LEARNING TO ACTIVELY LEARN: A ROBUST APPROACH

Anonymous authors
Paper under double-blind review

ABSTRACT

This work proposes a procedure for designing algorithms for specific adaptive data collection tasks like active learning and pure-exploration multi-armed bandits. Unlike the design of traditional adaptive algorithms that rely on concentration of measure and careful analysis to justify the correctness and sample complexity of the procedure, our adaptive algorithm is learned via adversarial training over equivalence classes of problems derived from information theoretic lower bounds. In particular, a single adaptive learning algorithm is learned that competes with the best adaptive algorithm learned for each equivalence class. Our procedure takes as input just the available queries, set of hypotheses, loss function, and total query budget. This is in contrast to existing meta-learning work that learns an adaptive algorithm relative to an explicit, user-defined subset or prior distribution over problems which can be challenging to define and be mismatched to the instance encountered at test time. This work is particularly focused on the regime when the total query budget is very small, such as a few dozen, which is much smaller than those budgets typically considered by theoretically derived algorithms. We perform synthetic experiments to justify the stability and effectiveness of the training procedure, and then evaluate the method on tasks derived from real data including a noisy 20 Questions game and a joke recommendation task.

1 INTRODUCTION

Closed-loop learning algorithms use previous observations to inform what measurements to take next in a closed loop in order to accomplish inference tasks far faster than any fixed measurement plan set in advance. For example, active learning algorithms for binary classification have been proposed that under favorable conditions require exponentially fewer labels than passive, random sampling to identify the optimal classifier (Hanneke et al., 2014; Katz-Samuels et al., 2021). And in the multi-armed bandits literature, adaptive sampling techniques have demonstrated the ability to identify the "best arm" that optimizes some metric with far fewer experiments than a fixed design (Garivier & Kaufmann, 2016; Fiez et al., 2019). Unfortunately, such guarantees often either require simplifying assumptions that limit robustness and applicability, or algorithmic use of concentration inequalities that are very loose unless the number of samples is very large.

This work proposes a framework for producing algorithms that are learned through simulated experience to be as effective and robust as possible, even on a tiny measurement budget (e.g., 20 queries) where most theoretical guarantees do not apply. Our work fits into a recent trend sometimes referred to as *learning to actively learn* and *differentiable meta-learning in bandits* (Konyushkova et al., 2017; Bachman et al., 2017; Fang et al., 2017; Boutilier et al., 2020; Kveton et al., 2020) which tune existing algorithms or learn entirely new active learning algorithms by policy optimization. Previous works in this area learn a policy by optimizing with respect to data observed through prior experience (e.g., meta-learning or transfer learning) or an assumed explicit prior distribution of problem parameters (e.g. a Gaussian prior over the true weight vector for linear regression). In contrast, our approach makes no assumptions about what parameters are likely to be encountered at test time, and therefore produces algorithms that *do not suffer from mismatching priors at test time*. Instead, our method learns a policy that attempts to mirror the guarantees of frequentist algorithms with instance dependent sample complexities: there is an intrinsic difficulty measure that orders problem instances and given a fixed budget, higher accuracies can be obtained for *all* easier instances than

harder instances. This difficulty measure is most naturally derived from information theoretic lower bounds.

But unlike information theoretic bounds that hand-craft adversarial instances, inspired by the robust reinforcement learning literature, we formulate a novel adversarial training objective that automatically train minimax policies and propose a tractable and computationally efficient relaxation. This allows our learned policies to be very aggressive while maintaining robustness over difficulty in problem instances, without resorting to using loose concentration inequalities in the algorithm. Indeed, this work is particularly useful in the setting where relatively few rounds of querying can be made. The learning framework is general enough to be applied to many active learning settings of interest and is intended to be used to produce robust and high performing algorithms. We implement the framework for the pure-exploration combinatorial bandit problem — a paradigm including problems such as active binary classification and the 20 question game. We empirically validate our framework on a simple synthetic experiment before turning our attention to datasets derived from real data including a noisy 20 Questions game and a joke recommendation task which are also embedded as combinatorial bandits. As demonstrated in our experiments, in the low budget setting, our learned algorithms are the only ones that *both enjoy robustness guarantees* (as opposed to greedy and existing learning to actively learn methods) *and perform non-vacuously and instance-optimally* (as opposed to statistically justified algorithms).

2 Proposed Framework for Robust Learning to Actively Learn

From a birds-eye perspective, whether learned or defined by an expert, any algorithm for active learning can be thought of as a policy from the perspective of reinforcement learning. To be precise, at time t, based on an internal state s_t, the policy π defines a distribution $\pi(s_t)$ over the set of potential actions $\mathcal{X}$. It then takes action $x_t \in \mathcal{X}, x_t \sim \pi(s_t)$ and receives observation y_t, updates the state and the process repeats.

Fix a horizon $T \in \mathbb{N}$, and a problem instance $\theta_* \in \Theta \subseteq \mathbb{R}^d$ which parameterizes the observation distribution. For $t = 1, 2, \ldots, T$

- state $s_t \in \mathcal{S}$ is a function of the history, $\{(x_i, y_i)\}_{i=1}^{t-1}$,
- action $x_t \in \mathcal{X}$ is drawn at random from the distribution $\pi(s_t)$ defined over $\mathcal{X}$, and
- next state $s_{t+1} \in \mathcal{S}$ is constructed by taking action x_t in state s_t and observing $y_t \sim f(\cdot|\theta_*, s_t, x_t)$

until the game terminates at time $t = T$ and the learner receives a loss L_T which is task specific. Note that L_T is a random variable that depends on the tuple $(\pi, \{(x_i, y_i)\}_{i=1}^{T}, \theta_*)$. We assume that f is a known parametric distribution to the policy but the parameter θ is unknown to the policy. Let $\mathbb{P}_{\pi,\theta}, \mathbb{E}_{\pi,\theta}$ denote the probability and expectation under the probability law induced by executing policy π in the game with $\theta_* = \theta$ to completion. Note that $\mathbb{P}_{\pi,\theta}$ includes any internal randomness of the policy π and the random observations $y_t \sim f(\cdot|\theta, s_t, x_t)$. Thus, $\mathbb{P}_{\pi,\theta}$ assigns a probability to any trajectory $\{(x_i, y_i)\}_{i=1}^{T}$. For a given policy π and $\theta_* = \theta$, the metric of interest we wish to minimize is the expected loss $\ell(\pi, \theta) := \mathbb{E}_{\pi,\theta}[L_T]$ where L_T as defined above is the loss observed at the end of the episode. For a fixed policy π, $\ell(\pi, \theta)$ defines a loss surface over all possible values of θ. This loss surface captures the fact that some values of θ are just intrinsically harder than others, but also that a policy may be better suited for some values of θ versus others.

Finally, we assume we are equipped with a positive function $\mathcal{C} : \Theta \to (0, \infty)$ that assigns a score to each $\theta \in \Theta$ that intuitively captures the "difficulty" of a particular θ, and can be used as a partial ordering of Θ. Ideally, $\mathcal{C}(\theta)$ is a monotonic transformation of $\ell(\pi^*, \theta)$ for some "best" policy π^* that we will define shortly. Our plan is now as follows, in Section 2.1, we ground the discussion and describe $\mathcal{C}(\theta)$ for the combinatorial bandit problem. Then in Section 2.2, we zoom out to define our main objective of finding a *min-gap* optimal policy, finally providing an adversarial training approach in Section 3.

2.1 Complexity for Combinatorial Bandits

A concrete example of the framework above is the combinatorial bandit problem. The learner has access to sets $\mathcal{X} = \{e_1, \cdots, e_d\} \subset \mathbb{R}^d$, where e_i is the i-th standard basis vector, and $\mathcal{Z} \subset \{0, 1\}^d$. In each round the learner chooses an $x_t \in \mathcal{X}$ according to a policy $\pi(\{(x_i, y_i)\}_{i=1}^{t-1})$ and observes

y_t with $\mathbb{E}[y_t|x_t, \theta_*] = \langle x_t, \theta_* \rangle$ for some unknown $\theta_* \in \mathbb{R}^d$. The goal of the learner is BEST ARM IDENTIFICATION. Denote $z_*(\theta_*) = \arg\max_{z \in \mathcal{Z}} \langle z, \theta_* \rangle$, then at time T the learner outputs a recommendation $\hat{z}$ and incurs loss $L_{BAI,T} = \mathbf{1}\{z_* \neq \hat{z}\}$. This setting naturally captures the 20 question game. Indeed assume there are $d \gg T = 20$ potential yes/no questions that can be asked at each time, corresponding to the elements of $\mathcal{X}$, and that each element of $\mathcal{Z}$ is a binary vector representing the answers to these questions for a given item. If answers y_t are deterministic then $\theta_* \in \{-1, 1\}^d$, but this framework also captures the case $\theta_* \in [-1, 1]^d$ when answers are stochastic, or answered incorrectly with some probability. Then a policy π at each time decides which question to ask based on the answers so far to determine the item closest to an unknown vector θ_*.

As described in Sections 5 and Appendix A, combinatorial bandits generalizes standard multi-armed bandits, and all of binary classification, and thus has received a tremendous amount of interest in recent years. A large portion of this work has focused on providing precise characterization of the information theoretic limit on the mimimal number of samples needed to identify $z_*(\theta_*)$ with high probability a quantity denoted as $\rho_*(\theta_*)$ which is the solution to an optimization problem (Soare et al., 2014; Fiez et al., 2019; Degenne et al., 2020) $\rho_*(\theta_*)^{-1} := \max_{\lambda \in \triangle_{\mathcal{X}}} \min_{\substack{\theta' \in \Theta \\ z_*(\theta') \neq z_*(\theta_*)}} \sum_{x \in \mathcal{X}} \lambda_x \langle x, \theta_* - \theta' \rangle^2$ for some set of alternatives Θ. This quantity provides a natural complexity measure $\mathcal{C}(\theta_*) = \rho_*(\theta_*)$ for a given instance θ_* and we describe it in a few specific cases below.

As a warmup example, consider the standard best-arm identification problem where $\mathcal{Z} = \mathcal{X} = \{\mathbf{e}_i : i \in [d]\}$ and choosing action $x_t \in \mathcal{X}$ results in reward $y_t \sim \text{Bernoulli}(\theta_{i_t})$. Let $i_*(\theta) = \arg\max_{z \in \mathcal{Z}} z^\top \theta = \arg\max_i \theta_i$. Then in this case $\rho_*(\theta) \approx \sum_{i \neq i_*(\theta)} (\theta_{i_*(\theta)} - \theta_i)^{-2}$ and it's been shown that there exists a constant $c_0 > 0$ such that for any sufficiently large $\nu > 0$ we have

$$\min_{\pi} \max_{\theta : \rho_*(\theta) \leq \nu} \ell_{BAI}(\pi, \theta) \geq \exp(-c_0 T/\nu)$$

In other words, more difficult instances correspond to θ with a small gap between the best arm and any other arm. Moreover, for any $\theta \in \mathbb{R}^d$ there exists a policy $\widetilde{\pi}$ that achieves $\ell(\widetilde{\pi}, \theta) \leq c_1 \exp(-c_2 T/\rho_*(\theta_*))$ where c_1, c_2 capture constant and low-order terms (Carpentier & Locatelli, 2016; Karnin et al., 2013; Garivier & Kaufmann, 2016). Said plainly, the above correspondence between the lower bound and the upper bound for the multi-armed bandit problem shows that $\rho_*(\theta_*)$ is a natural choice for $\mathcal{C}(\theta)$ in this setting.

