FIRST-EXPLORE PLUS: SAMPLE-EFFICIENT TRAINING OF IN-CONTEXT EXPLORATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Exploration is essential for learning, yet common strategies—such as random action selection, generic exploration heuristics, or priors acquired through imitation learning—often require orders of magnitude more interactions than humans to systematically explore the same space. In-context meta-reinforcement learning can learn to explore adaptively and as efficiently as a human, but can still fail entirely in environments with deceptive rewards. First-Explore addresses this by partitioning a policy into exploration and exploitation modes with separate objectives, but its original instantiation is highly sample-inefficient, restricting experiments to simple environments and obscuring phenomena that emerge only at greater scale. We introduce First-Explore Plus (First-Explore+), which rebuilds the framework on a standardized Proximal Policy Optimization (PPO) backbone. Unlike the original First-Explore, First-Explore+ uses context to decide when to stop exploring, allowing it to achieve higher meta-RL returns in environments where the amount of exploration required varies across tasks. First-Explore+ matches or exceeds the original method's performance on challenging meta-learning benchmarks with deceptive rewards while using orders of magnitude fewer training samples. This increased efficiency also allows us to evaluate the First-Explore framework in a substantially more complex environment, Deceptive XLand-MiniGrid, where First-Explore+ achieves high scores while RL^2 and E-RL^2 completely fail to learn. Intriguingly, First-Explore+ also achieves substantial return in the first episode of each trial, before receiving any cross-episode context, and outperforms standard PPO in a single-episode evaluation despite PPO being trained directly to maximize single-episode return. These results suggest a broader role for meta-RL: training for cross-episode exploration can produce policies that achieve higher returns even when evaluated without cross-episode context.

1 INTRODUCTION

In complex environments, learning high-performing behavior often requires substantial exploration. Most reinforcement-learning (RL) methods explore with generic heuristics such as ϵ-greedy sampling (Sutton & Barto, 1998; Mnih et al., 2015; Osband et al., 2016), entropy regularization (Schulman et al., 2015a; Haarnoja et al., 2018), or novelty-based objectives (Sukhbaatar et al., 2017; Burda et al., 2018; Conti et al., 2018; Henaff et al., 2022). These methods adapt slowly to what has already been tried (e.g., through an incremental parameter update after a large batch of episodes). As a result, they cannot reliably switch between behaviors after a single episode. They thereby often require far more interactions to learn than a human might.

In-context meta-reinforcement learning (meta-RL; Duan et al., 2016) offers an alternative: a single policy can rapidly adapt its behavior to different environments—including environments not seen during training (Bauer et al., 2023)—via in-context adaptation. Yet when exploration is costly, meta-RL can learn to *avoid* exploring, particularly when exploration and exploitation require conflicting behavior (Norman & Clune, 2024). First-Explore sidesteps this conflict by training exploration and exploitation as separate modes of a policy, but is highly sample-inefficient, restricting it to relatively simple environments and potentially obscuring phenomena that emerge at greater scales.

We present First-Explore Plus (First-Explore+), which retains this separation while replacing First-Explore's custom training procedure with a standardized Proximal Policy Optimization (PPO, Schul-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

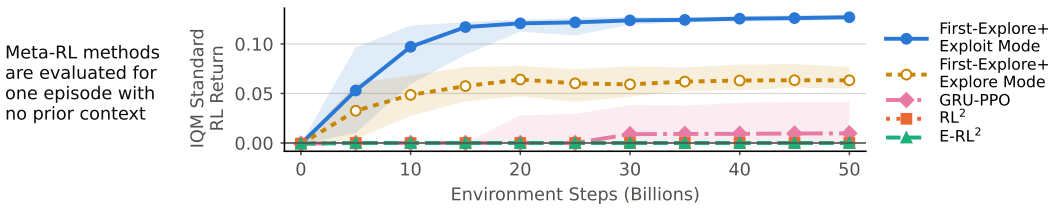

Figure 1: **On Deceptive XLand, First-Explore+ beats standard RL at its own game.** Single-episode return on new tasks with no prior context—the objective standard RL optimizes—plotted against environment steps. **First-Explore+ in exploit mode (•)** achieves far higher return than **GRU-PPO (♦)** with the same architecture and number of environment interactions; **RL² (■)** and **E-RL² (▲)** remain at zero, and even First-Explore+'s **explore mode (○)** outperforms **GRU-PPO**. IQM over 10 seeds with 95% bootstrap confidence intervals; greedy action selection, 4,096 tasks.

man et al., 2017) backbone. Unlike the original, First-Explore+ also uses context to decide when to switch between exploration and exploitation, enabling higher return in certain environments, see Section F. First-Explore+ also consistently outperforms the original while using orders of magnitude fewer samples, enabling us to train it in Deceptive XLand-MiniGrid, a substantially more complex environment in which First-Explore is impractical.

At this greater scale, we observe a new phenomenon: a learnt First-Explore+ policy outperforms an equivalent standard-RL policy even when both are evaluated in the same standard-RL setting, Figure 1. First-Explore+ in both **explore (yellow ○)** and **exploit (blue •)** mode achieves substantial return in the first episode of each trial, before receiving any cross-episode context, whereas single-episode **GRU-PPO (pink ♦)** with the same training budget and model architecture performs poorly.

This result suggests that when rewards are deceptive, in-context meta-RL may learn better single-episode policies than standard RL methods.

More broadly, First-Explore+ provides a path toward agents that can rapidly master new tasks through in-context learning and adaptive exploration. We release open-source JAX (Bradbury et al., 2018) and PyTorch/CleanRL (Paszke et al., 2019; Huang et al., 2022) implementations of First-Explore+, E-RL², and RL².

2 Background

2.1 In-Context Meta-Reinforcement Learning

In-context meta-reinforcement learning trains agents to adapt within a trial, which consists of several episodes, to a task drawn from a distribution of partially observable Markov decision processes (POMDPs), $p(\mathcal{M})$. For each trial, the agent interacts with a sampled environment instance $\mathcal{M}_i = (\mathcal{S}, \mathcal{A}, \mathcal{O}, P_i, R_i, \Omega_i)$. Here, $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, and $\mathcal{O}$ is the observation space. For environment instance $\mathcal{M}_i$, $P_i(s' \mid s, a)$ is the transition function specifying the probability of transitioning from state $s \in \mathcal{S}$ to state $s' \in \mathcal{S}$ after taking action $a \in \mathcal{A}$; $R_i(r \mid s, a)$ is the reward function specifying the distribution over rewards $r \in \mathbb{R}$; and $\Omega_i(o \mid s)$ is the observation function specifying the probability of receiving observation $o \in \mathcal{O}$ in state s. A trial consists of multiple episodes from the same sampled environment instance $\mathcal{M}_i$: the environment state is reset at the start of each episode, but the underlying transition, reward, and observation functions remain fixed throughout the trial. The objective is to maximize cumulative reward across the entire trial (meaning all episodes within the trial), $G_{\text{trial}} = \sum_{t=0}^{H-1} r_t$, where r_t is the reward received at timestep t and H is the total number of timesteps in the trial.

The agent's policy is implemented using a sequence model. At each timestep, the policy receives the current observation along with the previous action and reward, allowing the sequence model to remember this information for adaptation throughout a trial.

In the environments studied in this paper, each trial has an initially unknown reward structure. To achieve high cumulative trial return, agents must identify the reward structure during earlier episodes and use this experience to obtain more reward in later episodes of the same trial.

2.2 In-Context Meta-RL Methods

The canonical in-context meta-RL method is RL^2 (Duan et al., 2016; Wang et al., 2016). RL^2 applies standard RL algorithms, such as PPO, to optimize the total return of a memory-based policy over an entire trial, treating the trial as a single long episode. In principle, this objective allows RL^2 to learn how to balance exploration and exploitation across episodes to maximize trial return. However, when exploration that enables higher trial return comes at the cost of immediate episode return, RL^2 can fail to learn sufficient exploration and perform poorly (Stadie et al., 2018; Norman & Clune, 2024). E-RL^2 masks the returns of the first k episodes in the training objective, where k is a crucial hyperparameter of the method. By masking rewards, E-RL^2 does not penalize exploration, and so often sacrifices the possibility of maximizing cumulative trial return. Despite this, it can achieve significantly higher returns in certain environments.

First-Explore (Norman & Clune, 2024) similarly enables costly exploration using a dual-mode policy: an explore mode with masked rewards and an exploit mode that maximizes episodic return. The explore mode is trained to maximize the exploit-mode return in the *next* episode, encouraging it to collect information that enables strong exploit-mode performance. Training alternates between explore-mode and exploit-mode episodes, which we refer to as explore-exploit pairs. First-Explore has several advantages over RL^2 and E-RL^2. It has been shown to perform well in toy deceptive-reward environments where RL^2 fails to learn. Unlike E-RL^2, which requires the number of exploration episodes k to be selected before training via a costly hyperparameter sweep, First-Explore selects k after training by evaluating all values in a linear sweep.

However, First-Explore is highly sample-inefficient because it relies on full Monte Carlo return estimates rather than value-function bootstrapping. It also uses a fixed exploration budget of k episodes before switching to exploit mode, preventing it from adapting the amount of exploration to each environment instance. First-Explore+ addresses these limitations. It learns when to switch to exploit mode based on the current context, allowing it to adapt its exploration behavior to each problem. Implemented as an extension of PPO, it combines First-Explore's ability to handle deceptive rewards with the sample efficiency of modern RL methods.

