In-Context Learning for Pure Exploration

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this work, we study the active sequential hypothesis testing problem, also known as *pure exploration*, where the goal is to actively control a data collection process to efficiently identify the correct hypothesis underlying a decision problem. While relevant across multiple domains, devising adaptive exploration strategies remains challenging, particularly due to difficulties in encoding appropriate inductive biases. To address these limitations, we introduce *In-Context Pure Exploration* (`ICPE`), an in-context learning approach that uses Transformers to learn exploration strategies directly from experience. Numerical results across diverse benchmarks highlight `ICPE`'s capability to achieve satisfactory performance in stochastic and structured settings, demonstrating its ability to meta-learn exploration strategies.

1 Introduction

Modern artificial intelligence systems have achieved remarkable performance across specialized tasks such as image classification Krizhevsky et al. [2012], Super-human board-game play Silver et al. [2018], protein-structure prediction Jumper et al. [2021] and large-scale language modelling Brown et al. [2020]. Yet, there is still a lack in understanding how to autonomously discover meta-skills fundamental for sequential decision making, such as active testing or active learning Chernoff [1992], Cohn et al. [1996].

Consider an agent tasked with sequentially selecting samples to quickly improve its understanding of an underlying phenomenon. When the decision maker can exert some control over the collected samples' *information content*, this is a problem also known as the *active sequential hypothesis testing problem* Chernoff [1992], Ghosh [1991], Naghshvar and Javidi [2013], Naghshvar et al. [2012], Mukherjee et al. [2022] or *pure exploration problem* Degenne and Koolen [2019], Degenne et al. [2019, 2020]. Active hypothesis testing has become increasingly important nowadays, with applications ranging from medical diagnostics Berry et al. [2010], image identification Vaidhiyan et al. [2012], recommender systems Resnick and Varian [1997], etc. Nonetheless, devising an adaptive data collection strategy is notoriously difficult and highly problem-specific.

In this paper, we address the question: how can sequential decision-making agents autonomously discover and leverage hidden structure to enhance active exploration for hypothesis testing? We introduce *In-Context Pure Explorer* (`ICPE`), a novel method combining Supervised Learning and Deep RL Goodfellow et al. [2016], Murphy [2023] , which builds on the in-context learning and sequence modeling capabilities of Transformers Lee et al. [2023]–a meta-learning approach that uncovers underlying shared structure across a class of problems $\mathcal{M}$ Schaul and Schmidhuber [2010], Bengio et al. [1990].

`ICPE` operates by integrating two complementary neural networks: an inference (I) network, trained via supervised learning to infer the true hypothesis given current data, and an exploration (π) network, trained through reinforcement learning to select actions optimizing the inference accuracy of the I network.

Submitted to the First Exploration in AI Today Workshop at ICML (EXAIT at ICML 2025). Do not distribute.

We validate ICPE through different benchmarks, demonstrating its ability to efficiently explore in stochastic and structured environments. In particular, these results show that ICPE achieves performance comparable to optimal instance-dependent Best Arm Identification (BAI) algorithms Garivier and Kaufmann [2016], Audibert and Bubeck [2010], without requiring explicit problem-specific exploration strategies that often involve solving complex optimization problems. Thanks to the in-context capability of ICPE, it is effectively discovering active sampling techniques that at test time do not need much more computation than a forward pass. Consequently, ICPE emerges as a practical applicable method for data-efficient exploration.

1.1 Related Work

The problem of active sequential hypothesis testing Chernoff [1992], Ghosh [1991], Lindley [1956], Naghshvar and Javidi [2013], Naghshvar et al. [2012], Mukherjee et al. [2022], Gan et al. [2021], in which a learner is tasked with adaptively performing a sequence of actions to identify an unknown property of the environment, is closely related to the exploration problem in Reinforcement Learning (RL) Sutton and Barto [2018], where an agent needs to identify the optimal policy. This exploration problem has long centred on regret minimisation Sutton and Barto [2018], with techniques based on Upper-Confidence Bounds Auer et al. [2002, 2008], Cappé et al. [2013], Lattimore and Hutter [2012], Auer [2002], posterior-sampling Kaufmann et al. [2012], Osband et al. [2013], Russo and Van Roy [2014], Gopalan et al. [2014] and Information-Directed Sampling (IDS) Russo et al. [2018]; yet these schemes assume that minimizing regret is the sole objective and falter in identification problems.

A more closely related setting is that of pure exploration in bandits and Markov Decision Processes (MDPs), settings known as Best Arm/Policy Identification (BAI/BPI) Audibert and Bubeck [2010], Garivier and Kaufmann [2016], Degenne and Koolen [2019], Al Marjani et al. [2021], Russo and Proutiere [2023a], Russo et al. [2025]. In these problems the samples collected by the agent are no longer perceived as rewards, and the agent must actively optimize its exploration strategy to identify the optimal policy. BAI/BPI reframe the task as sequential hypothesis testing, yielding instance-adaptive algorithms in fixed-confidence settings such as Track-and-Stop (TaS) Garivier and Kaufmann [2016]. However, while BAI strategy are powerful, they may be suboptimal when the underlying information structure is not adequately captured within the hypothesis testing framework. Although IDS and BAI offer frameworks to account for such structure, extending these approaches to Deep Learning is difficult, particularly when the information structure is unknown.

Recently Transformers Vaswani et al. [2017], Chen et al. [2021] have demonstrated remarkable in-context learning capabilities Brown et al. [2020], Garg et al. [2022]. In-context learning Moeini et al. [2025] is a form of meta-RL Beck et al. [2023], where agents can solve new tasks without updating any parameters by simply conditioning on additional context, such as their action-observation histories. Building on this ability, Lee et al. [2023] recently showed that Transformers can be trained in a supervised manner using offline data to mimic posterior sampling in reinforcement learning. In Dai et al. [2024] the authors presente ICEE (In-Context Exploration Exploitation). ICEE uses Transformer architectures to perform in-context exploration-exploration for RL. ICEE tackles this challenge by expanding the framework of return conditioned RL with in-context learning Chen et al. [2021], Emmons et al. [2021]. Return conditioned learning is a type of technique where the agent learns the return-conditional distribution of actions in each state. Actions are then sampled from the distribution of actions that receive high return Srivastava et al. [2019], Kumar et al. [2019]. Lastly, we note the important contribution of RL^2 Duan et al. [2016], which proposes to represent an RL policy as the hidden state of an RNN, whose weights are learned via RL. ICPE employs a similar idea, but focuses on a different objective (identification), and splits the process into a supervised inference network that provides rewards to an RL-trained transformer network that selects actions to maximize information gain.

2 Learning to Explore: In-Context Pure Exploration

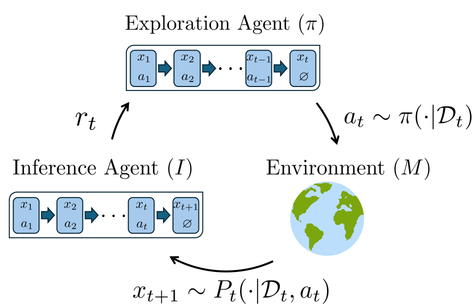
We introduce ICPE (In-Context Pure Exploration), a deep-learning framework that combines sequential architecture with supervised and reinforcement learning to automatically discover efficient exploration policies for active sequential hypothesis testing. Instead of explicitly encoding inductive biases, we use transformers to let the agent autonomously infer the problem structures from experiences.

Environment and Interaction Model. We consider a model class of environments $\mathcal{M}$ and a distribution $\mathcal{P}(\mathcal{M}) \in \Delta(\mathcal{M})$ from which the true environment M is sampled from. We model an environment as a tuple $M = (\mathcal{X}, \mathcal{A}, P, \rho)$, where $\mathcal{X}$ is a set of possible observations, $\mathcal{A}$ is a finite set of actions, $P = (P_t)_{t\in\mathbb{N}}$ denotes the transition functions, with $P_t : (\mathcal{X} \times \mathcal{A})^t \to \Delta(\mathcal{X})$ and $\rho \in \Delta(\mathcal{X})$ denotes the initial observation distribution. All the environments in a class $\mathcal{M}$ share the same set of observations $\mathcal{X}$ and set of actions $\mathcal{A}$. The learner interacts with the environment in a sequential manner: (1) an initial observation $x_1 \sim \rho$ is sampled from $\mathcal{X}$; (2) at time-step t, the learner chooses an action a_t and observes the next observation $x_{t+1} \sim P_t(\cdot|\mathcal{D}_t, a_t)$, meaning that x_{t+1} is drawn independently from $P_t(\cdot|\mathcal{D}_t, a_t)$ given a trajectory $\mathcal{D}_t = (x_1, a_1, \dots, x_{t-1}, a_{t-1}, x_t)$. Formally, the learner uses a *randomized policy* $\pi = (\pi_t)_{t\in\mathbb{N}}$, which is a sequence of deterministic functions, to select actions: action a_t is selected by sampling independently from $\pi_t(\mathcal{D}_t)$ (with $\mathcal{D}_t$ being a random variable), where $\pi_t(\mathcal{D}_t)$ specifies a probability distribution over $\mathcal{A}$.

We assume a task-specific ground-truth hypothesis $H_M^\star$ from a predefined class $\mathcal{H}$ of hypotheses for each environment, where our goal is to efficiently infer this hypothesis. Informally, we can state our objective as follows:

> *Given an environment M drawn from $\mathcal{P}(\mathcal{M})$, how can we learn a sampling strategy π that collects data $\mathcal{D}$ from M so the agent can reliably infer $H_M^\star$ solely from $\mathcal{D}$?*

An oracle $h(\hat{H}; M) = \mathbf{1}_{\{\hat{H}=H_M^\star\}}$ provides supervised feedback at training time (not test time), indicating correctness without revealing hidden structures. Using oracle feedback, we learn an inference mapping $I : \mathcal{D}_t \mapsto \Delta(\mathcal{H})$, yielding posterior distributions over hypotheses given collected data. The estimator $\hat{H}_t \sim I(\cdot|\mathcal{D}_t)$ guides exploration by providing a reward signal to an RL agent collecting the data $\mathcal{D}_t$ using an exploration policy π.



Example: Best Arm Identification A relevant example is that of Best-Arm Identification in MAB problems Garivier and Kaufmann [2016]. Recall that in a MAB problem the decision maker can choose between K different actions $a_1, \dots, a_K$ (we also say *arms*) at each time-step. Upon selecting an action a at time t, it observes a random reward r_t distributed according to a distribution ν_{a_t}. In BAI the goal is to identify the best action $a^\star = \arg\max_a \mathbb{E}_{R\sim\nu_a}[R]$ as quickly as possible (hence $H^\star = a^\star$). While several algorithms have been provided for different settings Soare et al. [2014], Jedra and Proutiere [2020], Russo and Proutiere [2023b], Kocák and Garivier [2020], Poiani et al. [2024], a major issue is that the algorithm design can drastically change if the assumptions change. Moreover, it is difficult to design efficient techniques for more complex settings such as MDPs (in fact, the problem becomes non-convex Marjani and Proutiere [2021], Russo and Pacchiano [2025]). Therefore, in this work we address the open question of whether it is possible to learn efficient exploration strategies directly from experience, avoiding the process of designing a BAI algorithm.

2.1 `ICPE` for Fixed Confidence Problems

In this work, we focus on the fixed confidence setting Garivier and Kaufmann [2016]. In this setting, the agent needs to learn to stop the data sampling process as soon as it is sufficiently confident to have correctly estimated $H^\star$ for an environment M. Let $\mathbb{P}_M^\pi$ be the underlying probability measure of the process $((\mathcal{D}_t, a_t))_t$ under a sampling strategy π. In the following we also write $\mathbb{P}^\pi_{M\sim\mathcal{P}(\mathcal{M})}(\cdot) = \mathbb{E}_{M\sim\mathcal{P}(\mathcal{M})}[\mathbb{P}_M^\pi(\cdot)]$ to denote the expected probability over the prior.

We equip the learner with the capability to stop the sampling process at any point in time. We denote such stopping rule by τ, which is a stopping time with respect to the filtration $(\sigma(\mathcal{D}_t))_t$. Then, the learner wishes to find an optimal stopping rule τ (with $\tau < \infty$ a.s.), exploration policy π and inference network I subject to a confidence level at the stopping time τ:

$$\min_{\tau,\pi,I} \quad \mathbb{E}_{M\sim\mathcal{P}(\mathcal{M})}[\tau] \quad \text{s.t.} \quad \mathbb{P}^\pi_{M\sim\mathcal{P}(\mathcal{M})}(h(\hat{H}_\tau; M) = 1) \geq 1 - \delta. \tag{1}$$

Algorithm 1 ICPE (In-Context Pure Exploration) - Fixed Confidence

```
Input: Tasks distribution $\mathcal{P}(\mathcal{M})$; confidence $\delta$; learning rates $\alpha, \beta$; initial $\lambda$ and hyper-parameters $T, N, \eta$.
Initialize buffer $\mathcal{B}$, networks $Q_\theta, I_\phi$ and set $\bar{\theta} \leftarrow \theta, \bar{\phi} \leftarrow \phi$.
while Training is not over do
    Sample environment $M \sim \mathcal{P}(\mathcal{M})$ with true hypothesis $H^\star$, observe $s_1 \sim \rho$ and set $t \leftarrow 1$.
    repeat
        Execute action $a_t = \arg\max_a Q_\theta(s_t, a)$ in $M$ and observe next state $s_{t+1}$.
        Add experience $z_t = (s_t, a_t, s_{t+1}, d_t = \mathbf{1}\{s_{t+1} \text{ is terminal }\}, H^\star)$ to $\mathcal{B}$.
        Set $t \leftarrow t + 1$.
    until $a_{t-1} = a_{\text{stop}}$ or $t > N$.
    Update variable $\lambda$ according to
```

$$\lambda \leftarrow \max\left(0, \lambda - \beta\left(I_\phi(H^\star|s_\tau) - 1 + \delta\right)\right). \tag{2}$$

```
    Sample batches $B, B' \sim \mathcal{B}$ and update $\theta, \phi$ as
```

$$\theta \leftarrow \theta - \alpha \nabla_\theta \frac{1}{|B|} \sum_{z \in B} \left[\mathbf{1}_{\{a \neq a_{\text{stop}}\}} \left(y_\lambda(z) - Q_\theta(s, a)\right)^2 + \left(r_\lambda(z_{\text{stop}}) - Q_\theta(s, a_{\text{stop}})\right)^2\right], \tag{3}$$

$$\phi \leftarrow \phi + \alpha \nabla_\phi \frac{1}{|B'|} \sum_{z \in B'} \left[\log(I_\phi(H^\star|s'))\right]. \tag{4}$$

```
    Update $\bar{\theta} \leftarrow (1 - \eta)\bar{\theta} + \eta\theta$ and every $T$ steps set $\bar{\phi} \leftarrow \phi$.
end while
```

Introducing a stopping action a_{stop} to π_t, we define $\tau = \min(N, \inf t : a_t = a_{\text{stop}})$ for a maximum horizon N (the horizon is introduced for practical reasons). We consider solving the dual formulation:

$$\min_{\lambda \geq 0} \max_{\pi, I} V_\lambda(\pi, I) = -\mathbb{E}^\pi_{M \sim \mathcal{P}(\mathcal{M})}[\tau] + \lambda\left[\mathbb{P}^\pi_{M \sim \mathcal{P}(\mathcal{M})}\left(h(\hat{H}_\tau; M) = 1\right) - 1 + \delta\right],$$

with $\hat{H}_\tau \sim I(\cdot|\mathcal{D}_\tau)$. To solve this problem, ICPE treats each optimization separately, and optimize using a descent-ascent scheme. ICPE leverages transformers to encode trajectories $\mathcal{D}_t$ as fixed-length states $s_t = (\mathcal{D}_t, \varnothing_{t:N})$ of an induced MDP M, padding with null tokens to horizon N. The resulting MDP formulation has actions $\mathcal{A} \cup a_{\text{stop}}$ and a reward structure penalizing each step until stopping, defined below.