In recent years, algorithms for the more general combinatorial bandit setting have been established with instance-dependent sample complexities matching $\rho_*(\theta_*)$ (up to logarithmic factors) (Karnin et al., 2013; Chen et al., 2014; Fiez et al., 2019; Chen et al., 2017; Degenne et al., 2020; Katz-Samuels et al., 2020). Another complexity term that appears in Cao & Krishnamurthy (2017) for combinatorial bandits is

$$\widetilde{\rho}(\theta) = \sum_{i=1}^{d} \max_{z : z_i \neq z_{*,i}(\theta)} \frac{\|z - z_*(\theta)\|_2^2}{\langle z - z_*(\theta), \theta \rangle^2}. \tag{1}$$

One can show $\rho_*(\theta) \leq \widetilde{\rho}(\theta)$ (Katz-Samuels et al., 2020) and in many cases track each other. Because $\widetilde{\rho}(\theta)$ can be computed much more efficiently compared to $\rho_*(\theta)$, we take $\mathcal{C}(\theta) = \widetilde{\rho}(\theta)$.

2.2 OBJECTIVE: RESPONDING TO ALL DIFFICULTIES

As described above, though there exists algorithms for the combinatorial bandit problem that are instance-optimal in the fixed-confidence setting along with algorithms for the fixed-budget, they do not work well with small budgets as they rely on statistical guarantees. Indeed, for their guarantees to be non-vacuous, we need the budget T to be sufficiently large enough to compare to the potentially large constants in upper bounds. In practice, they are so conservative that for the first 20 samples they would just sample uniformly. To overcome this, we now provide a different framework that for policy learning in a worst-case setting that is effective even in the small budget regime.

The challenge is in finding a policy that performs well across all potential problem instances simultaneously. It is common to consider minimax optimal policies which attempt to perform well on worst case instances — but as a result, may perform poorly on easier instances. Thus, an ideal policy π would perform uniformly well over a set of θ's that are all equivalent in "difficulty". Since

each $\theta \in \Theta$ is equipped with an inherent notion of difficulty, $C(\theta)$, we can stratify the space of all possible instances by difficulty. A good policy is one whose worst case performance over all possible *problem difficulties* is minimized. We formalize this idea below.

For any set of problem instances $\widetilde{\Theta} \subset \Theta$ and $r \geq 0$ define

$$\ell(\pi, \widetilde{\Theta}) := \max_{\theta \in \widetilde{\Theta}} \ell(\pi, \theta) \qquad \text{and} \qquad \Theta^{(r)} := \{\theta : \mathcal{C}(\theta) \leq r\}.$$

For a fixed $r > 0$ (including $r = \infty$), a policy π' that aims to minimize just $\ell(\pi', \Theta^{(r)})$ will be minimax for $\Theta^{(r)}$ and may not perform well on easy instances. To overcome this shortsightedness we introduce a new objective by focusing on $\ell(\pi, \Theta^{(r)}) - \min_{\pi'} \ell(\pi', \Theta^{(r)})$; the *sub-optimality gap* of a given policy π relative to an r-dependent baseline policy trained specifically for each r.

Objective: Return the policy

$$\pi_* := \arg\min_{\pi} \max_{r>0} \left(\ell(\pi, \Theta^{(r)}) - \min_{\pi'} \ell(\pi', \Theta^{(r)}) \right) \tag{2}$$

which minimizes the worst case sub-optimality gap over all $r > 0$.

Figure 1 illustrates these definitions. The blue curve (r-dependent baseline) captures the best possible performance $\min_{\pi'} \ell(\pi', \Theta^{(r)})$ that is possible for each difficulty level r. In other words, the r-dependent baseline defines a different policy for each value of r. Therefore, the blue curve may be unachievable with just a single policy. The green curve captures a policy that achieves the minima (r-dependent baseline) at a given r'. Though it is the ideal policy for this difficulty, it could be sub-optimal at any other difficulty. The orange curve is the performance of our optimal policy π^* — it is willing to sacrifice performance for any given r to achieve an overall better worst case gap from the baseline.

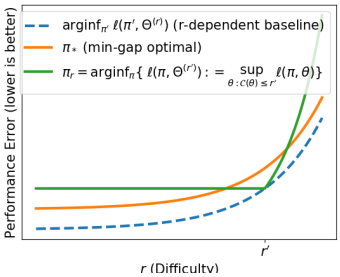

Figure 1: Performance curves for various policies.

3 MAPO: Adversarial Training Algorithm

Identifying π_* naively requires the computation of $\min_{\pi'} \ell(\pi', \Theta^{(r)})$ for all $r > 0$. However, in practice given an increasing sequence $r_1 < \cdots < r_K$ that indexes nested sets of problem instances of increasing difficulty, $\Theta^{(r_1)} \subset \Theta^{(r_2)} \subset \cdots \subset \Theta^{(r_K)}$, we wish to identify a policy $\widehat{\pi}$ that minimizes the maximum sub-optimality gap with respect to this sequence. Explicitly, we seek to learn

$$\widehat{\pi} = \arg\min_{\pi} \max_{k \leq K} \left(\ell(\pi, \Theta^{(r_k)}) - \ell(\pi_k, \Theta^{(r_k)}) \right) \text{ where } \quad \pi_k \in \arg\min_{\pi} \max_{\theta : \mathcal{C}(\theta) \leq r_k} \ell(\pi, \theta). \tag{3}$$

Note that as $K \to \infty$ and $\sup_k \frac{r_{k+1}}{r_k} \to 1$, equation 2 and equation 3 are essentially equivalent under benign smoothness conditions on $\mathcal{C}(\theta)$, in which case $\widehat{\pi} \to \pi_*$. In practice, we choose $\Theta^{(r_K)}$ contains all problems that can be solved within the budget T relatively accurately, and a small $\epsilon > 0$, where $\max_k \frac{r_{k+1}}{r_k} = 1 + \epsilon$. In Algorithm 1, our algorithm MAPO efficiently solves this objective by first computing π_k for all $k \in [K]$ to obtain $\ell(\pi_k, \Theta^{(r_k)})$ as benchmarks, and then uses these benchmarks to train $\widehat{\pi}$. The next section will focus on the challenges of the optimization problems in equation 4 and equation 5.

3.1 Differentiable Policy Optimization

The critical part of running MAPO (Algorithm 1) is to solve for equation 4 and equation 5. Note that equation 5 is an optimization of the same form with equation 4 after shifting the loss by the scalar value $b^{(r_{k(\theta)})}$. Consequently, to learn $\{\widetilde{\pi}_k\}_k$ and $\widehat{\pi}$, it suffices to develop a training procedure to solve $\min_\pi \max_{\theta \in \Omega} \ell'(\pi, \theta)$ for an arbitrary set Ω and generic loss function $\ell'(\pi, \theta)$.

We would like to solve this saddle-point problem using an alternating gradient descent/ascent method in Algorithm 2 that we describe now. Instead of optimizing over all possible policies, we

Algorithm 1 MAPO: Min-gap Adversarial Policy Optimization

Input: sequence $\{r_k\}_{k=1}^K$, complexity function $\mathcal{C}$.
Define $k(\theta) \in [K]$ such that $r_{k(\theta)-1} < \mathcal{C}(\theta) \leq r_{k(\theta)}$ for all θ with $\mathcal{C}(\theta) \leq r_K$.
for $k \in 1, ..., K$ **do**
Obtain policy π_k by solving:

$$\pi_k := \arg\min_{\pi} \ell(\pi, \Theta^{(r_k)}) = \arg\min_{\pi} \max_{\theta \in \Theta^{(r_k)}} \ell(\pi, \theta) \qquad \text{and} \qquad b^{(r_k)} := \ell(\pi_k, \Theta^{(r_k)}) \tag{4}$$

end for
Training for min-gap optimal policy: Solve the following:

$$\widehat{\pi} = \arg\min_{\pi} \max_{\theta \in \Theta^{(r_K)}} \left[\ell(\pi, \theta) - b^{(r_{k(\theta)})} \right] \tag{5}$$

Output: $\widehat{\pi}$ (a solution to equation 3).

restrict the policy class to neural networks that take state representation as input and output a probability distribution over actions, parameterized by weights ψ. In practice, $\ell'(\pi^\psi, \theta)$ may be poorly behaved in (ψ, θ) so a gradient descent/ascent procedure may get stuck in a neighborhood of a critical point that is not an optimal solution to the saddle point problem. To avoid this, we instead track over many different possible θ's (intuitively corresponding to different initializations):

$$\min_{\psi} \max_{\theta \in \Omega} \ell'(\pi^\psi, \theta) = \min_{\psi} \max_{\widetilde{\theta}_{1:N} \subset \Omega} \max_{i \in [N]} \ell'(\pi^\psi, \widetilde{\theta}_i). \tag{6}$$

$$= \min_{\psi} \max_{\widetilde{\theta}_{1:N} \subset \Omega} \max_{\lambda \in \Delta_N} \mathbb{E}_{i \sim \lambda} \ell'(\pi^\psi, \widetilde{\theta}_i). \tag{7}$$

$$= \min_{\psi} \max_{w \in \mathbb{R}^N, \widetilde{\theta}_{1:N} \subset \Omega} \mathbb{E}_{i \sim \text{SOFTMAX}(w)} \left[\ell'(\pi^\psi, \widetilde{\theta}_i) \right]. \tag{8}$$

In the first equality we replace the maximum over all Ω to a maximum over all subsets $\widetilde{\Theta} = \widetilde{\theta}_{1:N}$ of size N. The resulting maximum over the N points is still a discrete optimization. To smooth it out, we utilize the fact that a $\max$ over a set is just the same as the maximum over of the expectation over all distributions on that set. In the last equality, we reparameterize the set of distributions with the softmax to weight the different values of $\widetilde{\theta}$. In each round, we backpropagate through w and $\widetilde{\theta}_{1:N}$.

Now we discuss the optimization routine outlined in Algorithm 2. For the inside optimization, ideally, in each round we would build an estimate of the loss function at our current choice of π^ψ for each of the $\widetilde{\theta}_{1:N}$'s under consideration. To do so, we rollout the policy for each $\theta \in \widetilde{\theta}_{1:N}$ under consideration L times and then average the resulting losses (this also allows us to construct a stochastic gradient of the loss). In practice we can't consider all $\theta \in \widetilde{\theta}_{1:N}$, so instead we sample M of them from w. This has a computational benefit by allowing us to be strategic by considering θ's each round that are closest to the $\arg\max_{\widetilde{\theta}_{1:N}} \ell'(\pi^\psi, \theta)$.

After this we then backpropagate through w and $\widetilde{\Theta}$ using the stochastic gradients learned from the rollouts. Finally, we then update π by backpropagation through the neural network under consideration. The gradient steps are taken with unbiased gradient estimates $g^w(i, \tau)$, $g^{\widetilde{\Theta}}(i, \tau)$ and $g^\psi(i, \tau)$, which are computed by using the score-function identity and is described in detail in Appendix C. We outline more implementation details in Appendix B along with the below algorithm with explicit gradient estimate formulas. Hyperparamters can be found in Appendix D.