3 Method

3.1 First-Explore+

First-Explore+ replaces First-Explore's Monte Carlo gradient estimator with a bootstrapped PPO update (Schulman et al., 2017) computed jointly over explore-mode and exploit-mode. During optimization, we treat each explore–exploit pair as a single episode by masking the termination signal at the end of exploration. This allows the critic and Generalized Advantage Estimation (GAE) (Schulman et al., 2015b) to propagate value across the explore–exploit boundary, as illustrated in Figure 2. We also mask exploration rewards from the PPO objective, so explore-mode actions are trained only according to their effect on subsequent exploit-mode returns. The agent still observes these rewards, however, and can use them to inform later actions. This preserves First-Explore's core objective while replacing high-variance Monte Carlo return estimates with bootstrapped estimates from a learned critic.

A single policy and value network is shared across exploration and exploitation modes and trained with the standard PPO objective (Schulman et al., 2017). We append a binary mode flag to each observation, allowing the agent to condition its behavior on the mode. Observations also include the previous one-hot action and reward, giving the policy and value function access to the immediately preceding transition. Our main experiments use recurrent architectures, which represent longer-term context through hidden states. We include experiments with transformer architectures in Section I.

Consistent with First-Explore, each episode begins from the agent context at the end of the preceding exploration episode. Thus, information gathered during exploration persists into subsequent episodes, while exploitation mode context is discarded at the end of an episode. After K explore–exploit pairs, a new task instance is sampled and the agent's context is reset, completing one trial.

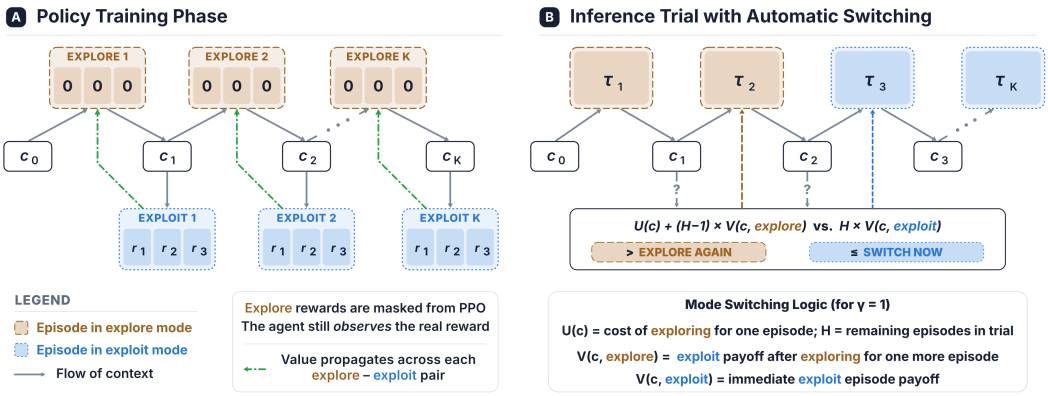

Figure 2: Full diagram of the First-Explore+ algorithm.

3.2 Exploit Context Injection

First-Explore trains exploit mode only on context generated by exploration mode, so it performs poorly when adapting from its own experience. Norman & Clune (2024) address this by freezing context after k exploration episodes at inference time, preventing subsequent exploitation experience from contributing to adaptation. First-Explore+ instead introduces *exploit context injection*: with probability ρ, the policy switches to exploit mode while generating context during training. These episodes otherwise use the standard exploration-mode training procedure, including reward masking and value propagation. This exposes the exploit policy to self-generated experience, allowing it to adapt from exploitation trajectories at inference time.

3.3 Critic-based Mode Switching and Auxiliary U Critic

The original First-Explore uses a fixed number of exploration episodes and cannot dynamically choose how much to explore based on the task. This may limit performance (Section F) and requires k to be tuned after training. First-Explore+ instead learns to compare the value of additional exploration against immediate exploitation, allowing it to decide when to switch to exploit-mode at inference time. Determining whether further exploration is worthwhile requires estimating its cost (i.e. expected return), which is—by design—masked from the value function in explore-mode. To address this, we introduce an auxiliary critic U that estimates the unmasked return of an exploration episode:

$$U(c) = \mathbb{E}(R(\tau) \mid \tau \sim \pi_{\text{explore mode}})$$

U is not used to update the policy or main critic, although it shares parameters with the policy so learning to predict U may affect representation learning. U is evaluated in combination with the mode-conditioned value function V, to estimate the value of additional exploration. In explore mode, $V(c_i, \text{explore})$ estimates the value of an exploration episode given the current context. Importantly, due to reward masking in explore-mode, this is equivalent to the value of the subsequent exploit episode, $V(c_{i+1}, \text{exploit})$. Together, V and U provide the quantities needed to approximate a simple optimal-stopping rule. Eliding discounting—detailed in Section E—given a context c and h remaining episodes, exploration should stop when the expected return from exploiting for all remaining episodes—$V(c, \text{exploit}) \times h$—exceeds that from taking one additional exploration episode—$U(c)$—and then exploiting thereafter—$V(c, \text{explore}) \times (h-1)$:

$$\text{stop exploring iff } V(c, \text{exploit}) \times h \geq V(c, \text{explore}) \times (h-1) + U(c)$$

We apply this rule in our experiments and find that it consistently outperforms the fixed k estimated by First-Explore's sweep procedure in Bandit, Dark Treasure Room and Ray Maze ($p < 0.002$), and matches in XLand-MiniGrid. We also find evidence that learning U itself may aid learning, even without using U at inference time (see Section E).

4 EXPERIMENTS

In this section, we compare First-Explore+ with RL^2, E-RL^2, and the original First-Explore on the three deceptive-reward meta-RL environments introduced by Norman & Clune (2024)—Bandits with One Fixed Arm, Dark Treasure Room, and Ray Maze—and on a new deceptive variant of XLand-MiniGrid (Nikulin et al., 2024). First, we show that FE+ matches or outperforms every baseline on these environments while improving the sample efficiency of FE by orders of magnitude (Section 4.1). Second, we examine the behaviors of our trained agents in RayMaze, and find that FE+ agents learn to intentionally seek out environment information, while baselines employ simpler, less effective strategies (Section 4.2). Finally, we demonstrate that training agents with First-Explore+ improves context-free performance, suggesting that in-context meta-learning may provide benefits to broader non-meta-RL settings (Section 4.3).

We train FE+, RL^2, and E-RL^2 with a GRU RNN architecture (Chung et al., 2014; Cho et al., 2014), while First-Explore is tested with the custom transformer model from the original code base. For deceptive XLand-MiniGrid experiments, we instead use the custom GRU model introduced by Nikulin et al. (2024). Results using a Gated-TransformerXL (Parisotto et al., 2020) architecture for FE+ and RL^2 can be found in Section I (Table 7). We do not evaluate the original First-Explore on deceptive XLand-MiniGrid because it is cost prohibitive to train for 50B steps, and because the original code does not support variable length episodes or GRU training.

The deceptive variant of XLand-MiniGrid introduces an exploration cost to the original environment. In XLand-MiniGrid an agent picks up and combines objects to discover "production rules" and create a goal object. We introduce a dead-end rule that consumes one of the necessary ingredients for the goal object. Triggering this rule renders the level unsolvable, and incurs a penalty of -0.1. Agents can achieve higher trial return by discovering this dead-end rule in early episodes and avoiding it throughout the trial. Full details of the environments are provided in Appendix B.

For each method, we perform a random hyperparameter search and train the final models using ten independent random seeds. Additional training details are explained in Section H. For the original First-Explore, we sweep the number of exploration episodes k after training and select the value that maximizes mean return on the training distribution (Section G). This value is then held constant for all subsequent evaluations. Each trained policy is evaluated on 4,096 independently sampled environments for 1 trial each. We report the interquartile mean (IQM) cumulative trial return across the 10 seeds, with 95% bootstrap confidence intervals over seeds.

4.1 MAIN RESULTS

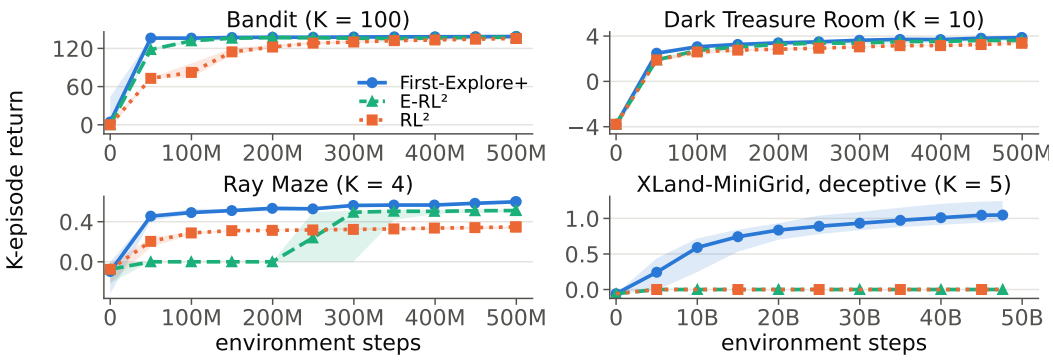

Figure 3: Training curves for RL^2, E-RL^2, and FE+ on all four environments. At each checkpoint during training, we evaluate the agent's performance, with 95% bootstrap confidence intervals, on 4,096 tasks sampled independently of training.

Training curves for all baselines are shown in Figure 3, and Table 1 summarizes the main experimental results. **First-Explore+ achieves the highest score in all four environments**, reaching more than twice the score of the original First-Explore in Dark Treasure Room. In XLand-MiniGrid, the most complex environment, First-Explore+ achieves positive returns whereas RL^2 and E-RL^2 completely fail to learn.

