summarized by (3) while θ remains fixed.

A policy-gradient interpretation. The policy-gradient theorem on the augmented-MDP objective J, with ϕ as the inner-loop variables, gives

$$\nabla_{\{\phi_t\}} J = \mathbb{E}\left[\sum_{t \geq 0} \nabla_{\phi_t} \log \pi(a_t \mid s_t, \phi_t)\, \widehat{G}_t \right], \tag{4}$$

i.e., the ideal direction for improving expected return is a return-weighted score-function step. We interpret the architecture-determined update (3) as a return-aware specialization of this direction—a policy-gradient-style step on ϕ:

$$\phi_{t+1} \approx \phi_t + \alpha\, \nabla_{\phi_t} \log \pi(a_t \mid s_t, \phi_t)\, \widehat{G}_t, \tag{5}$$

under which actions associated with higher credited returns become more likely in subsequent predictions. This PG-like correspondence is a working interpretation rather than a formal equivalence; it helps interpret potential variance bottlenecks in return-based credit assignment and motivate the actor-critic remedy that follows.

Monte-Carlo credit and variance. Without providing an explicit value/baseline signal for credit assignment, a common choice for credit assignment is the trajectory-level Monte-Carlo return,

$$\widehat{G}_t \triangleq \sum_{k \geq 0} \gamma^k r_{t+k}. \tag{6}$$

However, $\widehat{G}_t$ aggregates all downstream stochasticity into a single scalar and does not bootstrap from any estimate of future outcomes, which can yield high-variance credit assignment under short-horizon, data-scarce adaptation. This motivates incorporating a value-informed signal that enables bootstrapping within the context.

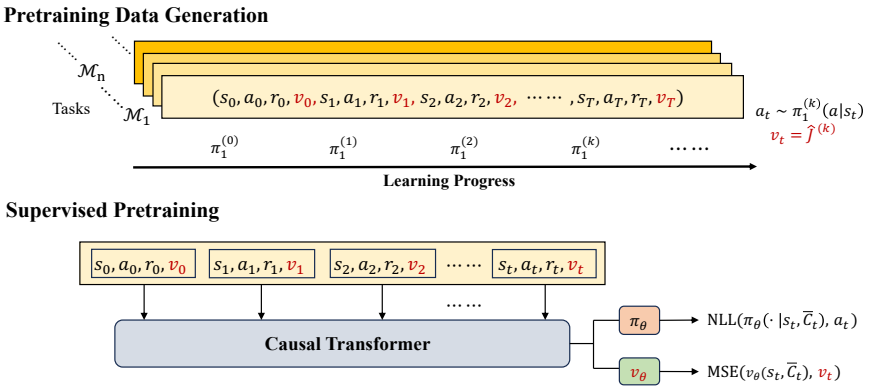


Figure 2: **Supervised pretraining of CV-ICRL from online RL learning progress.** *Top: Pretraining data generation.* We collect trajectories from training tasks along an online RL learning run, recording checkpoint policies $\{\pi^{(k)}\}$ and annotating each transition with a value token $v_t = \hat{J}^{(k)}$ (Monte-Carlo evaluated). *Bottom: Supervised pretraining.* A causal Transformer is pretrained to predict both the next action $\pi_\theta(\cdot \mid s_t, \bar{C}_t)$ and the context value $v_\theta(s_t, \bar{C}_t)$ from value-augmented history $\bar{C}_t$ using a joint loss: action NLL and value MSE.

4 Context Value Informed In-Context RL

We propose an actor–critic-like ICRL framework, **Context Value Informed ICRL (CV-ICRL)**, to reduce the variance of credit assignment. The key idea is to guide the in-context policy with an explicit value signal that reflects the agent's *expected performance given the currently available context* C, enabling TD-style bootstrapping within the context. We formalize this signal as the **context value**.

4.1 Context Value

Let $\pi(a \mid s, C)$ be an in-context policy and let the context evolve according to the update rule (1). We define the **context value** as the standard discounted value on the augmented state (s, C):

$$V^\pi(s, C) \triangleq \mathbb{E}_{\pi, \mathcal{M}_{\text{test}}}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \,\middle|\, s_t = s,\, C_t = C\right],$$

where future actions follow $a_{t+k} \sim \pi(\cdot \mid s_{t+k}, C_{t+k})$ and the context evolves by $C_{t+k+1} = U(C_{t+k}, (s_{t+k}, a_{t+k}, r_{t+k}))$. Because C can summarize interaction history across episodes, $V^\pi(s, C)$ measures the expected performance enabled by the information already inferred from context, rather than a state-only value. In our method we take $\pi = \pi_\theta$ (the pretrained Transformer policy) and write $V(s, C) \triangleq V^{\pi_\theta}(s, C)$ for brevity. Predicting and writing this signal back into the context is what enables the actor–critic-style, lower-variance test-time updates we describe next.

4.2 CV-ICRL: Actor–Critic-Style In-Context Updates

CV-ICRL augments a pretrained causal Transformer with an explicit *context critic* that predicts the context value and, to make the critic information accessible to subsequent in-context computation, writes the value estimate back into the context. This *augmented* context is $\bar{C}_t \triangleq \big((s_0, a_0, r_0, v_0), \ldots, (s_{t-1}, a_{t-1}, r_{t-1}, v_{t-1})\big)$.

Given the current state s_t and augmented context $\bar{C}_t$, the frozen Transformer outputs (i) an action distribution (actor) $\pi_\theta(\cdot \mid s_t, \bar{C}_t)$ and (ii) a scalar value estimate (critic) $v_t \triangleq v_\theta(s_t, \bar{C}_t) \approx V(s_t, \bar{C}_t)$. After sampling a_t and observing r_t and s_{t+1}, the context is updated as $\bar{C}_{t+1} \triangleq U(\bar{C}_t, (s_t, a_t, r_t, v_t))$. Context is retained across episode resets; once the FIFO window is full, the oldest transition is removed. Algorithm 1 (Appendix A) summarizes the resulting training and test-time procedure.

From return-only credit to actor–critic credit. Replacing the base context C_t in (3) by the augmented $\bar{C}_t$, the same fast-weight evolution applies, and the policy-gradient-style interpretation (5)

carries over with $\phi_t \equiv \Phi_\theta(\bar{C}_t)$. Compared with the Monte-Carlo credit signal in this interpretation (Equation (6)), CV-ICRL equips the context with a learned context-value estimate, enabling a bootstrapped one-step return target

$$\widehat{G}_t^{\mathrm{TD}} \triangleq r_t + \gamma v_{t+1}. \tag{7}$$

Plugging (7) into (5) yields an actor–critic-style update,

$$\phi_{t+1} \approx \phi_t + \alpha \, \nabla_{\phi_t} \log \pi(a_t \mid s_t, \phi_t) \, \widehat{G}_t^{\mathrm{TD}}, \tag{8}$$

where the value token makes v_{t+1} explicitly available in the context, allowing in-context updates to condition on a bootstrapped credit signal. The following result formalizes this variance reduction as a Rao–Blackwellization of the Monte-Carlo return when using the true conditional value.

Property 4.1 (Variance reduction for PG-style updates with a bootstrapped one-step TD return)**.** Let $G_t = \sum_{k\geq 0} \gamma^k r_{t+k}$ be the Monte-Carlo return and let $V_{t+1} \triangleq V(s_{t+1}, C_{t+1})$ denote the (true) context value at the next step. Define the transition summary $H_t \triangleq (C_t, s_t, a_t, r_t, s_{t+1})$. Let $\psi_t = \nabla_{\phi_t} \log \pi(a_t \mid s_t, \phi_t)$ and assume ψ_t is deterministic given (C_t, s_t, a_t). Define the Rao–Blackwellized (bootstrapped) return target $\tilde{G}_t \triangleq \mathbb{E}[G_t \mid H_t]$. If V_{t+1} is measurable w.r.t. H_t (equivalently, C_{t+1} is determined by H_t), then $\tilde{G}_t = r_t + \gamma V_{t+1}$, and $\tilde{G}_t$ is a Rao–Blackwellization of G_t. Consequently,

$$\mathrm{Var}(G_t) \;\geq\; \mathrm{Var}\Big(\tilde{G}_t\Big), \tag{9}$$

$$\mathrm{Var}(\psi_t G_t \mid C_t, s_t, a_t) \;\geq\; \mathrm{Var}\Big(\psi_t \tilde{G}_t \mid C_t, s_t, a_t\Big). \tag{10}$$

Replacing V_{t+1} with its in-context estimate v_{t+1} yields the practical one-step TD target $\widehat{G}_t^{\mathrm{TD}} \triangleq r_t + \gamma v_{t+1}$ in the main text. The appendix also gives an approximation-error condition under which the practical target retains the variance advantage. The proof is provided in Appendix B.

4.3 Supervised Pretraining from RL Learning Progress

Following prior ICRL works [16, 13], we obtain supervised targets by distilling online RL training progress into a single Transformer. We run RL algorithm on training tasks and record policy checkpoints $\{\pi^{(k)}\}_{k=1}^{K}$ along the training progress. For each checkpoint $\pi^{(k)}$, we sample trajectory segments and form context–state pairs (C_t, s_t) from the same task. We use the checkpoint action distribution as the supervision signal, treating $\pi^{(k)}(\cdot \mid s_t)$ as a practical proxy for the context-conditioned response after observing C_t.

To supervise the value token, we use a stable long-horizon performance proxy rather than step-wise state values. Concretely, for each checkpoint $\pi^{(k)}$ we estimate its expected discounted return by Monte-Carlo evaluation, $\hat{J}^{(k)} \triangleq \frac{1}{N}\sum_{i=1}^{N}\sum_{t=0}^{T-1}\gamma^t r_t^{(i)}$, and use $\hat{J}^{(k)}$ to supervise the value head for samples collected under $\pi^{(k)}$. This provides a checkpoint-level competence proxy aligned with the context–policy correspondence induced by distillation, preserving learning-stage differences while averaging within-checkpoint state variation. Our analysis in Theorem C.5 gives a sufficient value-error condition for a tighter performance bound under its policy-identification model.

Supervised pretraining objectives. During pretraining, the augmented history is $\bar{C}_t = ((s_i, a_i, r_i, \hat{J}^{(k_i)}))_{i<t}$, where k_i identifies each transition's source checkpoint. Given a context-state-action pair $(\bar{C}_t, s_t, a_t)$ collected under checkpoint $\pi^{(k)}$ with evaluated-return estimate $\hat{J}^{(k)}$, the training loss is

$$\mathcal{L} = \mathbb{E}_{k,t}\Big[\underbrace{\mathcal{L}_\pi\big(\pi_\theta(\cdot \mid s_t, \bar{C}_t), a_t\big)}_{\text{policy distillation}} + \lambda \underbrace{\big(v_\theta(s_t, \bar{C}_t) - \hat{J}^{(k)}\big)^2}_{\text{value regression}}\Big],$$

where $\mathcal{L}_\pi$ is implemented as negative log-likelihood, and λ balances the two parts of loss. The source PPO generates action labels without consuming the added value annotation, whereas the Transformer is trained on the augmented context. At test time, the same temporal input structure uses the Transformer's own previous value predictions and interaction history on unseen tasks.

Table 1: Minigrid results over 6 task types. We report equally weighted task means ± population standard deviations over 20 unseen tasks per task type. AER and LER are higher-is-better; Instability Frequency (IF, %) is lower-is-better. IDT's IF is omitted for BlockedUnlockPickup because it fails on the task. Best results are **bolded**; second-best results are underlined.