4 EXPERIMENTS

We now evaluate the approach described in the previous section for combinatorial bandits with $\mathcal{X} = \{\mathbf{e}_i : i \in [d]\}$ and $\mathcal{Z} \subset \{0, 1\}^d$. This setting generalizes both binary active classification for arbitrary model class and active recommendation, which we evaluate by conducting experiments on two respective real datasets. We evaluated based on two criteria: *instance-dependent worst-case* and *average-case*. For instance-dependent worst-case, we measure, for each r_k and policy π,

Algorithm 2 Gradient Based Optimization of equation 8

Input: partition Ω, number of iterations N_{it}, number of problem samples M, number of rollouts per problem L, and loss variable L_T at horizon T (see beginning of Section 2).
Goal: Compute the optimal policy $\arg\min_\pi \max_{\theta\in\Omega} \ell'(\pi,\theta) = \arg\min_\pi \max_{\theta\in\Omega} \mathbb{E}_{\pi,\theta}[L_T]$. Note in the case of $\ell'(\pi,\theta) = \ell(\pi,\theta) - b^{(r_{k(\theta)})}$, L_T is inherently subtracting the scalar value $b^{(r_{k(\theta)})}$.
Initialization: w, finite set $\widetilde{\Theta} = \widetilde{\theta}_{1:N}$ and ψ.
for $t = 1, ..., N_{\text{it}}$ **do**
  for $m = 1, ..., M$ **do**
    Sample $I_m \overset{i.i.d.}{\sim}$ SOFTMAX(w).
    Collect L independent rollout trajectories, denoted as $\tau_{m,1:L}$, by the policy π^ψ for θ_{I_m}.
  end for
  Update the generating distribution by taking ascending steps on gradient estimates:

$$\widetilde{\Theta}, w \leftarrow \widetilde{\Theta} + \frac{1}{ML}\sum_{m=1}^{M}\left(\nabla_{\widetilde{\Theta}}\mathcal{L}_{\text{barrier}}(\widetilde{\theta}_{I_m},\Omega) + \sum_{l=1}^{L} g^{\widetilde{\Theta}}(I_m,\tau_{m,l})\right), w + \frac{1}{ML}\sum_{m=1}^{M}\sum_{l=1}^{L} g^{w}(I_m,\tau_{m,l})$$

  where $\mathcal{L}_{\text{barrier}}$ is a differentiable barrier loss that heavily penalizes the $\widetilde{\theta}_{I_m}$'s outside Ω.
  Update the policy by taking descending step on gradient estimate:

$$\psi \leftarrow \psi - \frac{1}{ML}\sum_{m=1}^{M}\sum_{l=1}^{L} g^{\psi}(I_m,\tau_{m,l})$$

end for

$\ell(\pi,\Theta^{(r_k)}) := \max_{\theta\in\Theta^{(r_k)}} \ell(\pi,\theta)$ and plot this value as a function of r_k. We note that our algorithm is designed to optimize for such a metric. For the secondary average-case metric, we instead measure, for policy π and some collected set Θ, $\frac{1}{|\Theta|}\sum_{\theta\in\Theta}\ell(\pi,\theta)$. Performances of instance-dependent worst-case metric are reported in Figures 2, 3, 4, 6, and 7 below while the average case performances are reported in the tables and Figure 5. Full scale of the figures can also be found in Appendix F.

4.1 ALGORITHMS

We compare against a number of baseline active learning algorithms (see Section 5 for a review). UNCERTAINTY SAMPLING at time t computes the empirical maximizer of $\langle z,\widehat{\theta}\rangle$ and the runner-up, and samples an index uniformly from their symmetric difference (i.e thinking of elements of $\mathcal{Z}$ as subsets of $[d]$); if either are not unique, an index is sampled from the region of disagreement of the winners (see Appendix G for details). The greedy methods are represented by soft generalized binary search (SGBS) (Nowak, 2011) which maintains a posterior distribution over $\mathcal{Z}$ and samples to maximize information gain. A hyperparameter $\beta \in (0, 1/2)$ of SGBS determines the strength of the likelihood update. We plot or report a range of performance over $\beta \in \{.01, .03, .1, .2, .3, .4\}$. The agnostic algorithms for classification (Dasgupta, 2006; Hanneke, 2007b;a; Dasgupta et al., 2008; Huang et al., 2015; Jain & Jamieson, 2019) or combinatorial bandits (Chen et al., 2014; Gabillon et al., 2016; Chen et al., 2017; Cao & Krishnamurthy, 2017; Fiez et al., 2019; Jain & Jamieson, 2019) are so conservative that given just $T = 20$ samples, they are all exactly equivalent to uniform sampling and hence represented by UNIFORM. To represent a policy based on learning to actively learn with respect to a prior, we employ the method of Kveton et al. (2020), denoted BAYES-LAL, with a fixed prior $\widetilde{\mathcal{P}}$ constructed by drawing a z uniformly at random from $\mathcal{Z}$ and defining $\theta = 2z - 1 \in [-1, 1]^d$ (details in Appendix H). When evaluating each policy, we use the successive halving algorithm (Li et al., 2017; 2018) for optimizing our non-convex objective with randomly initialized gradient descent and restarts (details in Appendix E).

4.2 SYNTHETIC DATASET: THRESHOLDS

We begin with a very simple instance to demonstrate the instance-dependent performance achieved by our learned policy. For $d = 25$, let $\mathcal{X} = \{\mathbf{e}_i : i \in [d]\}$, $\mathcal{Z} = \{\sum_{i=1}^{k}\mathbf{e}_i : k = 0, 1, \ldots, d\}$, and $f(\cdot|\theta, x)$ is a Bernoulli distribution over $\{-1, 1\}$ with mean $\langle x,\theta\rangle \in [-1, 1]$. Appendix A shows that $z_*(\theta_*) = \arg\max_z\langle z,\theta_*\rangle$ is the best threshold classifier for a label distribution induced

by $(\theta_* + 1)/2$. We trained baseline policies $\{\pi_k\}_{k=1}^9$ for the BEST IDENTIFICATION metric with $\mathcal{C}(\theta) = \widetilde{\rho}(\mathcal{X}, \mathcal{Z}, \theta)$ and $r_k = 2^{3+i/2}$ for $i \in \{0, \dots, 8\}$.

First we compare the base policies π_k to $\widehat{\pi}$. Figure 2 presents $\ell(\pi, \Theta^{(r)}) = \max_{\theta:\widetilde{\rho}(\theta)\le r} \ell(\pi, \theta) = \max_{\theta:\widetilde{\rho}(\theta)\le r} \mathbb{P}_{\pi,\theta}(\widehat{z} \neq z_*(\theta))$ as a function of r for our base policies $\{\pi_k\}_k$ and the global policy $\widehat{\pi}$, each as an individual curve. Figure 3 plots the same information in terms of gap: $\ell(\pi, \Theta^{(r)}) - \min_{k:r_{k-1}<r\le r_k} \ell(\pi_k, \Theta^{(r_k)})$. We observe that each π_k performs best in a particular region and $\widehat{\pi}$ performs almost as well as the r-dependent baseline policies over the range of r.

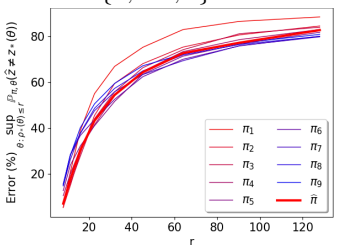

Figure 2: Learned policies, lower is better

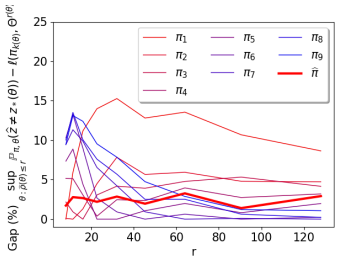

Figure 3: Sub-optimality of individual policies, lower is better

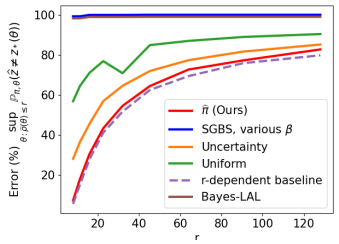

Figure 4: Max $\{\theta : \widetilde{\rho}(\theta) \le r\}$, lower is better

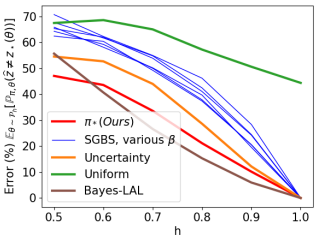

Figure 5: Average $\mathbb{E}_{\theta\sim\mathcal{P}_h}[\cdot]$, lower is better

Under the same conditions as Figure 2, Figure 4 compares the performance of $\widehat{\pi}$ to the algorithm benchmarks. Since SGBS and Bayes-LAL are deterministic, the adversarial training finds a θ that tricks them into catastrophic failure. Figure 5 trades adversarial evaluation for evaluating with respect to a parameterized prior: For each $h \in \{0.5, 0.6, \dots, 1\}$, $\theta \sim \mathcal{P}_h$ is defined by drawing a z uniformly at random from $\mathcal{Z}$ and then setting $\theta_i = (2z_i - 1)(2\alpha_i - 1)$ where $\alpha_i \sim \text{Bernoulli}(h)$. Thus, each sign of $2z - 1$ is flipped with probability h. We then compute $\mathbb{E}_{\theta\sim\mathcal{P}_h}[\mathbb{P}_{\pi,\theta}(\widehat{z} = z_*(\theta))] = \mathbb{E}_{\theta\sim\mathcal{P}_h}[\ell(\pi, \theta)]$. While SGBS now performs much better than uniform and uncertainty sampling, our policy $\widehat{\pi}$ is still superior to these policies. However, Bayes-LAL is best overall which is expected since the support of $\mathcal{P}_h$ is essentially a rescaled version of the prior used in Bayes-LAL.

4.3 REAL DATASETS

20 Questions. Our dataset is constructed from the real data of Hu et al. (2018). Summarizing how we used the data, 100 yes/no questions were considered for 1000 celebrities. Each question $i \in [100]$ for each person $j \in [1000]$ was answered by several annotators to construct an empirical probability $\bar{p}_i^{(j)} \in [0, 1]$ denoting the proportion of annotators that answered "yes." To construct our instance, we take $\mathcal{X} = \{\mathbf{e}_i : i \in [100]\}$ to encode questions and $\mathcal{Z} = \{z^{(j)} : [z^{(j)}]_i = \mathbf{1}\{\bar{p}_i^{(j)} > 1/2\}\} \subset \{0, 1\}^{1000}$. Just as before, we trained $\{\pi_k\}_{k=1}^4$ for the BEST IDENTIFICATION metric with $\mathcal{C}(\theta) = \widetilde{\rho}(\mathcal{X}, \mathcal{Z}, \theta)$ and $r_i = 2^{3+i/2}$ for $i \in \{1, \dots, 4\}$. See Appendix I for details.

Jester Joke Recommendation. We now turn our attention away from BEST IDENTIFICATION to SIMPLE REGRET where $\ell(\pi, \theta) = \mathbb{E}_{\pi,\theta}[\langle z_*(\theta) - \widehat{z}, \theta\rangle]$. We consider the Jester jokes dataset of Goldberg et al. (2001) that contains jokes ranging from innocent puns to grossly offensive jokes. We filter the dataset to only contain users that rated all 100 jokes, resulting in 14116 users. A rating of each joke was provided on a $[-10, 10]$ scale which was rescaled to $[-1, 1]$ and observations were simulated as Bernoulli's like above. We then clustered the ratings of these users (see Appendix J for details) to 10 groups to obtain $\mathcal{Z} = \{z^{(k)} : k \in [10], z^{(k)} \in \{0, 1\}^{100}\}$ where $z_i^{(k)} = 1$ corresponds to recommending the ith joke in user cluster $z^{(k)} \in \mathcal{Z}$. See Appendix J for details.

4.3.1 INSTANCE-DEPENDENT WORST-CASE

Figure 6 and Figure 7 are analogous to Figure 4 but for the 20 questions and Jester joke instances, respectively. The two deterministic policies, SGBS and Bayes-LAL, fail on these datasets as well against the worst-case instances.

On the Jester joke dataset, our policy alone nearly achieves the r-dependent baseline for all r. But on 20 questions, uncertainty sampling performs remarkably well. These experiments on real datasets demonstrate that our policy obtains near-optimal instance dependent sample complexity.