Table 1: Cumulative trial return of each method with greedy action selection. First-Explore+ outperforms all other evaluated methods, and learns with an order of magnitude fewer steps than First-Explore. First-Explore is not evaluated on XLand due to its sample inefficiency and design incompatibilities, e.g., not handling variable-length episodes. We report the interquartile mean over 10 seeds and 95% bootstrap confidence intervals in brackets, evaluated on 4,096 tasks.

	500M environment steps (XLand: 50B)			5–10B steps
Environment	First-Explore+	E-RL^2	RL^2	Original First-Explore
Bandit	**138.7** [136.7, 138.8]	137.7 [137.4, 137.9]	135.4 [132.0, 136.9]	128.7 [128.1, 129.4]
Dark Treasure Room	**3.86** [3.79, 3.96]	3.65 [3.56, 3.79]	3.37 [3.25, 3.45]	2.45 [2.27, 2.58]
Ray Maze	**0.60** [0.58, 0.61]	0.50 [0.25, 0.50]	0.35 [0.33, 0.36]	0.46 [0.44, 0.48]
XLand-MiniGrid	**1.25** [1.10, 1.43]	0.00 [0.00, 0.00]	0.00 [0.00, 0.00]	–

First-Explore+ significantly outperforms both RL^2 and E-RL^2 in every environment with both greedy and stochastic action selection (exact two-sided Mann–Whitney U tests, 10 seeds per method; Section J). In Ray Maze and XLand-MiniGrid, every First-Explore+ seed outperforms every RL^2 and E-RL^2 seed ($p = 1.1 \times 10^{-5}$). In Dark Treasure Room, every First-Explore+ seed outperforms every RL^2 seed ($p = 1.1 \times 10^{-5}$). Full details of the statistical analyses are provided in Section J.

We further compare the sample and time efficiency of First-Explore+ with the original First-Explore in Section D. Overall, First-Explore+ is **37–96× more sample-efficient** and **44–469× more time-efficient**, with the largest gains in the more complex Dark Treasure Room and Ray Maze environments. These efficiency gains make it practical to train First-Explore+ on XLand-MiniGrid, a substantially more challenging environment in which agents begin to make progress only after several billion environment steps.

4.2 Learnt Exploration Behavior Analysis

Ray Maze allows us to analyze cross-episode exploration behaviors because it has a simple reward structure with a clear exploration–exploitation tradeoff. The agent must navigate a procedurally generated maze using 11 lidar-like observations to reach goals. In each episode, the agent can achieve at most one goal. 30% of the goals give $+1$ reward, while 70% give -1, and the agent can only determine a goal's value by visiting it. Therefore, visiting goals has negative expected return for the current episode. In Ray Maze, optimal exploration and exploitation are easy to identify: before a positive goal is found, exploration requires visiting a new goal of unknown value; once a positive goal is found, exploitation requires returning to it. If only two of the four episodes remain and no positive goal has been found, no strategy improves cumulative trial return in expectation; the agent should therefore stop exploring and avoid touching any goal (see Figure 4, **C**).



Figure 4: Mean behavior across episodes of the Ray Maze trial. **A:** RL^2 (front), E-RL^2 (middle), and First-Explore (back); **B:** First-Explore+; and **C:** an optimal upper bound assuming all goals are reachable within the episode time limit. Each bar shows the mean proportion of outcomes: exploration corresponds to visiting previously unseen bad (**red crosshatch**) or good (**blue left diagonal**) goals, while exploitation corresponds to revisiting a previously identified good goal (**green right diagonal**). Among the trained methods, only First-Explore+ continues exploring in episode 2 (**red** and **blue**), enabling substantially more exploitation and higher returns in episodes 3 and 4 (**more green shading**).

As shown in Figure 4, **A**, RL^2, E-RL^2, and First-Explore explore only in the first episode, while the optimal behavior performs a variable number of explorations: if the goal found in the first episode was negative, the agent should explore once more—by visiting a new goal—but if the first goal was positive, it should exploit instead by revisiting the good goal. The original First-Explore cannot adapt this way because it fixes the number of explorations before observing the task, making higher returns unobtainable. Although RL^2 and E-RL^2 could in principle learn the optimal strategy, they fail to do so empirically. Only First-Explore+ learns to adjust its exploration across the trial and achieves a trial return above 0.5, the highest achievable return under the original First-Explore objective (Section F).

Next, we examine how exploration changes as goals are placed farther from the agent's starting location. RL^2 explores only locally (Figure 5, orange lines), rarely reaching goals more than two cells away. First-Explore learns to explore farther than RL^2 but fails to consistently reach goals at distance six, and only E-RL^2 and First-Explore+ reliably visit distant goals. Despite learning effective first-episode exploration, E-RL^2 and First-Explore underperform First-Explore+ in Ray Maze because they never explore in the second episode (Figure 4).

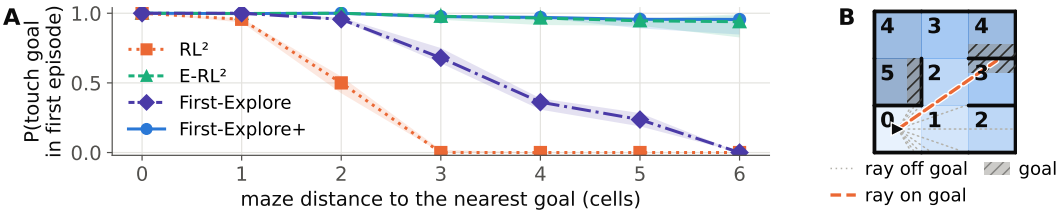

Figure 5: **A:** Mean goal-reaching success rate in the first episode of each trial, plotted against the distance from the agent's starting position to the nearest goal. **B:** A visualization of a procedurally generated maze. Maze cells are labeled and shaded according to their distance from the agent's starting location; goal regions are shaded grey; the agent's 11 lidar observations are shown as dotted lines; and the on-goal lidar observation is shown in orange. **RL^2 (orange ■)** learns to explore only far enough to reach nearby goals, rarely reaching goals at distance 3 or greater, such as the goal shown in subfigure B. **First-Explore (purple ♦)** explores significantly farther but fails to reach goals at distance 6. In contrast, **E-RL^2 (green ▲)** and **First-Explore+ (blue ●)** both reliably reach distant goals in the first episode.

These results highlight a potentially counterintuitive failure mode of deceptive returns and costly exploration: even when methods such as RL^2 and E-RL^2 learn to explore, they can remain locked out of deeper exploration. RL^2 does not learn to explore distant goals despite prolonged training (Figure 3, **orange ■ line low**), and both RL^2 and E-RL^2 fail to explore at all in the second episode (Figure 4, **A**: the episode 2 bars contain no **blue** or **red**). First-Explore+ learns to consistently reach distant goals and to adapt its exploration to the current task and context, enabling deeper exploration.

4.3 Meta Learning Exploration can Improve Context-Free Performance

In the three environments introduced by Norman & Clune (2024), exploration and exploitation directly conflict. Agents can achieve high cumulative trial returns only by accepting negative returns in early exploration episodes. Deceptive XLand-MiniGrid differs: exploration and exploitation are not fully opposed. Effective exploration can yield higher immediate episode return than avoiding exploration, despite the risk of triggering a dead-end rule and incurring a penalty.

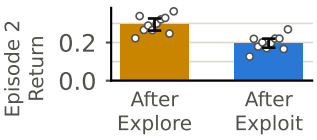

Figure 6: First-Explore+ exploit return after an initial **explore** or **exploit** episode. IQM, 10 seeds, 95% CIs.

However, ineffective exploration, such as random action, remains costly. As such, RL^2 and E-RL^2 learn to avoid the penalty rather than solve the task, yielding a return of 0.0 (Figure 3, bottom-right flat lines). In contrast, both First-Explore+ modes learn effective behavior: exploration increases subsequent exploit return (Figure 6, **orange** bar tops **exploit**; higher for all 10 seeds, paired Wilcoxon $p = 0.002$) and overall trial return (Figure 3, high **blue ● line**).

This advantage extends to the first episode of a trial, when no cross-episode context is available. Surprisingly, under identical training conditions[1], First-Explore+ significantly outperforms single-episode GRU-PPO (Figure 1, blue and yellow lines above pink, total rank separation; $p = 1.1 \times 10^{-5}$). GRU-PPO fails to learn in 7 of 10 independent runs, and in the remaining 3 runs, the *lowest-return* First-Explore+ agent doubles the *highest-return* PPO agent's score (0.113 vs. 0.056).

First-Explore+ receives no information from previous episodes during evaluation, so this advantage cannot result from cross-episode adaptation at inference time. Instead, *learning cross-episode exploration* produces an exploit-mode policy that explores more effectively within a single episode. These results provide preliminary evidence that meta-RL can serve not only as a mechanism for *inference-time* adaptation, but also as a training procedure for learning exploration strategies that improve context-free single-episode performance.

5 Discussion

In many settings, actions are costly, making exploration difficult to learn when its benefits are uncertain. Robots expend energy to act, businesses spend capital to generate revenue, and researchers invest time before knowing whether an idea will succeed. Importantly, in such settings agents cannot simply ignore costs; balancing rewards and penalties is integral to the task. In simpler domains, First-Explore does exactly this; it learns effective exploration even when exploration has a high cost.