Task Type	Metric	AD	AD-ϵ	IDT	CV-ICRL
SimpleCrossing S9N3	AER	0.929 ± 0.067	0.152 ± 0.119	0.218 ± 0.181	$\mathbf{0.945 \pm 0.015}$
	LER	0.941 ± 0.029	0.396 ± 0.252	0.529 ± 0.226	$\mathbf{0.949 \pm 0.012}$
	IF (%)	3.428 ± 8.615	81.378 ± 10.588	70.875 ± 15.810	$\mathbf{1.561 \pm 1.880}$
SimpleCrossing S11N5	AER	0.865 ± 0.124	0.138 ± 0.107	0.323 ± 0.296	$\mathbf{0.902 \pm 0.076}$
	LER	0.886 ± 0.142	0.329 ± 0.242	0.587 ± 0.331	$\mathbf{0.921 \pm 0.066}$
	IF (%)	15.754 ± 15.686	73.744 ± 12.312	67.190 ± 26.947	$\mathbf{13.209 \pm 13.108}$
BlockedUnlock Pickup	AER	0.911 ± 0.160	0.219 ± 0.370	0.000 ± 0.000	$\mathbf{0.955 \pm 0.010}$
	LER	0.911 ± 0.209	0.057 ± 0.134	0.000 ± 0.000	$\mathbf{0.961 \pm 0.005}$
	IF (%)	2.493 ± 7.373	83.978 ± 10.186	–	$\mathbf{1.825 \pm 2.732}$
Unlock	AER	0.940 ± 0.025	0.042 ± 0.026	0.258 ± 0.157	$\mathbf{0.966 \pm 0.008}$
	LER	0.941 ± 0.029	0.552 ± 0.255	0.492 ± 0.239	$\mathbf{0.969 \pm 0.007}$
	IF (%)	4.885 ± 4.890	86.588 ± 10.775	72.213 ± 11.375	$\mathbf{0.217 \pm 0.172}$
LavaCrossing S9N3	AER	0.918 ± 0.035	0.902 ± 0.059	0.915 ± 0.057	$\mathbf{0.921 \pm 0.051}$
	LER	0.933 ± 0.051	0.936 ± 0.052	$\mathbf{0.945 \pm 0.020}$	0.939 ± 0.026
	IF (%)	$\mathbf{3.706 \pm 4.456}$	6.149 ± 9.108	5.590 ± 8.810	5.228 ± 8.787
LavaCrossing S9N2	AER	0.937 ± 0.027	0.918 ± 0.032	0.898 ± 0.097	$\mathbf{0.943 \pm 0.015}$
	LER	0.954 ± 0.012	0.946 ± 0.034	0.927 ± 0.060	$\mathbf{0.954 \pm 0.011}$
	IF (%)	2.101 ± 2.501	3.524 ± 3.735	8.213 ± 12.041	$\mathbf{1.557 \pm 1.289}$

We also analyze CV-ICRL from an imitation-learning perspective. Under the appendix's policy-identification model and value-error condition, conditioning on the predicted value token $v(s, C)$ can reduce the worst-case deviation from the reference behavior and yields a tighter performance bound. The formal statement and proof are provided in Appendix C.

5 Experiments

5.1 Experimental Setup

Environments and Tasks. We evaluate our methods on three testbeds: Dark Room [16], Minigrid [5], and Procgen [6]. Dark Room is a commonly used environment for ICRL algorithms. Minigrid presents a more challenging and diverse benchmark, with tasks whose underlying MDPs differ in both reward functions and observation spaces; the consistent symbolic observation format across task types lets us rigorously test the model's cross-task generalization. Procgen substantially raises the visual and dynamic complexity beyond gridworlds: each task type uses 64×64 RGB observations, procedurally generated levels with diverse layouts and enemy placements, and a 15-dimensional discrete action space. We use three Procgen task types: Bigfish, Starpilot, and Miner.

For Dark Room that has only one task type, we test ICRL policies on 20 unseen tasks that did not appear in training datasets. For Minigrid and Procgen, within each task type (e.g. LavaCrossingS9N3), we train baselines and CV-ICRL methods in 400 tasks and test on 20 unseen tasks. Additionally, we test 4 unseen Minigrid task types to further demonstrate the generalization capabilities of ICRL policies. More details of unseen tasks and task types are provided in Appendix D.1.

Preparations of Pretraining Datasets. We collect training datasets from the PPO algorithm training process [23]. For Dark Room, we train PPO in the same way as AD. For Minigrid, the PPO policy faces instability when training directly from scratch, so we adopt a pretrain–finetune strategy: we first pretrain the model and then finetune it on hard tasks. For Procgen, we similarly train PPO checkpoints per task type to generate the trajectory dataset. For Minigrid value annotation, evaluating 40 checkpoints in 20 environments for 500 steps each requires 400,000 transitions per task type. More details can be found in Appendix D.2.

Baselines. We use AD [16], AD-ϵ [32], and IDT [13] as baselines. We additionally report DPT [17] results in Appendix D.4 as a complementary ICRL baseline. We also compare with cross-episode adaptations of Reinformer and IC-CQL on four Minigrid task types; their protocols are provided

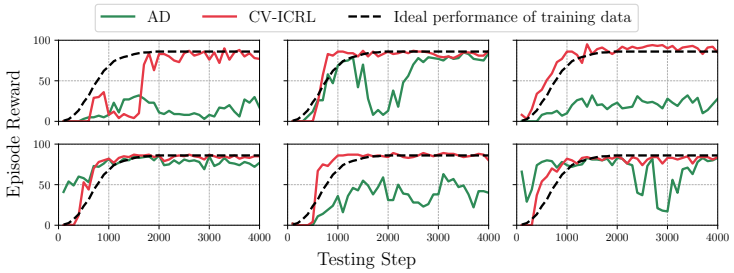


Figure 3: Across 6 different Dark Room tasks, AD struggles to maintain performance stability, whereas CV-ICRL substantially improves stability.

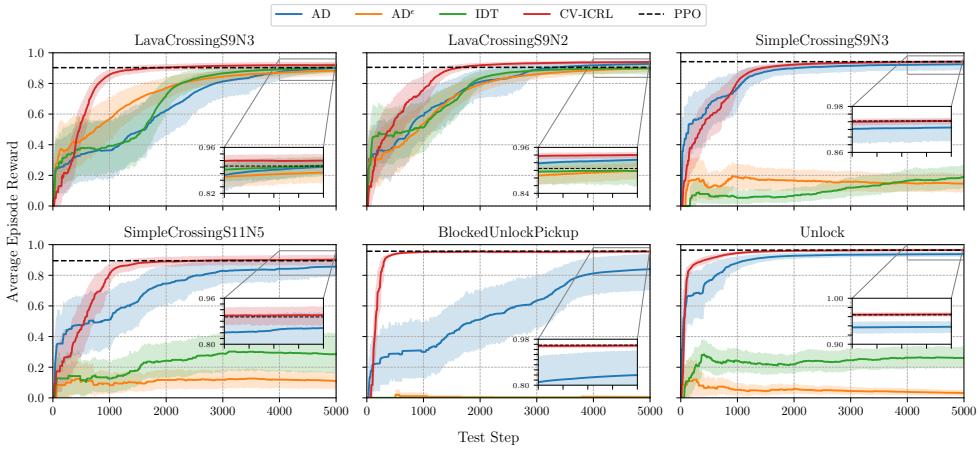


Figure 4: Test time AER in Minigrid. Curves show the mean over 20 unseen tasks with 95% confidence intervals; the black dashed line is a single-task PPO reference trained directly on each evaluation task.

in Appendix D.7. We implement AD, AD-ϵ, IDT, and CV-ICRL on a GPT-2 [21] based backbone, ensuring comparable parameter scales and closely matched architectural hyperparameters. The AD/CV-ICRL comparison also matches data ordering and training budget. Added-baseline architectures and protocols are described separately. More details can be found in Appendix D.3.

Metrics. To evaluate the overall performance, we use **Average Episode Return** (AER) as our main metric. In addition, we also consider the **Last Episode Return** (LER), which is used in previous work [26], as it reflects the final performance, indicating the ultimate in-context learning outcomes. In addition, to quantify the performance stability of ICRL policy, we propose a metric named **Instability Frequency**, which is calculated as the proportion of episodes in which the episode reward decreases by at least 5% compared to the previous one. The Instability Frequency IF is given by $IF = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(r_i \leq 0.95 \cdot r_{i-1})$, where N is the total number of episodes, r_i is the reward of the i-th episode, and $\mathbb{I}(\cdot)$ is the indicator function.

5.2 Main Results

Case study: performance degradation following occasional low-return rollouts. To examine performance instability in data-scarce test time adaptation, we conduct a case study on Dark Room. We observe that episode returns can occasionally drop sharply, which is consistent with sampling noise and the stochasticity of the policy/environment. Under the PG-like interpretation, such low-return outliers can perturb return-based credit assignment. Crucially, after a bad rollout occurs, AD [16] often exhibits degraded performance over subsequent rollouts (and recovers slowly), consistent with sensitivity to noisy return feedback during adaptation (shown in Figure 3).

The performance instability shown in case study is prevalent in many scenarios. CV-ICRL proves effective in alleviating this issue, leading to significant improvements in overall performance. Building on the initial observations from the case study, We then examine the prevalence of

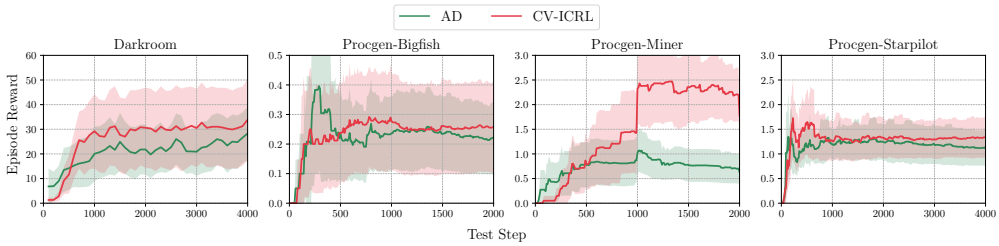


Figure 5: Test time episode reward in Dark Room and three Procgen task types. Curves show the mean over 20 unseen tasks/levels with 95% confidence intervals.

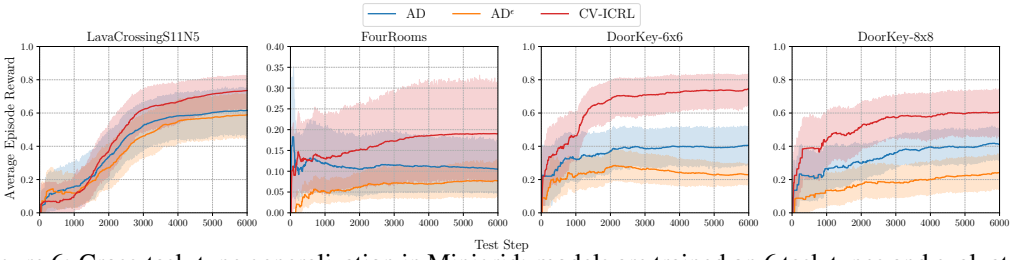


Figure 6: Cross-task-type generalization in Minigrid: models are trained on 6 task types and evaluated on 4 unseen task types.

this behavior across a diverse set of tasks in Minigrid. Our experiments across 6 diverse tasks in the Minigrid environment confirm this prevalence. As presented in Table 1, the results for baseline methods show frequent low-return episodes and performance degradation, reflected in a high Instability Frequency (IF) and lower AER. Against this backdrop, CV-ICRL delivers consistent gains in both stability and performance. As reported in Table 1, our method reduces IF relative to AD on five of the six task types, indicating less frequent performance degradation. In Figure 4, CV-ICRL reaches high returns earlier within the same interaction budget. LER gains are smaller on tasks where both methods approach the performance ceiling; AER and IF more clearly characterize differences along the adaptation path. We observe the same trend on the Dark Room and Procgen benchmarks (Figure 5). The temporal-variance analysis in Appendix D.6 shows that the wider aggregate confidence bands in some plots mainly reflect cross-task difficulty heterogeneity. We further provide a small-scale continuous-control evaluation on MetaWorld in Appendix D.5, where CV-ICRL improves over AD on both ML1-reach-v3 and ML1-push-v3.