Learning I. The distribution I is modeled using a transformer with parameter ϕ, and we denote it by I_ϕ. Then, considering a fixed (π, λ), the maximization with respect to I amounts to solving

$$\max_\phi \mathbb{E}^\pi_{M \sim \mathcal{P}(\mathcal{M})}[h(\hat{H}_\tau; M)], \qquad \hat{H}_\tau \sim I_\phi(\cdot|s_\tau).$$

Therefore, we can train ϕ via a cross-entropy loss $-\sum_{H'} h(H'; M) \log(I_\phi(H'|s_\tau))$.

Learning π. The policy π is learnt using RL techniques. We define a reward r that penalizes the agent at all time-steps, that is $r_t = -1$, while at the stopping-time we have $r_\tau = -1 + \lambda \mathbb{E}_{H \sim I(\cdot|s_\tau)}[h(H; M)]$. Accordingly, one can define the Q-value of (π, I, λ) in a state-action pair (s, a) as $Q^{\pi, I}_\lambda(s, a) = \mathbb{E}^\pi_{M \sim \mathcal{P}(\mathcal{M})}\left[\sum_{n=t}^\tau r_n \middle| s_t = s, a_t = a\right]$, with $a_n \sim \pi_n(\cdot|s_n)$.

We model π with a transformer of parameter θ, and train it using DQN Mnih et al. [2015], Van Hasselt et al. [2016] with a replay buffer $\mathcal{B}$ and a target network $Q_{\bar{\theta}}$ parameterized by $\bar{\theta}$. To maintain timescale separation, we introduce a separate target inference network $I_{\bar{\phi}}$, parameterized by $\bar{\phi}$, which provides feedback for training θ. Note that, as discussed earlier, we introduce a dedicated stop-action a_{stop} whose value depends solely on history. Thus, its Q-value can be updated at any time, allowing retrospective evaluation of stopping. For learning the Q-values, we define the reward for a transition $z = (s, a, s', d, H^\star)$ as:

$$r_\lambda(z) := -1 + d\lambda \log I_{\bar{\phi}}(H^\star|s'), \quad d = \mathbf{1}\{z \text{ terminal}\},$$

where we set $s' \leftarrow s$ if $a = a_{\text{stop}}$, and terminal means either $a = a_{\text{stop}}$ or the last step in the horizon. We also define the transition z_{stop} by replacing (a, s') with (a_{stop}, s) in z. Then, for $a \neq a_{\text{stop}}$, the Q-values can be learned using a target value:

$$y_\lambda(z) = r_\lambda(z) + (1 - d) \max_i Q_{\bar{\theta}}(s', a_i).$$

Instead, for the stopping action, we use the loss $(r_\lambda(z_{\text{stop}}) - Q_\theta(s, a_{\text{stop}}))^2$. Therefore, the overall loss used for training θ on a transition z is:

$$\mathbf{1}_{\{a \neq a_{\text{stop}}\}} \left(y_\lambda(z) - Q_\theta(s,a)\right)^2 + \left(r_\lambda(z_{\text{stop}}) - Q_\theta(s, a_{\text{stop}})\right)^2,$$

where $\mathbf{1}_{\{a \neq a_{\text{stop}}\}}$ avoids double accounting for the stopping action.

Last steps. Then, to train (θ, ϕ), we sample two independent batches $(B, B') \sim \mathcal{B}$ from the buffer, and compute the gradient updates as in eqs. (3) and (4) of algorithm 1. We periodically update target networks, setting $\bar{\phi} \leftarrow \phi$ every T steps and using a Polyak averaging $\bar{\theta} \leftarrow (1-\eta)\bar{\theta} + \eta\theta$, with $\eta \in (0,1)$.

Finally, we update λ by assessing the confidence of I_ϕ at the stopping time (2) for a fixed (π, I). Thus, for sufficiently small learning rates, optimizing (λ, θ, ϕ) resembles an ascent-descent scheme.

3 Empirical Evaluation

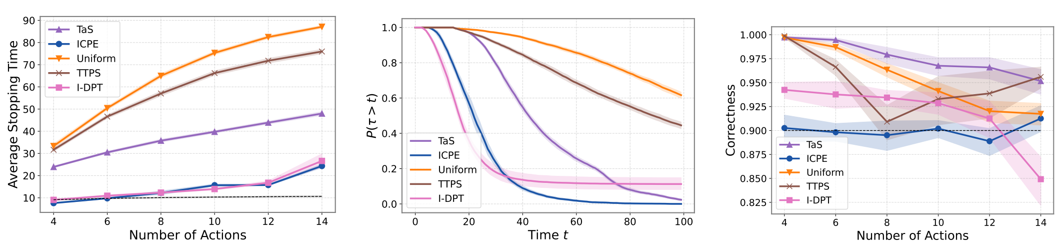
We evaluate our approach across various tasks: stochastic bandits with or without latent structure; learning a probabilistic version of binary search. Due to space limitations, we refer the reader to appendix C for more details and more experiments on MAB problem with feedback graphs Russo et al. [2025], MDPs with hidden information and an an analysis of `ICPE` in the borader setting of classifying images by sequentially revealing image patches.

Algorithms. In our evaluations we compare to different algorithms, depending on the problem. Some of the algorithms include: uniform sampling, TaS (Track and Stop) Garivier and Kaufmann [2016], TTPS (Top Two Sampling) Russo et al. [2018]. We also include a variant of IDS Russo and Van Roy [2018] based on the I-mapping, which uses the observation that I defines a posterior distribution over $\mathcal{H}$. Always based on this idea, we also introduce I-DPT, a variant of DPT Lee et al. [2023], based on the fact that I can be used to explore a problem à-la Thompson Sampling. More information about these methods, and their hyper-parameters, can be found in appendix B [1].

3.1 Bandit Problems

We now apply `ICPE` to the classical BAI problem within MAB tasks. For the MAB setting we have a finite number of actions $\mathcal{A} = \{1, \dots, K\}$, corresponding to the actions in the MAB problem M. For each action a, we define a corresponding reward distribution ν_a from which rewards are sampled i.i.d. Then, $\mathcal{P}(\mathcal{M})$ is a prior distribution on the actions' rewards distributions $(\nu_a)_a$ and for BAI we let $H^\star = \arg\max_a \mathbb{E}_{r\sim\nu_a}[r]$, so that we need to identify the best action. Lastly, the observation at time t is $x_t = (a_t, r_t)$, where a_t is the chosen action at time t and r_t is a reward sampled from ν_{a_t}.

Stochastic Bandit Problems. We evaluate `ICPE` on stochastic bandit environments with $\delta = 0.1$ and $N = 100$. Each action's reward distribution is normally distributed $\nu_a = \mathcal{N}(\mu_a, 0.5^2)$, with $(\mu_a)_{a\in\mathcal{A}}$ drawn from $\mathcal{P}(\mathcal{M})$. In this case $\mathcal{P}(\mathcal{M})$ is a uniform distribution over problems with minimum gap $\max_a \mu_a - \max_{b\neq a}\mu_a \geq \Delta_0$, with $\Delta_0 = 0.4$. Hence, an algorithm could exploit this property to infer $H^\star$ more quickly. For this case, we also derive some sample complexity bounds in appendix A. Figure 1 summarizes the results for this setting. We compare to TaS and



(a) (b) (c)

Figure 1: Results for stochastic MABs with fixed confidence $\delta = 0.1$ and $N = 100$: (a) average stopping time τ; (b) survival function of τ; (c) probability of correctness $\mathbb{P}^\pi_{M\sim\mathcal{P}(\mathcal{M})}(h(\hat{H}; M) = 1)$.

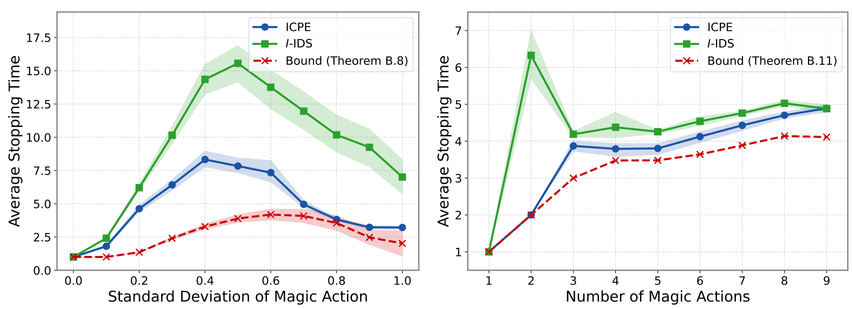
TTPS, and use the stopping rule of TaS also for Uniform and TTPS (the stopping rule is based on a

[1] In the results, shaded areas indicate 95% confidence intervals, computed via hierarchical bootstrapping.

self-normalized process, compared with a threshold function $\beta(t, \delta)$; see also appendix B for more details). Overall, we see in fig. 1a how ICPE is able to find a more efficient strategy compared to classical techniques. Interestingly, also I-DPT seems to achieve relatively small sample complexities. However, its tail distribution of τ is rather large compared to ICPE (fig. 1b) and the correctness is smaller than $1 - \delta$ for large values of K. Methods like TaS and TTPS achieve larger sample complexity, but also larger correctness values (fig. 1c). This is due to the fact that it is hard to define stopping rules. In fact, it is well known that current theoretically sound stopping rules are overly conservative Garivier and Kaufmann [2016]. Nonetheless, even using a less conservative rule such as $\beta(t, \delta) = \log((1 + \log(t))/\delta)$, which is what we use (and, yet, has not been proven to guarantee δ-correctness), is still conservative. The fact that ICPE can achieve the right value of confidence can help discover potential ways to define stopping rules. Lastly, in fig. 1a in black we show a complexity bound (proof in appendix A.1). While seemingly constant, it is actually *slowly* increasing in the number of arms.

Bandit Problems with Hidden Information. To evaluate ICPE in structured settings, we introduce bandit environments with latent informational dependencies, termed *magic actions*. In the single magic action case, the magic action a_m's reward is distributed according to $\mathcal{N}(\mu_{a_m}, \sigma_m^2)$, where $\sigma_m \in (0, 1)$ and $\mu_{a_m} := \phi(\arg\max_{a \neq a_m} \mu_a)$ encodes information about the optimal action's identity through an invertible mapping ϕ that is unknown to the learner. The index a_m is fixed, and the mean rewards of the other actions $(\mu_a)_{a \neq a_m}$ are sampled from $\mathcal{P}(\mathcal{M})$, a uniform distribution over models guaranteeing that a_m, as defined above, is not optimal (see appendices A.2 and C.1.2 for more details). Then, we define the reward distribution of the non-magic actions as $\mathcal{N}(\mu_a, (1 - \sigma_m)^2)$.

In our first experiment, we vary the standard deviation σ_m in $[0, 1]$. Thus, agents must balance sampling between informative and noisy actions based on varying uncertainty levels. We evaluate ICPE in a fixed-confidence setting with error rate $\delta = 0.1$. Figure 2a compares ICPE's sample complexity against a theoretical lower bound (see appendix A) and an informed baseline, denoted as I-IDS, which performs standard IDS leveraging ICPE's trained inference network I for exploiting the magic action (details in Appendix B). ICPE achieves sample complexities close to the theoretical

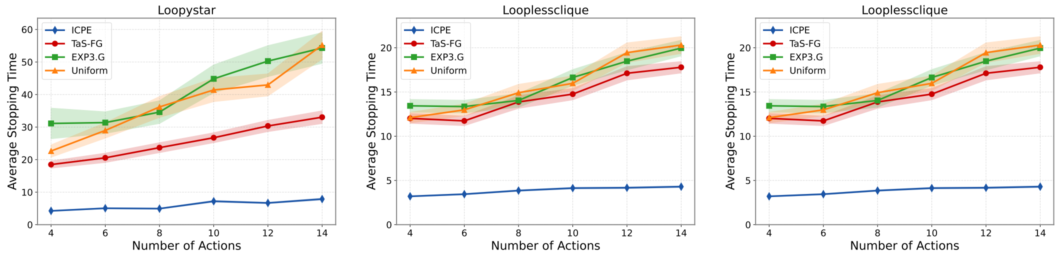


(a) (b)

Figure 2: (a) Single magic action: average stopping time and the theoretical lower bound across varying σ_m. (b) Magic chain: average stopping time between ICPE, I-IDS vs. number of magic actions.

bound across all tested noise levels, consistently outperforming I-IDS. To further challenge ICPE, we introduce a multi-layered "magic chain" bandit environments, where there is a sequence of n magic actions $\mathcal{A}_m := \{a_{i_1}, \ldots, a_{i_n}\} \subset \mathcal{A}$ such that $\mu_{a_{i_j}} = \phi(\mu_{a_{i_{j+1}}})$, and $\mu_{a_{i_n}} = \phi(\arg\max_{a \notin \mathcal{A}_m} \mu_a)$. The first index i_1 is known, and by following the chain, an agent can uncover the best action in n steps. However, the optimal sample complexity depends on the ratio of magic actions to non-magic arms. Varying the number of magic actions from 1 to 9 in a 10-actions environment, Figure 2b demonstrates ICPE's empirical performance, outperforming I-IDS.

Bandit Problems with Feedback Graphs. In bandit problems, playing action u yields its reward, while full-information settings reveal all rewards. Feedback graphs interpolate between these extremes: a directed graph $G \in [0, 1]^{K \times K}$ specifies that choosing u reveals the reward of v with probability $G_{u,v}$. Although feedback graphs have been extensively studied for regret minimization Mannor and Shamir [2011], their role in pure exploration remains underexplored Russo et al. [2025]; here we use them as structured testbeds, where latent relational and stochastic dependencies



(a) (b) (c)

Figure 3: Sample complexity comparison under the fixed-confidence setting for: (a) Loopy Star, (b) Loopless Clique, and (c) Ring graphs.

must be inferred to explore efficiently. Formally, upon playing u the learner observes for each $v \in [K]$:

$$r_v \sim \begin{cases} \mathcal{N}(\mu_v, \sigma^2), & \text{with probability } G_{u,v}, \\ \text{no observation}, & \text{otherwise.} \end{cases}$$

We tested ICPE on 3 different graph families with $\delta = 0.1$: the loopy star graph, the ring graph and the loopless clique Russo et al. [2025]. We set the optimal arm's mean to 1 and all others to 0.5 to facilitate faster convergence. We compared it to Uniform Sampling, EXP3.G, and Tas-FG using a shared stopping rule from Russo et al. [2025].

As shown in Figure 3, ICPE consistently achieves significantly lower sample complexity, suggesting that that ICPE is able to meta-learn and leverage the underlying structure of the graph.