In Section 4.3, we show that agents trained with First-Explore+ can outperform standard PPO even when both are evaluated in a single episode with no prior context, despite PPO being directly optimized for this setting. Other RL paradigms similarly exploit privileged information available only during training to improve the resulting policy, including centralized training with decentralized execution (Gronauer & Diepold, 2022) for multi-agent RL (MARL) and on-policy self-distillation (Zhao et al., 2026; Zhang et al., 2026) for RL with verifiable rewards (RLVR). Our results suggest that meta-RL can similarly serve as a form of privileged training: access to cross-episode context during training can induce better exploratory behavior even when that context is absent at inference. To our knowledge, this is the first clear evidence that in-context meta-RL can substantially improve zero-context, single-episode exploration.

Looking forward, scaling First-Explore+ across sufficiently diverse task distributions could yield a general-purpose foundation model for exploration – a policy that has learned reusable priors for efficiently gathering information in unfamiliar environments. Rather than relying on a hand-designed notion of novelty or uncertainty, such a model could transfer exploration knowledge across tasks and adapt its information-gathering strategy to the structure of a new problem. This could provide a strong exploratory prior for downstream reinforcement learning, substantially reducing the amount of task-specific interaction required to discover effective behavior in sparse-reward, deceptive-reward, or otherwise difficult environments.

6 Related Works

Our work focuses on black-box in-context meta-learning, in which agents adapt through context rather than updating parameters. Many non-meta-RL methods address exploration, including intrinsic motivation (Chentanez et al., 2004; Bellemare et al., 2016; Pathak et al., 2017; Burda et al., 2018; Henaff et al., 2022), curriculum learning (Portelas et al., 2020; Narvekar et al., 2020; Jiang et al., 2021), and search-based methods (Ecoffet et al., 2021). These approaches are complementary to in-context meta-RL and could be combined with the methods studied here. For instance, HyperX introduces a novelty bonus for the state and context of an in-context learner (Zintgraf et al., 2021). However, these approaches typically encode specific exploration biases, such as visual novelty (Pathak et al., 2017; Burda et al., 2018) or value-prediction error (Jiang et al., 2021; Parker-Holder et al., 2022). In contrast, in-context meta-RL aims to learn effective exploratory behavior from experience without prescribing an exploration heuristic.

Other branches of meta-learning can also learn exploration. Initialization-based meta-learning learns a policy initialization that can be adapted to a new task with few gradient updates (Finn et al.,

[1]Identical sweep and training environment interaction budget on identical architectures, except that First-Explore+ receives an additional mode flag—held constant during this evaluation.

2017; Gurumurthy et al., 2020). This requires task-specific gradient updates at test time, whereas in-context meta-learning adapts within a trial without parameter updates. Optimization-based meta-learning instead learns an update rule that generalizes across tasks (Xu et al., 2018; Lu et al., 2022; Oh et al., 2025). However, these methods generally optimize how the policy is updated rather than explicitly learning exploration, which is only indirectly shaped by the optimization objective.

Within black-box in-context meta-RL, several works learn separate exploration and exploitation policies. Some train an exploration policy by directly rewarding information gain for task identification (Fu et al., 2021; Zhang et al., 2021), which may learn to seek information that is not relevant to solving the task. Like First-Explore+, DREAM (Liu et al., 2021) instead uses exploitation returns to train exploratory behavior that gathers information useful for maximizing future return. DREAM first performs task identification and then trains a task-conditioned exploitation policy. This separation allows exploration and exploitation to be optimized independently, but requires each task to have a unique identifier, introduces additional complexity, and has not been shown to scale to many exploration episodes.

Finally, Bauer et al. (2023) demonstrate that RL^2-style meta-RL can scale to broad task distributions. They introduce the Adaptive Agent (AdA), which uses accumulated context to rapidly adapt within XLand 2.0, a procedurally generated environment. However, the reported results do not show that AdA deliberately gathers informative context when exploration and exploitation conflict. The authors explicitly note that, "knowing that there are 8 trials in total [instead of 1], a Bayes-optimal policy [might choose] to display a less rewarding and more exploratory behaviour in trial 1," but AdA does not exhibit this behavior.[2] In contrast, First-Explore directly targets this ability to sacrifice short-term reward for information: with U, First-Explore+ chooses when to explore based on the remaining episodes h in the trial. In Deceptive XLand-MiniGrid, with $h = 5$, it explores in the first episode of 99.7% of trials, whereas with $h = 1$ it exploits on 99.7% of tasks.

7 Limitations

Our experiments focus on deceptive-reward or high-exploration-cost environments and primarily compare First-Explore+ with what we call implicit-inference meta-RL methods, namely RL^2, E-RL^2, and First-Explore. We do not claim state-of-the-art exploration performance, but instead test three narrower claims: (1) First-Explore+ performs well in deceptive-reward environments where RL^2 and E-RL^2 fail, (2) it is substantially more sample efficient than the original First-Explore algorithm, and (3) it can train policies with better zero-context performance than standard RL methods. We evaluate sample efficiency on three relatively simple environments and zero-context performance only on XLand-MiniGrid, so the generality of these findings to other environments, training systems, and exploration methods remains open.

8 Conclusion

First-Explore+ separates exploration from exploitation, enabling effective exploration when information gathering is costly or immediate rewards are deceptive. We show that First-Explore+ outperforms standard in-context meta-RL baselines while dramatically improving the sample efficiency of the original First-Explore.

These efficiency gains allow First-Explore+ to learn in the substantially more complex Deceptive XLand-MiniGrid environment, where existing baselines fail to achieve positive returns. In this setting, First-Explore+ learns effective exploratory behavior and, strikingly, improves performance even in single-episode rollouts with no context from previous episodes. This suggests that training agents to explore across episodes can distill useful exploration strategies into the unconditioned policy, improving behavior even before any task-specific information has been gathered.

More broadly, First-Explore+ makes it more practical to study learned exploration at scale. We hope that further research and development of these methods will enable in-context meta-RL agents to learn increasingly general exploration priors that transfer to unfamiliar environments. Rather than prescribing a fixed exploration heuristic, such policies could provide learned, task-agnostic biases for information gathering that accelerate adaptation and policy learning in complex environments.

[2] AdA's "trials" and "episodes" terminology is flipped, corresponding to our episodes and trials, respectively.

AI USE STATEMENT

In this work, we use generative AI tools to assist with refactoring training code, to implement standard hyperparameter sweeps, to produce code for figures, and to check for writing errors. We have not used generative AI tools for any other tasks with required disclosure. No AI was used for tasks such as formulating or providing key ingredients for mathematical claims, proof writing, translation, and dataset formatting or generation. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies exploration in simulated meta-reinforcement-learning benchmarks (bandits, grid worlds, and mazes). It involves no human subjects, personal data, or deployed systems, and we do not foresee direct ethical concerns beyond those general to reinforcement learning research.

REPRODUCIBILITY STATEMENT

We release open-source JAX and PyTorch/CleanRL implementations of First-Explore+, the RL^2 and E-RL^2 baselines, the three benchmark environments of Norman & Clune (2024), and our deceptive XLand-MiniGrid variant. Hyperparameter search ranges and selected values for every method and environment are given in Section H (GRU) and Section I (Transformer-XL); the evaluation protocol is described in Section 4; statistical test details are in Section J; and compute requirements are listed in Section A and Table 2.

REFERENCES

Jakob Bauer, Kate Baumli, Satinder Baveja, Feryal Behbahani, Avishkar Bhoopchand, Nathalie Bradley-Schmieg, Michael Chang, Natalie Clay, Adrian Collister, et al. Human-timescale adaptation in an open-ended task space. *arXiv preprint arXiv:2301.07608*, 2023.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/afda332245e2af431fb7b672a68b659d-Paper.pdf.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Yash Katariya, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/jax-ml/jax.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

Nuttapong Chentanez, Andrew Barto, and Satinder Singh. Intrinsically motivated reinforcement learning. In L. Saul, Y. Weiss, and L. Bottou (eds.), *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004. URL https://proceedings.neurips.cc/paper_files/paper/2004/file/4be5a36cbaca8ab9d2066debfe4e65c1-Paper.pdf.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, eighth workshop on syntax, semantics and structure in statistical translation*, pp. 103–111, 2014.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Edoardo Conti, Vashisht Madhavan, Felipe Petroski Such, Joel Lehman, Kenneth Stanley, and Jeff Clune. Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. *Advances in neural information processing systems*, 31, 2018.

Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. First return, then explore. *Nature*, 590(7847):580–586, February 2021. ISSN 1476-4687. doi: 10.1038/s41586-020-03157-9. URL https://doi.org/10.1038/s41586-020-03157-9.

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

Haotian Fu, Hongyao Tang, Jianye Hao, Chen Chen, Xidong Feng, Dong Li, and Wulong Liu. Towards effective context for meta-reinforcement learning: an approach based on contrastive learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7457–7465, 2021.

Sven Gronauer and Klaus Diepold. Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review*, 55(2):895–943, February 2022. ISSN 1573-7462. doi: 10.1007/s10462-021-09996-w. URL https://doi.org/10.1007/s10462-021-09996-w.

Swaminathan Gurumurthy, Sumit Kumar, and Katia Sycara. Mame: Model-agnostic meta-exploration. In *Conference on Robot Learning*, pp. 910–922. PMLR, 2020.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. Pmlr, 2018.

Mikael Henaff, Roberta Raileanu, Minqi Jiang, and Tim Rocktäschel. Exploration via elliptical episodic bonuses. *Advances in Neural Information Processing Systems*, 35:37631–37646, 2022.

Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, Jeff Braga, Dipam Chakraborty, Kinal Mehta, and João G.M. Araújo. Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18, 2022. URL http://jmlr.org/papers/v23/21-1342.html.

