Optimal Regret Bounds for Policy Optimization in Contextual Bandits

Orin Levy
Blavatnik School of Computer Science
Tel Aviv University
orinlevy@mail.tau.ac.il

Yishay Mansour
Blavatnik School of Computer Science
Tel Aviv University
and Google Research
mansour.yishay@gmail.com

Abstract

We present the first high-probability optimal regret bound for a policy optimization technique applied to the problem of stochastic contextual multi-armed bandit (CMAB) with general function approximation. Our algorithm is both efficient and achieves an optimal regret bound of $\widetilde{O}(\sqrt{K|\mathcal{A}|\log|\mathcal{F}|})$, where K is the number of rounds, $\mathcal{A}$ is the set of arms, and $\mathcal{F}$ is the function class used to approximate the losses. Our results bridge the gap between theory and practice, demonstrating that the widely used policy optimization methods for the contextual bandits problem can achieve a rigorously-proved optimal regret bound. We support our theoretical results with an empirical evaluation of our algorithm.

1 Introduction

Policy Optimization (PO) methods are among the most practical techniques in Reinforcement Learning (RL), with impressive empirical success across a wide range of tasks [46]. The applications of policy optimization span various domains, from recommendations [26] to training robots [41, 42, 32, 25] and control tasks [37, 45, 34], to its notable successes in Large Language Models (LLMs) fine-tuning [52, 24, 40]. Motivated by the great success of context-based policy optimization methods in aligning LLMs with human preferences [43], we revisit the widely studied model of *Contextual Multi-Armed Bandits* (CMAB).

CMAB can model many real-life online tasks where external factors influence the outcome of a selected strategy. This includes online advertisement and recommendation systems, where user preferences affect whether they click on the proposed item or not; healthcare, where a patient's medical history impacts their reaction to a given treatment, and more.

The CMAB problem describes an online decision-making scenario, where external factors affect the decision. We refer to these factors as the *context*. In each round k of K-rounds game, the agent first observes a new context c_k selected from a huge context space $\mathcal{C}$. Given the current context, the agent chooses an action a_k from a finite set of actions $\mathcal{A}$, and suffers a loss ℓ_k associated with the context c_k and the action a_k. We emphasize that the context determines the loss for each action, meaning that for different contexts, the optimal action-selection strategy might be completely different.

The agent aims to minimize the cumulative loss collected throughout a K-round game. Since, in CMAB, the action-selection strategy is context-dependent, we consider it as a contextual *policy*. We measure the performance of the agent in terms of *regret*, which is the cumulative loss of the agent when compared to the cumulative loss of the best contextual policy. Hence, the agent's goal is to minimize regret.

2nd Workshop on Aligning Reinforcement Learning Experimentalists and Theorists (ARLET 2025).

Due to its high relevance, CMAB has been extensively studied, both theoretically and empirically. On the theoretical side, CMAB has been studied under several assumptions and different learning setups, which we will elaborate on later. On the empirical side, the work of Bietti et al. [8] is the most notable.

In this paper, we consider *stochastic* CMAB, in which the context in each round is sampled from an unknown distribution. In this setting, the context space is typically very large, making it likely that the agent will never observe the same context more than once. Since the optimal action depends on the context, achieving sublinear regret is impossible without further assumptions. To address this, the stochastic CMAB literature has focused on the *offline function approximation* framework; a minimal and realistic setting in which the desired regret rate of $\widetilde{O}(\sqrt{K})$ becomes achievable.

In this framework, the agent is provided with a *realizable* and finite loss function class $\mathcal{F}$, where each function maps context-action pairs to an expected loss value. Realizability means that the true loss function is in the class $\mathcal{F}$. Moreover, the agent does not have direct access to the function class, but instead interacts with it through an offline regression oracle, which returns a candidate function that best fits the dataset of observed losses at each round.