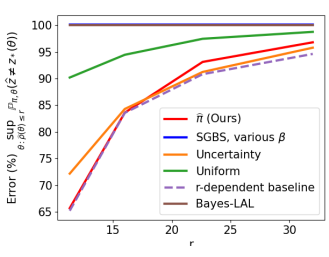

Figure 6: 20 Questions

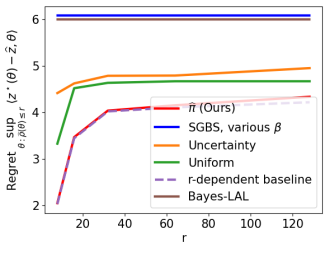

Figure 7: Jester Joke

4.3.2 Average Case Performance

While the metric of the previous section rewarded algorithms that perform uniformly well over all possible environments that could be encountered, in this section we consider the performance of an algorithm with respect to a distribution over environments, which we denote as average case.

Table 1: 20 Questions, higher the better

Method	Accuracy(%)
π^* (Ours)	17.9
SGBS	{26.5, 26.2, 27.2, 26.5, 21.4, 12.8}
Uncertainty	14.3
Bayes-LAL	4.1
Uniform	6.9

Table 2: Jester Joke, lower the better

Method	Regret
π^* (Ours)	3.209
SGBS	{3.180, 3.224, 3.278, 3.263, 3.153, 3.090}
Uncertainty	3.027
Bayes-LAL	3.610
Uniform	3.877

While heuristic based algorithms (such as SGBS, uncertainty sampling and Bayes-LAL) can perform catastrophically for worst-case instances, they can perform very well with respect to a benign distribution over instances. Here we demonstrate that our policy not only performs optimally under the instance-dependent worst-case metric but also remain comparable even when evaluated under the average case metric. To measure the average performance, we construct prior distributions $\widehat{\mathcal{P}}$ based on the individual datasets:

- For the 20 questions dataset, to draw a $\theta \sim \widehat{\mathcal{P}}$, we uniformly at random select a $j \in [1000]$ and sets $\theta_i = 2\bar{p}_i^{(j)} - 1$ for all $i \in [d]$.
- For the Jester joke recommendation dataset, to draw a $\theta \sim \widehat{\mathcal{P}}$, we uniformly sample a user and employ their ratings to each joke.

On the 20 questions dataset, as shown in Table 1, SGBS and $\widehat{\pi}$ are the winners. Bayes-LAL performs much worse in this case, potentially because of the distribution shift from $\widetilde{\mathcal{P}}$ (prior we train on) to $\widehat{\mathcal{P}}$ (prior at test time). The strong performance of SGBS may be due to the fact that $\text{sign}(\theta_i) = 2z_*(\theta)_i - 1$ for all i and $\theta \sim \widehat{\mathcal{P}}$, a realizability condition under which SGBS has strong guarantees (Nowak, 2011). On the Jester joke dataset, Table 2 shows that despite our policy not being trained for this setting, its performance is still among the top.

5 Related Work

Learning to actively learn. Previous works vary in how the parameterize the policy, ranging from parameterized mixtures of existing expertly designed active learning algorithms (Baram et al., 2004; Hsu & Lin, 2015; Agarwal et al., 2016), parameterizing hyperparameters (e.g., learning rate, rate of forced exploration, etc.) in an existing popular algorithm (e.g, EXP3) (Konyushkova et al., 2017; Bachman et al., 2017; Cella et al., 2020), and the most ambitious, policies parameterized end-to-end like in this work (Boutilier et al., 2020; Kveton et al., 2020; Sharaf & Daumé III, 2019; Fang et al., 2017; Woodward & Finn, 2017). These works take an approach of defining a prior distribution either through past experience (meta-learning) or expert created (e.g., $\theta \sim \mathcal{N}(0, \Sigma)$), and then evaluate their policy with respect to this prior distribution. Defining this prior can be difficult, and moreover, if the θ encountered at test time did not follow this prior distribution, performance could suffer significantly. Our approach, on the other hand, takes an adversarial training approach and can

be interpreted as learning a parameterized least favorable prior (Wasserman, 2013), thus gaining a much more robust policy as an end result.

Robust and Safe Reinforcement Learning. Our work is also highly related to the field of robust and safe reinforcement learning, where our objective can be considered as an instance of *minimax criterion under parameter uncertainty* (Garcıa & Fernández, 2015). Widely applied in applications such as robotics (Mordatch et al., 2015; Rajeswaran et al., 2016), these methods train a policy in a simulator like Mujoco (Todorov et al., 2012) to minimize a defined loss objective while remaining robust to uncertainties and perturbations to the environment (Mordatch et al., 2015; Rajeswaran et al., 2016). Ranges of these uncertainty parameters are chosen based on potential values that could be encountered when deploying the robot in the real world. In our setting, however, defining the set of environments is far less straightforward and is overcome by the adoption of the $\mathcal{C}(\theta)$ function.

Active Binary Classification Algorithms. The literature on active learning algorithms can be partitioned into *model-based heuristics* like uncertainty sampling, query by committee, or model-change sampling (Settles, 2009), *greedy binary-search* like algorithms that typically rely on a form of bounded noise for correctness (Dasgupta, 2005; Kääriäinen, 2006; Golovin & Krause, 2011; Nowak, 2011), and *agnostic* algorithms that make no assumptions on the probabilistic model (Dasgupta, 2006; Hanneke, 2007b;a; Dasgupta et al., 2008; Huang et al., 2015; Jain & Jamieson, 2019; Katz-Samuels et al., 2020; 2021). Though the heuristics and greedy methods can perform very well for some problems, it is typically easy to construct counter-examples (e.g., outside the assumptions) in which they catastrophically fail as demonstrated in our experiments. The agnostic algorithms have strong robustness guarantees but rely on concentration inequalities, and consequently require at least hundreds of labels to observe any deviation from random sampling (see Huang et al. (2015) for comparison). Therefore, they were implicitly represented by uniform in our experiments.

Pure-exploration Multi-armed Bandit Algorithms. In the linear structure setting, for sets $\mathcal{X}, \mathcal{Z} \subset \mathbb{R}^d$ known to the player, pulling an "arm" $x \in \mathcal{X}$ results in an observation $\langle x, \theta_* \rangle +$ zero-mean noise, and the objective is to identify $\arg\max_{z \in \mathcal{Z}} \langle z, \theta_* \rangle$ for a vector θ_* unknown to the player (Soare et al., 2014; Karnin, 2016; Tao et al., 2018; Xu et al., 2017; Fiez et al., 2019). A special case of linear bandits is combinatorial bandits where $\mathcal{X} = \{\mathbf{e}_i : i \in [d]\}$ and $\mathcal{Z} \subset \{0,1\}^d$ (Chen et al., 2014; Gabillon et al., 2016; Chen et al., 2017; Cao & Krishnamurthy, 2017; Fiez et al., 2019; Jain & Jamieson, 2019). Active binary classification is a special case of combinatorial pure-exploration multi-armed bandits (Jain & Jamieson, 2019), which we exploit in the threshold experiments. While the above works have made great theoretical advances in deriving algorithms and information theoretic lower bounds that match up to constants, the constants are so large that these algorithms only behave well when the number of measurements is very large. When applied to the instances of our paper (only 20 queries are made), these algorithms behave no differently than random sampling.

6 Discussion and Future Directions

We see this work as an exciting but preliminary step towards realizing the full potential of this general approach. Although our experiments has been focusing on applications of combinatorial bandit, we see this framework generalizing with minor changes to many more widely applicable settings such as multi-class active classification, contextual bandits, etc. To generalize $\mathcal{C}(\theta)$ to these settings, one can refer to existing literature for instance-dependent lower bounds (Katz-Samuels et al., 2021; Agarwal et al., 2014). Alternatively, when such a lower bound does not exist, we conjecture that a heuristic scoring function could also serve as $\mathcal{C}(\theta)$. For example, in a chess game, one could simply use the scoring function of the pieces left on board as a proxy for difficulty.

From a practical perspective, training a $\widehat{\pi}$ can take many hours of computational resources for even these small instances. Scaling these methods to larger instances is an important next step. While training time scales linearly with the horizon length T, we note that one can take multiple samples per time step. With minimal computational overhead, this could enable training on problems that require larger sample complexities. In our implementation we hard-coded the decision rule for $\widehat{z}$ given s_T, but it could also be learned as in (Luedtke et al., 2020). Likewise, the parameterization of the policy and generator worked well for our purposes but was chosen somewhat arbitrarily–are there more natural choices? Finally, while we focused on stochastic settings, this work naturally extends to constrained fully adaptive adversarial sequences which is an interesting direction of future work.

REFERENCES

Alekh Agarwal, Daniel Hsu, Satyen Kale, John Langford, Lihong Li, and Robert Schapire. Taming the monster: A fast and simple algorithm for contextual bandits. In *International Conference on Machine Learning*, pp. 1638–1646. PMLR, 2014.

Alekh Agarwal, Haipeng Luo, Behnam Neyshabur, and Robert E Schapire. Corralling a band of bandit algorithms. *arXiv preprint arXiv:1612.06246*, 2016.

VM Aleksandrov, VI Sysoyev, and SHEMENEV. VV. Stochastic optimization. *Engineering Cybernetics*, (5):11–+, 1968.

Philip Bachman, Alessandro Sordoni, and Adam Trischler. Learning algorithms for active learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 301–310. JMLR. org, 2017.

Yoram Baram, Ran El Yaniv, and Kobi Luz. Online choice of active learning algorithms. *Journal of Machine Learning Research*, 5(Mar):255–291, 2004.

Craig Boutilier, Chih-Wei Hsu, Branislav Kveton, Martin Mladenov, Csaba Szepesvari, and Manzil Zaheer. Differentiable bandit exploration. *arXiv preprint arXiv:2002.06772*, 2020.

Tongyi Cao and Akshay Krishnamurthy. Disagreement-based combinatorial pure exploration: Efficient algorithms and an analysis with localization. *stat*, 2017.

Alexandra Carpentier and Andrea Locatelli. Tight (lower) bounds for the fixed budget best arm identification bandit problem. In *Conference on Learning Theory*, pp. 590–604, 2016.

Leonardo Cella, Alessandro Lazaric, and Massimiliano Pontil. Meta-learning with stochastic linear bandits. *arXiv preprint arXiv:2005.08531*, 2020.

Lijie Chen, Anupam Gupta, Jian Li, Mingda Qiao, and Ruosong Wang. Nearly optimal sampling algorithms for combinatorial pure exploration. In *Conference on Learning Theory*, pp. 482–534, 2017.

Shouyuan Chen, Tian Lin, Irwin King, Michael R Lyu, and Wei Chen. Combinatorial pure exploration of multi-armed bandits. In *Advances in Neural Information Processing Systems*, pp. 379–387, 2014.

Sanjoy Dasgupta. Analysis of a greedy active learning strategy. In *Advances in neural information processing systems*, pp. 337–344, 2005.

Sanjoy Dasgupta. Coarse sample complexity bounds for active learning. In *Advances in neural information processing systems*, pp. 235–242, 2006.

Sanjoy Dasgupta, Daniel J Hsu, and Claire Monteleoni. A general agnostic active learning algorithm. In *Advances in neural information processing systems*, pp. 353–360, 2008.

Rémy Degenne, Pierre Ménard, Xuedong Shang, and Michal Valko. Gamification of pure exploration for linear bandits. In *International Conference on Machine Learning*, pp. 2432–2442. PMLR, 2020.

Meng Fang, Yuan Li, and Trevor Cohn. Learning how to active learn: A deep reinforcement learning approach. *arXiv preprint arXiv:1708.02383*, 2017.

Tanner Fiez, Lalit Jain, Kevin G Jamieson, and Lillian Ratliff. Sequential experimental design for transductive linear bandits. In *Advances in Neural Information Processing Systems*, pp. 10666–10676, 2019.