3.2 Algorithm Discovery: Meta-Learning Binary Search

To test ICPE's ability to recover classical exploration algorithms, we evaluate whether it can autonomously meta-learn binary search. We define an action space of $\mathcal{A} = \{1, \dots, K\}$, where K is the upper bound on the possible location of the hidden target $H^\star \sim \mathcal{A}$. Pulling an arm above or below $H^\star$ yields a observation $x_t = -1$ or $x_t = +1$, respectively—providing directional feedback. We train ICPE under the fixed-confidence setting for $K = 2^3, \dots, 2^8$ using a target error rate of $\delta = 0.01$. In table 1 we report results on 100 held-out tasks per setting. ICPE consistently achieves perfect accuracy with worst-case stopping times that match the optimal $\log_2(K)$ rate, demonstrating that it has successfully rediscovered binary search purely from experience. While simple, this task illustrates ICPE's broader potential to learn efficient search strategies in domains where no hand-designed algorithm is available.

K (Actions)	Min Accuracy	Mean Stop Time	Max Stop Time	$\log_2 K$
8	1.00	2.13 ± 0.12	3	3
16	1.00	2.93 ± 0.12	4	4
32	1.00	3.71 ± 0.15	5	5
64	1.00	4.50 ± 0.21	6	6
128	1.00	5.49 ± 0.23	7	7
256	1.00	6.61 ± 0.26	8	8

Table 1: ICPE performance on the binary search task as the number of actions K increases.

4 Conclusions

In this work, we addressed the design of efficient pure-exploration strategies for the *active sequential hypothesis testing* problem, where an agent sequentially selects samples to rapidly identify the true hypothesis. While particularly relevant across different domains, it is difficult to design optimal strategies in the presence of hidden structure, and most of the existing optimal strategies are restricted to simple cases for unstructured multi-armed bandit problems. To overcome these limitations, we introduced ICPE, an in-context learning framework that leverages Transformers to learn exploration policies directly from experience. Our results demonstrate that ICPE is able to autonomously discovering task-specific adaptive exploration strategies. We believe our work makes a fundamental contribution to active testing, and in particular to the sub-field of best-arm identification. Future directions include several directions, including a theoretical analysis of ICPE's guarantees and scaling ICPE to larger, higher-dimensional problems.

References

Aymen Al Marjani, Aurélien Garivier, and Alexandre Proutiere. Navigating to the best policy in markov decision processes. *Advances in Neural Information Processing Systems*, 34:25852–25864, 2021.

Jean-Yves Audibert and Sébastien Bubeck. Best arm identification in multi-armed bandits. In *COLT-23th Conference on learning theory-2010*, pages 13–p, 2010.

Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47:235–256, 2002.

Peter Auer, Thomas Jaksch, and Ronald Ortner. Near-optimal regret bounds for reinforcement learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 21, 2008.

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.

Yoshua Bengio, Samy Bengio, and Jocelyn Cloutier. *Learning a synaptic learning rule*. Citeseer, 1990.

Scott M Berry, Bradley P Carlin, J Jack Lee, and Peter Muller. *Bayesian adaptive methods for clinical trials*. CRC press, 2010.

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.

Olivier Cappé, Aurélien Garivier, Odalric-Ambrym Maillard, Rémi Munos, and Gilles Stoltz. Kullback-leibler upper confidence bounds for optimal sequential allocation. *The Annals of Statistics*, pages 1516–1541, 2013.

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.

Herman Chernoff. Sequential design of experiments. *The Annals of Mathematical Statistics*, 30(3): 755–770, 1959.

Herman Chernoff. *Sequential design of experiments*. Springer, 1992.

David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 4:129–145, 1996.

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

Rémy Degenne and Wouter M Koolen. Pure exploration with multiple correct answers. *Advances in Neural Information Processing Systems*, 32, 2019.

Rémy Degenne, Wouter M Koolen, and Pierre Ménard. Non-asymptotic pure exploration by solving games. *Advances in Neural Information Processing Systems*, 32, 2019.

Rémy Degenne, Pierre Ménard, Xuedong Shang, and Michal Valko. Gamification of pure exploration for linear bandits, 2020.

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.

Scott Emmons, Benjamin Eysenbach, Ilya Kostrikov, and Sergey Levine. Rvs: What is essential for offline rl via supervised learning? *arXiv preprint arXiv:2112.10751*, 2021.

Kyra Gan, Su Jia, and Andrew Li. Greedy approximation algorithms for active sequential hypothesis testing. *Advances in Neural Information Processing Systems*, 34:5012–5024, 2021.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems*, 35:30583–30598, 2022.

Aurélien Garivier and Emilie Kaufmann. Optimal best arm identification with fixed confidence. In *Conference on Learning Theory*, pages 998–1027. PMLR, 2016.

Bashkar K Ghosh. A brief history of sequential analysis. *Handbook of sequential analysis*, 1, 1991.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

Aditya Gopalan, Shie Mannor, and Yishay Mansour. Thompson sampling for complex online problems. In *International conference on machine learning*, pages 100–108. PMLR, 2014.

Yassir Jedra and Alexandre Proutiere. Optimal best-arm identification in linear bandits. *Advances in Neural Information Processing Systems*, 33:10007–10017, 2020.

Marc Jourdan, Rémy Degenne, Dorian Baudry, Rianne de Heide, and Emilie Kaufmann. Top two algorithms revisited. *Advances in Neural Information Processing Systems*, 35:26791–26803, 2022.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

Emilie Kaufmann and Wouter M Koolen. Mixture martingales revisited with applications to sequential tests and confidence intervals. *Journal of Machine Learning Research*, 22(246):1–44, 2021.

Emilie Kaufmann, Nathaniel Korda, and Rémi Munos. Thompson sampling: An asymptotically optimal finite-time analysis. In *International conference on algorithmic learning theory*, pages 199–213. Springer, 2012.

Emilie Kaufmann, Olivier Cappé, and Aurélien Garivier. On the complexity of best-arm identification in multi-armed bandit models. *The Journal of Machine Learning Research*, 17(1):1–42, 2016.

Tomáš Kocák and Aurélien Garivier. Best arm identification in spectral bandits. *arXiv preprint arXiv:2005.09841*, 2020.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Aviral Kumar, Xue Bin Peng, and Sergey Levine. Reward-conditioned policies. *arXiv preprint arXiv:1912.13465*, 2019.

Tor Lattimore and Marcus Hutter. Pac bounds for discounted mdps. In *Algorithmic Learning Theory: 23rd International Conference, ALT 2012, Lyon, France, October 29-31, 2012. Proceedings 23*, pages 320–334. Springer, 2012.

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.

Dennis V Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 27(4):986–1005, 1956.

Shie Mannor and Ohad Shamir. From bandits to experts: On the value of side-observations. *Advances in neural information processing systems*, 24, 2011.

Aymen Al Marjani and Alexandre Proutiere. Adaptive sampling for best policy identification in markov decision processes. In *International Conference on Machine Learning*, pages 7459–7468. PMLR, 2021.

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.

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.

Subhojyoti Mukherjee, Ardhendu S Tripathy, and Robert Nowak. Chernoff sampling for active testing and extension to active regression. In *International Conference on Artificial Intelligence and Statistics*, pages 7384–7432. PMLR, 2022.

Kevin P Murphy. *Probabilistic machine learning: Advanced topics*. MIT press, 2023.

Mohammad Naghshvar and Tara Javidi. Active sequential hypothesis testing. *The Annals of Statistics*, 41(6):2703–2738, 2013.

Mohammad Naghshvar, Tara Javidi, and Kamalika Chaudhuri. Noisy bayesian active learning. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1626–1633. IEEE, 2012.

Ian Osband, Daniel Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via posterior sampling. *Advances in Neural Information Processing Systems (NeurIPS)*, 26, 2013.

Riccardo Poiani, Marc Jourdan, Emilie Kaufmann, and Rémy Degenne. Best-arm identification in unimodal bandits. *arXiv preprint arXiv:2411.01898*, 2024.

Paul Resnick and Hal R Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

Alessio Russo and Aldo Pacchiano. Adaptive exploration for multi-reward multi-policy evaluation. *arXiv preprint arXiv:2502.02516*, 2025.

Alessio Russo and Alexandre Proutiere. Model-free active exploration in reinforcement learning. *Advances in Neural Information Processing Systems*, 36:54740–54753, 2023a.

Alessio Russo and Alexandre Proutiere. On the sample complexity of representation learning in multi-task bandits with global and local structure. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 9658–9667, 2023b.

Alessio Russo and Filippo Vannella. Multi-reward best policy identification. *Advances in Neural Information Processing Systems*, 37:105583–105662, 2025.

Alessio Russo, Yichen Song, and Aldo Pacchiano. Pure exploration with feedback graphs. In *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research. PMLR, 2025.

Daniel Russo and Benjamin Van Roy. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243, 2014.

Daniel Russo and Benjamin Van Roy. Learning to optimize via information-directed sampling. *Operations Research*, 66(1):230–252, 2018.

Daniel J Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, Zheng Wen, et al. A tutorial on thompson sampling. *Foundations and Trends® in Machine Learning*, 11(1):1–96, 2018.

Tom Schaul and Jürgen Schmidhuber. Metalearning. *Scholarpedia*, 5(6):4650, 2010.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419): 1140–1144, 2018.

Marta Soare, Alessandro Lazaric, and Rémi Munos. Best-arm identification in linear bandits. *Advances in neural information processing systems*, 27, 2014.

Rupesh Kumar Srivastava, Pranav Shyam, Filipe Mutz, Wojciech Jaskowski, and Jürgen Schmidhuber. Training agents using upside-down reinforcement learning. *CoRR*, abs/1912.02877, 2019. URL http://arxiv.org/abs/1912.02877.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Nidhin Koshy Vaidhiyan, SP Arun, and Rajesh Sundaresan. Active sequential hypothesis testing with application to a visual search problem. In *2012 IEEE International Symposium on Information Theory Proceedings*, pages 2201–2205. IEEE, 2012.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Appendix

Contents

Appendix

A Theoretical Results

In this section we provide different theoretical results, mainly for the sample complexity of different MAB problems with structure.

A.1 Sample Complexity Bounds for MAB Problems with Fixed Minimum Gap

We now derive a sample complexity lower bound for a MAB problem where the minimum gap is known and the rewards are normally distributed.

Consider a MAB problem wit K arms $\{1, \dots, K\}$. To each arm a is associated a reward distribution $\nu_a = \mathcal{N}(\mu_a, \sigma^2)$ that is simply a Gaussian distribution. Let $a^\star(\mu) = \arg\max_a \mu_a$, and define the gap in arm a to be $\Delta_a(\mu) = \mu_{a^\star(\mu)} - \mu_a$. In the following, without loss of generality, we assume that $a^\star(\mu) = 1$.

We define the minimum gap to be $\Delta_{\min}(\mu) = \min_{a \neq a^\star(\mu)} \Delta_a(\mu)$. Assume now to know that $\Delta_{\min} \geq \Delta_0 > 0$.

Then, for any δ-correct algorithm, guaranteeing that at some stopping time τ the estimated optimal arm $\hat{a}_\tau$ is δ-correct, i.e., $\mathbb{P}_\mu(\hat{a}_\tau \neq a^\star(\mu)) \leq \delta$, we have the following result.

Theorem A.1. *Consider a model μ satisfying $\Delta_{\min} \geq \Delta_0 > 0$. Then, for any δ-probably correct method* Alg*, with $\delta \in (0, 1/2)$, we have that the optimal sample complexity is bounded as*

$$\frac{1}{\max\left(\Delta_0^2, \frac{1}{\sum_{a \neq 1} 1/\Delta_a^2}\right)} \leq \inf_{\tau:\mathrm{Alg} \textit{ is } \delta\textit{-correct}} \frac{\mathbb{E}_\mu[\tau]}{2\sigma^2 \mathrm{kl}(1-\delta, \delta)} \leq 2 \sum_a \frac{1}{(\Delta_a + \Delta_0)^2},$$

with $\Delta_1 = 0$ and $\mathrm{kl}(x, y) = x \log(x/y) + (1-x)\log((1-x)/(1-y))$. In particular, the solution $\omega_a \propto 1/(\Delta_a + \Delta_0)^2$ (up to a normalization constant) achieves the upper bound.

Proof. **Step 1: Log-likelihood ratio.** The initial part of the proof is rather standard, and follows the same argument used in the Best Arm Identification and Best Policy Identification literature Garivier and Kaufmann [2016], Russo and Vannella [2025].

Define the set of models

$$\mathcal{S} = \left\{\mu' \in \mathbb{R}^K : \Delta_{\min}(\mu') \geq \Delta_0\right\},$$

and the set of alternative models

$$\mathrm{Alt}(\mu) = \left\{\mu' \in \mathcal{S} : \arg\max_a \mu'_a \neq 1\right\}.$$

Take the expected log-likelihood ratio between μ and $\mu' \in \mathrm{Alt}(\mu)$ of the data observed up to τ $\Lambda_\tau = \log \frac{\mathrm{d}\mathbb{P}_\mu(A_1, R_1, \dots, A_\tau, R_\tau)}{\mathrm{d}\mathbb{P}_{\mu'}(A_1, R_1, \dots, A_\tau, R_\tau)}$, where A_t is the action taken in round t, and R_t is the reward observed upon selecting A_t. Then, we can write

$$\Lambda_t = \sum_a \sum_{n=1}^t \mathbf{1}_{\{A_n = a\}} \log \frac{f_a(R_n)}{f'_a(R_n)}$$

where f_a, f'_a, are, respectively, the reward density for action a in the two models μ, μ' with respect to the Lebesgue measure. Letting $N_a(t)$ denote the number of times action a has been selected up to round t, by an application of Wald's lemma the expected log-likelihood ratio can be shown to be

$$\mathbb{E}_\mu[\Lambda_\tau] = \sum_a \mathbb{E}_\mu[N_a(\tau)] \mathrm{KL}(\mu_a, \mu'_a)$$

where $\mathrm{KL}(\mu_a, \mu'_a)$ is the KL divergence between two Gaussian distributions $\mathcal{N}(\mu_a, \sigma)$ and $\mathcal{N}(\mu'_a, \sigma)$ (note that we have σ_1 instead of σ for $a = 1$).