Under these assumptions, state-of-the-art algorithms [23, 51, 53] achieve an optimal regret bound of $\widetilde{O}(\sqrt{K|\mathcal{A}|\log|\mathcal{F}|})$ efficiently, assuming access to an efficient offline regression oracle. Xu and Zeevi [53] construct confidence bounds and use a deterministic action selection rule that plays the best optimistic arm in each round. In contrast, Simchi-Levi and Xu [51], Foster et al. [23] introduce the inverse gap weighting (IGW) technique, which defines a stochastic policy where the probability of selecting each action is proportional to its estimated suboptimality under the current loss predictor. They prove, respectively, optimal worst-case and instance-dependent regret bounds.

However, despite the success of PO methods in other domains, the existing stochastic CMAB literature lacks an efficient algorithm that leverages PO within the offline function approximation framework to minimize regret.

PO methods have proven highly effective in practice, as they offer closed-form and intuitive policy update rules that do not require solving optimization problems. Moreover, they naturally encourage exploration due to the stochastic nature of the policy. In addition, PO updates are typically based on exponential weighting (i.e., softmaxing), which results in smoother and more delicate adjustments with respect to the loss predictor updates, compared to the IGW policies. This leads to more stable policy updates with respect to the oracle's outputs.

For all these reasons, we believe that developing PO-based regret minimization algorithms for stochastic CMAB represents an important and currently unaddressed gap in literature, which this paper aims to close.

Summary of our main contributions. Our main goal is to prove a concept: *policy optimization updates on top of function approximation based loss estimators obtain optimal regret bound.* Following that, our main result is a regret minimization algorithm for stochastic CMAB with general offline function approximation, based on policy optimization updates. Our algorithm is both computationally efficient (assuming efficient oracle implementation) and achieves an optimal regret bound of $\widetilde{O}(\sqrt{K|\mathcal{A}|\log|\mathcal{F}|})$, which holds with high probability.

On the technical side, we address the challenge of ensuring sufficient exploration when the agent plays a PO stochastic policy that relies on function approximation-based loss estimators. To address this limitation, we generalize the counterfactual confidence bounds of Xu and Zeevi [53] to stochastic policies and derive compatible counterfactual exploration bonuses that integrate into the policy-optimization update rule. To the best of our knowledge, this is the first work to achieve such guarantees without making additional assumptions regarding the function class (e.g., Eluder dimension [33]).

On the empirical side, we implemented our algorithm and evaluated its performance on the Vowpal Wabbit CMAB benchmark suite [8], demonstrating competitive results compared to state-of-the-art baselines.

Our results show that policy optimization can be adapted to the stochastic CMAB setting with general function approximation, achieving both provable optimal regret bounds and competitive empirical performance.

1.1 Related Literature Review

Contextual Multi-Armed Bandits. The study of Contextual Multi-Armed Bandits (CMAB) has gained significant attention in the last decade, with various assumptions made about the contexts (i.e., adversarially or stochastically chosen), the function classes (policy or loss/rewards classes), and the oracles used, if any. Existing literature can be categorized to two main streams.

The first stream focuses on learning a close-to-best policy within specific given finite policy class Π. It began with the well-known EXP4 algorithm for adversarial CMAB [6], followed by Dudik et al. [16] and Agarwal et al. [5], who explored computationally efficient methods for stochastic CMAB. These two works proved an optimal regret bound of $\widetilde{O}(\sqrt{K|\mathcal{A}|\log|\Pi|})$.

The second stream, which is also the stream this work belongs to, pertains to the realizable function approximation setting. In which, the agent has access to a realizable rewards or losses function class (used to approximate the rewards/losses for each context and action), which she accesses via an optimization oracle. Langford and Zhang [31] considered stochastic contexts and obtained suboptimal regret. Later, Agarwal et al. [3] presented a regressor elimination algorithm and obtained an optimal regret bound of $\widetilde{O}(\sqrt{K|\mathcal{A}|\log|\mathcal{F}|})$, where $\mathcal{F}$ represents a finite set of realizable contextual reward functions used to approximate the reward. The downside of this algorithm is that the runtime complexity of the algorithm scales with $|\mathcal{F}|$. Foster et al. [20] initiated the research of regression oracle based efficient algorithms for stochastic CMAB. Finally, Simchi-Levi and Xu [51] use inverse gap weighting (IGW) techniques to derive optimal worst-case regret and Foster et al. [23] extends them to obtain instance-dependent regret bound. Xu and Zeevi [53] use upper confidence bounds for deterministic policies to also derive optimal worst-case regret. All of the previously mentioned algorithms access the function class using a standard offline least-squares regression oracle, and derive algorithms that obtain optimal regret and can be implemented efficiently, where the efficiency is depending on the run-time complexity of the oracle in use. We, in contrast, apply a policy optimization technique over an optimistic loss approximation, computed for stochastic polices. Similarly, our algorithm is efficient, assuming an efficient offline regression oracle, and obtains optimal regret bounds.