Additional baseline comparisons. CV-ICRL achieves the highest mean AER on all four task types, while Reinformer is competitive except on LavaCrossingS9N3. Full AER, LER, and IF results are provided in Table 9, with the baseline protocols in Appendix D.7.

ICRL algorithm demonstrates generalization across varied tasks and scenes, and CV-ICRL enhances this ability. In prior works, ICRL methods have typically been validated in scenarios where the differences between tasks lie mainly in reward function or simple changes in the transition. Such simple modifications often result in tasks that are not significantly different from those seen in the training set, leading some work to suggest that these ICRL algorithms cannot address out-of-distribution generalization problems [22]. In this study, we aim to validate the OOD generalization capability in the Minigrid environment. We combine data from 6 types of tasks mentioned previously into a single dataset, train a general ICRL policy, and then evaluate its performance on 4 novel types of tasks (introduced in Appendix D.1).

As shown in Figure 6, the performance on these four novel task types demonstrates generalization to scenarios not encountered during pretraining. Here, "learn-to-learn" refers to improvement from early to later episodes through evolving test-time context while model parameters remain frozen. For example, retained context can record a failed DoorKey attempt or an unsuccessful Lava route and inform later episodes. Moreover, CV-ICRL outperforms other AD-like methods, showing faster adaptation to novel tasks and achieving a higher average episode return.

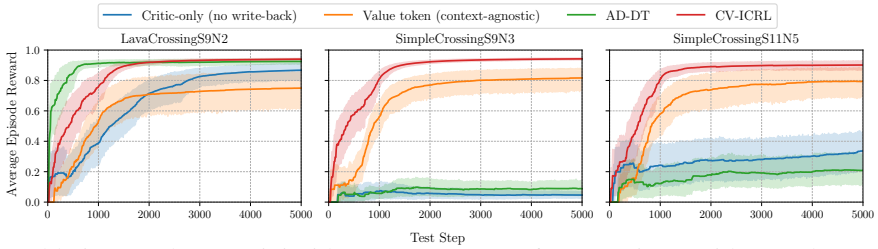


Figure 7: Ablation results on Minigrid. CV-ICRL outperforms variants without value write-back, without task-dependent value information, or with RTG conditioning, showing that gains come from feeding a meaningful context-value estimate back into the policy.

5.3 Ablations

We conduct ablations to separate three possible sources of improvement: auxiliary value supervision, the presence of an additional scalar token, and DT-style scalar conditioning. **Critic-only (no write-back)** trains the value head with the same regression target but removes v_t from the context, so the policy cannot condition on the critic prediction. **Value token (context-agnostic)** keeps the value-token channel but replaces the predicted context value with a fixed timestep function $v_t \leftarrow g(t)$, independent of the task, state, and rewards. **AD-DT (RTG conditioning)** follows the AD training protocol but replaces reward tokens with return-to-go (RTG) tokens, testing whether a DT-style scalar conditioning signal is sufficient.

As shown in Figure 7, all three variants underperform full CV-ICRL. The critic-only variant supports a contribution beyond multitask supervision from writing value predictions back into the context. The context-agnostic token variant shows that an additional timestep channel alone does not match full CV-ICRL: it exposes interaction progress but cannot distinguish histories at the same step. AD-DT is competitive on LavaCrossingS9N2 but weaker on the two SimpleCrossing task types. With sparse terminal rewards and desired return $G^* = 1$, its RTG remains nearly constant before termination, providing little graded feedback before success. Together, these task-dependent results support our central design choice of feeding a context-conditioned value estimate back into the in-context policy.

Deployment value-token interventions. Over 8,000 interaction steps, zeroing or freezing the written value token reduces AER on both SimpleCrossingS9N3 and BlockedUnlockPickup, supporting behavioral reliance on token content and continued write-back. Full results and intervention definitions are provided in Table 11 and Appendix D.9.

Alternative value supervision. We also train with the source PPO critic target $y_{\mathrm{PPO}}(s_t, C_t) = V^{\pi_{k(C_t)}}(s_t)$, where the context identifies the source checkpoint. It improves AER on the two SimpleCrossing task types, while checkpoint-return supervision performs better on the other four. The complete six-task comparison is provided in Table 10 (Appendix D.8).

6 Conclusion

In this paper, we frame ICRL as a policy-gradient-like learning process and relate its performance instability to high-variance return-based credit signals. To mitigate this issue, we propose **CV-ICRL**, which equips the in-context policy with a learned *context value* prediction and writes it back into the context, enabling TD-style, actor–critic credit assignment during inference time. Across Dark Room, Minigrid, and Procgen, CV-ICRL yields more stable test time adaptation and improves cross-episode returns. **Limitations.** Our policy-gradient view is supported by recent results on implicit weight adaptation in Transformers but is not yet a formal equivalence. Our cross-task-type generalization experiments are mainly within Minigrid, with small-scale MetaWorld evidence; broader environment-family shifts remain future work. More refined context-value estimation may further improve performance.

Impact Statement

This paper presents work whose goal is to advance the field of in-context reinforcement learning. Our method and algorithm do not involve any adversarial attack and will not endanger human security. All of our experiments are performed in the simulation environment, which does not involve ethical and fair issues.

Acknowledgment

This work was supported by the Exploratory Research Grant of the Shanghai Innovation Institute. The authors declare no competing interests.

References

[1] Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shimon Whiteson. A survey of meta-reinforcement learning. *arXiv preprint arXiv:2301.08028*, 2023.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[3] Stephanie Chan, Adam Santoro, Andrew Lampinen, Jane Wang, Aaditya Singh, Pierre Richemond, James McClelland, and Felix Hill. Data distributional properties drive emergent in-context learning in transformers. *Advances in neural information processing systems*, 35:18878–18891, 2022.

[4] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

[5] Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo Perez-Vicente, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. In *Advances in Neural Information Processing Systems 36, New Orleans, LA, USA*, December 2023.

[6] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International conference on machine learning*, pages 1282–1289. PMLR, 2019.

[7] Zhenwen Dai, Federico Tomasi, and Sina Ghiassian. In-context exploration-exploitation for reinforcement learning. *arXiv preprint arXiv:2403.06826*, 2024.

[8] Can Demircan, Tankred Saanum, Akshay Jagadish, Marcel Binz, and Eric Schulz. Sparse autoencoders reveal temporal difference learning in large language models. In *International Conference on Learning Representations*, volume 2025, pages 23743–23768, 2025.

[9] Benoit Dherin, Michael Munn, Hanna Mazzawi, Michael Wunder, and Javier Gonzalvo. Learning without training: The implicit dynamics of in-context learning. *arXiv preprint arXiv:2507.16003*, 2025.

[10] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, et al. A survey on in-context learning. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pages 1107–1128, 2024.

[11] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[12] Haoran He, Peilin Wu, Chenjia Bai, Hang Lai, Lingxiao Wang, Ling Pan, Xiaolin Hu, and Weinan Zhang. Bridging the sim-to-real gap from the information bottleneck perspective. In *8th Annual Conference on Robot Learning*, 2024.

[13] Sili Huang, Jifeng Hu, Hechang Chen, Lichao Sun, and Bo Yang. In-context decision transformer: Reinforcement learning via hierarchical chain-of-thought. In *Forty-first International Conference on Machine Learning*, 2024.

[14] Yuheng Jing, Bingyun Liu, Kai Li, Yifan Zang, Haobo Fu, Qiang Fu, Junliang Xing, and Jian Cheng. Opponent modeling with in-context search. *Advances in Neural Information Processing Systems*, 37:61549–61591, 2024.

[15] Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

[16] Michael Laskin, Luyu Wang, Junhyuk Oh, Emilio Parisotto, Stephen Spencer, Richie Steigerwald, DJ Strouse, Steven Stenberg Hansen, Angelos Filos, Ethan Brooks, et al. In-context reinforcement learning with algorithm distillation. In *The Eleventh International Conference on Learning Representations*, 2023.

[17] Jonathan Lee, Annie Xie, Aldo Pacchiano, Yash Chandak, Chelsea Finn, Ofir Nachum, and Emma Brunskill. Supervised pretraining can learn in-context reinforcement learning. *Advances in Neural Information Processing Systems*, 36:43057–43083, 2023.

[18] Hao Liu and Pieter Abbeel. Emergent agentic transformer from chain of hindsight experience. In *International Conference on Machine Learning*, pages 21362–21374. PMLR, 2023.

[19] Jinmei Liu, Fuhong Liu, Jianye Hao, Bo Wang, Huaxiong Li, Chunlin Chen, and Zhi Wang. Scalable in-context q-learning. *arXiv preprint arXiv:2506.01299*, 2025.

[20] Amir Moeini, Jiuqi Wang, Jacob Beck, Ethan Blaser, Shimon Whiteson, Rohan Chandra, and Shangtong Zhang. A survey of in-context reinforcement learning. *arXiv preprint arXiv:2502.07978*, 2025.

[21] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI Technical Report*, 2019.

[22] Sharath Chandra Raparthy, Eric Hambro, Robert Kirk, Mikael Henaff, and Roberta Raileanu. Generalization to new sequential decision making tasks with in-context learning. In *International Conference on Machine Learning*, pages 42138–42158. PMLR, 2024.

[23] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[24] Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

[25] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

[26] Denis Tarasov, Alexander Nikulin, Ilya Zisman, Albina Klepach, Andrei Polubarov, Lyubaykin Nikita, Alexander Derevyagin, Igor Kiselev, and Vladislav Kurenkov. Yes, q-learning helps offline in-context rl. In *NeurIPS 2024 Workshop on Scaling Self-Improving Foundation Models without Human Supervision*, 2024.

[27] Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pages 35151–35174. PMLR, 2023.

[28] Jiuqi Wang, Ethan Blaser, Hadi Daneshmand, and Shangtong Zhang. Transformers can learn temporal difference methods for in-context reinforcement learning. In *International Conference on Learning Representations*, 2025. URL `https://proceedings.iclr.cc/paper_files/paper/2025/file/2e3435554b430bd8fe92a60c509929a0-Paper-Conference.pdf`.

[29] Zhijie Wang, Bo Jiang, and Shuai Li. In-context learning on function classes unveiled for transformers. In *Forty-first International Conference on Machine Learning*, 2024.

[30] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

[31] Zifeng Zhuang, Dengyun Peng, Jinxin Liu, Ziqi Zhang, and Donglin Wang. Reinformer: Max-return sequence modeling for offline RL. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 62707–62722. PMLR, 2024. URL `https://proceedings.mlr.press/v235/zhuang24b.html`.

[32] Ilya Zisman, Vladislav Kurenkov, Alexander Nikulin, Viacheslav Sinii, and Sergey Kolesnikov. Emergence of in-context reinforcement learning from noise distillation. In *Forty-first International Conference on Machine Learning*, 2024.

A Algorithm Pseudocode

Algorithm 1 summarizes the supervised pretraining and the test-time deployment of CV-ICRL. During pretraining, augmented contexts are constructed by teacher-forcing each transition with the source-checkpoint return $\hat{J}^{(k)}$ as the value token. At test time, the model produces v_t from its critic head and writes it back into the context for subsequent steps; the in-context distribution thus evolves under the model's own value predictions.