Minqi Jiang, Edward Grefenstette, and Tim Rocktäschel. Prioritized level replay. In *International Conference on Machine Learning*, pp. 4940–4950. PMLR, 2021.

Evan Z Liu, Aditi Raghunathan, Percy Liang, and Chelsea Finn. Decoupling exploration and exploitation for meta-reinforcement learning without sacrifices. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6925–6935. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/liu21s.html.

Chris Lu, Jakub Kuba, Alistair Letcher, Luke Metz, Christian Schroeder de Witt, and Jakob Foerster. Discovered policy optimisation. *Advances in Neural Information Processing Systems*, 35:16455–16468, 2022.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E. Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21(181):1–50, 2020. URL http://jmlr.org/papers/v21/20-212.html.

Alexander Nikulin, Vladislav Kurenkov, Ilya Zisman, Artem Agarkov, Viacheslav Sinii, and Sergey Kolesnikov. Xland-minigrid: Scalable meta-reinforcement learning environments in jax. *Advances in Neural Information Processing Systems*, 37:43809–43835, 2024.

Ben Norman and Jeff Clune. First-explore, then exploit: Meta-learning to solve hard exploration-exploitation trade-offs. *Advances in Neural Information Processing Systems*, 37:27490–27528, 2024.

Junhyuk Oh, Gregory Farquhar, Iurii Kemaev, Dan A. Calian, Matteo Hessel, Luisa Zintgraf, Satinder Singh, Hado van Hasselt, and David Silver. Discovering state-of-the-art reinforcement learning algorithms. *Nature*, 648(8093):312–319, December 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09761-x. URL https://doi.org/10.1038/s41586-025-09761-x.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. *Advances in neural information processing systems*, 29, 2016.

Emilio Parisotto, Francis Song, Jack Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, et al. Stabilizing transformers for reinforcement learning. In *International conference on machine learning*, pp. 7487–7498. PMLR, 2020.

Jack Parker-Holder, Minqi Jiang, Michael Dennis, Mikayel Samvelyan, Jakob Foerster, Edward Grefenstette, and Tim Rocktäschel. Evolving curricula with regret-based environment design. In *International Conference on Machine Learning*, pp. 17473–17498. PMLR, 2022.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pp. 2778–2787. PMLR, 2017.

Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. Teacher algorithms for curriculum learning of deep rl in continuously parameterized environments. In *Conference on Robot Learning*, pp. 835–853. PMLR, 2020.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015a.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015b.

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

Bradly C Stadie, Ge Yang, Rein Houthooft, Xi Chen, Yan Duan, Yuhuai Wu, Pieter Abbeel, and Ilya Sutskever. Some considerations on learning to explore via meta-reinforcement learning. *arXiv preprint arXiv:1803.01118*, 2018.

Sainbayar Sukhbaatar, Zeming Lin, Ilya Kostrikov, Gabriel Synnaeve, Arthur Szlam, and Rob Fergus. Intrinsic motivation and automatic curricula via asymmetric self-play. *arXiv preprint arXiv:1703.05407*, 2017.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, MA, 1998.

Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

Zhongwen Xu, Hado van Hasselt, and David Silver. Meta-gradient reinforcement learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/2715518c875999308842e3455eda2fe3-Paper.pdf.

Guibin Zhang, Jiayang Lyu, Ran Sun, Xinlei Yu, Haoyu Zhao, Qibing Ren, and Shuicheng Yan. Latent on-policy self-distillation. *arXiv preprint arXiv:2608.13040*, 2026.

Jin Zhang, Jianhao Wang, Hao Hu, Tong Chen, Yingfeng Chen, Changjie Fan, and Chongjie Zhang. Metacure: Meta reinforcement learning with empowerment-driven exploration. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12600–12610. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/zhang21w.html.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

Luisa M Zintgraf, Leo Feng, Cong Lu, Maximilian Igl, Kristian Hartikainen, Katja Hofmann, and Shimon Whiteson. Exploration in approximate hyper-state space for meta reinforcement learning. In *International Conference on Machine Learning*, pp. 12991–13001. PMLR, 2021.

APPENDIX CONTENTS

A Experimental Details **15**

B Environments **15**

C First-Explore Background **16**

D Training Efficiency **16**

E Critic-based Mode Switching versus a Fixed Number of Explore Episodes **17**

F First-Explore's Provable Limits **19**

G Sweeping over the Number of Explore Episodes **21**

H Hyperparameters **21**

I Gated Transformer XL Baselines **22**

J Statistical Tests **25**

A Experimental Details

For efficiency, rather than generating a new environment instance at every trial reset, we maintain a pool of pre-generated instances (size given in Table 5) and start each new trial from one drawn uniformly at random from the pool. The pool is regenerated from fresh random draws after every PPO update, so training is not restricted to a fixed subset of instances. For Deceptive XLand-MiniGrid, tasks come from a fixed set of one million rulesets; each instance pairs a ruleset drawn from this set with a randomly generated layout.

Each run uses a single NVIDIA L40S GPU and 8 CPUs. With the GRU architecture, a 500M-step run takes roughly 0.5–1 GPU-hour in Bandit, Dark Treasure Room, and Ray Maze, so the hyperparameter searches and final runs for these three environments total approximately 300 GPU-hours. Deceptive XLand-MiniGrid is substantially more expensive: each 5B-step search run takes about 2.3 GPU-hours and each 50B-step final run about two GPU-days, for a total of approximately 2,800 GPU-hours. The Transformer-XL experiments required approximately 400 GPU-hours. The original First-Explore required 4.3, 46.5, and 27.2 GPU-hours per run in Bandit, Dark Treasure Room, and Ray Maze respectively, or approximately 800 GPU-hours across ten seeds. In total, the experiments reported in this paper used approximately 4,300 GPU-hours, excluding development and preliminary experiments.

B Environments

Bandit Bandit is a ten-armed Gaussian bandit. The agent selects an arm $a \in \{1, \dots, 10\}$ at each timestep. The first arm always gives the fixed reward $\mu_1 = 0.5$, while the remaining arms have instance-specific means $v_a \sim \mathcal{N}(0, 1)$. Pulling one of these arms produces reward $v_a + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \frac{1}{2})$. The arm means are fixed for the duration of the trial, but the noise term is resampled on each pull.

This environment is deceptive because arm 1 provides a reliable positive reward but reveals no information about the remaining arms. Since the other arms have a mean reward of zero across instances, sampling them for exploration has lower expected reward than repeatedly pulling arm 1. However, high trial return requires identifying an arm whose sampled mean exceeds μ_1, then exploiting that arm for the remainder of the trial.

Dark Treasure Room Dark Treasure Room is a 9×9 grid world containing hidden treasures and traps. The agent starts in the center of the grid and observes only its current (x, y) coordinate. It has five actions: move in one of the four cardinal directions or remain still. Each environment contains eight objects placed uniformly at random. When the agent reaches an object, the object is consumed for that episode and the agent receives its associated reward. Object reward values are sampled uniformly from the range $[-4, 2]$, and overlapping objects have additive rewards. This environment is deceptive because visiting an unseen object has negative expected reward. The agent must nevertheless move blindly into previously unvisited coordinates to discover which locations contain positive rewards.

Ray Maze Ray Maze is a partially observable navigation task in a randomly generated maze with impassable walls and three goal locations. The agent has three actions: turn left, turn right, or move forward. It observes the environment through 11 LiDAR-like rays, each of which reports the distance to the nearest wall along that ray, the wall orientation, and whether the ray intersects a goal location. These observations reveal where goals are, but not whether a goal is beneficial or harmful.

Each goal independently gives reward $+1$ with probability 0.3 and reward -1 otherwise. Only the first goal reached in an episode produces reward, preventing the agent from evaluating multiple goals within a single episode. Since an unknown goal has negative expected value, the agent must initially risk visiting goals in order to determine their rewards. High return then requires remembering which goals are positive and navigating back to them in later episodes, while avoiding both known negative goals and untested goals.

Deceptive XLand-MiniGrid XLand-MiniGrid (Nikulin et al., 2024) is a grid world in which agents must discover interaction rules to produce a goal object, mimicking the dynamics of the

XLand environment introduced by (Bauer et al., 2023). Each level contains randomly generated objects of varying shapes and colors, and a random set of "production" rules that decide how objects interact. The agent can move, pick up, and place objects. The AgentHold rule causes objects to transform into a new object when held, and the TileNear rule causes an object to transform when placed next to another object specified by the production rule. Agents must interact with objects to discover the production rules and trigger the goal rule, which may be AgentHold or TileNear. Triggering the goal rule grants a reward of $1 - t/T$ where t is the number of steps taken and T is the maximum episode length.

We create a new deceptive-reward XLand-MiniGrid environment (Deceptive XLand-MiniGrid) that guarantees a "dead-end" rule in every rule-set. Triggering this rule consumes an object that is necessary for the goal rule, rendering the level unsolvable, and resulting in a penalty of -0.1. The dead-end rules are easier to reach than the goal-rule, so the agent is likely to first discover this rule then learn to avoid it to achieve a non-negative score. Importantly, the dead-end rules can be discovered in an early episode to reliably avoid them and reach the goal in later trials. We train on 7×7 grids with an episode length of 50 to focus on rule discovery instead of navigation. Agents require many more samples to perform well in this environment, so we are not able to test the expensive original implementation of First-Explore in this domain.