Victor Gabillon, Alessandro Lazaric, Mohammad Ghavamzadeh, Ronald Ortner, and Peter Bartlett. Improved learning complexity in combinatorial pure exploration bandits. In *Artificial Intelligence and Statistics*, pp. 1004–1012, 2016.

Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

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

Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *information retrieval*, 4(2):133–151, 2001.

Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

Steve Hanneke. A bound on the label complexity of agnostic active learning. In *Proceedings of the 24th international conference on Machine learning*, pp. 353–360, 2007a.

Steve Hanneke. Teaching dimension and the complexity of active learning. In *International Conference on Computational Learning Theory*, pp. 66–81. Springer, 2007b.

Steve Hanneke et al. Theory of disagreement-based active learning. *Foundations and Trends® in Machine Learning*, 7(2-3):131–309, 2014.

Botao Hao, Tor Lattimore, and Csaba Szepesvari. Adaptive exploration in linear contextual bandit. *arXiv preprint arXiv:1910.06996*, 2019.

Wei-Ning Hsu and Hsuan-Tien Lin. Active learning by learning. In *Twenty-Ninth AAAI conference on artificial intelligence*, 2015.

Huang Hu, Xianchao Wu, Bingfeng Luo, Chongyang Tao, Can Xu, Wei Wu, and Zhan Chen. Playing 20 question game with policy-based reinforcement learning. *arXiv preprint arXiv:1808.07645*, 2018.

Tzu-Kuo Huang, Alekh Agarwal, Daniel J Hsu, John Langford, and Robert E Schapire. Efficient and parsimonious agnostic active learning. In *Advances in Neural Information Processing Systems*, pp. 2755–2763, 2015.

Lalit Jain and Kevin G Jamieson. A new perspective on pool-based active classification and false-discovery control. In *Advances in Neural Information Processing Systems*, pp. 13992–14003, 2019.

Matti Kääriäinen. Active learning in the non-realizable case. In *International Conference on Algorithmic Learning Theory*, pp. 63–77. Springer, 2006.

Zohar Karnin, Tomer Koren, and Oren Somekh. Almost optimal exploration in multi-armed bandits. In *International Conference on Machine Learning*, pp. 1238–1246, 2013.

Zohar S Karnin. Verification based solution for structured mab problems. In *Advances in Neural Information Processing Systems*, pp. 145–153, 2016.

Julian Katz-Samuels, Lalit Jain, Zohar Karnin, and Kevin Jamieson. An empirical process approach to the union bound: Practical algorithms for combinatorial and linear bandits. *arXiv preprint arXiv:2006.11685*, 2020.

Julian Katz-Samuels, Jifan Zhang, Lalit Jain, and Kevin Jamieson. Improved algorithms for agnostic pool-based active classification. *arXiv preprint arXiv:2105.06499*, 2021.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. Learning active learning from data. In *Advances in Neural Information Processing Systems*, pp. 4225–4235, 2017.

Branislav Kveton, Martin Mladenov, Chih-Wei Hsu, Manzil Zaheer, Csaba Szepesvari, and Craig Boutilier. Differentiable meta-learning in contextual bandits. *arXiv preprint arXiv:2006.05094*, 2020.

Tor Lattimore and Csaba Szepesvari. The end of optimism? an asymptotic analysis of finite-armed linear bandits. *arXiv preprint arXiv:1610.04491*, 2016.

Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. Massively parallel hyperparameter tuning. *arXiv preprint arXiv:1810.05934*, 2018.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, 18(1):6765–6816, 2017.

Alex Luedtke, Marco Carone, Noah Simon, and Oleg Sofrygin. Learning to learn from data: Using deep adversarial learning to construct optimal statistical procedures. *Science Advances*, 6 (9), 2020. doi: 10.1126/sciadv.aaw2140. URL `https://advances.sciencemag.org/content/6/9/eaaw2140`.

Shie Mannor and John N Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *Journal of Machine Learning Research*, 5(Jun):623–648, 2004.

Igor Mordatch, Kendall Lowrey, and Emanuel Todorov. Ensemble-cio: Full-body dynamic motion planning that transfers to physical humanoids. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5307–5314. IEEE, 2015.

Robert D Nowak. The geometry of generalized binary search. *IEEE Transactions on Information Theory*, 57(12):7893–7906, 2011.

Jungseul Ok, Alexandre Proutiere, and Damianos Tranos. Exploration in structured reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 8874–8882, 2018.

Aravind Rajeswaran, Sarvjeet Ghotra, Balaraman Ravindran, and Sergey Levine. Epopt: Learning robust neural network policies using model ensembles. *arXiv preprint arXiv:1610.01283*, 2016.

Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

Amr Sharaf and Hal Daumé III. Meta-learning for contextual bandit exploration. *arXiv preprint arXiv:1901.08159*, 2019.

Max Simchowitz and Kevin G Jamieson. Non-asymptotic gap-dependent regret bounds for tabular mdps. In *Advances in Neural Information Processing Systems*, pp. 1151–1160, 2019.

Max Simchowitz, Kevin Jamieson, and Benjamin Recht. The simulator: Understanding adaptive sampling in the moderate-confidence regime. *arXiv preprint arXiv:1702.05186*, 2017.

Marta Soare, Alessandro Lazaric, and Rémi Munos. Best-arm identification in linear bandits. In *Advances in Neural Information Processing Systems*, pp. 828–836, 2014.

Chao Tao, Saúl Blanco, and Yuan Zhou. Best arm identification in linear bandits with linear dimension dependency. In *International Conference on Machine Learning*, pp. 4877–4886, 2018.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

Alexandre B Tsybakov. *Introduction to nonparametric estimation*. Springer Science & Business Media, 2008.

Bart Van Parys and Negin Golrezaei. Optimal learning for structured bandits. 2020.

Larry Wasserman. *All of statistics: a concise course in statistical inference*. Springer Science & Business Media, 2013.

Mark Woodward and Chelsea Finn. Active one-shot learning. *arXiv preprint arXiv:1702.06559*, 2017.

Liyuan Xu, Junya Honda, and Masashi Sugiyama. Fully adaptive algorithm for pure exploration in linear bandits. *arXiv preprint arXiv:1710.05552*, 2017.

A INSTANCE DEPENDENT SAMPLE COMPLEXITY

Identifying forms of $\mathcal{C}(\theta)$ is not as difficult a task as one might think due to the proliferation of tools for proving lower bounds for active learning (Mannor & Tsitsiklis, 2004; Tsybakov, 2008; Garivier & Kaufmann, 2016; Carpentier & Locatelli, 2016; Simchowitz et al., 2017; Chen et al., 2014). One can directly extract values of $\mathcal{C}(\theta)$ from the literature for regret minimization of linear or other structured bandits (Lattimore & Szepesvari, 2016; Van Parys & Golrezaei, 2020), contextual bandits (Hao et al., 2019), and tabular as well as structured MDPs (Simchowitz & Jamieson, 2019; Ok et al., 2018). Moreover, we believe that even reasonable surrogates of $\mathcal{C}(\theta)$ should result in a high quality policy π_*.

We review some canonical examples:

- **Multi-armed bandits.** In the best-arm identification problem, there are $d \in \mathbb{N}$ Gaussian distributions where the ith distribution has mean $\theta_i \in \mathbb{R}$ for $i = 1, \dots, d$. In the above formulation, this problem is encoded as action $x_t = i_t$ results in observation $y_t \sim \text{Bernoulli}(\theta_{i_t})$ and the loss $\ell(\pi, \theta) := \mathbb{E}_{\pi,\theta}[\mathbf{1}\{\widehat{i} \neq i_*(\theta)\}]$ where $\widehat{i}$ is π's recommended index and $i_*(\theta) = \arg\max_i \theta_i$. It's been shown that there exists a constant $c_0 > 0$ such that for any sufficiently large $\nu > 0$ we have

$$\min_{\pi} \max_{\theta : \mathcal{C}_{MAB}(\theta) \leq \nu} \ell(\pi, \theta) \geq \exp(-c_0 T/\nu)$$
$$\text{where} \quad \mathcal{C}_{MAB}(\theta) := \sum_{i \neq i_*(\theta)} (\theta_{i_*(\theta)} - \theta_i)^{-2}$$

  Moreover, for any $\theta \in \mathbb{R}^d$ there exists a policy $\widetilde{\pi}$ that achieves $\ell(\widetilde{\pi}, \theta) \leq c_1 \exp(-c_2 T/\mathcal{C}_{MAB}(\theta))$ where c_1, c_2 capture constant and low-order terms (Carpentier & Locatelli, 2016; Karnin et al., 2013; Simchowitz et al., 2017; Garivier & Kaufmann, 2016).

The above correspondence between the lower bound and the upper bound suggests that $\mathcal{C}_{MAB}(\theta)$ plays a critical role in determining the difficulty of identifying $i_*(\theta)$ for *any* θ. This exercise extends to more structured settings as well:

- **Content recommendation / active search.** Consider n items (e.g., movies, proteins) where the ith item is represented by a feature vector $x_i \in \mathcal{X} \subset \mathbb{R}^d$ and a measurement $x_t = x_i$ (e.g., preference rating, binding affinity to a target) is modeled as a linear response model such that $y_t \sim \mathcal{N}(\langle x_i, \theta \rangle, 1)$ for some unknown $\theta \in \mathbb{R}^d$. If $\ell(\pi, \theta) := \mathbb{E}_{\pi,\theta}[\mathbf{1}\{\widehat{i} \neq i_*(\theta)\}]$ as above then nearly identical results to that of above hold for an analogous function of $\mathcal{C}_{MAB}(\theta)$ (Soare et al., 2014; Karnin, 2016; Fiez et al., 2019).
- **Active binary classification.** For $i = 1, \dots, d$ let $\phi_i \in \mathbb{R}^p$ be a feature vector of an unlabeled item (e.g., image) that can be queried for its binary label $y_i \in \{-1, 1\}$ where $y_i \sim \text{Bernoulli}(\theta_i)$ for some $\theta \in \mathbb{R}^d$. Let $\mathcal{H}$ be an *arbitrary set of classifiers* (e.g., neural nets, random forest, etc.) such that each $h \in \mathcal{H}$ assigns a label $\{-1, 1\}$ to each of the items $\{\phi_i\}_{i=1}^d$ in the pool. If items are chosen sequentially to observe their labels, the objective is to identify the true risk minimizer $h_*(\theta) = \arg\min_{h \in \mathcal{H}} \sum_{i=1}^d \mathbb{E}_\theta[\mathbf{1}\{h(\phi_i) \neq y_i\}]$ using as few requested labels as possible and $\ell(\pi, \theta) := \mathbb{E}_{\pi,\theta}[\mathbf{1}\{\widehat{h} \neq h_*(\theta)\}]$ where $\widehat{h} \in \mathcal{H}$ is π's recommended classifier. Many candidates for $\mathcal{C}(\theta)$ have been proposed from the agnostic active learning literature (Dasgupta, 2006; Hanneke, 2007b;a; Dasgupta et al., 2008; Huang et al., 2015; Jain & Jamieson, 2019) but we believe the most granular candidates come from the combinatorial bandit literature (Chen et al., 2017; Fiez et al., 2019; Cao & Krishnamurthy, 2017; Jain & Jamieson, 2019). To make the reduction, for each $h \in \mathcal{H}$ assign a $z^{(h)} \in \{0, 1\}^d$ such that $[z^{(h)}]_i := \mathbf{1}\{h(\phi_i) = 1\}$ for all $i = 1, \dots, d$ and set $\mathcal{Z} = \{z^{(h)} : h \in \mathcal{H}\}$. It is easy to check that $z_*(\theta) := \arg\max_{z \in \mathcal{Z}} \langle z, \theta \rangle$ satisfies $z_*(\theta) = z^{(h_*(\theta))}$. Thus, requesting the label of example i is equivalent to sampling from $\text{Bernoulli}(\langle \mathbf{e}_i, \theta \rangle) \in \{-1, 1\}$, completing the reduction to combinatorial bandits: $\mathcal{X} = \{\mathbf{e}_i : i \in [d]\}$, $\mathcal{Z} \subset \{0, 1\}^d$. We then apply the exact same $\mathcal{C}(\theta)$ as above for linear bandits.