Algorithm 1: CV-ICRL: supervised pretraining and test-time deployment.

Input: Trajectory dataset $\mathcal{D}$ collected along online RL checkpoints $\{\pi^{(k)}\}_{k=1}^{K}$, each annotated with the checkpoint return $\hat{J}^{(k)}$; loss weight λ.

Output: Pretrained Transformer with parameters θ, providing actor head π_θ and critic head v_θ.

Stage 1: Supervised pretraining.

Form augmented contexts $\bar{C}_t = ((s_0, a_0, r_0, v_0), \ldots, (s_{t-1}, a_{t-1}, r_{t-1}, v_{t-1}))$ with teacher-forced $v_i = \hat{J}^{(k_i)}$ from each sample's source checkpoint.

while *not converged* **do**

  Sample $(\bar{C}_t, s_t, a_t, \hat{J}^{(k_t)}) \sim \mathcal{D}$.

  $\mathcal{L} \leftarrow -\log \pi_\theta(a_t \mid s_t, \bar{C}_t) + \lambda \left(v_\theta(s_t, \bar{C}_t) - \hat{J}^{(k_t)}\right)^2$.

  Update θ by gradient descent on $\mathcal{L}$.

end

Stage 2: Test-time deployment on an unseen task.

Reset environment to obtain s_0; initialize $\bar{C}_0 \leftarrow \emptyset$.

for $t = 0, 1, \ldots, T-1$ **do**

  $v_t \leftarrow v_\theta(s_t, \bar{C}_t)$; `// predict context value`

  $a_t \sim \pi_\theta(\cdot \mid s_t, \bar{C}_t)$; `// sample action`

  Execute a_t, observe r_t and s_{t+1}.

  $\bar{C}_{t+1} \leftarrow U\big(\bar{C}_t, (s_t, a_t, r_t, v_t)\big)$; `// write` v_t `back`

  Retain only the most recent context window in $\bar{C}_{t+1}$.

  if *the episode terminates or is truncated* **then**

    Reset the environment to obtain the next state s_{t+1}, retaining $\bar{C}_{t+1}$.

  end

end

B Proof of Property 4.1

Full proof of Property 4.1. **Step 1: Rao–Blackwell variance reduction for G_t.** Throughout this proof, C_t denotes the current context; for CV-ICRL it should be read as the augmented context $\bar{C}_t$ that contains previously written value tokens. We omit the bar for readability. Let $H_t = (C_t, s_t, a_t, r_t, s_{t+1}, C_{t+1})$, where C_{t+1} is deterministically obtained by the relevant context update, and define $\tilde{G}_t \triangleq \mathbb{E}[G_t \mid H_t]$. By the law of total variance,

$$\mathrm{Var}(G_t) = \mathrm{Var}(\mathbb{E}[G_t \mid H_t]) + \mathbb{E}[\mathrm{Var}(G_t \mid H_t)]. \tag{11}$$

Since $\mathbb{E}[\mathrm{Var}(G_t \mid H_t)] \geq 0$, it follows immediately that

$$\mathrm{Var}(G_t) \geq \mathrm{Var}(\mathbb{E}[G_t \mid H_t]) = \mathrm{Var}(\tilde{G}_t). \tag{12}$$

Under assumption (7), the ideal conditional expectation admits the explicit form $\tilde{G}_t = r_t + \gamma V(s_{t+1}, C_{t+1})$. CV-ICRL implements this ideal target with the critic prediction $v_{t+1} \approx V(s_{t+1}, C_{t+1})$. This identification is not needed for the inequality (12) itself, but is used to connect the Rao–Blackwellized target to the bootstrapped TD form.

Step 2: Conditional Rao–Blackwellization given (C_t, s_t, a_t). Let $Z_t \triangleq (C_t, s_t, a_t)$. Because $\tilde{G}_t = \mathbb{E}[G_t \mid H_t]$ and H_t contains Z_t as a subvector, we also have the conditional Rao–Blackwell identity

$$\mathbb{E}[G_t \mid H_t] = \mathbb{E}[\,\mathbb{E}[G_t \mid H_t] \mid H_t\,] \quad \text{and} \quad \mathbb{E}[G_t \mid Z_t] = \mathbb{E}[\tilde{G}_t \mid Z_t]. \tag{13}$$

Moreover, applying the law of total variance conditionally on Z_t yields

$$\mathrm{Var}(G_t \mid Z_t) = \mathrm{Var}(\mathbb{E}[G_t \mid H_t] \mid Z_t) + \mathbb{E}[\mathrm{Var}(G_t \mid H_t) \mid Z_t] \geq \mathrm{Var}(\tilde{G}_t \mid Z_t). \tag{14}$$

Step 3: Multiplying by the score term. By definition, $\psi_t = \nabla_\theta \log \pi_\theta(a_t \mid C_t, s_t)$ is a function of (C_t, s_t, a_t) only, hence ψ_t is Z_t-measurable. Therefore, conditioned on Z_t, ψ_t is deterministic. For any scalar random variable X (or for each coordinate of a vector), we have

$$\mathrm{Var}(\psi_t X \mid Z_t) = \|\psi_t\|^2 \,\mathrm{Var}(X \mid Z_t), \tag{15}$$

where $\|\psi_t\|^2$ denotes the squared norm (and the identity holds componentwise in the vector case). Applying (15) with $X = G_t$ and $X = \tilde{G}_t$, and using (14), gives

$$\mathrm{Var}(\psi_t G_t \mid Z_t) = \|\psi_t\|^2 \,\mathrm{Var}(G_t \mid Z_t) \geq \|\psi_t\|^2 \,\mathrm{Var}(\tilde{G}_t \mid Z_t) = \mathrm{Var}(\psi_t \tilde{G}_t \mid Z_t), \tag{16}$$

which is exactly (10) (recalling $Z_t = (C_t, s_t, a_t)$).

This completes the proof. □

Approximate value predictions. Let $v_{t+1} = V_{t+1} + e_{t+1}$ and $\tilde{G}_t = r_t + \gamma V_{t+1}$. The practical target is

$$\widehat{G}_t^{\mathrm{TD}} = \tilde{G}_t + \gamma e_{t+1}.$$

Under the same distribution as the ideal target, its mean shift is $\gamma\mathbb{E}[e_{t+1}]$, and

$$\mathrm{Var}(\widehat{G}_t^{\mathrm{TD}}) = \mathrm{Var}(\tilde{G}_t) + \gamma^2 \,\mathrm{Var}(e_{t+1}) + 2\gamma \,\mathrm{Cov}(\tilde{G}_t, e_{t+1}).$$

Thus the practical target retains a variance advantage over G_t whenever

$$\gamma^2 \,\mathrm{Var}(e_{t+1}) + 2\gamma \,\mathrm{Cov}(\tilde{G}_t, e_{t+1}) < \mathrm{Var}(G_t) - \mathrm{Var}(\tilde{G}_t).$$

Controlled approximation error can therefore preserve the ideal Rao–Blackwell variance gain; a nonzero mean error introduces a mean shift.

C Improved Performance Bound Conditioned on Context Value

Notation in this appendix. We carefully distinguish the *ideal* (true) context value from the *estimated* value token used by CV-ICRL. For a policy π, the context value keeps the definition from Appendix 4.1: $V^\pi(s; C)$ is the expected discounted return conditioned on the current state and context. In the distillation analysis below, however, the scalar value token is supervised by a checkpoint-level return estimate rather than by exact per-state values. We denote this proxy target by $y^*(C)$. For the reference source policy $\pi^* \in \Pi$ associated with the current training context,

$$y^*(C) \triangleq J(\pi^*),$$

a deterministic scalar functional of π^* alone, where $J(\pi^*)$ is the Monte-Carlo evaluated expected return of that checkpoint policy. Thus $y^*(C)$ should be understood as the ideal value-token target used in this analysis, not as a separate definition of the state–context value $V^{\pi^*}(s; C)$. Lowercase $v(C)$ denotes the *estimated* context value produced by our value head and written into the context; its estimation error is

$$d_v \triangleq |\, v(C) - y^*(C) \,|.$$

The ideal regime $v = y^*$ (i.e., $d_v = 0$) is treated first in Theorem C.4; the realistic regime $d_v > 0$ is handled in Theorem C.5.

Property C.1 (Improved Performance Bound)**.** Let $\pi(\cdot \mid s, C)$ denote the base ICRL policy and let $\pi_v(\cdot \mid s, C, v)$ be its value-conditioned variant (CV-ICRL), with $v(C)$ the predicted context-value token. Let $\pi^*(\cdot \mid s, C)$ be the reference (source) policy. Define the worst-case TV deviations

$$\epsilon_{\mathrm{base}} \triangleq \sup_C D_{TV}\big(\pi^*(\cdot \mid s, C) \,\|\, \pi(\cdot \mid s, C)\big), \qquad \epsilon_v \triangleq \sup_C D_{TV}\big(\pi^*(\cdot \mid s, C) \,\|\, \pi_v(\cdot \mid s, C, v)\big).$$

Then for any state s,

$$\sup_C \big|V^{\pi^*}(s; C) - V^{\pi}(s; C)\big| \leq \frac{2r_{\max}}{(1-\gamma)^2}\,\epsilon_{\mathrm{base}}, \tag{17}$$

$$\sup_C \big|V^{\pi^*}(s; C) - V^{\pi_v}(s; C, v)\big| \leq \frac{2r_{\max}}{(1-\gamma)^2}\,\epsilon_v. \tag{18}$$

Furthermore, under the posterior model of Theorem C.4, when the estimation error of v satisfies $d_J - 2d_v > 0$ (with d_J defined in (30); Theorem C.5), conditioning on v strictly increases the posterior mass on π^*, yielding $\epsilon_v < \epsilon_{\text{base}}$ and a strictly tighter performance bound for π_v. The proof combines Theorems C.2 to C.4 and is given at the end of this appendix.

To formally quantify the advantage of incorporating value information, we must first establish the baseline performance. We begin by presenting the theoretical guarantee on the performance error for a standard ICRL policy, which is learned without value side-information. This result, adapted from imitation learning works [12], serves as the foundation for our subsequent comparison.

Here, for different pretrained data paris (C_t, s_t, a_t) (with v_t for our methods), their source policies $\pi_t^{(k)}$ are different. Here, we formulate the learning target policy condition on C_t to be $\pi^*(a|s, C_t)$, where $\pi^*(a|s, C_t) = \pi_t^{(k)}(a|s)$.

Lemma C.2 (Performance Bound of ICRL Policy)**.** *The suboptimality gap for the ICRL policy, defined as the worst-case difference in policy value between the learned ICRL policy $\pi(\cdot|s, C)$ and the empirical optimal policy (target policy) $\pi^*(\cdot|s, C)$, is bounded as follows:*

$$\sup_C |V^{\pi^*}(s; C) - V^{\pi}(s; C)| \leq \frac{2r_{max}}{(1-\gamma)^2}\epsilon \tag{19}$$

where ϵ represents the worst-case statistical divergence between the two policies:

$$\epsilon = \sup_C D_{TV}(\pi^*(\cdot|s, C)\|\pi(\cdot|s, C)) \tag{20}$$

For notational simplicity, we use $V(s; C)$ to denote $V^{\pi}(s; C)$, and use $V^*(s, C)$ to denote $V^{\pi^*}(s; C)$ in the remainder of this proof.

Proof. Expanding $V(s; C)$ for one step:

$$V(s; C) = \sum_{a,s'} T(s'|s, a)\pi(a|s)[R(s, a) + \gamma V(s'; C'))] \tag{21}$$

Here, $C' = C \cup \{(a, r, s')\}$.