C First-Explore Background

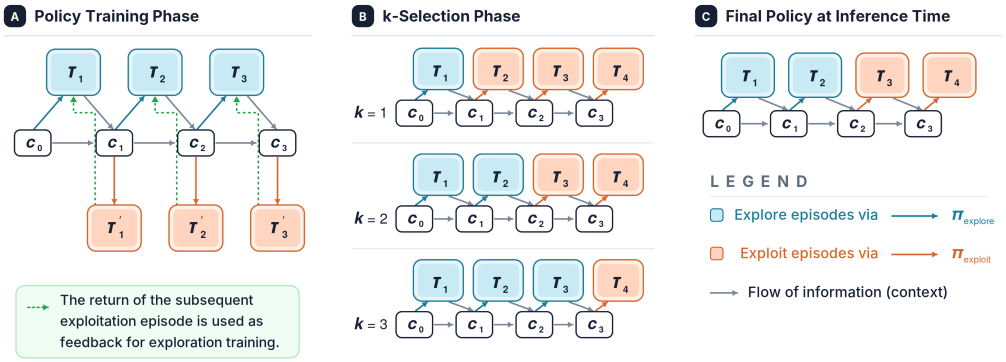

Figure 7: A full diagram of the First-Explore algorithm. a) The training phase trains an agent in explore-exploit episode pairs, propagating rewards through the pair. b) After training, k is swept to find the best number of exploration episodes. c) At inference time, the best k value is held fixed and used for rollouts.

D Training Efficiency

Table 2: Training time comparison across environments, evaluated on NVIDIA L40S GPUs. Parentheses indicate speedup relative to the original FE Transformer.

Method	Bandit	Dark Treasure Room	Ray Maze
FE Original Transformer	4.3 hours	46.5 hours	27.2 hours
FE+ Transformer-XL	23 min (11×)	2.0 hours (23×)	1.1 hours (24×)
FE+ GRU	6 min (44×)	6 min (469×)	10 min (165×)

We compare FE+ to First-Explore in Figure 9 for sample efficiency and Figure 10 for time efficiency, using log scales for environment interactions and time respectively. First-Explore+ is **37 to 96**

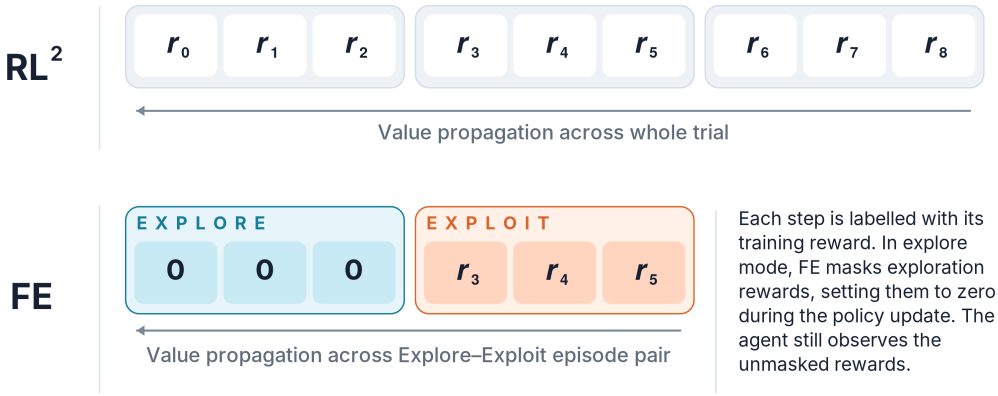

Figure 8: The value propagation used to train First-Explore and RL^2. RL^2 propagates credit assignment and gradients across an entire trial, while First-Explore only propagates across a single explore-exploit pair.

times more sample-efficient and **44-469 times more time efficient** than the original First-Explore. This improved time efficiency allows First-Explore+ (GRU) to train in about 6 minutes on Dark Treasure Room and 10 minutes on Ray Maze, compared with 46.5 and 27.2 hours for the original First-Explore (Table 2). These dramatic efficiency improvements make experimentation and future development of First-Explore+ cheaper and more accessible. Importantly, faster training allows us to apply First-Explore+ to more challenging environments that require many samples to solve. We demonstrate this by applying First-Explore+ to XLand-MiniGrid, an environment where agents only begin to make progress after several billion steps.

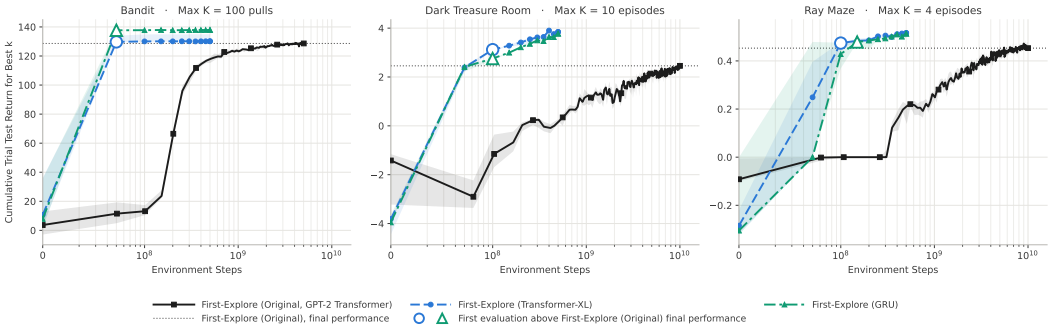

Figure 9: Sample efficiency of First-Explore+ (**GRU green triangles**, **Transformer blue circles**) compared to the original First-Explore algorithm (**black squares**), plotting cumulative return against training steps across three environments on a symmetric logarithmic (symlog) x-axis: Bandit, left; Dark Treasure Room, middle; and Ray Maze, right. First-Explore+ implements the First-Explore framework with markedly improved sample efficiency and performance—**green** and **blue** lines above **black**—exceeding (hollow markers) the original's final performance (dotted **gray**) early during training.

E Critic-based Mode Switching versus a Fixed Number of Explore Episodes

We compare the switching rule of Section 3.3 against the fixed-k schedule used by First-Explore.

Switching rule. At the start of each episode, with h episodes remaining in the trial and current context c, the agent takes one more exploration episode if

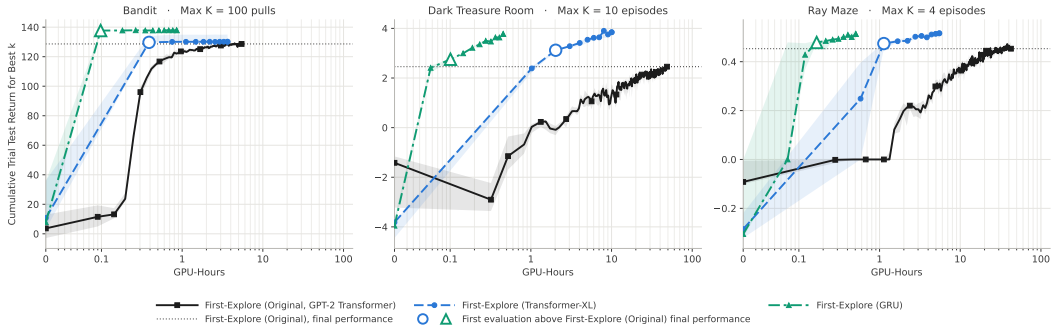

Figure 10: Time efficiency of First-Explore+ compared to the original First-Explore algorithm, plotting the same metrics as Figure 9 on a symlog x-axis but against time rather than step count. Though **GRU (green triangles)** and **Transformer (blue circles)** exhibit similar sample efficiency (Figure 9, **blue** and **green** lines close), the **GRU** implementation allows significantly greater time efficiency in suitable environments (**green** lines above and left of **blue** ones).

$$\mathbb{E}(\text{value of exploring once and then exploiting thereafter}) > \mathbb{E}(\text{value of exploiting from now})$$
$$\equiv U(c) + (h-1)\,\frac{V(c,\text{explore})}{\gamma^{L}} > h\,V(c,\text{exploit}),$$

and otherwise exploits for the remainder of the trial. $V(c,\text{explore})$ is divided by γ^{L}, where L is the episode length, because it values the next exploitation episode from the start of the exploration episode, L discounted steps earlier. This correction is exact for environments with a fixed episode length, such as those of Norman & Clune (2024). For variable-length episodes we use a fixed L, which makes the correction approximate; in XLand we use the maximum episode length, $L = 50$.

Fixed k. For each trained policy, we evaluate every $k \in \{0, \ldots, K\}$ on 4096 tasks. We select the k with the highest mean trial return, then re-evaluate that k on 4096 new tasks, so the reported return is not biased by the selection. The switching rule is evaluated on these same 4096 new tasks, so both numbers for a policy come from identical tasks. In both cases, each exploitation episode starts from the context at the end of exploration.

Table 3: The U rule against a fixed number of explore episodes. Cumulative trial return, interquartile mean with 95% bootstrap confidence interval in brackets, on 4096 held-out tasks shared by every column. First-Explore+ columns: the same 10 trained policies per environment, run with the U rule and at the best fixed k (chosen on a separate 4096-task draw). Without U critic: 5 policies trained without the U critic, at their best fixed k. p_{same}: exact two-sided Wilcoxon signed-rank test, U rule vs best fixed k on the same policies, paired by seed. $p_{\text{no U}}$: exact two-sided Mann–Whitney U test, U rule vs the policies trained without the U critic.