We also know from the information processing inequality Kaufmann et al. [2016] that $\mathbb{E}_\mu[\Lambda_\tau] \geq \sup_{\mathcal{E} \in \mathcal{M}_\tau} \mathrm{kl}(\mathbb{P}_\mu(\mathcal{E}), \mathbb{P}_{\mu'}(\mathcal{E}))$, where $\mathcal{M}_t = \sigma(A_1, R_1, \dots, A_t, R_t)$. We use the fact that the algorithm is δ-correct: by choosing $\mathcal{E} = \{\hat{a}_\tau = a^\star\}$ we obtain that $\mathbb{E}_\mu[\Lambda_\tau] \geq \mathrm{kl}(1-\delta, \delta)$, since

$\mathbb{P}_\mu(\mathcal{E}) \geq 1-\delta$ and $\mathbb{P}_{\mu'}(\mathcal{E}) = 1 - \mathbb{P}_{\mu'}(\hat{a}_\tau \neq a^\star) \leq 1 - \mathbb{P}_{\mu'}(\hat{a}_\tau = \arg\max_a \mu'_a) \leq \delta$ (we also used the monotonicity properties of the Bernoulli KL divergence). Hence

$$\sum_a \mathbb{E}_\mu[N_a(\tau)]\mathrm{KL}(\mu_a, \mu'_a) \geq \mathrm{kl}(1-\delta, \delta).$$

Letting $\omega_a = \mathbb{E}_\mu[N_a(\tau)]/\mathbb{E}_\mu[\tau]$, we have that

$$\mathbb{E}_\mu[\tau] \sum_a \omega_a \mathrm{KL}(\mu_a, \mu'_a) \geq \mathrm{kl}(1-\delta, \delta).$$

Lastly, optimizing over $\mu' \in \mathrm{Alt}(\mu)$ and $\omega \in \Delta(K)$ yields the bound:

$$\mathbb{E}_\mu[\tau] \geq T^\star(\mu)\mathrm{kl}(1-\delta, \delta),$$

where $T^\star(\mu)$ is defined as

$$(T^\star(\mu))^{-1} = \sup_{\omega \in \Delta(K)} \inf_{\mu' \in \mathrm{Alt}(\mu)} \sum_a \omega_a \mathrm{KL}(\mu_a, \mu'_a).$$

Step 2: Optimization over the set of alternative models. We now face the problem of optimizing over the set of alternative models.

Defining $\mathrm{Alt}_a = \left\{\mu' \in \mathbb{R}^K : \mu'_a - \mu'_b \geq \Delta_0 \ \forall b \neq a\right\}$, the set of alternative models can be decomposed as

$$\begin{aligned}\mathrm{Alt}(\mu) &= \left\{\mu' \in \mathbb{R}^K : \arg\max_a \mu'_a \neq 1,\ \Delta_{\min}(\mu') \geq \Delta_0\right\}, \\ &= \cup_{a\neq 1} \mathrm{Alt}_a.\end{aligned}$$

Hence, the optimization problem over the alternative models becomes

$$\inf_{\mu' \in \mathrm{Alt}(\mu)} \sum_a \omega_a \mathrm{KL}(\mu_a, \mu'_a) = \min_{\bar{a}\neq 1} \inf_{\mu' \in \mathrm{Alt}_{\bar{a}}} \sum_a \omega_a \frac{(\mu_a - \mu'_a)^2}{2\sigma^2}.$$

The inner infimum over μ' can then be written as

$$\begin{aligned} P^\star_{\bar{a}}(\omega) \coloneqq \inf_{\mu' \in \mathbb{R}^K} \quad & \sum_a \omega_a \frac{(\mu_a - \mu'_a)^2}{2\sigma^2}. \\ \text{s.t.} \quad & \mu'_{\bar{a}} - \mu'_b \geq \Delta_0 \quad \forall b \neq \bar{a}. \end{aligned} \tag{5}$$

While the problem is clearly convex, it does not yield an immediate closed form solution.

To that aim, we try to derive a lower bound and an upper bound of the value of this minimization problem.

Step 3: Upper bound on $P^\star_{\bar{a}}$. Note that an upper bound on $\min_{\bar{a}\neq 1} P^\star_{\bar{a}}(\omega)$ can be found by finding a feasible solution μ'. Consider then the solution $\mu'_1 = \mu_1 - \Delta$, $\mu'_{\bar{a}} = \mu_1$ and $\mu'_b = \mu_b$ for all other arms. Clearly We have that $\mu'_{\bar{a}} - \mu'_b \geq \Delta_0$ for all $b \neq \bar{a}$. Hence, we obtain

$$\min_{\bar{a}\neq 1} P^\star_{\bar{a}}(\omega) \leq \omega_1 \frac{\Delta_0^2}{2\sigma^2} + \min_{\bar{a}\neq 1} \omega_{\bar{a}} \frac{\Delta_{\bar{a}}^2}{2\sigma^2}.$$

At this point, one can easily note that if $\frac{\Delta_0^2}{2\sigma^2} \geq \frac{1}{2\sigma^2 \sum_{a\neq 1} \frac{1}{\Delta_a^2}}$, then $\sup_{\omega\in\Delta(K)} \min_{\bar{a}\neq 1} P^\star_{\bar{a}}(\omega) \leq \frac{\Delta_0^2}{2\sigma^2}$. This corresponds to the case where all the mass is given to $\omega_1 = 1$. Otherwise, the solution is to set $\omega_1 = 0$ and $\omega_a = \frac{1/\Delta_a^2}{\sum_b 1/\Delta_b^2}$ for $a \neq 1$.

Hence, we conclude that

$$(T^\star(\mu))^{-1} = \sup_{\omega\in\Delta(K)} \min_{\bar{a}\neq 1} P^\star_{\bar{a}}(\omega) \leq \frac{1}{2\sigma^2} \max\left(\Delta_0^2, \frac{1}{\sum_{a\neq 1} 1/\Delta_a^2}\right).$$

Step 4: Lower bound on $P^\star_{\bar a}$. For the lower bound, note that we can relax the constraint to only consider $\mu'_{\bar a} - \mu'_1 \geq \Delta_0$. This relaxation enlarges the feasible set, and thus the infimum of this new problem lower bounds $P^\star_{\bar a}(\omega)$.

By doing so, since the other arms are not constrained, by convexity of the KL divergence at the infimum we have $\mu'_b = \mu_b$ for all $b \notin \{1, \bar a\}$. Therefore

$$P^\star_{\bar a}(\omega) \geq \inf_{\mu' : \mu'_{\bar a} - \mu'_1 \geq \Delta_0} \sum_a \omega_a \frac{(\mu_a - \mu'_a)^2}{2\sigma^2} = \inf_{\mu' : \mu'_{\bar a} - \mu'_1 \geq \Delta_0} \omega_1 \frac{(\mu_1 - \mu'_1)^2}{2\sigma^2} + \omega_{\bar a} \frac{(\mu_{\bar a} - \mu'_{\bar a})^2}{2\sigma^2}.$$

Solving the KKT conditions we find the equivalent conditions $\mu'_{\bar a} = \mu'_1 + \Delta_0$ and

$$\omega_1(\mu_1 - \mu'_1) + \omega_{\bar a}(\mu_{\bar a} - \mu'_1 - \Delta_0) = 0 \Rightarrow \mu'_1 = \frac{\omega_1 \mu_1 + \omega_{\bar a}\mu_{\bar a} - \omega_{\bar a}\Delta_0}{\omega_1 + \omega_{\bar a}}.$$

Therefore

$$\mu'_{\bar a} = \frac{\omega_1 \mu_1 + \omega_{\bar a}\mu_{\bar a} - \omega_{\bar a}\Delta_0}{\omega_1 + \omega_{\bar a}} + \Delta_0 = \frac{\omega_1 \mu_1 + \omega_{\bar a}\mu_{\bar a} + \omega_1\Delta_0}{\omega_1 + \omega_{\bar a}}.$$

Plugging these solutions back in the value of the problem, we obtain

$$\begin{aligned}
P^\star_{\bar a}(\omega) &\geq \frac{\omega_1 \omega_{\bar a}^2}{(\omega_1 + \omega_{\bar a})^2} \frac{(\mu_1 - \mu_{\bar a} + \Delta_0)^2}{2\sigma^2} + \frac{\omega_{\bar a}\omega_1^2}{(\omega_1 + \omega_{\bar a})^2} \frac{(\mu_{\bar a} - \mu_1 - \Delta_0)^2}{2\sigma^2}, \\
&= \frac{\omega_1 \omega_{\bar a}}{\omega_1 + \omega_{\bar a}} \frac{(\mu_1 - \mu_{\bar a} + \Delta_0)^2}{2\sigma^2}, \\
&= \frac{\omega_1 \omega_{\bar a}}{\omega_1 + \omega_{\bar a}} \frac{(\Delta_{\bar a} + \Delta_0)^2}{2\sigma^2}.
\end{aligned}$$

Let $\theta_a = \Delta_a + \Delta_0$, with $\theta_1 = \Delta_0$. We plug in a feasible solution $\omega_a = \frac{1/\theta_a^2}{\sum_b 1/\theta_b^2}$, yielding

$$\begin{aligned}
(T^\star(\mu))^{-1} = \sup_{\omega \in \Delta(K)} \min_{\bar a \neq 1} P^\star_{\bar a}(\omega) &\geq \min_{\bar a \neq 1} \frac{1/(\theta_1\theta_{\bar a})^2}{\sum_b 1/\theta_b^2 (1/\theta_1^2 + 1/\theta_{\bar a}^2)} \frac{\theta_{\bar a}^2}{2\sigma^2}, \\
&= \min_{\bar a \neq 1} \frac{1}{\sum_b 1/\theta_b^2 (1 + \theta_1^2/\theta_{\bar a}^2)} \frac{1}{2\sigma^2}, \\
&= \frac{1}{2\sigma^2 \sum_b 1/\theta_b^2} \min_{\bar a \neq 1} \frac{1}{1 + \theta_1^2/\theta_{\bar a}^2}, \\
&\geq \frac{1}{2\sigma^2 \sum_b 1/\theta_b^2} \frac{1}{1 + \theta_1^2/\Delta_0^2}, \\
&= \frac{1}{4\sigma^2 \sum_b 1/\theta_b^2}.
\end{aligned}$$

□

A.2 Sample Complexity Lower Bound for the Magic Action MAB Problem

We now consider a special class of models that embeds information about the optimal arm in the mean reward of some of the arms. Let $\phi : \mathbb{R} \to \mathbb{R}$ be a strictly decreasing function over $\{2, \ldots, K\}$[2].

Particularly, we make the following assumptions:

1. We consider mean rewards μ satisfying $\mu_1 = \phi(\arg\max_{a \neq 1} \mu_a)$, and $\mu^\star = \max_a \mu_a > \phi(2)$. Arm 1 is called "magic action", and with this assumption we are guaranteed that the magic arm is not optimal, since

$$\mu_1 \frac{1}{\max_a \mu_a} = \phi(\arg\max_{a \neq 1} \mu_a) \frac{1}{\max_a \mu_a} \leq \phi(2) \frac{1}{\max_a \mu_a} < 1 \Rightarrow \max_a \mu_a > \mu_1.$$

2. The rewards are normally distributed, with a fixed known standard deviation σ_1 for the magic arm, and fixed standard deviation σ for all the other arms.

[2] One could also consider strictly increasing functions.

Hence, define the set of models

$$\mathcal{S} = \left\{ \mu \in \mathbb{R}^K : \mu_1 = \phi(\arg\max_{a \neq 1} \mu_a), \max_a \mu_a > \phi(2) \right\},$$

and the set of alternative models

$$\mathrm{Alt}(\mu) = \left\{ \mu' \in \mathcal{S} : \arg\max_a \mu'_a \neq a^\star \right\},$$

where $a^\star = \arg\max_a \mu_a$.

Then, for any δ-correct algorithm, guaranteeing that at some stopping time τ the estimated optimal arm $\hat{a}_\tau$ is δ-correct, i.e., $\mathbb{P}_\mu(\hat{a}_\tau \neq a^\star) \leq \delta$, we have the following result.

Theorem A.2. *For any δ-correct algorithm, the sample complexity lower bound on the magic action problem is*

$$\mathbb{E}_\mu[\tau] \geq T^\star(\mu)\mathrm{kl}(1-\delta, \delta), \tag{6}$$

where $\mathrm{kl}(x,y) = x\log(x/y) + (1-x)\log((1-x)/(1-y))$ and $T^\star(\mu)$ is the characteristic time of μ, defined as

$$(T^\star(\mu))^{-1} = \max_{\omega \in \Delta(K)} \min_{a \neq 1, a^\star} \omega_1 \frac{(\phi(a^\star) - \phi(a))^2}{2\sigma_1^2} + \sum_{b \in \mathcal{K}_a(\omega)} \omega_b \frac{(\mu_b - m(\omega; \mathcal{K}_a(\omega))^2}{2\sigma^2}, \tag{7}$$

where $m(\omega;\mathcal{C}) = \frac{\sum_{a\in\mathcal{C}} \omega_a \mu_a}{\sum_{a\in\mathcal{C}} \omega_a}$ and the set $\mathcal{K}_a(\omega)$ is defined as

$$\mathcal{K}_a(\omega) = \{a\} \cup \{b \in \{2, \dots, K\} : \mu_b \geq m(\omega; \mathcal{C}_b \cup \{a\}) \text{ and } \mu_b \geq \phi(2)\}.$$

with $\mathcal{C}_x = \{b \in \{2,\dots,K\} : \mu_b \geq \mu_x\}$ for $x \in [K]$.

Proof. **Step 1: Log-likelihood ratio.** The initial part of the proof is rather standard, and follows the same argument used in the Best Arm Identification and Best Policy Identification literature Garivier and Kaufmann [2016], Russo and Vannella [2025].

Take the expected log-likelihood ratio between μ and $\mu' \in \mathrm{Alt}(\mu)$ of the data observed up to τ $\Lambda_\tau = \log \frac{\mathrm{d}\mathbb{P}_\mu(A_1,R_1,\dots,A_\tau,R_\tau)}{\mathrm{d}\mathbb{P}_{\mu'}(A_1,R_1,\dots,A_\tau,R_\tau)}$, where A_t is the action taken in round t, and R_t is the reward observed upon selecting A_t. Then, we can write

$$\Lambda_t = \sum_a \sum_{n=1}^t \mathbf{1}_{\{A_n = a\}} \log \frac{f_a(R_n)}{f'_a(R_n)}$$

where f_a, f'_a, are, respectively, the reward density for action a in the two models μ, μ' with respect to the Lebesgue measure. Letting $N_a(t)$ denote the number of times action a has been selected up to round t, by an application of Wald's lemma the expected log-likelihood ratio can be shown to be

$$\mathbb{E}_\mu[\Lambda_\tau] = \sum_a \mathbb{E}_\mu[N_a(\tau)]\mathrm{KL}(\mu_a, \mu'_a)$$

where $\mathrm{KL}(\mu_a, \mu'_a)$ is the KL divergence between two Gaussian distributions $\mathcal{N}(\mu_a, \sigma)$ and $\mathcal{N}(\mu'_a, \sigma)$ (note that we have σ_1 instead of σ for $a = 1$).