Thus,

$$\begin{aligned}
&V^*(s; C) - V(s; C) \\
=&\sum_{a,s'} T(s'|s, a)[\pi^*(a|s)[R(s, a) + \gamma V^*(s'; C'))] - \pi(a|s)[R(s, a) + \gamma V(s'; C'))]] \\
=&\sum_{a,s'} T(s'|s, a)R(s, a)[\pi^*(a|s) - \pi(a|s)] \\
&+ \gamma \sum_{a,s'} T(s'|s, a)[\pi^*(a|s)V^*(s'; C') - \pi(a|s)V(s'; C')] \\
=&\sum_{a,s'} T(s'|s, a)R(s, a)[\pi^*(a|s) - \pi(a|s)] \\
&+ \gamma \sum_{a,s'} T(s'|s, a)\pi^*(a|s)[V^*(s'; C') - V(s'; C')] + V(s'; C')[\pi^*(a|s) - \pi(a|s)] \\
=&\sum_{a,s'} T(s'|s, a)[R(s, a) + \gamma V(s'; C')][\pi^*(a|s) - \pi(a|s)] \\
&+ \gamma \sum_{a,s'} T(s'|s, a)\pi^*(a|s)[V^*(s'; C') - V(s'; C')] \\
\leq& \|T(R + \gamma V)\|_\infty \|\pi^* - \pi\|_\infty + \gamma\|V^* - V\|_\infty && \triangleright \text{ Hölder's inequality} \\
\leq& 2\|(R + \gamma V)\|_\infty \epsilon + \gamma\|V^* - V\|_\infty && \triangleright \text{Equation (20)} \\
\leq& \frac{2r_{\max}}{1-\gamma}\epsilon + \gamma\|V^* - V\|_\infty && \triangleright V(s) \leq \frac{r_{\max}}{1-\gamma}
\end{aligned}$$

Then,

$$\|V^* - V\|_\infty \le \frac{2r_{\max}}{1-\gamma}\epsilon + \gamma\|V^* - V\|_\infty$$
$$(1-\gamma)\|V^* - V\|_\infty \le \frac{2r_{\max}}{1-\gamma}\epsilon$$

Then we have

$$\sup_C |V^*(s;C) - V(s;C)| = \|V^* - V\|_\infty \le \frac{2r_{\max}}{(1-\gamma)^2}\epsilon \tag{22}$$

□

We treat the inference process of an ICRL model as a two-stage sampling procedure: first, sampling a candidate source policy $\pi \in \Pi$ from the posterior $P(\pi \mid C)$ induced by the context, and second, sampling an action a from the chosen policy. The resulting in-context distribution is the posterior mixture

$$\pi(\cdot|s,C) = \sum_{\pi\in\Pi} P(\pi|C)\,\pi(\cdot|s). \tag{23}$$

Analogously, conditioning on the value token v yields $\pi(\cdot|s,C,v) = \sum_{\pi\in\Pi} P(\pi \mid C,v)\,\pi(\cdot|s)$.

Lemma C.3 (TV bound via posterior mass)**.** *Assume the in-context distribution decomposes via the posterior mixture* (23)*, with $\pi^* \in \Pi$. Then for any value token $v(C)$ written into the context,*

$$\sup_C D_{TV}\big(\pi^*(\cdot|s,C) \,\|\, \pi(\cdot|s,C,v)\big) \le 1 - P(\pi^* \mid C,v), \tag{24}$$

$$\sup_C D_{TV}\big(\pi^*(\cdot|s,C) \,\|\, \pi(\cdot|s,C)\big) \le 1 - P(\pi^* \mid C). \tag{25}$$

Eq. (24) *holds for any value token v; the ideal regime $v = y^*$ feeds into Theorem C.4, while the realistic case $d_v > 0$ is treated in Theorem C.5.*

Proof. We prove (24); the proof of (25) is identical. By definition of total variation distance,

$$\begin{aligned}
&D_{TV}(\pi^*(\cdot \mid s,C) \,\|\, \pi(\cdot \mid s,C,v)) \\
=&\frac{1}{2}\sum_a \left|\pi^*(a \mid s,C) - \sum_{\pi\in\Pi} P(\pi \mid C,v)\,\pi(a \mid s)\right| \\
=&\frac{1}{2}\sum_a \left|\pi^*(a \mid s,C) - P(\pi^* \mid C,v)\pi^*(a \mid s,C) - \sum_{\pi\in\Pi\setminus\{\pi^*\}} P(\pi \mid C,v)\,\pi(a \mid s)\right| \\
=&\frac{1}{2}\sum_a \left|(1 - P(\pi^* \mid C,v))\pi^*(a \mid s,C) - \sum_{\pi\in\Pi\setminus\{\pi^*\}} P(\pi \mid C,v)\,\pi(a \mid s)\right| \\
\le&\frac{1}{2}\sum_a \Big((1 - P(\pi^* \mid C,v))\pi^*(a \mid s,C) + \sum_{\pi\in\Pi\setminus\{\pi^*\}} P(\pi \mid C,v)\,\pi(a \mid s)\Big) \\
=&\frac{1}{2}\Big((1 - P(\pi^* \mid C,v))\sum_a \pi^*(a \mid s,C) + \sum_{\pi\in\Pi\setminus\{\pi^*\}} P(\pi \mid C,v)\sum_a \pi(a \mid s)\Big) \\
=&\frac{1}{2}\Big((1 - P(\pi^* \mid C,v)) + \sum_{\pi\in\Pi\setminus\{\pi^*\}} P(\pi \mid C,v)\Big) \\
=&\frac{1}{2}\Big((1 - P(\pi^* \mid C,v)) + (1 - P(\pi^* \mid C,v))\Big) \\
=&1 - P(\pi^* \mid C,v),
\end{aligned}$$

where we used $\sum_a \pi^*(a \mid s,C) = 1$, $\sum_a \pi(a \mid s) = 1$, and $P(\pi^* \mid C,v) + \sum_{\pi\neq\pi^*} P(\pi \mid C,v) = 1$.

Similarly, for the naive ICRL policy $\pi(\cdot|C)$ we can also have

$$\sup_C D_{TV}(\pi^*(\cdot|s,C)\|\pi(\cdot|s,C)) \leq 1 - P(\pi^*|C) \tag{26}$$

□

Now we consider the posterior term $P(\pi|C)$. We will prove that $P(\pi^*|C, y^*)$ is larger than $P(\pi^*|C)$, so the upper bound of $D_{TV}(\pi^*(\cdot|C)\|\pi(\cdot|C, y^*))$ is tighter than the corresponding bound without the value token.

Lemma C.4 (Comparison of two upper bounds)**.** *Let $\Pi = \{\pi_i\}_{i=1}^N$ be the finite set of source-policy checkpoints used during distillation, with a uniform prior $P(\pi_i) = 1/N$ (each checkpoint is sampled with equal frequency in the ICRL training pipeline). Assume the joint likelihood factorizes as*

$$P(C, y^* \mid \pi) = P(C \mid \pi)\, P(y^* \mid \pi), \tag{27}$$

which is justified by noting that $y^*(C) = J(\pi^*)$ is a deterministic functional of π^* alone (it does not depend on the realized context C given π); hence, conditional on a candidate π, the data C and the value-token observation y^* act as separate likelihood channels. We model the value likelihood by

$$P(y^* \mid \pi) \;\propto\; \exp\big(-\beta\,|y^*(C) - J(\pi)|\big), \qquad \beta > 0, \tag{28}$$

where $J(\pi)$ is the (true) expected return of π in the environment.

Fix a source policy $\pi^* \in \Pi$ *and take the observed value target to be*

$$y^*(C) = J(\pi^*). \tag{29}$$

Define the return separation

$$d_J \triangleq \min_{i \neq *} |J(\pi^*) - J(\pi_i)|. \tag{30}$$

Then

$$\frac{P(\pi^* \mid C, y^*)}{P(\pi^* \mid C)} \;\geq\; 1 + \delta_{\mathrm{rel}}, \tag{31}$$

where

$$\delta_{\mathrm{rel}} = \frac{\Big(\sum_{i \neq *} P(C \mid \pi_i)\Big)\,\big(e^{\beta d_J} - 1\big)}{P(C \mid \pi^*)e^{\beta d_J} + \sum_{i \neq *} P(C \mid \pi_i)} \;>\; 0 \quad \textit{whenever } d_J > 0. \tag{32}$$

Proof. By Bayes' rule and the uniform prior,

$$P(\pi \mid C) = \frac{P(C \mid \pi)}{\sum_j P(C \mid \pi_j)}. \tag{33}$$

Using the factorization assumption (27),

$$P(\pi \mid C, y^*) = \frac{P(C \mid \pi)P(y^* \mid \pi)}{\sum_j P(C \mid \pi_j)P(y^* \mid \pi_j)}. \tag{34}$$

Therefore,

$$\frac{P(\pi^* \mid C, y^*)}{P(\pi^* \mid C)} = \frac{P(y^* \mid \pi^*)\sum_j P(C \mid \pi_j)}{\sum_i P(C \mid \pi_i)P(y^* \mid \pi_i)}. \tag{35}$$

Now we lower bound the denominator using the value-likelihood model. Under (28) and $y^*(C) = J(\pi^*)$, we have

$$\frac{P(y^* \mid \pi^*)}{P(y^* \mid \pi_i)} = \exp\big(\beta\,|J(\pi^*) - J(\pi_i)|\big) \;\geq\; \exp(\beta d_J) \;\triangleq\; \Gamma_{\min}, \qquad \forall i \neq *, \tag{36}$$

where d_J is defined in (30). Equivalently, for all $i \neq *$,

$$P(y^* \mid \pi_i) \;\leq\; \frac{P(y^* \mid \pi^*)}{\Gamma_{\min}}. \tag{37}$$

Plugging (37) into the denominator of (35) yields

$$\frac{P(\pi^* \mid C, y^*)}{P(\pi^* \mid C)} = \frac{P(y^* \mid \pi^*) \sum_j P(C \mid \pi_j)}{P(C \mid \pi^*)P(y^* \mid \pi^*) + \sum_{i \neq *} P(C \mid \pi_i)P(y^* \mid \pi_i)}$$

$$\geq \frac{P(y^* \mid \pi^*) \sum_j P(C \mid \pi_j)}{P(C \mid \pi^*)P(y^* \mid \pi^*) + \sum_{i \neq *} P(C \mid \pi_i) \frac{P(y^* \mid \pi^*)}{\Gamma_{\min}}} \tag{38}$$

$$= \frac{\sum_j P(C \mid \pi_j)}{P(C \mid \pi^*) + \sum_{i \neq *} P(C \mid \pi_i)/\Gamma_{\min}}. \tag{39}$$

Let $S_* \triangleq P(C \mid \pi^*)$ and $S_{\text{other}} \triangleq \sum_{i \neq *} P(C \mid \pi_i)$. Then (39) becomes

$$\frac{P(\pi^* \mid C, y^*)}{P(\pi^* \mid C)} \geq \frac{S_* + S_{\text{other}}}{S_* + S_{\text{other}}/\Gamma_{\min}} = 1 + \frac{S_{\text{other}}(\Gamma_{\min} - 1)}{S_* \Gamma_{\min} + S_{\text{other}}}. \tag{40}$$

Substituting $\Gamma_{\min} = e^{\beta d_J}$ from (36) gives (32), and $\delta_{\text{rel}} > 0$ whenever $d_J > 0$. □

In practice the value head is not exact: the value token $v(C)$ deviates from the ideal value-token target $y^*(C) = J(\pi^*)$ by d_v, defined in the appendix notation. The next corollary shows that the posterior-ratio improvement of Theorem C.4 survives this error as long as d_v is not too large.