		First-Explore+ (10 seeds)			Without U critic (5 seeds)	
	Environment	U rule	Best fixed k	p_{same}	Best fixed k	$p_{\text{no U}}$
Greedy	Bandit	138.7 [136.7, 138.8]	137.9 [135.6, 137.9]	0.002	137.9 [137.9, 137.9]	0.075
	Dark Treasure Room	3.86 [3.79, 3.96]	3.73 [3.64, 3.84]	0.002	3.73 [3.60, 3.77]	0.013
	Ray Maze	0.60 [0.58, 0.61]	0.51 [0.51, 0.53]	0.002	0.51 [0.49, 0.53]	0.0007
	XLand	1.25 [1.10, 1.43]	1.24 [1.09, 1.43]	0.695	0.99 [0.32, 1.06]	0.019
Stochastic	Bandit	138.7 [136.3, 138.8]	137.9 [135.6, 137.9]	0.002	137.8 [137.8, 137.9]	0.075
	Dark Treasure Room	3.85 [3.77, 3.92]	3.70 [3.61, 3.76]	0.002	3.65 [3.53, 3.71]	0.003
	Ray Maze	0.59 [0.57, 0.60]	0.51 [0.51, 0.52]	0.002	0.50 [0.47, 0.53]	0.0007
	XLand	1.09 [0.94, 1.24]	1.09 [0.94, 1.26]	0.160	0.90 [0.29, 0.97]	0.040

Policies. We evaluate the 10 First-Explore+ policies per environment from Section 4.1 both with the switching rule and at their best fixed k. We also train 5 policies per environment without the U critic and evaluate them at their best fixed k.

Results. Table 3 reports the results.

- **Same policies.** In Bandit, Dark Treasure Room and Ray Maze, the switching rule achieves a higher return than the best fixed k for every one of the 10 policies, under both greedy and stochastic action selection ($p = 0.002$, the smallest attainable with 10 seeds). The rule adapts the amount of exploration to each task. On average it explores fewer episodes than the best fixed k in Bandit (7.6 vs. a median of 9), about as many in Dark Treasure Room (5.0 vs. 5), and more in Ray Maze (1.5 vs. 1). In XLand, the switching rule matches the best fixed k ($p = 0.70$ greedy, $p = 0.16$ stochastic).
- **Without the U critic.** The switching rule outperforms the best fixed k of policies trained without the U critic in Dark Treasure Room ($p \leq 0.013$), Ray Maze ($p = 0.0007$) and XLand ($p \leq 0.040$). In Bandit the difference is not significant ($p = 0.075$).
- **Effect of training with U.** In XLand, training with the U critic also improves the policy itself. At their best fixed k, First-Explore+ policies achieve 1.24 vs. 0.99 under greedy selection ($p = 0.019$) and 1.09 vs. 0.90 under stochastic selection ($p = 0.055$), with all 10 seeds learning against 4 of 5 without the U critic. In Bandit, Dark Treasure Room and Ray Maze, both variants use the same hyperparameters. At their best fixed k they perform alike there ($p \geq 0.37$), so the gains in these environments come from the switching decision.

F First-Explore's Provable Limits

As originally presented in Norman & Clune (2024), First-Explore sets how many episodes it spends exploring in a trial before inference: it cannot adaptively choose to explore for more episodes on some tasks but not on others. Assuming both modes of the policy fit their training objectives, this constraint provably limits the return that First-Explore's objectives guarantee in certain environments.

To show this, we consider Ray Maze as described in Sections B and 4.2: a trial of $K = 4$ episodes with three goals, each independently worth $+1$ with probability 0.3 and -1 otherwise, where only the first goal touched in an episode is rewarded. As in Figure 4C, we further assume that all goals are reachable within the episode time limit. We analyze the optimal behavior associated with each fixed k in turn.

- In the $k = 0$ case: First-Explore exploits each episode, which requires visiting known positive-value goals and otherwise avoiding goals. As the agent does not know the value of any goal in the first episode, the optimal single-episode exploit behavior is to avoid all goals in the first episode. The same logic then applies to the subsequent episodes. Perfectly learned, this behavior achieves exactly 0 trial return.
- In the $k = 1$ case:
  - In the first episode, exploration constitutes visiting a goal.
  - Exploitation in the subsequent episodes constitutes returning to that goal iff it was found to be positive, and otherwise avoiding unknown goals.
  - The maximum achievable trial return under this behavior is $4\mathbb{P}(\text{goal positive}) - \mathbb{P}(\text{goal negative}) = 4 \times 0.3 - 0.7 = 0.5$.
- In the $k = 2$ case:
  - In the first episode, exploration constitutes visiting a goal.
  - In the second episode, exploration requires visiting a new goal iff the first goal was found to be negative. If the first goal was positive, the next exploit return is already $+1$, so the explore objective is indifferent to second-episode behavior; we take the objective-optimal policy that touches no goal.
  - Exploitation in the subsequent episodes constitutes returning to a goal found to be positive, and otherwise avoiding unknown goals.

  - The trial return under this behavior is then

    $$3\mathbb{P}(\text{first goal positive}) + \mathbb{P}(\text{first goal negative})\big(3\mathbb{P}(\text{second goal positive}) - \mathbb{P}(\text{second goal negative}) - 1\big)$$
    $$= 3 \times 0.3 + 0.7 \times (3 \times 0.3 - 0.7 - 1) = 0.34.$$

- Higher values of k score even lower by the same analysis (-0.22 for $k = 3$ and -0.88 for $k = 4$).

Thus no fixed k guarantees more than 0.5 on Ray Maze: for $k = 1$ every objective-optimal policy attains exactly 0.5, and for every other k some objective-optimal policy attains less. In contrast, a policy that explores a second time only after a negative first goal attains $0.3 \times 4 + 0.7 \times (-1 + 0.3 \times 3 - 0.7) = 0.64$.

The critic-based switching rule of Section 3.3 recovers this adaptive policy. With exact critics, at the start of a trial the rule compares 0×4 against $0.3 \times 3 + (0.3 - 0.7) = 0.5$ and explores; after a negative first goal it compares 0 against $0.3 \times 2 - 0.4 = 0.2$ and explores again; after two negative goals it compares 0 against $0.3 - 0.4 = -0.1$ and stops; and after a positive goal it compares 3 against $2 + U(c) \leq 3$ and stops, regardless of what the explore mode would do. First-Explore+ with exact critics therefore attains 0.64.

Empirically, the original First-Explore stays below the 0.5 bound (0.46, 95% CI $[0.44, 0.48]$), while First-Explore+ exceeds it (0.60, 95% CI $[0.58, 0.61]$; Table 1).

G SWEEPING OVER THE NUMBER OF EXPLORE EPISODES

After training First-Explore+, we sweep over the number of explore episodes to find the optimal tradeoff between exploration and exploitation. As we see in Figure 11, additional exploration episodes improve performance up to a point. However, each episode spent searching for information decreases the amount of time the agent can spend exploiting previously discovered rewards. This tradeoff is most visually apparent in the Bandit environment, where each extra exploration episode results in exactly 1 less pull of the best-known arm. This leads to a nearly linear decrease in cumulative reward as the number of exploration episodes increases past the optimal number.

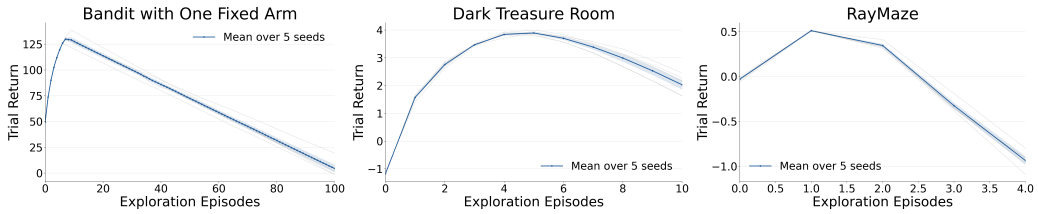

Figure 11: Cumulative return achieved by the agent across all episodes in a trial for different numbers of exploration episodes. The mean and standard deviation over 5 models is shown in blue and the returns for individual models are shown in gray. This shows the tradeoff between collecting more information and exploiting currently known sources of rewards.

H HYPERPARAMETERS

All GRU results in the main text use the hyperparameters in Tables 4 to 6. For each method and environment we ran a random search of 50 configurations over the space in Table 4, each trained on one seed for 500M environment steps (on Deceptive XLand-MiniGrid: two seeds and 5B environment steps). The searches use identically seeded draws, so First-Explore+ and RL^2 evaluate the same 50 candidate configurations in every environment; E-RL^2 additionally samples its number of masked episodes and so draws a different set. The selected configuration for each method and environment was trained on ten seeds.

On Deceptive XLand-MiniGrid, no configuration in the RL^2, E-RL^2 or GRU-PPO searches achieved a statistically significant positive return within the 5B-step search budget, so all three were trained with the best hyperparameters from a First-Explore+ hyperparameter sweep without the (U) critic or exploit context injection. With these hyperparameters, GRU-PPO reached positive return in 3 of 10 seeds when trained for 50B steps.

Settings that were not searched—architecture, batch shape, the number of episodes per trial K, and the exploit-context injection probability ρ—are fixed per environment (Tables 5 and 6). Exploit-context injection is sustained: once triggered in a trial, it applies to every remaining exploration episode of that trial.

Table 4: Random-search space, identical for First-Explore+, E-RL^2, RL^2 and single-episode PPO in every environment. Each search draws 50 configurations, trained on 1 seed for 500M environment steps (Deceptive XLand-MiniGrid: 50 configurations, 2 seeds, 5B steps). E-RL^2 additionally searches the number of initial episodes whose return is masked from its objective.