We also know from the information processing inequality Kaufmann et al. [2016] that $\mathbb{E}_\mu[\Lambda_\tau] \geq \sup_{\mathcal{E}\in\mathcal{M}_\tau} \mathrm{kl}(\mathbb{P}_\mu(\mathcal{E}), \mathbb{P}_{\mu'}(\mathcal{E}))$, where $\mathcal{M}_t = \sigma(A_1, R_1, \dots, A_t, R_t)$. We use the fact that the algorithm is δ-correct: by choosing $\mathcal{E} = \{\hat{a}_\tau = a^\star\}$ we obtain that $\mathbb{E}_\mu[\Lambda_\tau] \geq \mathrm{kl}(1-\delta, \delta)$, since $\mathbb{P}_\mu(\mathcal{E}) \geq 1-\delta$ and $\mathbb{P}_{\mu'}(\mathcal{E}) = 1 - \mathbb{P}_{\mu'}(\hat{a}_\tau \neq a^\star) \leq 1 - \mathbb{P}_{\mu'}(\hat{a}_\tau = \arg\max_a \mu'_a) \leq \delta$ (we also used the monotonicity properties of the Bernoulli KL divergence). Hence

$$\sum_a \mathbb{E}_\mu[N_a(\tau)]\mathrm{KL}(\mu_a, \mu'_a) \geq \mathrm{kl}(1-\delta, \delta).$$

Letting $\omega_a = \mathbb{E}_\mu[N_a(\tau)]/\mathbb{E}_\mu[\tau]$, we have that

$$\mathbb{E}_\mu[\tau] \sum_a \omega_a \mathrm{KL}(\mu_a, \mu'_a) \geq \mathrm{kl}(1-\delta, \delta).$$

Lastly, optimizing over $\mu' \in \mathrm{Alt}(\mu)$ and $\omega \in \Delta(K)$ yields the bound:

$$\mathbb{E}_\mu[\tau] \geq T^\star(\mu)\mathrm{kl}(1-\delta, \delta),$$

where $T^\star(\mu)$ is defined as

$$(T^\star(\mu))^{-1} = \sup_{\omega \in \Delta(K)} \inf_{\mu' \in \mathrm{Alt}(\mu)} \sum_a \omega_a \mathrm{KL}(\mu_a, \mu_a').$$

Step 2: Optimization over the set of alternative models. We now face the problem of optimizing over the set of alternative models. First, we observe that $\mathcal{S} = \cup_{a \neq a^\star}\{\mu : \mu_1 = \phi(a), \mu_a > \phi(2)\}$. Therefore, we can write

$$\mathrm{Alt}(\mu) = \cup_{a \notin \{1, a^\star\}} \left\{\mu' : \mu_1' = \phi(a), \mu_a' > \max(\phi(2), \mu_b') \; \forall b \neq a\right\}.$$

Hence, for a fixed $a \notin \{1, a^\star\}$, the inner infimum becomes

$$\begin{aligned} \inf_{\mu' \in \mathbb{R}^K} \quad & \omega_1 \frac{(\phi(a^\star) - \phi(a))^2}{2\sigma_1^2} + \sum_{a \neq 1} \omega_a \frac{(\mu_a - \mu_a')^2}{2\sigma^2} \\ \text{s.t.} \quad & \mu_a' \geq \max\left(\phi(2), \mu_b'\right) \quad \forall b, \\ & \mu_1' = \phi(a). \end{aligned} \tag{8}$$

To solve it, we construct the following Lagrangian

$$\ell(\mu', \theta) = \omega_1 \frac{(\phi(a^\star) - \phi(a))^2}{2\sigma_1^2} + \sum_{b \neq 1} \omega_b \frac{(\mu_b - \mu_b')^2}{2\sigma^2} + \sum_b \theta_b \left(\max\left(\phi(2), \mu_b'\right) - \mu_a'\right),$$

where $\theta \in \mathbb{R}_+^K$ is the multiplier vector. From the KKT conditions we already know that $\theta_1 = 0, \theta_a = 0$ and $\theta_b = 0$ if $\mu_b' \leq \phi(2)$, with $b \in \{2, \dots, K\}$. In particular, we also know that either we have $\mu_b' = \mu_a'$ or $\mu_b' = \mu_b$. Therefore, for $\mu_b \leq \phi(2)$ the solution is $\mu_b' = \mu_b$, while for $\mu_b > \phi(2)$ the solution depends also on ω.

To fix the ideas, let $\mathcal{K}$ be the set of arms for which $\mu_b' = \mu_a'$ at the optimal solution. Such set must necessarily include arm a. Then, note that

$$\frac{\partial \ell}{\partial \mu_a'} = \omega_a \frac{\mu_a' - \mu_a}{\sigma^2} - \sum_{b \in [K]} \theta_b = 0.$$

and

$$\frac{\partial \ell}{\partial \mu_b'} = \omega_b \frac{\mu_b' - \mu_b}{\sigma^2} + \theta_b = 0 \quad \text{ for } b \neq (1, a).$$

Then, using the observations derived above, we conclude that

$$\mu_a' = \frac{\sum_{b \in \mathcal{K}} \omega_b \mu_b}{\sum_{b \in \mathcal{K}} \omega_b},$$

with $\mu_b' = \mu_a'$ if $b \in \mathcal{K}$, and $\mu_b' = \mu_b$ otherwise. However, how do we compute such set $\mathcal{K}$?

First, $\mathcal{K}$ includes arm a. However, in general we have $\mathcal{K} \neq \{a\}$: if that were not true we would have $\mu_a' = \mu_a$ and $\mu_b' = \mu_b$ for the other arms – but if any μ_b is greater than μ_a, then a is not optimal, which is a contradiction. Therefore, also arm $a^\star$ is included in $\mathcal{K}$, since any convex combination of $\{\mu_a\}$ is necessarily smaller than $\mu_{a^\star}$. We apply this argument repeatedly for every arm b to obtain $\mathcal{K}$.

Hence, for some set $\mathcal{C} \subseteq [K]$ define the average reward

$$m(\omega; \mathcal{C}) = \frac{\sum_{a \in \mathcal{C}} \omega_a \mu_a}{\sum_{a \in \mathcal{C}} \omega_a},$$

and the set $\mathcal{C}_x = \{a\} \cup \{b \in \{2, \dots, K\} : \mu_b \geq \mu_x\}$ for $x \in [K]$. Then,

$$\mathcal{K} \coloneqq \mathcal{K}(\omega) = \{a\} \cup \left\{b \in \{2, \dots, K\} : \mu_b \geq m(\omega; \mathcal{C}_b) \text{ and } \mu_b \geq \phi(2)\right\}.$$

In other words, $\mathcal{K}$ is the set of *confusing arms* for which the mean reward in the alternative model changes. An arm b is *confusing* if the average reward m, taking into account b, is smaller than μ_b. If this holds for b, then it must also hold all the arms b' such that $\mu_{b'} \geq \mu_b$. $\square$

Finally, to get a better intuition of the main result, we can look at the 3-arms case: it is optimal to only sample the magic arm iff $|\phi(a^\star) - \phi(a)| > \frac{\sigma_1(\mu_{a^\star} - \mu_a)}{2\sigma}$.

Lemma A.3. *With $K = 3$ we have that $\omega_1 = 1$ if and only if*

$$|\phi(a^\star) - \phi(a)| > \frac{\sigma_1(\mu_{a^\star} - \mu_a)}{2\sigma},$$

and $\omega_1 = 0$ if the reverse inequality holds.

Proof. With 3 arms, from the proof of the theorem we know that $\mathcal{K}_a(\omega) = \{a, a^\star\}$ for all ω. Letting $m(\omega) = \frac{\omega_a\mu_a + \omega_{a^\star}\mu_{a^\star}}{\omega_a + \omega_{a^\star}}$, we obtain

$$(T^\star(\mu))^{-1} = \max_{\omega \in \Delta(3)} \omega_1 \frac{(\phi(a^\star) - \phi(a))^2}{2\sigma_1^2} + \frac{\omega_a(\mu_a - m(\omega))^2 + \omega_{a^\star}(\mu_{a^\star} - m(\omega))^2}{2\sigma^2}.$$

Clearly the solution is $\omega_1 = 1$ as long as

$$\frac{(\phi(a^\star) - \phi(a))^2}{2\sigma_1^2} > \max_{\omega : \omega_a + \omega_{a^\star} = 1} \frac{\omega_a(\mu_a - m(\omega))^2 + \omega_{a^\star}(\mu_{a^\star} - m(\omega))^2}{2\sigma^2}.$$

To see why this is the case, let $f_1 = \frac{(\phi(a^\star) - \phi(a))^2}{2\sigma_1^2}$, $f_2(\omega_a, \omega_{a^\star}) = \frac{\omega_a(\mu_a - m(\omega))^2}{2\sigma^2}$ and $f_3(\omega_a, \omega_{a^\star}) = \frac{\omega_{a^\star}(\mu_{a^\star} - m(\omega))^2}{2\sigma^2}$. Then, we can write

$$\omega_1 f_1 + \omega_a f_2(\omega_a, \omega_{a^\star}) + \omega_{a^\star} f_3(\omega_a, \omega_{a^\star}) = \omega_1 f_1 + (1 - \omega_1)\left[\frac{\omega_a f_2}{1 - \omega_1} + \frac{\omega_{a^\star} f_3}{1 - \omega_1}\right].$$

Being a convex combination, this last term can be upper bounded as

$$\omega_1 f_1 + \omega_a f_2(\omega_a, \omega_{a^\star}) + \omega_{a^\star} f_3(\omega_a, \omega_{a^\star}) \le \max\left(f_1, \frac{\omega_a f_2}{1 - \omega_1} + \frac{\omega_{a^\star} f_3}{1 - \omega_1}\right).$$

Now, note that also the term inside the bracket is a convex combination. Threfore, let $\omega_a = (1 - \omega_1)\alpha$ and $\omega_{a^\star} = (1 - \omega_1)(1 - \alpha)$ for some $\alpha \in [0, 1]$. We have that

$$m(\omega) = \frac{(1 - \omega_1)\alpha\mu_a + (1 - \omega_1)(1 - \alpha)\mu_{a^\star}}{1 - \omega_1} = \alpha\mu_a + (1 - \alpha)\mu_{a^\star}.$$

Hence, we obtain that

$$\begin{aligned}
\frac{\omega_a(\mu_a - m(\omega))^2 + \omega_{a^\star}(\mu_{a^\star} - m(\omega))^2}{2(1 - \omega_1)\sigma^2} &= \frac{\omega_a f_2 + \omega_{a^\star} f_3}{1 - \omega_1}, \\
&= \frac{\alpha(1 - \alpha)^2(\mu_a - \mu_{a^\star})^2 + (1 - \alpha)\alpha^2(\mu_{a^\star} - \mu_a)^2}{2\sigma^2}, \\
&= \alpha(1 - \alpha)\frac{(1 - \alpha)(\mu_a - \mu_{a^\star})^2 + \alpha(\mu_{a^\star} - \mu_a)^2}{2\sigma^2}, \\
&= \alpha(1 - \alpha)\frac{(\mu_a - \mu_{a^\star})^2}{2\sigma^2}.
\end{aligned}$$

Since this last term is maximized for $\alpha = 1/2$, we obtain

$$\omega_1 f_1 + \omega_a f_2(\omega_a, \omega_{a^\star}) + \omega_{a^\star} f_3(\omega_a, \omega_{a^\star}) \le \max\left(f_1, \frac{(\mu_a - \mu_{a^\star})^2}{8\sigma^2}\right).$$

Since f_1 is attained for $\omega_1 = 1$, we have that as long as $f_1 > \frac{(\mu_a - \mu_{a^\star})^2}{8\sigma^2}$, then the solution is $\omega_1 = 1$.

On the other hand, if $\frac{(\mu_a - \mu_{a^\star})^2}{8\sigma^2} > f_1$, then we can set $\omega_a = (1 - \omega_1)/2$ and $\omega_{a^\star} = (1 - \omega_1)/2$, leading to

$$\omega_1 f_1 + \omega_a f_2(\omega_a, \omega_{a^\star}) + \omega_{a^\star} f_3(\omega_a, \omega_{a^\star}) = \omega_1 f_1 + (1 - \omega_1)\frac{(\mu_a - \mu_{a^\star})^2}{8\sigma^2},$$

which is maximized at $\omega_1 = 0$. $\square$

A.3 Sample Complexity Bound for the Multiple Magic Actions MAB Problem

We now extend our analysis to the case where multiple magic actions can be present in the environment. In contrast to the single magic action setting, here a *chain* of magic actions sequentially reveals information about the location of the optimal action. Without loss of generality, assume that the first n arms (with indices $1, \dots, n$) are the magic actions, and the remaining $K - n$ arms are non–magic. The chain structure is such that pulling magic arm j (with $1 \le j < n$) yields information about only the location of the next magic arm $j+1$, while pulling the final magic action (arm n) reveals the identity of the optimal action. As before, we assume that the magic actions are informational only and are never optimal.

To formalize the model, let $\phi : \{1, \dots, n\} \to \mathbb{R}$ be a strictly decreasing function. We assume that the magic actions have fixed means given by

$$\mu_j = \begin{cases} \phi(j+1), & \text{if } j = 1, \dots, n-1, \\ \phi\Big(\arg\max_{a \notin \{1,\dots,n\}} \mu_a\Big), & \text{if } j = n. \end{cases}$$

and that the non–magic arms satisfy

$$\mu^\star = \max_{a \notin \{1,\dots,n\}} \mu_a > \phi(n).$$

Thus, the optimal arm lies among the non–magic actions. Considering the noiseless case where the rewards of all actions are fixed and the case where we can identify if an action is magic once revealed, we have the following result.

Theorem A.4. *Consider noiseless magic bandit problem with K arms and n magic actions. The optimal sample complexity is upper bounded as*

$$\inf_{\mathrm{Alg}} \mathbb{E}_{\mathrm{Alg}}[\tau] \le \min\left(n, \sum_{j=1}^{K-n} \left(\prod_{i=j+1}^{K-n} \frac{i}{n-1+i}\right)\left(1 + \frac{n-1}{n-1+j} \min\left(\frac{n-2}{2}, \frac{j(n-1+j)}{j+1}\right)\right)\right).$$

Proof. In the proof we derive a sample complexity bound for a policy based on some insights. We use the assumption that upon observing a reward from a magic arm, the learner can almost surely identify that the pulled arm is a magic arm.

Let us define the state (m, r, l), where m denotes the number of remaining unrevealed magic actions ($m_0 = n - 1$), r denotes the number of remaining unrevealed non-magic actions ($r_0 = K - n$), and l is the binary indicator with value 1 if we have revealed any hidden magic action and 0 otherwise.

Before any observation the learner has no information about which $n-1$ indices among $\{2, \dots, K\}$ form the chain of intermediate magic arms. Hence, one can argue that at the first time-step is optimal to sample uniformly at random an action in $\{2, \dots, K\}$.

Upon observing a magic action, and thus we are in state $(m, r, 1)$, we consider the following candidate policies: (1) start from the revealed action and follow the chain, or (2) keep sampling unrevealed actions uniformly at random until all non-magic actions are revealed. As previously discussed, starting the chain from the initial magic action would be suboptimal and we do not consider it.