Corollary C.5 (Robustness of Theorem C.4 to value-estimation error)**.** *Suppose that, in the setting of Theorem C.4, the value token observed by the model is the estimate $v(C)$ in place of the ideal target $y^*(C)$, with error $d_v \triangleq |v(C) - y^*(C)|$, and adopt the value-likelihood model*

$$P(v \mid \pi) \propto \exp\big(-\beta\,|v(C) - J(\pi)|\big), \qquad \beta > 0. \tag{41}$$

Then the posterior-ratio inequality of Theorem C.4 continues to hold with the modified constant

$$\Gamma_{\min} \triangleq \exp\big(\beta(d_J - 2d_v)\big), \tag{42}$$

and in particular,

$$\frac{P(\pi^* \mid C, v)}{P(\pi^* \mid C)} > 1 \qquad \textit{whenever} \qquad d_J - 2d_v > 0. \tag{43}$$

Proof. For any $\pi_i \neq \pi^*$, the triangle inequality gives

$$|v(C) - J(\pi_i)| = |(v(C) - J(\pi^*)) + (J(\pi^*) - J(\pi_i))|$$
$$\geq |J(\pi^*) - J(\pi_i)| - |v(C) - J(\pi^*)| = |J(\pi^*) - J(\pi_i)| - d_v. \tag{44}$$

Plugging into (41), the likelihood ratio satisfies

$$\Gamma_i \triangleq \frac{P(v \mid \pi^*)}{P(v \mid \pi_i)} = \exp\Big(\beta\big(|v(C) - J(\pi_i)| - |v(C) - J(\pi^*)|\big)\Big)$$
$$\geq \exp\Big(\beta\big(|J(\pi^*) - J(\pi_i)| - 2d_v\big)\Big). \tag{45}$$

Taking the minimum over $i \neq *$ yields the uniform lower bound $\Gamma_i \geq \exp(\beta(d_J - 2d_v)) = \Gamma_{\min}$. Substituting this $\Gamma_{\min}$ into the proof of Theorem C.4 preserves the chain leading to (31), and the resulting δ_{rel} is strictly positive whenever $\Gamma_{\min} > 1$, i.e., $d_J - 2d_v > 0$. □

Proof of Theorem C.1. The two suboptimality-gap inequalities follow directly from Theorem C.2 applied to π and to π_v respectively, with the worst-case TV deviations ϵ_{base} and ϵ_v as defined in the statement of Theorem C.1.

For the strictness claim, Theorem C.3 gives

$$\epsilon_{\text{base}} \leq 1 - P(\pi^* \mid C), \qquad \epsilon_v \leq 1 - P(\pi^* \mid C, v).$$

Under the posterior model of Theorem C.4 with the ideal value token ($v = y^*$, $d_v = 0$), we have $P(\pi^* \mid C, v) \geq (1 + \delta_{\text{rel}})P(\pi^* \mid C) > P(\pi^* \mid C)$ whenever $d_J > 0$. Theorem C.5 extends this to the realistic case in which the value head incurs error d_v, with the strict inequality preserved as long as $d_J - 2d_v > 0$. In either case,

$$P(\pi^* \mid C, v) > P(\pi^* \mid C) \qquad \Longrightarrow \qquad \epsilon_v < \epsilon_{\text{base}},$$

and multiplying through by $2r_{\max}/(1 - \gamma)^2$ yields a strictly tighter performance bound for the value-conditioned policy π_v. □

D Details of Experiments

D.1 Environments

D.1.1 Dark Room

The Dark Room environment is a challenging testbed for an agent's ability to perform efficient exploration under conditions of extreme reward sparsity and partial observability. The task places an agent in a large gridworld with a single goal state, but the agent's perception is limited to its local vicinity. A positive reward is only granted upon reaching the goal, meaning the agent must conduct a systematic, memory-based search to explore the space without any guiding signals. It is therefore highly effective at evaluating an agent's capacity to use memory for long-term navigation and exploration.

D.1.2 Minigrid

The environments commonly employed in current methods, such as Dark Room and MiniWorld, are relatively simple. For instance, the state space in Dark Room is merely two-dimensional. More importantly, the task variations in these settings are largely confined to changes in the reward function or minor shifts in dynamics. This is insufficient to demonstrate the generalization capability of an ICRL algorithm to unseen tasks. To address this, we use Minigrid, a more complex and diverse environment, to demonstrate that our ICRL algorithm can indeed generalize to completely new types of tasks.

Minigrid environment features a collection of gridworld scenarios where an agent must infer and accomplish a goal through exploration. Tasks range from simple navigation, such as bypassing obstacles to reach a goal, to complex sequential decision-making, like opening a series of doors. Critically, Minigrid provides a fixed-size (7x7), uniform symbolic observation across all its diverse tasks. This standardization removes the challenge of unifying observation representations, making it an ideal platform to directly test how effectively an In-Context Reinforcement Learning (ICRL) policy can adapt to new tasks.

Below are the introductions of 6 task types used for both training and testing.

LavaCrossingS9N3. In this task type, the agent must navigate through a room with deadly lava streams running horizontally or vertically. The agent must reach the green goal square while avoiding lava, with a single safe crossing point for each stream. A path to the goal is guaranteed.

LavaCrossingS9N2. Similar to LavaCrossingS9N3, but the challenge is slightly less difficult compared to LavaCrossingS9N3.

SimpleCrossingS9N3. In this task type, the agent must navigate through a room with walls instead of lava. The objective is to reach the green goal square while avoiding walls. This task is easier compared to the LavaCrossing tasks and is useful for quickly testing algorithms.

SimpleCrossingS11N5. Similar to SimpleCrossingS9N3, the agent must reach the green goal square while avoiding walls. The task involves a larger environment with more walls, making it moderately more challenging than SimpleCrossingS9N3 but still easier than the LavaCrossing tasks.

BlockedUnlockPickup. In this task type, the agent must pick up an object placed in another room, behind a locked door. The door is blocked by a ball that the agent must first move to unlock the door. The agent must learn to move the ball, pick up the key, open the door, and then pick up the object in the other room.

Unlock. This task type is a simplified version of BlockedUnlockPickup. The agent only needs to pick up the key and then unlock the door.

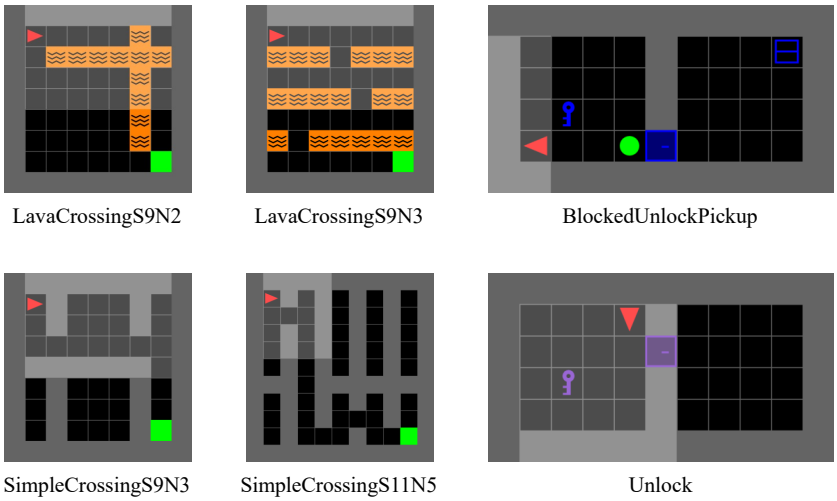


Figure 8: The 6 tasks for the main experiments, with different seeds corresponding to different layouts.

Below are the introductions of 4 task types used for the cross-task-type generalization experiments, and only for test.

LavaCrossingS11N5. Similar to LavaCrossingS9N3, LavaCrossingS11N5 introduces greater difficulty by increasing the number of lava streams and the complexity of the room layout. This makes it more challenging for the agent to find a safe path to the goal.

FourRooms. Agent must navigate a maze composed of four rooms, interconnected by gaps in the walls. The agent's goal is to reach the green goal square to receive a reward. Both the agent and the goal square are randomly placed in any of the four rooms.

DoorKey-6x6. In this task type, the agent must pick up a key to unlock a door and then reach the green goal square.

DoorKey-8x8. Similar to DoorKey-6x6, but is more complex due to the larger environment size.

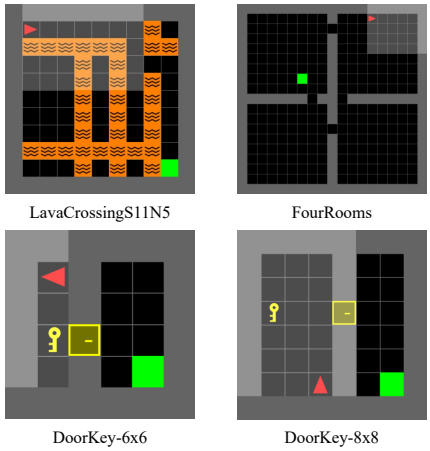


Figure 9: The 4 tasks for the cross-task generalization experiments, with different seeds corresponding to different layouts.

D.1.3 Procgen

Procgen [6] is a suite of procedurally generated arcade-style environments designed to evaluate generalization in deep RL. Each task type uses 64×64 RGB observations, a 15-dimensional discrete action space, and procedurally generated levels that vary in layout, dynamics, and visual appearance. Compared with gridworld testbeds (Dark Room, Minigrid), Procgen poses substantially greater perceptual and dynamical challenges. We evaluate on three Procgen task types described below.

Bigfish. The agent controls a small fish that must consume smaller fish to grow while avoiding larger fish that would consume it; the goal is to maximize cumulative growth.

Starpilot. A side-scrolling shoot-em-up where the agent dodges enemies and projectiles while shooting incoming targets to maximize the score.

Miner. The agent navigates a tile-based mining environment, collects diamonds, and exits without being killed by falling boulders.

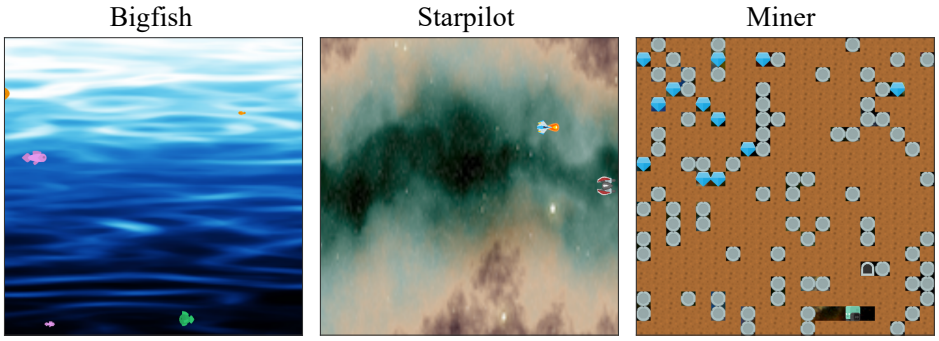


Figure 10: The three Procgen task types used in our experiments: Bigfish, Starpilot, and Miner.

D.1.4 MetaWorld

MetaWorld is a benchmark of robotic manipulation tasks designed for continuous-control reinforcement learning. Unlike Dark Room, Minigrid, and Procgen, where the policy selects from a discrete action set, MetaWorld requires the agent to output continuous motor commands for controlling a simulated robotic arm. This makes the environment qualitatively different from our main testbeds: adaptation must account for smooth control, contact dynamics, and precise object manipulation rather than only selecting among symbolic or discrete actions.