Hyperparameter	Distribution	Range
Learning rate	Log-uniform	$[2.5 \times 10^{-4}, 5.0 \times 10^{-3}]$
Entropy coefficient	Uniform	$[0, 0.05]$
PPO clipping ratio	Uniform	$[0.1, 0.3]$
Discount factor γ	Categorical	$\{0.99, 0.995, 0.999\}$
GAE λ	Uniform	$[0.9, 0.99]$
E-RL^2 only: number of masked episodes		
Bandit	Uniform integer	$\{7, \ldots, 12\}$
Dark Treasure Room	Uniform integer	$\{3, \ldots, 7\}$
Ray Maze	Uniform integer	$\{1, 2\}$
Deceptive XLand-MiniGrid	Uniform integer	$\{1, 2\}$

Table 5: Settings fixed per environment (not searched), shared by every method. The training task pool is redrawn after every update. The bottom block is shared by all runs.

	Bandit	Dark Treasure Room	Ray Maze	Deceptive XLand MiniGrid
Episodes per trial K	100	10	4	5 (PPO: 1)
Parallel environments	512	1,024	1,024	16,384
Rollout length (steps)	200	378	240	256
Minibatches per update	4	4	4	32
Update epochs	4	4	4	1
GRU hidden size	256	256	256	1024
Actor/critic head hidden size	128	128	256	256
Training task pool size	4,096	4,096	4,096	2,048
Total environment steps	500M	500M	500M	50B
Value-function loss coefficient	0.5			
Maximum gradient norm	0.5			
Adam ϵ	1×10^{-8}			
Learning-rate schedule	linear decay to 0			
U-critic loss coefficient (First-Explore+)	0.5			

I GATED TRANSFORMER XL BASELINES

Hyperparameters for the Transformer-XL runs (five seeds per method and environment) are given in Tables 8 and 9.

Table 6: Selected hyperparameters per environment and method. First-Explore+ lists its exploit-context injection probability ρ; E-RL^2 lists the number k of initial episodes whose return it masks. On Deceptive XLand-MiniGrid, no configuration in the RL^2, E-RL^2 or GRU-PPO searches achieved a statistically significant positive return within the 5B-step search budget, so all three were trained with the best hyperparameters from an earlier First-Explore+ hyperparameter sweep. With these hyperparameters, GRU-PPO reached positive return in 3 of 10 seeds when trained for 50B steps.

Environment	Method	Learning rate	Entropy coef.	Clip ratio	γ	GAE λ	Method-specific
Bandit	First-Explore+	1.74×10^{-3}	1.33×10^{-2}	0.188	0.999	0.931	$\rho = 0.00161$
	E-RL^2	1.59×10^{-3}	2.55×10^{-2}	0.241	0.99	0.939	$k = 7$
	RL^2	6.43×10^{-4}	2.24×10^{-2}	0.281	0.99	0.960	–
Dark Treasure Room	First-Explore+	2.38×10^{-3}	2.21×10^{-3}	0.127	0.99	0.902	$\rho = 0.09$
	E-RL^2	2.01×10^{-3}	6.12×10^{-3}	0.110	0.995	0.933	$k = 6$
	RL^2	3.53×10^{-3}	2.10×10^{-2}	0.244	0.999	0.913	–
Ray Maze	First-Explore+	1.34×10^{-3}	2.07×10^{-2}	0.105	0.999	0.987	$\rho = 0.09$
	E-RL^2	7.32×10^{-4}	3.85×10^{-2}	0.169	0.99	0.974	$k = 1$
	RL^2	1.34×10^{-3}	2.07×10^{-2}	0.105	0.999	0.987	–
Deceptive XLand MiniGrid	First-Explore+	2.30×10^{-3}	2.27×10^{-2}	0.104	0.995	0.978	$\rho = 0.035$
	E-RL^2	3.44×10^{-3}	1.90×10^{-2}	0.212	0.995	0.977	$k = 1$
	RL^2	3.44×10^{-3}	1.90×10^{-2}	0.212	0.995	0.977	–
	GRU-PPO	3.44×10^{-3}	1.90×10^{-2}	0.212	0.995	0.977	–

Table 7: Cumulative trial return of the Transformer-XL runs after 500M environment steps (interquartile mean over 5 seeds, 95% bootstrap confidence interval in brackets, 4096 held-out tasks). First-Explore+ matches RL^2 in Bandit under greedy action selection and outperforms it in every other environment and action selection. RL^2 scores approximately zero in Ray Maze on every seed.

	Greedy action selection		Stochastic action selection	
Environment	First-Explore+	RL^2	First-Explore+	RL^2
Bandit	131.7 [127.2, 136.9]	128.4 [124.7, 130.8]	**131.4** [127.0, 136.3]	84.8 [82.5, 85.5]
Dark Treasure Room	**3.84** [3.66, 3.95]	3.39 [3.21, 3.53]	**3.81** [3.65, 3.98]	2.17 [2.12, 2.24]
Ray Maze	**0.510** [0.494, 0.523]	0.000 [0.000, 0.000]	**0.473** [0.462, 0.490]	0.000 [-0.001, 0.000]

Table 8: Transformer-XL random-search space, shared by First-Explore+ and RL^2. Each search draws 50 configurations, trained on one seed for 100M environment steps; the Ray Maze First-Explore+ configuration comes from a search of 50 configurations on two seeds with a 5B-step budget. Every configuration uses 8 attention heads with query/key/value width equal to the embedding size.

Hyperparameter	Distribution	Range
Learning rate	Log-uniform	$[5 \times 10^{-5}, 5 \times 10^{-4}]$
Entropy coefficient	Log-uniform	$[1 \times 10^{-3}, 1 \times 10^{-1}]$
PPO clipping ratio	Uniform	$[0.1, 0.3]$
Discount factor γ	Categorical	$\{0.99, 0.995, 0.999\}$
GAE λ	Uniform	$[0.9, 0.99]$
Transformer-XL architecture		
Embedding size	Categorical	$\{128, 256\}$
Number of layers	Categorical	$\{1, 2, 3\}$
Memory window (steps)	Categorical	$\{32, 64, 128\}$
MLP hidden size	Categorical	$\{128, 256\}$
Attention gating	Categorical	$\{\texttt{False}, \texttt{True}\}$

Table 9: Selected Transformer-XL hyperparameters (five seeds each). RL^2 uses one configuration in every environment. The bottom block is shared by all runs.

	First-Explore+			RL^2
Hyperparameter	Bandit	Dark Treasure Room	Ray Maze	All
Learning rate	1.75×10^{-4}	2.68×10^{-4}	1.69×10^{-4}	2.11×10^{-4}
Entropy coefficient	9.15×10^{-2}	2.85×10^{-3}	6.67×10^{-2}	4.93×10^{-2}
PPO clipping ratio	0.131	0.278	0.158	0.275
Discount factor γ	0.999	0.995	0.995	0.999
GAE λ	0.940	0.926	0.950	0.926
Embedding size	256	256	128	256
Number of layers	1	2	2	3
Memory window (steps)	32	64	64	32
MLP hidden size	128	128	128	256
Attention gating	`True`	`False`	`False`	`False`
Episodes per trial K	100	10	4	100 / 10 / 4
Attention heads	8 (query/key/value width = embedding size)			
Activation	ReLU			
Parallel environments	512			
Rollout length (steps)	200			
Minibatches per update / update epochs	4 / 4			
Total environment steps	500M			
Training task pool size (fixed per run)	100,000			
Value-function loss coefficient	0.5			
Maximum gradient norm	0.5			
Adam ϵ	1×10^{-5}			
Learning-rate schedule	linear decay to 0			

J STATISTICAL TESTS

We compare First-Explore+ with RL^2 and with E-RL^2 in each environment, under both greedy and stochastic action selection, using exact two-sided Mann–Whitney U tests on the final cumulative trial returns of the 10 seeds per method. The results are shown in Table 10. With 10 seeds per method, the largest possible statistic is $U_{\mathrm{FE}} = 100$, attained under complete rank separation, when every First-Explore+ seed outperforms every baseline seed; the corresponding p-value, 1.1×10^{-5}, is the smallest attainable.

Table 10: Mann–Whitney U tests comparing First-Explore+ with RL^2 and E-RL^2 (exact, two-sided; 10 seeds per method). U_{FE} is First-Explore+'s U statistic (maximum 100). Bolded p-values are statistically significant for $\alpha = 0.05$. *Complete rank separation: every First-Explore+ seed outperforms every baseline seed.

		vs. RL^2		vs. E-RL^2	
Sampling	Environment	U_{FE}	p-value	U_{FE}	p-value
Greedy	Bandit	86	**0.0052**	80	**0.023**
Greedy	Dark Treasure Room	100	$\mathbf{1.1 \times 10^{-5}}^{*}$	82	**0.015**
Greedy	Ray Maze	100	$\mathbf{1.1 \times 10^{-5}}^{*}$	100	$\mathbf{1.1 \times 10^{-5}}^{*}$
Greedy	XLand-MiniGrid	100	$\mathbf{1.1 \times 10^{-5}}^{*}$	100	$\mathbf{1.1 \times 10^{-5}}^{*}$
Stochastic	Bandit	93	$\mathbf{4.9 \times 10^{-4}}$	90	**0.0015**
Stochastic	Dark Treasure Room	100	$\mathbf{1.1 \times 10^{-5}}^{*}$	94	$\mathbf{3.2 \times 10^{-4}}$
Stochastic	Ray Maze	100	$\mathbf{1.1 \times 10^{-5}}^{*}$	100	$\mathbf{1.1 \times 10^{-5}}^{*}$
Stochastic	XLand-MiniGrid	100	$\mathbf{1.1 \times 10^{-5}}^{*}$	100	$\mathbf{1.1 \times 10^{-5}}^{*}$