Upon drawing a hidden magic arm, let its chain index be $j \in \{2, \dots, n\}$ (which is uniformly distributed). The remaining cost to complete the chain is $n - j$, and hence its expected value is

$$\mathbb{E}[n-j] = \frac{n-2}{2}.$$

Therefore, the total expected cost for strategy (1) is

$$T_1 = \frac{n-2}{2}.$$

We can additionally compute the expected cost for strategy (2) as follows: if the last non-magic action is revealed at step i, then among the first $i-1$ draws there are exactly $r-1$ non-magic arms. Since there are $\binom{m+r}{r}$ ways to place all r non-magic arms $m+r$ slots, we have

$$\begin{aligned}
T_2 &= \mathbb{E}[\text{Draws until all non-magic revealed}] \\
&= \sum_{i=r}^{m+r} i \cdot \mathbb{P}[\text{Last non-magic revealed at step } i] \\
&= \sum_{i=r}^{m+r} i \cdot \frac{\binom{i-1}{r-1}}{\binom{m+r}{r}} \\
&= \frac{r! \cdot m!}{(m+r)!} \sum_{i=r}^{m+r} i \binom{i-1}{r-1} \\
&= \frac{r! \cdot m!}{(m+r)!} \sum_{i=r}^{m+r} \frac{i!}{(r-1)!(i-r)!} \\
&= \frac{r! \cdot m!}{(m+r)!} \sum_{i=r}^{m+r} r \binom{i}{r} \\
&= \frac{r \cdot r! \cdot m!}{(m+r)!} \binom{m+r+1}{r+1} \\
&= \frac{r \cdot r! \cdot m!}{(m+r)!} \cdot \frac{(m+r+1) \cdot (m+r)!}{(r+1) \cdot r! \cdot m!} \\
&= \frac{r(m+r+1)}{r+1}
\end{aligned}$$

Finally, we define a policy in $(m, r, 1)$ as the one choosing between strategy 1 and strategy 2, depending on which one achieves the minimum cost. Hence, the complexity of this policy is

$$V(m, r, 1) = \min\left(\frac{n-2}{2}, \frac{r(m+r+1)}{r+1}\right).$$

Now, before finding a magic arm, consider a policy that uniformly samples between the non-revealed arms. Therefore, in $(m, r, 0)$ we can achieve a complexity of $1 + \frac{m}{m+r} V(m-1, r, 1) + \frac{r}{m+r} V(m, r-1, 0)$. Since we can always achieve a sample complexity of n, we can find a policy with the following complexity:

$$\begin{aligned}
V(m, r, 0) &= \min\left(n, 1 + \frac{m}{m+r} V(m-1, r, 1) + \frac{r}{m+r} V(m, r-1, 0)\right) \\
&= \min\left(n, 1 + \frac{m}{m+r} \min\left(\frac{n-2}{2}, \frac{r(m+r)}{r+1}\right) + \frac{r}{m+r} V(m, r-1, 0)\right)
\end{aligned}$$

Given we always start with $n-1$ hidden magic actions we can define a recursion in terms of just the variable r as follows:

$$V(r) = 1 + \frac{n-1}{n-1+r} T(r) + \frac{r}{n-1+r} V(r-1),$$

where $T(r) = \min\left(\frac{n-2}{2}, \frac{r(n-1+r)}{r+1}\right)$. Letting $A(r) = \frac{r}{n-1+r}$ and $B(r) = 1 + \frac{n-1}{n-1+r} T(r)$, we can write

$$V(r) = B(r) + A(r) V(r-1),$$

Clearly $V(0) = 0$ since if all non-magic actions are revealed, then we know the optimal action deterministically. Unrolling the recursion we get

$$
\begin{aligned}
&V(1) = B(1), \\
&V(2) = B(2) + A(2)B(1), \\
&V(3) = B(3) + A(3)B(2) + A(3)A(2)B(1), \\
&\dots \\
&V(r) = \sum_{j=1}^{r} \left(\prod_{i=j+1}^{r} A(i) \right) B(j).
\end{aligned}
$$

Substituting back in our expression, we get

$$V(r) = \sum_{j=1}^{r} \left(\prod_{i=j+1}^{r} \frac{i}{n-1+i} \right) \left(1 + \frac{n-1}{n-1+j} T(j) \right).$$

Thus starting at $r = K - n$ we get the following expression:

$$\min \left(n, \sum_{j=1}^{K-n} \left(\prod_{i=j+1}^{K-n} \frac{i}{n-1+i} \right) \left(1 + \frac{n-1}{n-1+j} \min \left(\frac{n-2}{2}, \frac{j(n-1+j)}{j+1} \right) \right) \right),$$

which is also an upper bound on the optimal sample complexity.

□

To get a better intuition of the result, we also have the following corollary, which shows that we should expect a scaling linear in n for small values of n (for large values the complexity tends instead to "flatten").

Corollary A.5. *Let T be the scaling in theorem A.4. We have that*

$$\min(n, (K-n)/2) \lesssim T \lesssim C \min(n, K/2).$$

Proof. First, observe the scaling

$$\left(1 + \frac{n-1}{n-1+j} \min \left(\frac{n-2}{2}, \frac{j(n-1+j)}{j+1} \right) \right) = O(n/2).$$

At this point, note that

$$\prod_{i=j+1}^{K-n} \frac{i}{n-1+i} = \prod_{i=j+1}^{K-n} \left(1 + \frac{n-1}{i} \right)^{-1}.$$

Using that $\frac{x}{1+x} \leq \log(1+x) \leq x$, we have

$$\log \prod_{i=j+1}^{K-n} \frac{i}{n-1+i} = \sum_{i=j+1}^{K-n} -\log \left(1 + \frac{n-1}{i} \right) \geq -(n-1) \sum_{i=j+1}^{K-n} \frac{1}{i}.$$

and

$$\log \prod_{i=j+1}^{K-n} \frac{i}{n-1+i} = \sum_{i=j+1}^{K-n} -\log \left(1 + \frac{n-1}{i} \right) \leq -(n-1) \sum_{i=j+1}^{K-n} \frac{1}{n-1+i}.$$

Define $H_n = \sum_{i=1}^{n} 1/i$ to be the n-th Harmonic number, we also have

$$\sum_{i=j+1}^{K-n} \frac{1}{i} = H_{K-n} - H_j.$$

Therefore

$$-(n-1)(H_{K-n} - H_j) \leq \log \prod_{i=j+1}^{K-n} \frac{i}{n-1+i} \leq -(n-1)(H_{K-1} - H_{n+j-1})$$

Using that $H_\ell \sim \log(\ell) + \gamma + O(1/\ell)$, where γ is the Euler–Mascheroni constant, we get

$$\left(\frac{j}{K-n}\right)^{n-1} \lesssim \prod_{i=j+1}^{K-n} \frac{i}{n-1+i} \lesssim \left(\frac{n+j-1}{K-1}\right)^{n-1}.$$

Therefore, we can bound $\sum_{j=1}^{K-n} \left(\frac{n+j-1}{K-1}\right)^{n-1}$ using an integral bound

$$\sum_{j=1}^{K-n} \left(\frac{n+j-1}{K-1}\right)^{n-1} \leq \int_0^{K-n} \left(\frac{n+x}{K-1}\right)^{n-1} dx \leq \frac{e(K-1)}{n}.$$

From which follows that the original expression can be upper bounded by an expression scaling as $O(\min(n, (K-1)/2))$.

Similarly, using that $\sum_{j=1}^{K-n} \left(\frac{j}{K-n}\right)^{n-1} \geq (K-n)/n$, we have that the lower bound scales as $\min(n, (K-n)/2)$. $\square$

B Algorithms

In this section we present some of the algorithms more in detail. These includes: ICPE, TaS, I-DPT and I-IDS.

MDP Formulation for ICPE. Recall that in ICPE we treat trajectories of data $\mathcal{D}_t = (x_1, a_1, \ldots, x_t)$ as sequences to be given as input to sequential models, such as Transformers. We treat trajectories as states of an MDP M. An environment M can be then modeled as an MDP, which is a sequential model characterized by a tuple $M = (\mathcal{S}, \mathcal{A}, P', r, H^\star_M, \rho)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, $P' : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition function, $r : \mathcal{S} \to [0,1]$ defines the reward function (to be defined later), $H^\star \in \mathcal{H}$ is the true hypothesis in M and ρ is the initial state distribution.

We define the state at time-step t as $s_t = (\mathcal{D}_t, \varnothing_{t:N})$, with $\varnothing_{t:N}$ indicating a null sequence of tokens for the remaining steps up to some pre-defined horizon N, with $s_1 = (x_1, \varnothing_{1:N})$.

To be more precise, letting $(s_t^\varnothing, a_t^\varnothing)$ denote, respectively, the null elements in the state and action at time-step t, we have $\varnothing_{t:t+k} = \{s_t^\varnothing, a_{t+1}^\varnothing, s_{t+1}^\varnothing, \cdots, a_{t+k-1}^\varnothing, s_{t+k}^\varnothing\}$.

The limit N is a practical upper bound on the horizon that limits the dimensionality of the state, which is introduced for implementing the algorithm. The action space remains $\mathcal{A}$, and the transition dynamics P' are induced by (ρ, P).

B.1 ICPE with Fixed Confidence

Recall that $\mathcal{D}_t = (x_1, a_1, \ldots, x_{t-1}, a_{t-1}, x_t)$ and $\hat{H}_\tau \sim I(\cdot|D_\tau)$. In the fixed confidence setting, problems terminate at some random point in time τ, chosen by the learner, or when the maximum horizon N is reached. We model this by giving π_t an additional stopping action a_{stop} such that $\pi_t : \mathcal{D}_t \to \mathcal{A} \cup \{a_{\text{stop}}\}$ so that the data collection processes terminates at the stopping-time $\tau = \min(N, t_{\text{stop}})$, with $t_{\text{stop}} \coloneqq \inf\{t \in \mathbb{N} : a_t = a_{\text{stop}}\}$.

Optimizing the dual formulation

$$\min_{\lambda \geq 0} \max_{I, \pi} V_\lambda(\pi, I)$$

can be viewed as a multi-timescale stochastic optimization problem: the slowest timescale updates the variable λ, an intermediate timescale optimizes over I, and the fastest refines the policy π.

Algorithm 2 ICPE (In-Context Pure Exploration) - Fixed Confidence

1. **Input:** Tasks distribution $\mathcal{P}(\mathcal{M})$; confidence δ; learning rates α, β; initial λ and hyper-parameters T, N, η.
2. Initialize buffer $\mathcal{B}$, networks Q_θ, I_ϕ and set $\bar{\theta} \leftarrow \theta, \bar{\phi} \leftarrow \phi$.
3. **while** Training is not over **do**
4. Sample environment $M \sim \mathcal{P}(\mathcal{M})$ with hypothesis $H^\star$, observe $s_1 \sim \rho$ and set $t \leftarrow 1$.
5. **for** $t = 1, \ldots, N-1$ **do**
6. Execute action $a_t = \arg\max_a Q_\theta(s_t, a)$ in M and observe next state s_{t+1}.
7. Add experience $z_t = (s_t, a_t, s_{t+1}, d_t = \mathbf{1}\{s_{t+1} \text{ is terminal}\}, H^\star)$ to $\mathcal{B}$.
8. If $a_t = a_{\text{stop}}$, break the loop.
9. **end for**
10. Update variable λ according to

$$\lambda \leftarrow \max\left(0, \lambda - \beta\left(I_\phi(H^\star|s_{\tau+1}) - 1 + \delta\right)\right). \tag{9}$$

11. Sample batches $B, B' \sim \mathcal{B}$ and update θ, ϕ as

$$\theta \leftarrow \theta - \alpha \nabla_\theta \frac{1}{|B|} \sum_{z \in B} \left[\mathbf{1}_{\{a \neq a_{\text{stop}}\}} \left(y_\lambda(z) - Q_\theta(s,a)\right)^2 + \left(r_\lambda(z_{\text{stop}}) - Q_\theta(s, a_{\text{stop}})\right)^2\right], \tag{10}$$

$$\phi \leftarrow \phi + \alpha \nabla_\phi \frac{1}{|B'|} \sum_{z \in B'} \left[\log(I_\phi(H^\star|s)\right]. \tag{11}$$

12. Update $\bar{\theta} \leftarrow (1-\eta)\bar{\theta} + \eta\theta$ and every T steps set $\bar{\phi} \leftarrow \phi$.
13. **end while**

MDP Formulation. We can use the MDP formalism to define an RL problem: we define a reward r that penalizes the agent at all time-steps, that is $r_t = -1$, while at the stopping-time we have $r_\tau = -1 + \lambda \mathbb{E}_{H \sim I(\cdot|s_\tau)}[h(H; M)]$. Hence, a trajectory's return can be written as

$$G_\tau = \sum_{t=1}^{\tau} r_t = -\tau + 1 + \underbrace{r(s_\tau, a_\tau)}_{r_\tau} = -\tau + \lambda I(H^\star | s_\tau).$$

Accordingly, one can define the Q-value of (π, I, λ) in a state-action pair (s, a) at the t-th step as $Q_\lambda^{\pi,I}(s, a) = \mathbb{E}^\pi_{M \sim \mathcal{P}(\mathcal{M})} \left[\sum_{n=t}^{\tau} r_n \middle| s_t = s, a_t = a \right]$, with $a_n \sim \pi_n(\cdot|s_n)$

Optimization over ϕ. We treat each optimization separately, employing a descent-ascent scheme. The distribution I is modeled using a sequential architecture parameterized by ϕ, denoted by I_ϕ. Fixing (π, λ), the inner maximization in eq. (1) corresponds to

$$\max_\phi \mathbb{E}^\pi_{M \sim \mathcal{P}(\mathcal{M})}[h(\hat{H}_\tau; M)], \quad \text{with } \hat{H}_\tau \sim I_\phi(\cdot|s_\tau).$$

We train ϕ via cross-entropy loss:

$$-\sum_{H'} h(H'; M) \log I_\phi(H'|s_\tau) = -\log I_\phi(H^\star|s_\tau),$$

averaged across environments. Alternatively, a MAP estimator may be used with the same loss.

Optimization over π. The policy π is defined as the greedy policy with respect to learned Q-values. Therefore, standard RL techniques can learn the Q-function that maximizes the value in eq. (1) given (λ, I). Denoting this function by Q_θ, it is parameterized using a sequential architecture with parameters θ.

We train Q_θ using DQN Mnih et al. [2015], Van Hasselt et al. [2016], employing a replay buffer $\mathcal{B}$ and a target network $Q_{\bar{\theta}}$ parameterized by $\bar{\theta}$. To maintain timescale separation, we introduce an additional inference target network $I_{\bar{\phi}}$, parameterized by $\bar{\phi}$, which provides stable training feedback for θ. When (I, λ) are fixed, optimizing π reduces to maximizing:

$$-\tau + \lambda \log I_\phi(H^\star|s_\tau).$$

Hence, we define the reward at the transition $z = (s, a, s', d, H^\star)$ (with the convention that $s' \leftarrow s$ if $a = a_{\text{stop}}$) as:

$$r_\lambda(z) \coloneqq -1 + d\lambda \log I_{\bar{\phi}}(H^\star|s'),$$

where $d = \mathbf{1}\{z \text{ is terminal}\}$ (z is terminal if the transition corresponds to the last time-step in a horizon, or $a = a_{\text{stop}}$). Furthermore, for a transition $z = (s, a, s', d, H^\star)$ we define $z_{\text{stop}} \coloneqq z|_{(a,s') \leftarrow (a_{\text{stop}}, s)}$ as the same transition z with $a \leftarrow a_{\text{stop}}$ and $s' \leftarrow s$.

There is one thing to note: the logarithm in the reward is justified since the original problem can be equivalently written as:

$$\min_{\lambda \geq 0} \max_{I, \pi} -\mathbb{E}^\pi_{M \sim \mathcal{P}(\mathcal{M})}[\tau] + \lambda \left[\log \left(\mathbb{P}^\pi_{M \sim \mathcal{P}(\mathcal{M})}(h(\hat{H}_\tau; M) = 1) \right) - \log(1 - \delta) \right],$$

after noting that we can apply the logarithm to the constraint in eq. (1), before considering the dual. Thus the optimal solutions (I, π) remain the same.

Then, using classical TD-learning Sutton and Barto [2018], the training target for a transition $z = (s, a, s', d, H^\star)$ can be defined as:

$$y_\lambda(z) = r_\lambda(z) + (1 - d)\gamma \max_{a'} Q_{\bar{\theta}}(s', a'),$$

where $\gamma \in (0, 1]$ is the discount factor.