We use MetaWorld as a small-scale stress test for whether CV-ICRL can transfer beyond discrete-control domains. Specifically, we evaluate on two ML1 tasks, `ML1-reach-v3` and `ML1-push-v3`. In `ML1-reach-v3`, the agent must move the robot end-effector to a target position; in `ML1-push-v3`, it must push an object toward a goal. These tasks provide a compact continuous-control setting for probing whether context-value conditioning remains useful when the action space and dynamics differ substantially from the environments used in the main experiments.

D.2 Preparations of Training Datasets

For the Dark Room environment, we train PPO in the same way as Algorithm Distillation. We choose 20 different environments, each with different goal positions (x, y), and use these to train the PPO agent.

For Minigrid, the environment is more challenging than Dark Room. The PPO policy faces instability when trained from scratch for each layout. To address this, we adopt a pretrain-finetune strategy: we first pretrain a PPO model on a variety of layouts and then finetune it on specific layouts as needed.

For Procgen, we train a separate PPO model per task type (Bigfish, Starpilot, Miner) on procedurally generated levels, using a CNN-based policy network suited to the 64×64 RGB observations.

The PPO algorithm we use is based on the implementation provided by Stable-Baselines3, a popular library for reinforcement learning algorithms. This implementation ensures stability and efficiency during training. Below is a table summarizing the hyperparameters used for training our PPO model. The values not mentioned are set to the default values from Stable-Baselines3.

Table 2: Hyperparameters of PPO training for Dark Room

Hyperparameter	Value
Learning Rate	0.0001
Policy Network Architecture	MlpPolicy
Training Steps	1000000
Save Frequency	50000
N Steps	500

Table 3: Hyperparameters of PPO training for Minigrid Pre-train

Hyperparameter	Value
Learning Rate	0.0002
Number of Epochs	20
Policy Network Architecture	CustomCNN
Training Steps	2e7
Save Frequency	64000
N Steps	1600

Table 4: Hyperparameters of PPO training for Minigrid Fine-tune

Hyperparameter	Value
Learning Rate	2e-5
Policy Network Architecture	CustomCNN
Training Steps	400000
Save Frequency	16000
N Steps	1600

CustomCNN is a customized 2-layer convolutional neural network (CNN) feature extractor designed to process image inputs.

For the BlockedUnlockPickup task, due to its extreme difficulty, the model was unable to learn effectively from scratch. To address this, we used a pre-trained model from the UnlockPickup task and continued pretraining it on the BlockedUnlockPickup task.

Now, regarding the data collection process, for both Dark Room and Minigrid, we selected a series of models and arranged them to reflect their training progression: the earlier models are those that showed continuous performance improvement, while the later ones are those that had reached convergence. This arrangement was made with the intention that the ICRL policy learned from these models would also achieve stability after performance convergence. For each PPO process, we selected 40 models to sample data from a horizon of 400. In Minigrid, PPO checkpoints are saved every 64,000 environment steps during pretraining and every 16,000 steps during fine-tuning. For Dark Room, this process resulted in a total of 40,000 trajectories. For each task in Minigrid, we collected a total of 170,000 trajectories.

D.3 Details of Implementations

AD (Algorithm Distillation): AD [16] is trained on continuously improving trajectories generated from the agent's experience. In this framework, the model learns to predict the next action based on a history of states, actions, and rewards, effectively transforming reinforcement learning into a supervised learning problem.

AD$^\epsilon$: AD$^\epsilon$ [32] builds upon AD by replacing real online trajectories with simulated trajectories. These simulated trajectories are generated by sampling from a noised model, with the noise progressively reduced during training. This approach allows for greater flexibility in training the model by not relying strictly on real data. It also makes the model more robust to imperfections in the trajectory data.

IDT (In-Context Decision Transformer): IDT [13] extends the AD framework by reordering the context according to episode rewards and introducing a hierarchical decision-making structure. This hierarchical approach enables the model to handle longer horizons and more complex tasks. Specifically, IDT organizes the contexts into different decision levels, with the high-level model focusing on broader task goals, while the decision model focuses on more immediate decisions. This method allows for better handling of tasks with long temporal dependencies.

We implement all baselines as well as our method on a GPT-2 [21] based backbone, ensuring comparable parameter scales and closely matched architectural hyperparameters.

Due to the context length limitation of our GPT-2 model, we set the context length to 400, while GPT-2 has a maximum horizon of 1024. This constraint leads to a situation where each position in the Transformer corresponds to a time step's context, rather than having continuous three or four positions in the Minigrid environment (with a 7x7x3 observation) corresponding to one timestep's context. For this, we use 2 convolutional layers followed by 1 MLP layer, embedding the observations into 64-dimensional vectors. Actions are one-hot encoded into a 7-dimensional vector, while rewards and Context Values are represented as single-dimensional values.

For AD$^\epsilon$, the model sequence length is 40. For each position i in the model sequence, we set ϵ as a function: $\epsilon_i = \min\left(\frac{i}{30}, 1\right)$, which ensures consistency with AD and other algorithms in the Minigrid setting. The first 30 models correspond to trajectories with continuously improving performance, while the remaining 10 represent near-stable (optimal) models.

For IDT, we implement the hierarchical structure as outlined in the original paper. The structure includes three models: a decision model, a high-level decision model, and a reviewing decisions model. The high-level decision model's context timestep interval is set to 5, and both the decision model and high-level decision model share the same architecture and hyperparameters. The reviewing decisions model is a 2-layer MLP. The embedding size for the high-level decision model is set to 64.

For CV-ICRL, in Dark Room, we estimate the context-value target using the normalized average episode reward of the source policy. We consider the maximum average episode reward (AER) from the model sequence as 1. In Minigrid, we use the average episode reward of the source policy as the value-token target, as the max AER for these tasks is set to 1. For Minigrid, each of the 40 selected source checkpoints is evaluated in 20 environments for 500 steps, pooling completed episodes to estimate its mean return. This requires 10,000 transitions per checkpoint and 400,000 transitions per task type.

Checkpoint-level Monte Carlo evaluation estimates source-policy performance without an additional learned value estimator; its accuracy depends on return stochasticity and evaluation coverage. GAE and TD(λ) require state-value estimates, while Retrace additionally requires Q/value estimates and behavior/target policy information. Retrofitting an AD dataset requires the source checkpoints and their trajectory mapping. If checkpoint-return estimates were not retained, the checkpoint evaluations must be performed.

D.4 Additional DPT Baseline

Decision-Pretrained Transformer (DPT) [17] is a representative ICRL method based on supervised pretraining and posterior-sampling-style adaptation. To broaden the comparison beyond the AD-like baselines in the main text, we evaluate DPT on our Minigrid benchmark and report both success rate and Average Episode Return (AER) in Tables 5 and 6.

DPT achieves non-trivial performance on some navigation tasks but is less competitive on tasks requiring more structured sequential behavior or cross-task-type generalization. These results provide an additional reference point for ICRL methods beyond the main AD-like comparison.

Table 5: DPT comparison on Minigrid task types used in the main experiments. Each entry reports success rate / AER.

Environment	AD	CV-ICRL	DPT
LavaCrossingS9N2	100.00% / 0.9413	100.00% / 0.9445	85.00% / 0.8114
LavaCrossingS9N3	100.00% / 0.9319	100.00% / 0.9241	75.00% / 0.6688
SimpleCrossingS11N5	100.00% / 0.8746	100.00% / 0.9051	0.00% / 0.0000
SimpleCrossingS9N3	100.00% / 0.9308	100.00% / 0.9464	5.00% / 0.0034

Table 6: DPT comparison on unseen Minigrid task types for cross-task-type generalization. Each entry reports success rate / AER.

Environment	AD	CV-ICRL	DPT
DoorKey-6x6	90.00% / 0.4473	100.00% / 0.7592	0.00% / 0.0000
DoorKey-8x8	100.00% / 0.4381	100.00% / 0.6302	0.00% / 0.0000
FourRooms	75.00% / 0.1076	65.00% / 0.1898	25.00% / 0.2292
LavaCrossingS11N5	100.00% / 0.6373	100.00% / 0.8092	0.00% / 0.0000

D.5 Additional Continuous-Control Results on MetaWorld

To examine whether CV-ICRL also applies beyond discrete-control domains, we conduct a small-scale evaluation on two MetaWorld ML1 tasks. MetaWorld provides continuous-control manipulation environments, which differ substantially from Dark Room, Minigrid, and Procgen. As shown in Table 7, CV-ICRL outperforms AD on both tasks in terms of Average Episode Return (AER).

Table 7: Additional MetaWorld continuous-control results. We report Average Episode Return (AER); higher is better.

Method	ML1-reach-v3	ML1-push-v3
CV-ICRL	**426.0**	**55.2**
AD	253.2	18.2

D.6 Analysis of Confidence Bands

The confidence bands in the main figures aggregate performance across unseen tasks. A wider band can therefore reflect larger cross-task dispersion rather than unstable adaptation within each task. This distinction is important for CV-ICRL: the method often improves more on easier tasks while remaining comparable to AD on harder tasks, which can increase the standard deviation of task-level means.

To separate cross-task heterogeneity from within-task temporal instability, we compute statistics over the late stage of Dark Room evaluation (steps 3k–4k). As shown in Table 8, CV-ICRL has a larger standard deviation across task means, but substantially lower within-task temporal standard deviation. This supports our interpretation that wider aggregate bands in some plots come from task difficulty heterogeneity rather than unstable per-task adaptation.

D.7 Additional Baseline Comparisons

Table 9 reports the full comparison. We adapt Reinformer and IC-CQL to the retained cross-episode Minigrid setting. Reinformer represents max-return sequence modeling, while IC-CQL uses history-conditioned twin Q-functions. Both baselines use fixed checkpoint rules across the four task types, without task-specific selection. IC-CQL selects actions greedily from Q-values, whereas the sequence policies use categorical sampling. We report results over 20 unseen tasks per task type and an 8,000-step interaction budget.

Both adaptations use the original Minigrid trajectory corpus, the Minigrid CNN encoder, and a causal Transformer with four layers, four attention heads, embedding dimension 256, dropout 0, and a

Table 8: Decomposition of Dark Room variability in the late evaluation stage (steps 3k–4k). "Std across tasks" measures dispersion of task-level mean returns, while within-task std measures temporal fluctuation within each task.

Algorithm	Mean task reward	Std across tasks	Mean within-task std	Median within-task std
AD	24.56	23.04	7.13	6.64
CV-ICRL	**31.13**	**36.33**	**3.95**	**2.96**

400-step context window. Each model is trained for 15 epochs with effective batch size 32 and training seed 0. Reinformer uses AdamW with learning rate 10^{-4}, weight decay 10^{-4}, 5,000 warmup updates, gradient clipping at 0.25, expectile $\tau = 0.99$, and return-loss weight 1. Its episode-local RTG targets use $\gamma = 1$; incomplete episode suffixes are masked. A fixed training-corpus RTG scale is used throughout training and validation. The predicted RTG and the same causal hidden state feed a linear action head, requiring one Transformer pass per action. Its 1,799-parameter action head matches CV-ICRL's action head.

IC-CQL uses twin Q-functions with the TD target $r_t + \gamma(1 - d_t) \min_{j\in\{1,2\}} \max_a Q_j^{\text{target}}(h_{t+1}, a)$, where h_{t+1} is the history-conditioned representation and d_t indicates termination. Its conservative term is the sum of the two action cross-entropies with label smoothing 0.3 and weight 0.01. Optimization uses Adam with learning rate 3×10^{-4}, discount $\gamma = 0.9$, target-update coefficient 0.005, and gradient clipping at 1. The reported checkpoints are fixed at epoch 10 for Reinformer and epoch 5 for IC-CQL on every task type. IC-CQL acts using $\arg\max_a Q_1(h_t, a)$. Both added baselines use evaluator seeds 20000–20019, whereas the CV-ICRL reference uses seeds 0–19; comparisons are therefore not seed-paired. Reported means weight the 20 tasks equally, and standard deviations are population standard deviations across tasks.