Research has also considered adversarially chosen contexts, notably through the works of [18, 19, 22, 54], who introduced IGW-based policies using online regression oracles for accessing the function class $\mathcal{F}$. These efforts resulted in an optimal regret bound of $\widetilde{O}(\sqrt{K|\mathcal{A}|\mathcal{R}_K(\mathcal{O})})$, where $\mathcal{R}_K(\mathcal{O})$ denotes the regret of the oracle used.

An additional related model is linear CMAB, with Abe and Long [2] being the first to consider this model, and the current state-of-the-art regret minimization algorithms were introduced by Abbasi-Yadkori et al. [1] and Chu et al. [13]. Our framework is more general than linear function approximation.

PO Methods in Reinforcement Learning (RL). The theoretical analysis of PO techniques has been extensively studied, mostly in the basic RL setup of tabular Markov Decision Processes (MDPs), an RL environment with finite state and action spaces, dynamics, and rewards or losses associated with each state-action pair. Even-Dar et al. [17] initiated the theoretical research of policy optimization methods in RL by proposing and analyzing the weighted-majority algorithm for MDPs in the setup of known dynamics, adversarial losses, and full feedback. Later, Neu et al. [38] extended their technique to handle bandit feedback. Shani et al. [48] presented the optimistic policy optimization algorithm and analyzed it in the case of unknown dynamics and bandit feedback while considering both stochastic and adversarial losses. For stochastic losses, they obtained rate-optimal regret; however, in the case of adversarial losses, their regret bound was sub-optimal. Luo et al. [36] improved their result by applying policy optimization with more refined exploration bonuses and obtained rate-optimal regret in the case of unknown dynamics, adversarial losses, and bandit feedback.

Policy optimization has been applied to more complex setups in tabular MDPs, such as aggregated feedback [28], delayed feedback [29, 30], and many other setups with various assumptions.

Beyond the tabular setting, policy optimization has been applied to linear MDPs. Cai et al. [10] obtained optimal-rate regret in the unknown dynamics and adversarial losses model, assuming full feedback. Luo et al. [36] also presented a $\widetilde{O}(K^{2/3})$ regret for the linear case, assuming access to a simulator, where K is the number of episodes. Later, Dai et al. [14] improved the regret to $\widetilde{O}(\sqrt{K})$ under the same assumptions. Sherman et al. [49] obtained a $\widetilde{O}(K^{6/7})$ regret bound efficiently, without assuming access to a simulator, which was later improved by Liu et al. [35] to $\widetilde{O}(K^{4/5})$. Recently, Sherman et al. [50] obtained a rate-optimal regret of $\widetilde{O}(\sqrt{K})$ using a reward-free based warm-up. Later, Cassel and Rosenberg [11] obtained rate-optimal regret using policy optimization without warm-up.

To the best of our knowledge, pure policy optimization has not been studied in RL literature beyond tabular and linear MDPs. Our work is the first attempt to use policy optimization as an exploration method on top of general function approximation for CMABs.