As discussed earlier, we have a dedicated stopping action a_{stop}, whose value depends solely on history. Thus, its Q-value is updated retrospectively at any state s using an additional loss:

$$(r_\lambda(z_{\text{stop}}) - Q_\theta(s, a_{\text{stop}}))^2 .$$

Therefore, the overall loss that we consider for θ for a single transition z can be written as

$$\mathbf{1}_{\{a\neq a_{\text{stop}}\}}\left(y_\lambda(z)-Q_\theta(s,a)\right)^2+\left(r_\lambda(z_{\text{stop}})-Q_\theta(s,a_{\text{stop}})\right)^2,$$

where $\mathbf{1}_{\{a\neq a_{\text{stop}}\}}$ avoids double accounting for the stopping action.

To update parameters (θ,ϕ), we sample independent batches $(B,B')\sim\mathcal{B}$ from the replay buffer and apply gradient updates as specified in eqs. (3) and (4) of algorithm 1. Target networks are periodically updated, with $\bar\phi\leftarrow\phi$ every M steps, and $\bar\theta$ using Polyak averaging: $\bar\theta\leftarrow(1-\eta)\bar\theta+\eta\theta$, $\eta\in(0,1)$.

Optimization over λ. Finally, we update λ by assessing the confidence of I_ϕ at the stopping time according to eq. (2), maintaining a slow ascent-descent optimization schedule for sufficiently small learning rates.

Implementation with the MAP estimator. A practical implementation may consider to use the MAP estimator $\hat H_\tau=\arg\max_H I_\phi(H|s_\tau)$, which is what we do in practice, since it results in a lower variance estimator. We note that the loss function for I_ϕ, and the reward for Q_θ, as defined above, still yield the same optimal solution.

Cost implementation. Lastly, in practice, we optimize a reward $r_\lambda(z)=-c+dI_{\bar\phi}(H^\star|s')$, by setting $c=1/\lambda$, and noting that for a fixed λ the RL optimization remains the same. The reason why we do so is due to the fact that with this expression we do not have the product $\lambda\mathbb{E}_{H'\sim I_\phi}[h(H';M)]$, which makes the descent-ascent process more difficult.

We also use the following cost update

$$c_{t+1}=c_t-\beta(1-\delta-I_\phi(H_M^\star|s_{\tau+1})),$$

or $c_{t+1}=c_t-\beta(1-\delta-h(\hat H_\tau;M))$ if one uses the MAP estimator. To see why the cost can be updated in this way, define the parametrization $\lambda=e^{-x}$. Then the optimization problem becomes

$$\min_x\max_I\min_\pi -\mathbb{E}^\pi_{M\sim\mathcal{P}(\mathcal{M})}[\tau]+e^{-x}\left[\mathbb{P}^\pi_{M\sim\mathcal{P}(\mathcal{M})}\left(h(\hat H_\tau;M)=1\right)-1+\delta\right],$$

Letting $\rho=\mathbb{P}^\pi_{M\sim\mathcal{P}(\mathcal{M})}\left(h(\hat H_\tau;M)=1\right)-1+\delta$, the gradient update for x with a learning rate β simply is

$$x_{t+1}=x_t-\beta e^{-x_t}\rho,$$

implying that

$$-\log(\lambda_{t+1})=-\log(\lambda_t)-\beta\lambda_t\rho.$$

Defining $c_t=1/\lambda_t$, we have that

$$\log(c_{t+1})=\log(c_t)-(\beta\rho/c_t)\Rightarrow c_{t+1}=c_te^{\beta\rho/c_t}.$$

Using then the approximation $e^x\approx 1+x$, we find $c_{t+1}=c_t+\beta\rho=c_t-\beta(1-\delta-I_\phi(H_M^\star|s_{\tau+1}))$.

Training vs Deployment. Thus far, our discussion of `ICPE` has focused on the training phase. After training completes, the learned policy π and inference network I can be deployed directly: during deployment, π both collects data and determines when to stop—either by triggering its stopping action or upon reaching the horizon N.

B.2 Other Algorithms

In this section we describe Track and Stop (TaS) Garivier and Kaufmann [2016], and some variants such as I-IDS, I-DPT and the explore then commit variant of `ICPE`.

B.2.1 Track and Stop

Track and Stop (TaS, Garivier and Kaufmann [2016]) is an asymptotically optimal as $\delta\to 0$ for MAB problems. For simplicity, we consider a Gaussian MAB problem with K actions, where the reward of each action is normally distributed according to $\mathcal{N}(\mu_a,\sigma^2)$, and let $\mu=(\mu_a)_{a\in[K]}$ denote the model. The TaS algorithm consists of: (1) the model estimation procedure and recommender rule; (2) the sampling rule, dictating which action to select at each time-step; (3) the stopping rule, defining when enough evidence has been collected to identify the best action with sufficient confidence, and therefore to stop the algorithm.

Estimation Procedure and Recommender Rule The algorithm maintains a maximum likelihood estimate $\hat{\mu}_a(t)$ of the average reward for each arm based on the observations up to time t. Then, the recommender rule is defined as $\hat{a}_t = \arg\max_a \hat{\mu}_a(t)$.

Sampling Rule. The sampling rule is based on the observation that any δ-correct algorithm, that is an algorithm satisfying $\mathbb{P}(\hat{a}_\tau = a^\star) \geq 1-\delta$, with $a^\star = \arg\max_a \mu_a$, satisfies the following sample complexity

$$\mathbb{E}[\tau] \geq T^\star(\mu)\mathrm{kl}(1-\delta, \delta),$$

where $\mathrm{kl}(x,y) = x\log(x/y) + (1-x)\log((1-x)/(1-y))$ and

$$(T^\star(\mu))^{-1} = \sup_{\omega\in\Delta(K)} \min_{a\neq a^\star} \frac{\omega_{a^\star}\omega_a}{\omega_a + \omega_{a^\star}} \frac{\Delta_a^2}{2\sigma^2},$$

with $\Delta_a = \mu_{a^\star} - \max_{a\neq a^\star} \mu_a$. Interestingly, to design an algorithm with minimal sample complexity, we can look at the solution $\omega^\star = \arg\inf_{\omega\in\Delta(K)} T(\omega;\mu)$, with $(T(\omega))^{-1} = \min_{a\neq a^\star} \frac{\omega_{a^\star}\omega_a}{\omega_a+\omega_{a^\star}} \frac{\Delta_a^2}{2\sigma^2}$.

The solution $\omega^\star$ provides the best proportion of draws, that is, an algorithm selecting an action a with probability $\omega_a^\star$ matches the lower bound and is therefore optimal with respect to $T^\star$. Therefore, an idea is to ensure that N_t/t tracks $\omega^\star$, where N_t is the visitation vector $N(t) := \begin{bmatrix} N_1(t) & \dots & N_K(t) \end{bmatrix}^\top$.

However, the average rewards $(\mu_a)_a$ are initially unknown. A commonly employed idea [Garivier and Kaufmann, 2016, Kaufmann et al., 2016] is to track an estimated optimal allocation $\omega^\star(t) = \arg\inf_{\omega\in\Delta(K)} T(\omega;\hat{\mu}(t))$ using the current estimate of the model $\hat{\mu}(t)$.

However, we still need to ensure that $\hat{\mu}(t) \to \mu$. To that aim, we employ a D-tracking rule Garivier and Kaufmann [2016], whcih guarantees that arms are sampled at a rate of $\sqrt{t}$. If there is an action a with $N_a(t) \leq \sqrt{t} - K/2$ then we choose $a_t = a$. Otherwise, choose the action $a_t = \arg\min_a N_a(t) - t\omega_a^\star(t)$.

Stopping rule. The stopping rule determines when enough evidence has been collected to determine the optimal action with a prescribed confidence level. The problem of determining when to stop can be framed as a statistical hypothesis testing problem [Chernoff, 1959], where we are testing between K different hypotheses $(\mathcal{H}_a : (\mu_a > \max_{b\neq b} \mu_a))_a$.

We consider the following statistic $L(t) = tT(N(t)/t;\hat{\mu}(t))^{-1}$, which is a Generalized Likelihood Ratio Test (GLRT), similarly as in [Garivier and Kaufmann, 2016]. Comparing with the lower bound, one needs to stop as soon as $L(t) \geq \mathrm{kl}(\delta, 1-\delta) \sim \ln(1/\delta)$. However, to account for the random fluctuations, a more natural threshold is $\beta(t,\delta) = \ln((1+\ln(t))/\delta)$, thus we use $L(t) \geq \beta(t,\delta)$ for stochastic MAB problems. We also refer the reader to Kaufmann and Koolen [2021] for more details.

B.2.2 I-IDS

We implement a variant of Information Directed Sampling (IDS) Russo and Van Roy [2018], where we use the inference network I_ϕ learned during ICPE training as a posterior over optimal arms. This approach enables IDS to exploit latent structure in the environment without explicitly modeling it via a probabilistic model; instead, the learned I-network implicitly captures such structure.

By using the same inference network in both ICPE and I-IDS, we directly compare the exploration policy learned by ICPE to the IDS heuristic applied on top of the same posterior distribution. While computing the expected information gain in IDS requires intractable integrals, we approximate them using a Monte Carlo grid of 30 candidate reward values per action. The full pseudocode for I-IDS is given in Algorithm 3.

B.2.3 I-DPT

We implement a variant of DPT Lee et al. [2023] using the inference network. The idea is to act greedily with respect to the posterior distribution I at inference time.

First, we train I using `ICPE`. Then, at deployment we act with respect to I: in round t we selection action $a_t = \arg\max_H I(H|D_t)$. Upon observing x_{t+1}, we update D_{t+1} and stop as soon as $\arg\max_H I(H|D_t) \geq 1-\delta$.

B.3 Transformer Architecture

Here we briefly describe the architecture of the inference network I and of the network Q.

Both networks are implemented using a Transformer architecture. For the inference network, it is designed to predict a hypothesis H given a sequence of observations. Let the input tensor be denoted by $X \in \mathbb{R}^{B \times H \times m}$, where:

- B is the batch size,
- H is the sequence length (horizon), and
- $m = (d + |\mathcal{A}|)$, where d is the dimensionality of each observation x_t.

The inference network operates as follows:

1. **Embedding Layer**: Each observation vector $m_t = (x_t, a_t)$ is first embedded into a higher-dimensional space of size d_e using a linear transformation followed by a GELU activation: $h_t = \text{GELU}(W_{\text{embed}} m_t + b_{\text{embed}}), \quad h_t \in \mathbb{R}^{d_e}$.
2. **Transformer Layers**: The embedded sequence $h \in \mathbb{R}^{B \times H \times d_e}$ is then passed through multiple Transformer layers (specifically, a GPT-2 model configuration). The Transformer computes self-attention over the embedded input to model dependencies among observations:
$$h' = \text{Transformer}(h), \quad h' \in \mathbb{R}^{B \times H \times d_e}.$$
3. **Output Layer**: The final hidden state corresponding to the last element of the sequence ($h'_{:,-1,:}$) is fed into a linear output layer that projects it to logits representing the hypotheses:
$$o = W_{\text{out}} h'_{:,-1,:} + b_{\text{out}}, \quad o \in \mathbb{R}^{B \times |\mathcal{H}|}.$$
4. **Probability Estimation**: The output logits are transformed into log-probabilities via a log-softmax operation along the last dimension
$$\log p(H|X) = \text{log\_softmax}(o).$$

For Q, we use the same architecture, but do not take a log-softmax at the final step.

Algorithm 3 I-IDS

1: **Input:** Pre-trained inference network I_ϕ; prior means and variances μ_a, σ_a^2 for all $a \in \mathcal{A}$; target error threshold δ
2: **Initialize:** $f_a(x) = \mathcal{N}(x \mid \mu_a, \sigma_a^2)$ for each a
3: **for** $t = 1, 2, \ldots$ **do**
4:   **if** $\max_a I_\phi(a \mid \mathcal{D}_{t-1}) \geq 1 - \delta$ **then**
5:     **return** $\arg\max_a I_\phi(a \mid \mathcal{D}_{t-1})$
6:   **end if**
7:   **for** each arm $a \in \mathcal{A}$ **do**
8:     Approximate information gain:
$$g_t(a) = H\left(I_\phi(\cdot \mid \mathcal{D}_{t-1})\right) - \mathbb{E}_{r \sim p(r|a,\mathcal{D}_{t-1})}\left[H\left(I_\phi(\cdot \mid \mathcal{D}_{t-1}, a, r)\right)\right]$$
9:   **end for**
10:   Select action $a_t = \arg\max_a g_t(a)$
11:   Observe reward r_t
12:   Update dataset $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(a_t, r_t)\}$
13: **end for**

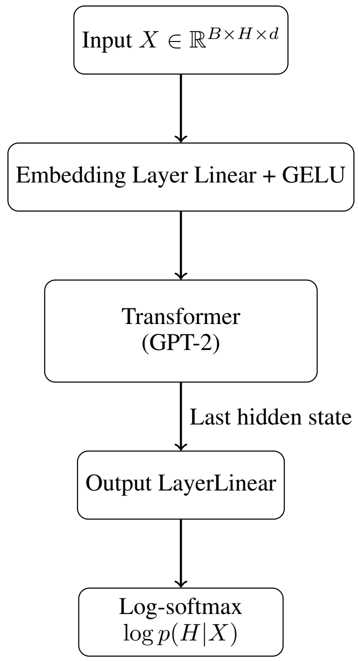


Figure 4: Model architecture for the inference network I (similarly for Q).

C Experiments

This section provides additional experimental results, along with detailed training and evaluation protocols to ensure clarity and reproducibility. All experiments were conducted using four NVIDIA A100 GPUs.

C.1 Bandit Problems

Here, we provide the implementation and evaluation details for the bandit experiments reported in Section 3.1, covering deterministic, stochastic, and structured settings.

Model Architecture and Optimization. For all bandit tasks, ICPE uses a Transformer with 3 layers, 2 attention heads, hidden dimension 256, GELU activations, and dropout of 0.1 applied to attention, embeddings, and residuals. Layer normalization uses $\epsilon = 10^{-5}$. Inputs are one-hot action-reward pairs with positional encodings. Models are trained using Adam with learning rates between 1×10^{-4} and 1×10^{-6}, and batch sizes from 128 to 1024 depending on task complexity.

C.1.1 Stochastic Bandits Problems

In the stochastic Gaussian bandit setting, we evaluate ICPE on best-arm identification tasks with $K \in \{4, 6, 8, \ldots, 14\}$. Arm means are sampled uniformly from $[0, 0.4K]$, with a guaranteed minimum gap of $1/K$ between the top two arms. All arms have a fixed reward standard deviation of 0.5. The target confidence level is set to $\delta = 0.1$.

We compare ICPE against several widely used baselines: *Top-Two Probability Sampling (TTPS)* Jourdan et al. [2022], *Track-and-Stop (TaS)* Garivier and Kaufmann [2016], *Uniform Sampling*, and *I-DPT*. For *I-DPT*, stopping occurs when the predicted optimal arm has estimated confidence at least $1 - \delta$. For *TTPS* and *TaS*, we apply the classical stopping rule based on the characteristic time $T^*(\hat{\mu}_t)$:

$$t \cdot T^*(\hat{\mu}_t) \geq \log\left(\frac{1 + \log t}{\delta}\right).$$

Each method is evaluated over three seeds, with 30 environments, and 30 trajectories per environment. 95% confidence intervals were computed with hierarchical bootstrapping.