Table 9: Additional baseline results over 20 unseen tasks per task type. AER and LER are higher-is-better; IF (%) is lower-is-better. Entries show mean ± population standard deviation. Best mean values are **bolded**.

Task type	Metric	CV-ICRL	IC-CQL	Reinformer
SimpleCrossingS9N3	AER	$\mathbf{0.945 \pm 0.015}$	0.883 ± 0.014	0.886 ± 0.020
	LER	$\mathbf{0.949 \pm 0.012}$	0.860 ± 0.287	0.801 ± 0.327
	IF (%)	$\mathbf{1.561 \pm 1.880}$	8.575 ± 1.381	10.735 ± 2.834
SimpleCrossingS11N5	AER	$\mathbf{0.902 \pm 0.076}$	0.303 ± 0.192	0.864 ± 0.225
	LER	$\mathbf{0.921 \pm 0.066}$	0.048 ± 0.210	0.895 ± 0.208
	IF (%)	13.209 ± 13.108	68.524 ± 17.792	$\mathbf{11.711 \pm 22.245}$
LavaCrossingS9N3	AER	$\mathbf{0.921 \pm 0.051}$	0.616 ± 0.420	0.094 ± 0.160
	LER	$\mathbf{0.939 \pm 0.026}$	0.572 ± 0.467	0.048 ± 0.208
	IF (%)	$\mathbf{5.228 \pm 8.787}$	35.376 ± 43.752	88.921 ± 16.946
BlockedUnlockPickup	AER	$\mathbf{0.955 \pm 0.010}$	0.195 ± 0.262	0.942 ± 0.021
	LER	$\mathbf{0.961 \pm 0.005}$	0.048 ± 0.209	0.934 ± 0.046
	IF (%)	$\mathbf{1.825 \pm 2.732}$	76.754 ± 25.521	7.078 ± 6.987

D.8 Alternative Value Supervision

To examine state-dependent supervision, we use the critic associated with the source PPO checkpoint:

$$y_{\text{PPO}}(s_t, C_t) = V^{\pi_{k(C_t)}}(s_t).$$

Here, $V^{\pi_{k(C_t)}}$ denotes the source critic estimate. The context identifies the teacher checkpoint, and the critic takes the state as input, making target construction context-indexed and state-dependent.

As shown in Table 10, PPO-state supervision has higher AER on both SimpleCrossing task types, while checkpoint-return supervision has higher AER on the other four. Neither target is uniformly superior.

Evaluation uses 20 unseen tasks per task type, 8,000 interaction steps per task, categorical action sampling, and context retention across episode resets. These evaluations were run on single RTX 3090 or RTX 4090 GPUs (24 GB). AER is computed by averaging episode returns within each task and then weighting the 20 tasks equally.

Table 10: Context-indexed PPO-state supervision. AER is reported as mean $\pm$ population standard deviation over 20 unseen tasks. The context selects the source PPO checkpoint, and its critic provides a state-dependent target.

Task type	CV-ICRL	PPO-state target
SimpleCrossingS9N3	0.945 ± 0.015	0.949 ± 0.001
SimpleCrossingS11N5	0.902 ± 0.076	0.916 ± 0.067
BlockedUnlockPickup	0.955 ± 0.010	0.922 ± 0.041
Unlock	0.966 ± 0.008	0.948 ± 0.015
LavaCrossingS9N3	0.921 ± 0.051	0.914 ± 0.011
LavaCrossingS9N2	0.943 ± 0.015	0.933 ± 0.052

D.9 Deployment Value-Token Interventions

We intervene on value-token feedback during deployment of the same pretrained CV-ICRL policy. For each evaluation stream, let $\hat{v}_t$ be the raw critic prediction. The Minigrid evaluator uses the running-maximum write-back rule

$$m_{-1} = 0, \qquad m_t = \max(m_{t-1}, \hat{v}_t).$$

The normal condition writes m_t into subsequent context; zero writes 0, and frozen holds all feedback tokens at the stream's first write-back value $m_0 = \max(0, \hat{v}_0)$. Each condition runs for 8,000 interaction steps across multiple episodes, retaining context and write-back state across episode resets.

The AER drops under zero and frozen feedback in Table 11 support behavioral reliance on token content and continued write-back. These experiments characterize the tested deployment behavior; they do not measure value calibration or robustness to longer per-episode horizons.

Table 11: Value-token interventions over 8,000 interaction steps across multiple episodes. AER is reported as mean $\pm$ population standard deviation across 20 evaluator streams. These results test behavioral reliance on token content and continued write-back.

Condition	SimpleCrossingS9N3	BlockedUnlockPickup
Normal	0.942 ± 0.020	0.936 ± 0.027
Zero token	0.090 ± 0.152	0.097 ± 0.124
Frozen token	0.286 ± 0.296	0.241 ± 0.219

D.10 Effect of Context Length

Table 12 compares the $H = 400$ main-table reference with a separately trained $H = 200$ model, using equally weighted AER across 20 evaluation tasks. The shorter-context model has lower AER on both LavaCrossingS9N3 and SimpleCrossingS9N3, with a larger difference on SimpleCrossingS9N3.

Context is retained across episode resets as a FIFO window. Defining C_t as the visible history preserves the conditional-value formulation, while information discarded outside the window is unavailable for later adaptation.

D.11 Compute Resources

The original experiments are run on a single NVIDIA RTX 3090 GPU (24 GB VRAM). Training proceeds in two stages: *Stage 1* runs PPO on each task type to produce a curriculum of policy checkpoints and the corresponding trajectory dataset; *Stage 2* supervised-pretrains each ICRL method (AD, AD$^\epsilon$, IDT, and CV-ICRL) on the resulting dataset. Approximate per-run wall-clock times are

Table 12: Context-length comparison over 20 unseen tasks per task type. The $H = 400$ column uses the main-table reference; $H = 200$ uses a separately trained model. AER is reported as the equally weighted task mean $\pm$ population standard deviation.

Task type	$H = 400$ (main-table reference)	$H = 200$
LavaCrossingS9N3	0.921 ± 0.051	0.832 ± 0.282
SimpleCrossingS9N3	0.945 ± 0.015	0.037 ± 0.053

Table 13: Approximate per-run compute for the original experiments on a single RTX 3090 (24 GB). Stage 1 is PPO pretraining and dataset generation; Stage 2 is supervised pretraining of an ICRL method. "per task type" multiplies over the number of task types in the environment; "per method" multiplies over AD, AD$^\epsilon$, IDT, and CV-ICRL. Added baseline, value-supervision, and deployment-intervention experiments are excluded.

Environment	Stage 1 (PPO)	Stage 2 (ICRL)
Dark Room	4 h (total)	4 h (total)
Minigrid (6 task types)	24 h per task type	8 h per (method, task type)
Procgen (3 task types)	16 h per task type	10 h per (method, task type)

summarized in Table 13. The added value-supervision evaluations also used RTX 4090 GPUs, as documented with their experimental protocol.

For the original experiments across all three environments, the total training compute is approximately $8 + 6 \times 24 + 6 \times 4 \times 8 + 3 \times 16 + 3 \times 4 \times 10 = 512$ GPU-hours. This estimate excludes added baseline and value-supervision experiments, deployment interventions, hyperparameter search, and earlier prototyping.

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: The abstract and the Introduction state our main contributions: a policy-gradient-like interpretation of ICRL that relates instability to high-variance return-based credit signals; the CV-ICRL framework, which predicts a context value and writes it back into the context to enable TD-style updates; and empirical gains in stability and returns on Dark Room, Minigrid, and Procgen, with additional out-of-distribution evidence. These claims are supported by the theoretical analysis in Sections 3 and 4 and the experiments in Section 5.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: The Conclusion includes an explicit limitations paragraph discussing the scope of the policy-gradient interpretation, the current scope of cross-task-type and continuous-control evaluations, and possible gains from more refined context-value estimation.

   Guidelines:

   - The answer [N/A] 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: Property 4.1 in Section 4 states the assumptions for variance reduction using the true conditional value, with a complete proof in Appendix B. The Improved Performance Bound and the sufficient value-error condition in Corollary C.5 are stated and proved in Appendix C.

   Guidelines:

   - The answer [N/A] 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: Section 4 describes the supervised pretraining procedure and loss; Section 5 specifies the benchmarks, baselines, and evaluation protocol; Appendices D.1–D.3 provide environment details, dataset preparation, and per-method architectural and training hyperparameters. Appendices D.7–D.10 document the added baseline comparisons, alternative value supervision, deployment interventions, and context-length comparison. The code repository and training-data access instructions are linked in the abstract for further reproducibility.

   Guidelines:

   - The answer [N/A] 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: [Yes]

   Justification: The repository linked in the abstract (`https://github.com/Bluixe/CV-ICRL`) provides training, evaluation, and data-collection code, and download instructions for six Minigrid and three Procgen CV-ICRL training files.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 5 describes the benchmarks, AER and LER metrics, Instability Frequency, and test-task evaluation protocol; Appendix D.3 reports the backbone configuration, context length, and observation/action embeddings. Appendices D.7–D.10 specify the added experiments' training and evaluation settings, including fixed checkpoint rules, action selection, interaction budgets, token interventions, and context retention. The added baseline comparisons use different evaluator seed sets and are not seed-paired.

   Guidelines:

   - The answer [N/A] 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: The Minigrid main table and added appendix tables report equally weighted means $\pm$ population standard deviations across 20 unseen tasks (or evaluator streams for the token interventions). These standard deviations describe variation across evaluation tasks, not variation across independent training seeds. Main figures (e.g., Figure 4) report task-mean returns with shaded 95% confidence intervals, as stated in their captions; Appendix D.6 distinguishes cross-task dispersion from within-task temporal instability.

   Guidelines:

   - The answer [N/A] 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: Appendix D.11 provides GPU specifications, approximate training times, and an estimate of training compute. Additional experimental settings are described in the corresponding appendix sections.

   Guidelines:

   - The answer [N/A] 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: The research is purely methodological reinforcement learning conducted in simulated, open-source environments (Dark Room, Minigrid, Procgen); it does not involve human subjects, personal or sensitive data, or deployments with foreseeable harm, and conforms with the NeurIPS Code of Ethics.

Guidelines:

- The answer [N/A] 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: [Yes]

Justification: The Impact Statement section discusses the scope and societal implications of the work; as foundational research on test-time adaptation in simulated RL, the work has limited direct path to negative applications, which is noted there.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The released code and training histories concern adaptation in simulated grid-world and procedural environments, which do not pose a high risk of misuse.

Guidelines:

- The answer [N/A] 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: All third-party assets used in the paper, including Minigrid [5] (Apache 2.0), the Procgen benchmark, the GPT-2 architecture [21] (MIT), and the PPO algorithm [23], are cited in the corresponding sections, and we use them in accordance with their licenses.

    Guidelines:

    - The answer [N/A] 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: The code repository linked in the abstract (`https://github.com/Bluixe/CV-ICRL`) includes a README documenting the environment setup, training scripts, and evaluation procedure for the main experiments. The Minigrid and Procgen CV-ICRL training histories are published at `https://huggingface.co/datasets/bluixe/CV-ICRL`, with the fixed release tag `camera-ready-v1`, under CC BY 4.0. The dataset card and manifest document the data formats, labels, and release scope, and a versioned downloader is provided. Protocols for the added experiments are described in Appendices D.7–D.10.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects; all experiments are conducted in simulated environments.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not involve human subjects; no IRB approval is required.

    Guidelines:

    - The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were used only for writing and editing assistance.

    Guidelines:

    - The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.