B GRADIENT BASED OPTIMIZATION ALGORITHM IMPLEMENTATION

First, we restate the algorithm with explicit gradient estimator formulas derived in Appendix C.

Algorithm 3 Gradient Based Optimization of equation 8 (Algorithm 2) with explicit gradient estimators.

Input: partition Ω, number of iterations N_{it}, number of problem samples M, number of rollouts per problem L, and loss variable L_T at horizon T (see beginning of Section 2).
Goal: Compute the optimal policy
$\arg\min_\pi \max_{\theta\in\Omega} \ell'(\pi,\theta) = \arg\min_\pi \max_{\theta\in\Omega} \mathbb{E}_{\pi,\theta}[L_T]$.
Initialization: w, finite set $\widetilde{\Theta}$ and ψ.
for $t = 1, ..., N_{\text{it}}$ **do**
  Collect rollouts of play:
  for $m = 1, ..., M$ **do**
    Sample problem index $I_m \overset{i.i.d.}{\sim} \text{SOFTMAX}(w)$.
    Collect L independent rollout trajectories ($\{\}$), denoted as $\tau_{m,1:L}$, by the policy π^ψ for problem instance θ_{I_m} and observe losses $\forall 1 \le l \le L, L_T(\pi^\psi, \tau_{m,l}, \widetilde{\theta}_{I_m})$.
  end for
  Optimize worst cases in Ω:
  Update the generating distribution by taking ascending steps on gradient estimates:

$$w \leftarrow w + \frac{1}{ML}\sum_{m=1}^{M} \nabla_w \log(\text{SOFTMAX}(w)_{I_m}) \cdot (\sum_{l=1}^{L} L_T(\pi^\psi, \tau_{m,l}, \widetilde{\theta}_{I_m})) \tag{9}$$

$$\widetilde{\Theta} \leftarrow \widetilde{\Theta} + \frac{1}{ML}\sum_{m=1}^{M}\sum_{l=1}^{L} \Big(\nabla_{\widetilde{\Theta}} \mathcal{L}_{\text{barrier}}(\widetilde{\theta}_{I_m}, \Omega) + \nabla_{\widetilde{\Theta}} L_T(\pi^\psi, \tau_{m,l}, \widetilde{\theta}_{I_m}) + L_T(\pi^\psi, \tau_{m,l}, \widetilde{\theta}_{I_m}) \cdot \nabla_{\widetilde{\Theta}} \log(\mathbb{P}_{\pi^\psi, \widetilde{\theta}_{I_m}}(\tau_{m,l})) \Big) \tag{10}$$

  where $\mathcal{L}_{\text{barrier}}$ is a differentiable barrier loss that heavily penalizes the $\widetilde{\theta}_{I_m}$'s outside Ω.
  Optimize policy:
  Update the policy by taking descending step on gradient estimate:

$$\psi \leftarrow \psi - \frac{1}{ML}\sum_{m=1}^{M}\sum_{l=1}^{L} L_T(\pi^\psi, \tau_{m,l}, \widetilde{\theta}_{I_m}) \cdot \nabla_\psi \log(\mathbb{P}_{\pi^\psi, \widetilde{\theta}_{I_m}}(\tau_{m,l})). \tag{11}$$

end for

In the above algorithm, the gradient estimates are unbiased estimates of the true gradients with respect to ψ, w and $\widetilde{\Theta}$ (shown in Appendix C). We choose N large enough to avoid *mode collapse*, and M, L as large as possible to reduce variance in gradient estimates while fitting the memory constraint. We then find the appropriate large number of optimization iterations so that the variance of the gradient estimates is reduced dramatically by averaging over time. We use Adam optimization (Kingma & Ba, 2014) in taking gradient updates.

Note the decomposition for $\log(\mathbb{P}_{\pi^\psi,\theta'}(\tau))$ in equation 10 and equation 11, where rollout $\tau = \{(x_t, y_t)\}_{t=1}^T$, and

$$\log(\mathbb{P}_{\pi^\psi,\theta'}(\{(x_t,y_t)\}_{t=1}^T)) = \log\Big(\pi^\psi(x_1)\cdot f(y_1|\theta', s_1)\cdot\textstyle\prod_{t=2}^T \pi^\psi(s_t,x_t)\cdot f(y_t|\theta',s_t,x_t)\Big).$$

Here π^ψ and f are only dependent on ψ and $\widetilde{\Theta}$ respectively. During evaluation of a fixed policy π, we are interested in solving $\max_{\theta\in\Omega}\ell'(\pi,\theta)$ by gradient ascent updates like equation 10. The decoupling of π^ψ and f thus enables us to optimize the objective without differentiating through a policy π, which could be non-differentiable like a deterministic algorithm.

B.1 Implementation Details

Training. When training our policies for Best Identification, we warm start the training with optimizing Simple Regret. This is because a random initialized policy performs so poorly that Best Identification is nearly always 1, making it difficult to improve the policy. After training $\pi_{1:K}$ in MAPO (Algorithm 1), we warm start the training of $\widehat{\pi}$ with $\widehat{\pi} = \pi_{\lfloor K/2\rfloor}$. In addition, our generating distribution parameterizations exactly follows from Section 3.1.

Loss functions. Instead of optimizing the approximated quantity from equation 8 directly, we add regularizers to the losses for both the policy and generator. First, we choose the $\mathcal{L}_{\text{barrier}}$ in equation 10 to be $\lambda_{\text{barrier}}\cdot\max\{0, \log(\mathcal{C}(\mathcal{X},\mathcal{Z},\theta)) - \log(r_k)\}$, for some large constant λ_{barrier}. To discourage the policy from over committing to a certain action and/or the generating distribution from covering only a small subset of particles (i.e., mode collapse), we also add negative entropy penalties to both policy's output distributions and $\text{SOFTMAX}(w)$ with scaling factors $\lambda_{\text{Pol-reg}}$ and $\lambda_{\text{Gen-reg}}$.

State representation. We parameterize our state space $\mathcal{S}$ as a flattened $|\mathcal{X}|\times 3$ matrix where each row represents a distinct $x\in\mathcal{X}$. Specifically, at time t the row of s_t corresponding to some $x\in\mathcal{X}$ records the number of times that action x has been taken $\sum_{s=1}^{t-1}\mathbf{1}\{x_s = x\}$, its inverse $(\sum_{s=1}^{t-1}\mathbf{1}\{x_s=x\})^{-1}$, and the sum of the observations $\sum_{s=1}^{t-1}\mathbf{1}\{x_s=x\}y_s$.

Policy MLP architecture. Our policy π^ψ is a multi-layer perceptron with weights ψ. The policy take a $3|\mathcal{X}|$ sized state as input and outputs a vector of size $|\mathcal{X}|$ which is then pushed through a soft-max to create a probability distribution over $\mathcal{X}$. At the end of the game, regardless of the policy's weights, we set $\widehat{z} = \arg\max_{z\in\mathcal{Z}}\langle z,\widehat{\theta}\rangle$ where $\widehat{\theta}$ is the minimum ℓ_2 norm solution to $\arg\min_\theta\sum_{s=1}^T(y_s - \langle x_s,\theta\rangle)^2$.

Our policy network is a simple 6-layer MLP, with layer sizes $\{3|\mathcal{X}|, 256, 256, 256, 256, |\mathcal{X}|\}$ where $3|\mathcal{X}|$ corresponds to the input layer and $|\mathcal{X}|$ is the size of the output layer, which is then pushed through a Softmax function to create a probability over arms. In addition, all intermediate layers are activated with the leaky ReLU activation units with negative slopes of .01. For the experiments for 1D thresholds and 20 Questions, they share the same network structure as mentioned above with $|\mathcal{X}| = 25$ and $|\mathcal{X}| = 100$ respectively.

C Gradient Estimate Derivation

Here we derive the unbiased gradient estimates equation 9, equation 10 and equation 11 in Algorithm 2. Since each the gradient estimates in the above averages over $M\cdot L$ identically distributed trajectories, it is therefore sufficient to show that our gradient estimate is unbiased for a single problem $\widetilde{\theta}_i$ and its rollout trajectory $\{(x_t,y_t)\}_{t=1}^T$.

For a feasible w, using the score-function identity (Aleksandrov et al., 1968)

$$\nabla_w \mathbb{E}_{i\sim\text{SOFTMAX}(w)}\left[\ell(\pi^\psi, \widetilde{\theta}_i)\right] = \mathbb{E}_{i\sim\text{SOFTMAX}(w)}\left[\ell(\pi^\psi, \widetilde{\theta}_i)\cdot\nabla_w \log(\text{SOFTMAX}(w)_i)\right].$$

Observe that if $i \sim \text{SOFTMAX}(w)$ and $\{(x_t, y_t)\}_{t=1}^T$ is the result of rolling out a policy π^ψ on $\widetilde{\theta}_i$ then

$$g^w(i, \{(x_t, y_t)\}_{t=1}^T) := L_T(\pi^\psi, \{(x_t, y_t)\}_{t=1}^T, \widetilde{\theta}_i)\cdot\nabla_w \log(\text{SOFTMAX}(w)_i)$$

is an unbiased estimate of $\nabla_w \mathbb{E}_{i\sim\text{SOFTMAX}(w)}\left[\ell(\pi^\psi, \widetilde{\theta}_i)\right]$.

For a feasible set $\widetilde{\Theta}$, by definition of $\ell(\pi, \theta)$,

$$\begin{aligned}
&\nabla_{\widetilde{\Theta}} \mathbb{E}_{i\sim\text{SOFTMAX}(w)}\left[\ell(\pi^\psi, \widetilde{\theta}_i)\right] \qquad (12)\\
&= \mathbb{E}_{i\sim\text{SOFTMAX}(w)}\left[\nabla_{\widetilde{\Theta}} \mathbb{E}_{\pi,\widetilde{\theta}_i}\left[L_T(\pi, \{(x_t, y_t)\}_{t=1}^T, \widetilde{\theta}_i)\right]\right]\\
&= \mathbb{E}_{i\sim\text{SOFTMAX}(w)}\left[\mathbb{E}_{\pi,\widetilde{\theta}_i}\left[\nabla_{\widetilde{\Theta}} L_T(\pi, \{(x_t, y_t)\}_{t=1}^T, \widetilde{\theta}_i) + L_T(\pi, \{(x_t, y_t)\}_{t=1}^T, \widetilde{\theta}_i)\cdot\nabla_{\widetilde{\Theta}} \log(\mathbb{P}_{\pi^\psi,\widetilde{\theta}_i}(\{(x_t, y_t)\}_{t=1}^T))\right]\right]
\end{aligned}$$

where the last equality follows from chain rule and the score-function identity (Aleksandrov et al., 1968). The quantity inside the expectations, call it $g^{\widetilde{\Theta}}(i, \{(x_t, y_t)\}_{t=1}^T)$, is then an unbiased estimator of $\nabla_{\widetilde{\Theta}} \mathbb{E}_{i\sim\text{SOFTMAX}(w)}\left[\ell(\pi^\psi, \widetilde{\theta}_i)\right]$ given i and $\{(x_t, y_t)\}_{t=1}^T$ are rolled out accordingly. Note that if $\mathcal{L}_{\text{barrier}} \neq 0$, $\nabla_{\widetilde{\Theta}}\mathcal{L}_{\text{barrier}}(\widetilde{\theta}_i, \Omega)$ is clearly an unbiased gradient estimator of $\mathbb{E}_{i\sim\text{SOFTMAX}(w)}[\mathbb{E}_{\pi,\widetilde{\theta}_i}[\mathcal{L}_{\text{barrier}}(\widetilde{\theta}_i, \Omega)]]$ given i and rollout are sampled accordingly.