C.1.2 Bandit Problems with Hidden Information

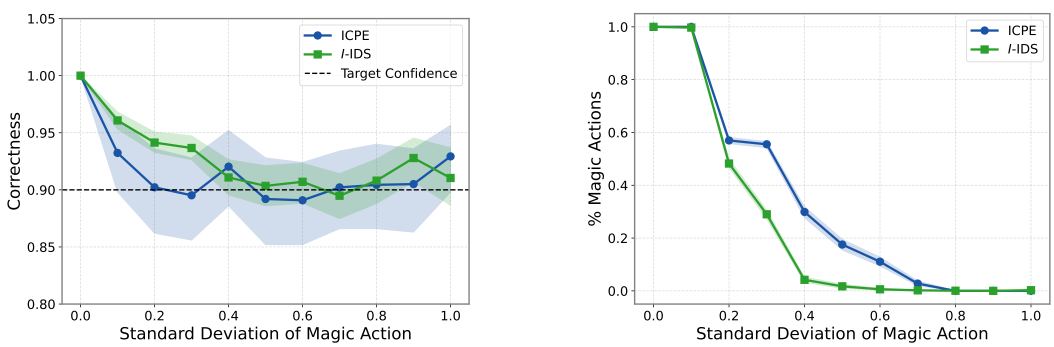
Magic Action Environments We evaluate ICPE in bandit environments where certain actions reveal information about the identity of the optimal arm, testing its ability to uncover and exploit latent structure under the fixed-confidence setting.

Each environment contains $K = 5$ arms. Non-magic arms have mean rewards sampled uniformly from $[1, 5]$, while the mean reward of the designated *magic action* (always arm 1) is defined as $\mu_1 = \phi(\arg\max_{a \neq 1} \mu_a)$ with $\phi(i) = i/K$. The magic action is not the optimal arm, but it encodes information about which of the other arms is. To control the informativeness of this signal, we vary the standard deviation of the magic arm $\sigma_1 \in \{0.0, 0.1, \ldots, 1.0\}$, while fixing the standard deviation of all other arms to $\sigma = 1 - \sigma_1$.

ICPE is trained under the fixed-confidence setting with a target confidence level of 0.9. For each σ_1, we compare ICPE's sample complexity to two baselines: (1) the average theoretical lower bound computed for the problem computed via averaging the result of Theorem A.2 over numerous environmental mean rewards, and (2) *I-IDS*, a pure-exploration information-directed sampling algorithm that uses ICPE's I-network for posterior estimation. All methods are over 500 environments, with 10 trajectories per environment. 95% confidence intervals are computed using hierarchical bootstrapping with two levels.

Beyond the exploration efficiency analysis shown in Figure 2a, we also assess the correctness of each method's final prediction and its usage of the magic action. As shown in Figure 5a, both ICPE and *I-IDS* consistently achieve the target accuracy of 0.9, validating their reliability under the fixed-confidence formulation.

Figure 5b plots the proportion of total actions that were allocated to the magic arm across different values of σ_1. While both methods adapt their reliance on the magic action as its informativeness degrades, *I-IDS* tends to abandon it earlier. This behavior suggests that ICPE is better able to retain and exploit structured latent information beyond what is captured by simple heuristics for expected information gain.



(a) (b)

Figure 5: (a) Final prediction accuracy across varying levels of noise in the magic action (σ_1). Both ICPE and *I-IDS* consistently achieve the target confidence threshold of 0.9. (b) Percentage of actions allocated to the magic arm as a function of σ_1. ICPE continues to exploit the magic action longer than *I-IDS*, suggesting more robust use of latent structure.

Magic Chain Environments To assess ICPE's ability to perform multi-step reasoning over latent structure, we evaluate it in environments where identifying the optimal arm requires sequentially uncovering a chain of informative actions. In these *magic chain* environments, each magic action reveals partial information about the next, culminating in identification of the best arm.

We use $K = 10$ arms and vary the number of magic actions $n \in \{1, 2, \ldots, 9\}$. Mean rewards for magic actions are defined recursively as:

$$\mu_{i_j} = \begin{cases} \phi(i_{j+1}), & \text{if } j = 1, \ldots, n-1, \\ \phi\Big(\arg\max_{a \notin \{i_1, \ldots, i_n\}} \mu_a\Big), & \text{if } j = n, \end{cases}$$

where $\phi(i) = i/K$, and the remaining arms have mean rewards sampled uniformly from $[1, 2]$. All rewards are deterministic (zero variance).

ICPE is trained under the fixed-confidence setting with $\delta = 0.99$. For each n, five models are trained across five seeds. We compare ICPE's average stopping time to the theoretical optimum (computed via Theorem A.4) and to the *I-IDS* baseline equipped with access to the I-network. Each model is evaluated over 1000 test environments per seed. 95% confidence intervals are computed using hierarchical bootstrapping.

In interpreting the results from Figure 2b, we observe that for environments with one or two magic actions, ICPE reliably learns the optimal policy of following the magic chain to completion. In these cases, the agent is able to identify the optimal arm without ever directly sampling it, by exploiting the structured dependencies embedded in the reward signals of the magic actions. Figure 6 illustrates a representative trajectory from the two-magic-arm setting, where the first magic action reveals the location of the second, which in turn identifies the optimal arm. The episode terminates without requiring the agent to explicitly sample the best arm itself.

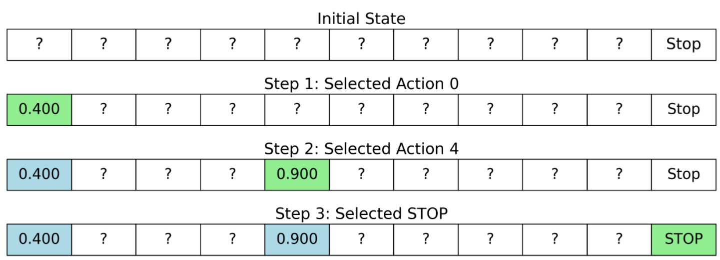


Figure 6: Example trajectory in the 2-magic-arm environment. ICPE follows the magic chain: the first magic action reveals the second, which identifies the optimal arm.

For environments with more than two magic actions, however, ICPE learns a different strategy. As the length of the magic chain increases, the expected sample complexity of following the chain from the start becomes suboptimal. Instead, ICPE learns to randomly sample actions until it encounters one of the magic arms and then proceeds to follow the chain from that point onward. This behavior represents an efficient, learned compromise between exploration cost and informativeness. Figure 7 shows an example trajectory from the six-magic-arm setting, where the agent initiates random sampling until it lands on a magic action, then successfully follows the remaining chain to identify the optimal arm.

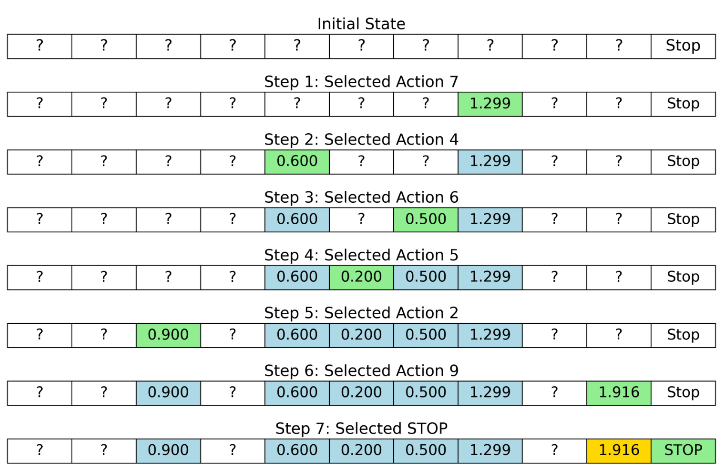


Figure 7: Example trajectory in the 6-magic-arm environment. Rather than starting from the first magic action, ICPE samples randomly until finding a magic action and then follows the chain to the optimal arm.

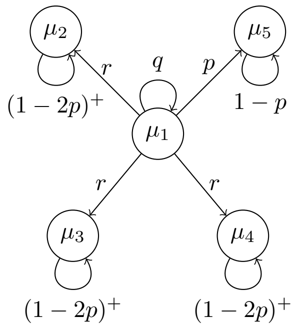


Figure 8: Loopy star graph.

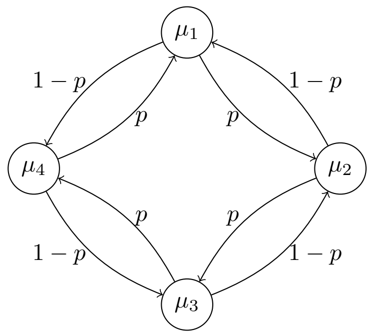


Figure 9: Ring graph.

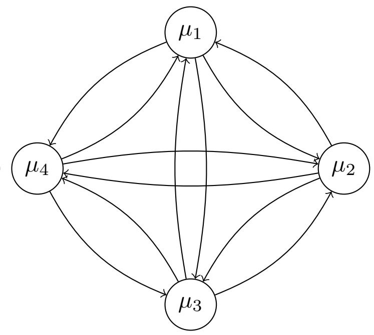


Figure 10: Loopless clique graph.

C.2 Exploration on Feedback Graphs

In the standard bandits setting we studied in Section 3.1, the learner observes the reward of the selected action, while in full-information settings, all rewards are revealed. Feedback graphs generalize this spectrum by specifying, via a directed graph G which additional rewards are observed when a particular action is chosen. Each node corresponds to an action, and an edge from u to v means that playing u may reveal feedback about v.

While feedback graphs have been widely studied for regret minimization Mannor and Shamir [2011], their use in pure exploration remains relatively underexplored Russo et al. [2025]. We study them here as a challenging and structured testbed for in-context exploration. Unlike unstructured bandits, these environments contain latent relational structure and stochastic feedback dependencies that must be inferred and exploited to explore efficiently.

Formally, we define a feedback graph as an adjacency matrix $G \in [0,1]^{K\times K}$, where $G_{u,v}$ denotes the probability that playing action u reveals the reward of action v. The learner observes a feedback vector $r \in \mathbb{R}^K$, where each coordinate is revealed independently with probability $G_{u,v}$:

$$r_v \sim \begin{cases} \mathcal{N}(\mu_v, \sigma^2), & \text{with probability } G_{u,v}, \\ \text{no observation}, & \text{otherwise.} \end{cases}$$

This setting allows us to test whether ICPE can learn to uncover and leverage latent graph structure to guide exploration. We evaluate performance on best-arm identification tasks across three representative feedback graph families:

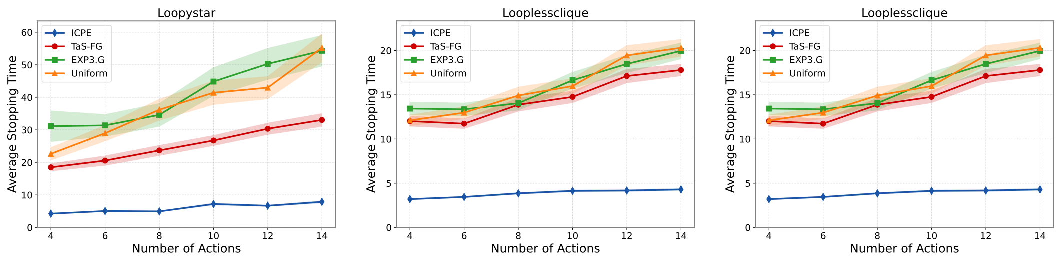
- **Loopy Star Graph** (Figure 8): A star-shaped graph with self-loops, parameterized by (p, q, r). The central node observes itself with probability q, one neighboring node with probability p, and all others with probability r. When p is small, it may be suboptimal to pull the central node, requiring the agent to adapt its strategy accordingly.
- **Ring Graph** (Figure 9): A cyclic graph where each node observes its right neighbor with probability p and its left neighbor with probability $1-p$. Effective exploration requires reasoning about which neighbors provide more informative feedback.
- **Loopless Clique Graph** (Figure 10): A fully connected graph with no self-loops. Edge probabilities are defined as:

  $$G_{u,v} = \begin{cases} 0 & \text{if } u = v, \\ \frac{p}{u} & \text{if } v \neq u \text{ and } v \text{ is odd}, \\ 1 - \frac{p}{u} & \text{otherwise.} \end{cases}$$

  Here, informativeness varies systematically with action index, requiring the learner to infer which actions are most useful.

These environments offer a diverse testbed for evaluating whether ICPE can uncover and exploit complex feedback structures without direct access to the underlying graph.

We tested ICPE in a fixed-confidence setting, using the same graph families but setting the optimal arm's mean to 1 and all others to 0.5 to facilitate faster convergence. ICPE was trained for $K = 4, 6, \ldots, 14$ with a target error rate of $\delta = 0.1$. We compared it to Uniform Sampling, EXP3.G, and Tas-FG using a shared stopping rule from Russo et al. [2025].



(a) (b) (c)

Figure 11: Sample complexity comparison under the fixed-confidence setting for: (a) Loopy Star, (b) Loopless Clique, and (c) Ring graphs.

As shown in Figure 11, ICPE consistently achieves significantly lower sample complexity than all baselines. This suggests that ICPE is able to meta-learn the underlying structure of the feedback graphs and leverage this knowledge to explore more efficiently than *uninformed* strategies. These results align with expectations: when environments share latent structure, learning to explore from experience offers a substantial advantage over fixed heuristics that cannot adapt across tasks.

C.3 Meta-Learning Binary Search

To test ICPE's ability to recover classical exploration algorithms, we evaluate whether it can autonomously meta-learn binary search.

We frame the task as a structured multi-armed bandit problem where the optimal arm (i.e., the target number) is uniformly drawn from $1, \ldots, K$. Pulling the correct arm yields a reward of $+10$, while pulling an arm above or below the target yields -1 or $+1$, respectively—providing directional feedback. The agent must learn to interpret and exploit this structure to efficiently locate the target.

We train ICPE under the fixed-confidence setting for $K = 2^3, \ldots, 2^8$, using $150{,}000$ in-context episodes and a target error rate of $\delta = 0.01$. Evaluation was conducted on 100 held-out tasks per setting. We report the minimum accuracy, mean stopping time, and worst-case stopping time, and compare against the theoretical binary search bound $O(\log_2 K)$.

Number of Actions (K)	**Minimum Accuracy**	**Mean Stopping Time**	**Max Stopping Time**	$\log_2 K$
8	1.00	2.13 ± 0.12	3	3
16	1.00	2.93 ± 0.12	4	4
32	1.00	3.71 ± 0.15	5	5
64	1.00	4.50 ± 0.21	6	6
128	1.00	5.49 ± 0.23	7	7
256	1.00	6.61 ± 0.26	8	8

Table 2: ICPE performance on the binary search task as the number of actions K increases.

As shown in Table 2, ICPE consistently achieves perfect accuracy with worst-case stopping times that match the optimal $\log_2(K)$ rate, demonstrating that it has successfully rediscovered binary search purely from data. While simple, this task illustrates ICPE's broader potential to learn efficient search strategies in domains where no hand-designed algorithm is available.