Likewise, for policy,

$$g^\psi(i, \{(x_t, y_t)\}_{t=1}^T) := L_T(\pi^\psi, \{(x_t, y_t)\}_{t=1}^T, \widetilde{\theta}_i)\cdot\nabla_\psi \log(\mathbb{P}_{\pi^\psi,\widetilde{\theta}_i}(\{(x_t, y_t)\}_{t=1}^T))$$

is an unbiased estimate of $\nabla_\psi \mathbb{E}_{i\sim\text{SOFTMAX}(w)}\left[\ell(\pi^\psi, \widetilde{\theta}_i)\right]$.

D HYPER-PARAMETERS

In this section, we list our hyperparameters. First we define λ_{binary} to be a coefficient that gets multiplied to binary loses, so instead of $\mathbf{1}\{z_*(\theta_*) \neq \widehat{z}\}$, we receive loss $\lambda_{\text{binary}}\cdot\mathbf{1}\{z_*(\theta_*) \neq \widehat{z}\}$. We choose λ_{binary} so that the recieved rewards are approximately at the same scale as SIMPLE REGRET. During our experiments, all of the optimizers are Adam. All budget sizes are $T = 20$. For fairness of evaluation, during each experiment (1D thresholds or 20 Questions), all parameters below are shared for evaluating all of the policies. To elaborate on training strategy proposed in MAPO (Algorithm 1) more, we divide our training into four procedures, as indicated in Table 3:

- **Init.** The initialization procedure takes up a rather small portion of iterations primarily for the purpose of optimizing for $\mathcal{L}_{\text{barrier}}$ so that the particles converge into the constrained difficulty sets. In addition, during the initialization process we initialize and freeze $w = \vec{0}$, thus putting an uniform distribution over the particles. This allows us to utilize the entire set of particles without w converge to only a few particles early on. To initialize $\widetilde{\Theta}$, we sample $2/3$ of the N particles uniformly from $[-1, 1]^{|\mathcal{X}|}$ and the rest $1/3$ of the particles by sampling, for each $i \in [|\mathcal{Z}|]$, $\frac{N}{3|\mathcal{Z}|}$ particles uniformly from $\{\theta : \arg\max_j\langle\theta, z_j\rangle = i\}$. We initialize our policy weights by Xavier initialization with weights sampled from normal distribution and scaled by .01.
- **Regret Training,** $\widetilde{\pi}_i$ Training with SIMPLE REGRET objective usually takes the longest among the Procedures. The primary purpose for this process is to let the policy converge to a reasonable warm start that already captures some essence of the task.
- **Fine-tune** π_i**.** Training with BEST IDENTIFICATION objective run multiple times for each π_i with their corresponding complexity set Θ_i. During each run, we start with a warm started policy, and reinitialize the rest of the models by running the initialization procedure followed by optimizing the BEST IDENTIFICATION objective.

Procedure	Hyper-parameter	Experiment: 1D Threshold $\lvert\mathcal{X}\rvert = 25$	20 Questions $\lvert\mathcal{X}\rvert = 100$	Jester Joke $\lvert\mathcal{X}\rvert = 100$
Init	N_{it}		20000 (all)	
	ψ learning rate		10^{-4} (all)	
	$\widetilde{\Theta}$ learning rate		10^{-3} (all)	
	w learning rate		0 (all)	
Regret Training	N_{it}		480000 (all)	
	ψ learning rate		10^{-4} (all)	
	$\widetilde{\Theta}$ learning rate		10^{-3} (all)	
	w learning rate		10^{-3} (all)	
Fine-tune	N_{it} for $\widetilde{\pi}_i$	200000	0	200000
	N_{it} for π_i	200000	1500000	N/A
	N_{it} for π_*	500000	250000	500000
	ψ learning rate		10^{-4} (all)	
	$\widetilde{\Theta}$ learning rate		10^{-3} (all)	
	w learning rate		10^{-3} (all)	
Adam Optimizer	β_1		.9 (all)	
	β_2		.999 (all)	

Table 3: Number of Iterations and Learning Rates

Procedure	Hyper-parameter	Experiment: 1D Threshold $\lvert\mathcal{X}\rvert = 25$	20 Questions $\lvert\mathcal{X}\rvert = 100$	Jester Joke $\lvert\mathcal{X}\rvert = 100$
Init + Train + Fine-tune	N	$1000 \times \lvert\mathcal{Z}\rvert$	$300 \times \lvert\mathcal{Z}\rvert$	$2000 \times \lvert\mathcal{Z}\rvert$
	M	1000	500	500
	L	10	30	30
	λ_{binary}	7.5	30	30
	$\lambda_{\text{Pol-reg}}$(regret)	.2	.8	.8
	$\lambda_{\text{Pol-reg}}$(fine-tune)	.3	.8	.8
	$\lambda_{\text{Gen-reg}}$	.05	.1	.05
	λ_{barrier}		10^3 (all)	

Table 4: Parallel Sizes and Regularization coefficients

- **Fine-tune** $\widehat{\pi}$ This procedure optimizes equation 3, with baselines $\min_k \ell(\pi_k, \Theta^{(r_k)})$ evaluated based on each π_i learned from the previous procedure. Similar to fine-tuning each individual π_i, we warm start a policy $\pi_{\lfloor K/2 \rfloor}$ and reinitialize w and Θ by running the initialization procedure again.

To provide a general strategy of choosing hyper-parameters, we note that L, firstly, λ_{binary}, $\lambda_{\text{Pol-reg}}$ are primarily parameters tuned for $|\mathcal{X}|$ as the noisiness and scale of the gradients, and entropy over the arms $\mathcal{X}$ grows with the size $|\mathcal{X}|$. Secondly, $\lambda_{\text{Gen-reg}}$ is primarily tuned for $|\mathcal{Z}|$ as it penalizes the entropy over the N arms, which is a multiple of $|\mathcal{Z}|$. Thirdly, learning rate of θ is primarily tuned for the convergence of constraint ρ^* into the restricted class, thus $\mathcal{L}_{\text{barrier}}$ becoming 0 after the specified number of iterations during initialization is a good indicator. Finally, we choose N and M by memory constraint of our GPU. The hyper-parameters for each experiment was tuned with less than 20 hyper-parameter assignments, some metrics to look at while tuning these hyper-parameters includes but are not limited to: gradient magnitudes of each component, convergence of each loss and entropy losses for each regularization term (how close it is to the entropy of a uniform probability), etc.

E POLICY EVALUATION

When evaluating a policy, we are essentially solving the following objective for a fixed policy π:

$$\max_{\theta \in \Omega} \ell(\pi, \theta)$$

where Ω is a set of problems. However, due to non-concavity of this loss function, gradient descent initialized randomly may converge to a local maxima. To reduce this possibility, we randomly initialize many initial iterates and take gradient steps round-robin, eliminating poorly performing trajectories. To do this with a fixed amount of computational resource, we apply the successive halving algorithm from Li et al. (2018). Specifically, we choose hyperparamters: $\eta = 4$, $r = 100$, $R = 1600$ and $s = 0$. This translates to:

- Initialize $|\widetilde{\Theta}| = 1600$, optimize for 100 iterations for each $\widetilde{\theta}_i \in \widetilde{\Theta}$
- Take the top 400 of them and optimize for another 400 iterations
- Take the top 100 of the remaining 400 and optimize for an additional 1600 iterations

We take gradient steps with the Adam optimizer (Kingma & Ba, 2014) with learning rate of 10^{-3} $\beta_1 = .9$ and $\beta_2 = .999$.

F FIGURES AT FULL SCALE

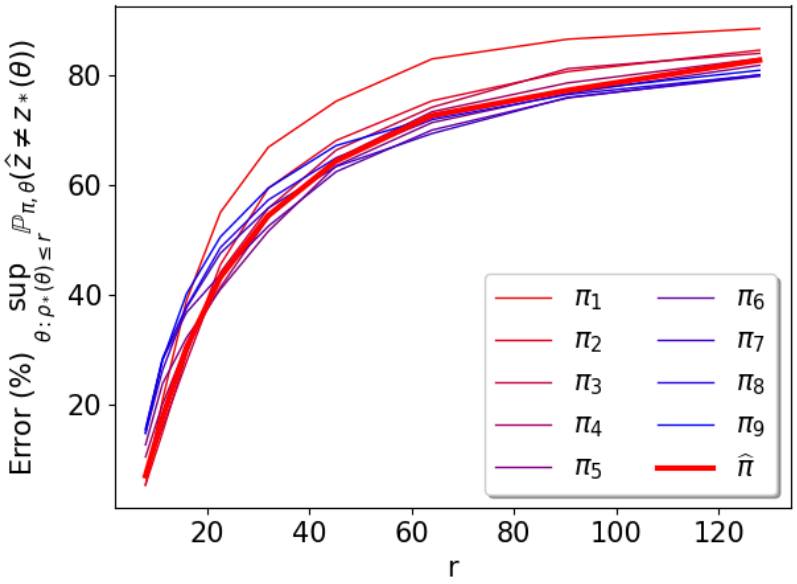

Figure 8: Full scale of Figure 2

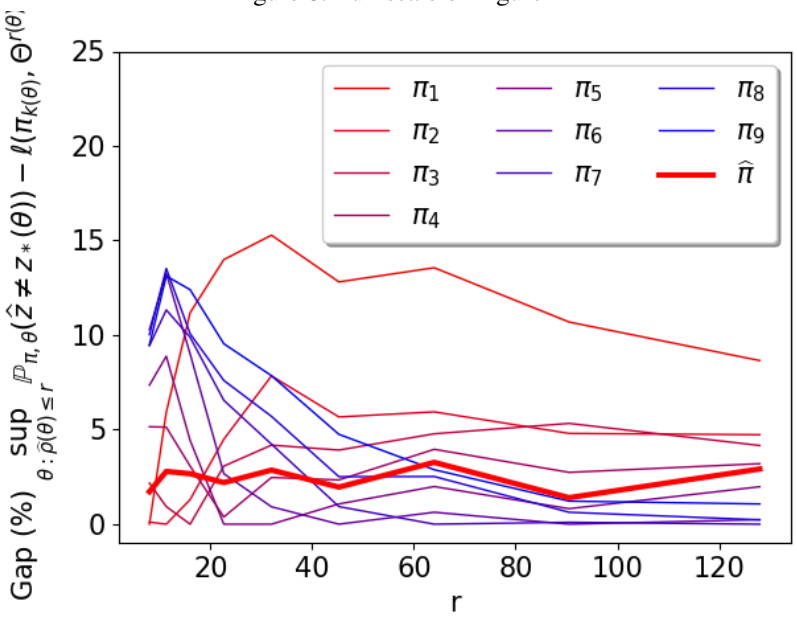

Figure 9: Full scale of Figure 3

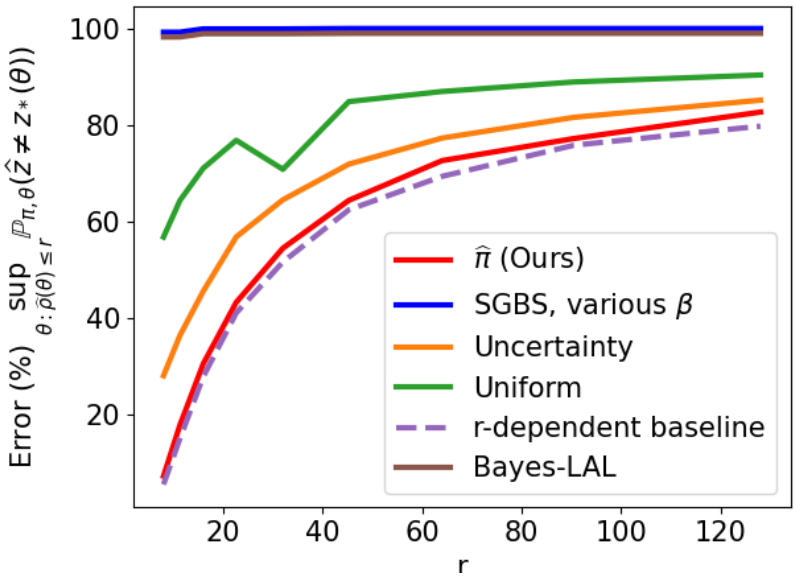

Figure 10: Full scale of Figure 4

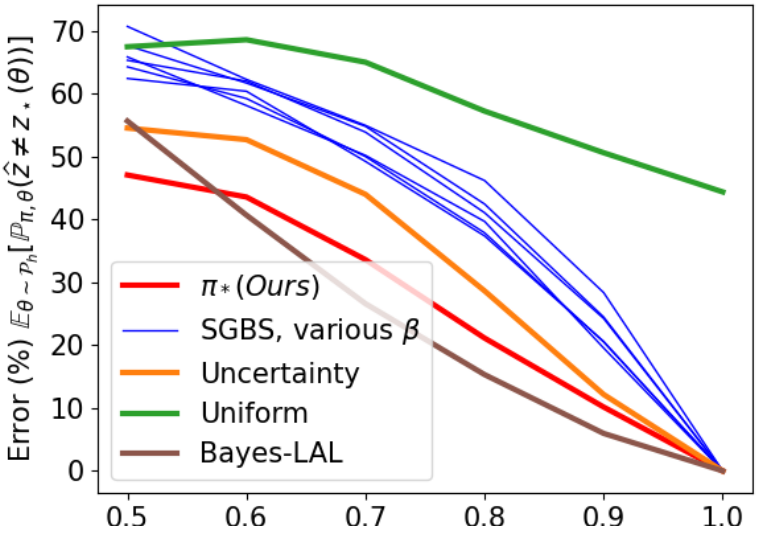

Figure 11: Full scale of Figure 5

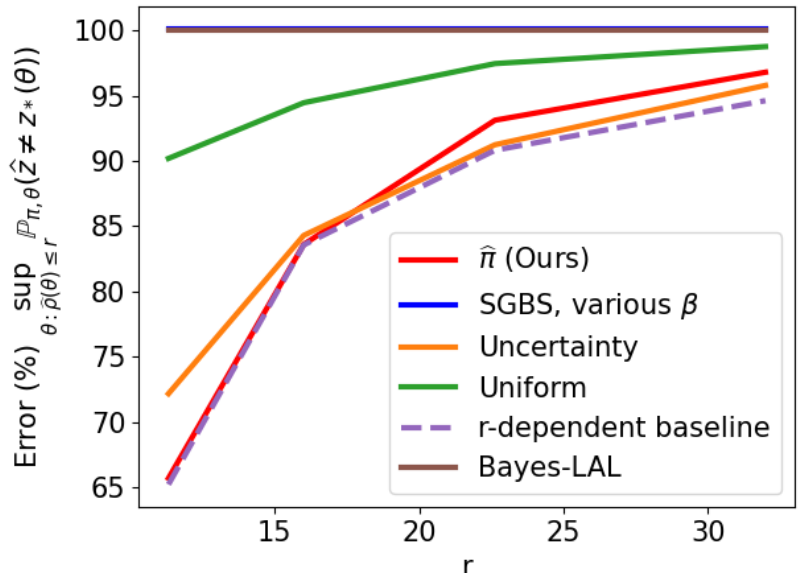

Figure 12: Full scale of Figure 6

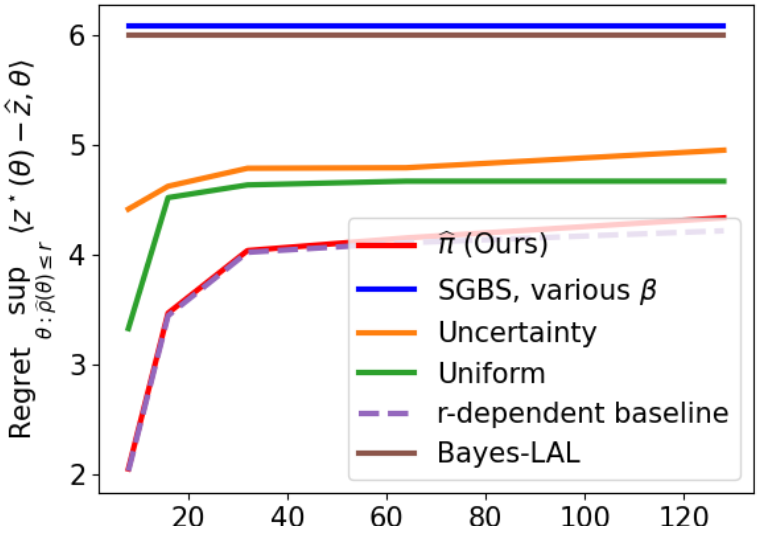

Figure 13: Full scale of Figure 7

G UNCERTAINTY SAMPLING

We define the symmetric difference of a set of binary vectors, SymDiff$(\{z_1, ..., z_n\}) = \{i : \exists j, k \in [n] \;\; s.t., z_j^{(i)} = 1 \wedge z_k^{(i)} = 0\}$, as the dimensions where inconsistencies exist.

Algorithm 4 Uncertainty sampling in very small budget setting

Input: $\mathcal{X}, \mathcal{Z}$
for $t = 1, ..., T$ **do**
  $\widehat{\theta}_{t-1} = \arg\min_\theta \sum_{s=1}^{T} (y_s - \langle x_s, \theta \rangle)^2$
  $\widehat{\mathcal{Z}} = \{z \in \mathcal{Z} : \max_{z' \in \mathcal{Z}} \langle z', \widehat{\theta}_{t-1} \rangle = \langle z, \widehat{\theta}_{t-1} \rangle\}$
  if $|\widehat{\mathcal{Z}}| = 1$ **then**
    $\widehat{\mathcal{Z}}_t = \widehat{\mathcal{Z}} \bigcup \{z \in \mathcal{Z} : \max_{z' \in (\mathcal{Z} \setminus \widehat{\mathcal{Z}})} \langle z', \widehat{\theta}_{t-1} \rangle = \langle z, \widehat{\theta}_{t-1} \rangle\}$
  else
    $\widehat{\mathcal{Z}}_t = \widehat{\mathcal{Z}}$
  end if
  Uniformly sample I_t from SymDiff($\widehat{\mathcal{Z}}_t$)
  Pull x_{I_t} and observe y_t
end for

H Learning to Actively Learn Algorithm

To train a policy under the learning to actively learn setting, we aim to solve for the objective

$$\min_{\psi} \mathbb{E}_{\theta \sim \widehat{\mathcal{P}}}[\ell(\pi^{\psi}, \theta)]$$

where our policy and states are parameterized the same way as Appendix **??** for a fair comparison. To optimize for the parameter, we take gradient steps like equation 11 but with the new sampling and rollout where $\widetilde{\theta}_i \sim \widehat{\mathcal{P}}$. This gradient step follows from both the classical policy gradient algorithm in reinforcement learning as well as from recent learning to actively learn work by Kveton et al. (2020).

Moreover, note that the optimal policy for the objective must be deterministic as justified by deterministic policies being optimal for MDPs. Therefore, it is clear that, under our experiment setting, the deterministic Bayes-LAL policy will perform poorly in the adversarial setting (for the same reason why SGBS performs poorly).

I 20 Questions Setup

Hu et al. (2018) collected a dataset of 1000 celebrities and 500 possible questions to ask about each celebrity. We chose 100 questions out of the 500 by first constructing $\bar{p}'$, $\mathcal{X}'$ and $\mathcal{Z}'$ for the 500 dimensions data, and sampling without replacement 100 of the 500 dimensions from a distribution derived from a static allocation. We down-sampled the number of questions so our training can run with sufficient M and L to de-noise the gradients while being prototyped with a single GPU.

Specifically, the dataset from Hu et al. (2018) consists of probabilities of people answering *Yes / No / Unknown* to each celebrity-question pair collected from some population. To better fit the combinatorial bandit scenario, we re-normalize the probability of getting *Yes / No*, conditioning on the event that these people did not answer *Unknown*. The probability of answering *Yes* to all 500 questions for each celebrity then constitutes vectors $\bar{p}'^{(1)}, ..., \bar{p}'^{(1000)} \in \mathbb{R}^{500}$, where each dimension of a give $\bar{p}_i'^{(j)}$ represents the probability of yes to the ith question about the jth person. The action set $\mathcal{X}'$ is then constructed as $\mathcal{X}' = \{\mathbf{e}_i : i \in [500]\}$, while $\mathcal{Z}' = \{z^{(j)} : [z_i^{(j)}] = \mathbf{1}\{\bar{p}_i^{(j)} > 1/2\}\} \subset \{0, 1\}^{1000}$ are binary vectors taking the majority votes.

To sub-sample 100 questions from the 500, we could have uniformly at random selected the questions, but many of these questions are not very discriminative. Thus, we chose a "good" set of queries based on the design recommended by ρ_* of Fiez et al. (2019). If questions were being answered noiselessly in response to a particular $z \in \mathcal{Z}'$, then equivalently we have that for this setting $\theta = 2z - 1$. Since ρ_* optimizes allocations λ over $\mathcal{X}'$ that would reduce the number of required queries as much as possible (according to the information theoretic bound of (Fiez et al., 2019)) if we want to find a single allocation for all $z' \in \mathcal{Z}$ simultaneously, we can perform the optimization

problem

$$\min_{\lambda \in \Delta^{(|\mathcal{X}|-1)}} \max_{z' \in \mathcal{Z}'} \max_{z \neq z'} \frac{\|z' - z\|^2_{(\sum_i \lambda_i x_i x_i^T)^{-1}}}{((z' - z)^T (2z' - 1))^2}.$$

We then sample elements from $\mathcal{X}'$ according to this optimal λ without replacement and add them to $\mathcal{X}$ until $|\mathcal{X}| = 100$.

J JESTER JOKE RECOMMENDATION SETUP

We consider the Jester jokes dataset of Goldberg et al. (2001) that contains jokes ranging from pun-based jokes to grossly offensive. We filter the dataset to only contain users that rated all 100 jokes, resulting in 14116 users. A rating of each joke was provided on a $[-10, 10]$ scale which was shrunk to $[-1, 1]$. Denote this set of ratings as $\hat{\Theta} = \{\theta_i : i \in [14116], \theta_i \in [-1, 1]^{100}\}$, where θ_i encodes the ratings of all 100 jokes by user i. To construct the set of arms $\mathcal{Z}$, we then clustered the ratings of these users to 10 groups to obtain $\mathcal{Z} = \{z_i : i \in [10], z_i \in \{0, 1\}^{100}\}$ by minimizing the following metric:

$$\min_{\mathcal{Z}:|\mathcal{Z}|=10} \sum_{i=1}^{14116} \max_{z_* \in \{0,1\}^{100}} \langle z_*, \theta_i \rangle - \max_{z \in \mathcal{Z}} \langle z, \theta_i \rangle.$$

To solve for $\mathcal{Z}$, we adapt the $k-means$ algorithm, with the metric above instead of the $L-2$ metric used traditionally.