2 Preliminaries and Notations

We consider the problem of *stochastic Contextual Multi-Armed Bandit (CMAB)*, in which we have a finite discrete set of arms $\mathcal{A}$, containing $|\mathcal{A}|$ arms. In the online learning scenario, in each round k of K rounds game, a fresh context $c \in \mathcal{C}$ is sampled from an unknown distribution $\mathcal{D}$ over $\mathcal{C}$. In general, the context space $\mathcal{C}$ can be infinite, but, for mathematical convenience, we assume it is huge but finite[1]. For each context $c \in \mathcal{C}$ and action $a \in \mathcal{A}$ there is an associated stochastic loss with expectation $\ell(c,a) = \mathbb{E}[L(c,a)|c,a]$, where $L(c,a) \in [0,1]$. After observing the context, the agent selects an action to play and suffers the related stochastic loss.

We refer to the agent's action selection strategy as a *(stochastic) contextual policy* $\pi : \mathcal{C} \to \Delta(\mathcal{A})$ that maps contexts to a distribution over actions. We refer to $\pi(c,a)$ as the probability dictated by π to play action a given the context is c. The *optimal policy* $\pi_\star$ satisfies for each $c \in \mathcal{C}$ that $\pi_\star(c,\cdot) \in \arg\min_{p \in \Delta(\mathcal{A})} \langle p, \ell(c,\cdot)\rangle$, where $\langle \cdot,\cdot \rangle$ denotes inner product between these two vectors. The interaction protocol with stochastic CMAB is as follows. In each round $k = 1, 2, \ldots, K$: (1) A fresh context c_k is sampled from $\mathcal{D}$; (2) The agent selects action $a_k \sim \pi_k(c_k,\cdot)$ to play; (3) The agent suffer loss $\ell_k = L(c_k, a_k)$.

Learning objective. We measure the performance of our algorithm in terms of the *(pseudo) regret* in comparison to the optimal policy $\pi_\star$, which is formally defined as $\mathcal{R}_K := \sum_{k=1}^K \langle \pi_k(c_k,\cdot) - \pi_\star(c_k,\cdot), \ell(c_k,\cdot)\rangle$. The expected regret is then $\mathbb{E}\mathcal{R}_K := \sum_{k=1}^K \mathbb{E}_{c_k}[\langle \pi_k(c_k,\cdot) - \pi_\star(c_k,\cdot), \ell(c_k,\cdot)\rangle]$, where the expectation is over the contexts sampled throughout the game. Our goal is to develop an efficient, policy optimization-based regret minimization algorithm for stochastic CMAB.

Additional notations. Throughout the paper, we denote by $|x|$ the absoulote value of any $x \in \mathbb{R}$. Also, for any two distributions p, q over (finite) support $\mathcal{X}$ we denote by $d_{KL}(p||q)$ the Kullback-Leibler (KL) divergence between p and q that is defined as $d_{KL}(p||q) := \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}$.

2.1 Offline Function Approximation

As previous literature for stochastic CMAB (e.g., [51, 53, 4, 31]) shows, an offline function approximation is necessary to obtain non-trivial regret for stochastic CMAB. Hence, in

[1]This assumption is standard in CMAB literature, as the extension to infinite context space is straightforward. The goal is to obtain a regret bound that is independent of the cardinality of $\mathcal{C}$.

compatible with previous works, we assume access to a realizable and finite[2] losses function class $\mathcal{F} \subseteq \mathcal{C} \times \mathcal{A} \to [0,1]$ via an offline least-squares regression oracle denoted $\mathcal{O}_{\text{sq}}^{\mathcal{F}}$.

Assumption 2.1 (Realizability)**.** There exist $f_\star \in \mathcal{F}$ such that for all $(c,a) \in \mathcal{C} \times \mathcal{A}$ it holds that $f_\star(c,a) = \ell(c,a)$.

The function class is being accessed via an offline least-squares regression oracle, next defined.

Assumption 2.2 (Offline least-squares regression oracle)**.** Given a dataset $D_n = \{(c_i, a_i, \ell_i)\}_{i=1}^n$ we assume access to an offline oracle $\mathcal{O}_{\text{sq}}^{\mathcal{F}}$ that returns a candidate solution $\hat{f}_{n+1}$ to the following Empirical Risk Minimization (ERM) problem with respect to the square loss: $\hat{f}_{n+1} \in \arg\min_{f \in \mathcal{F}} \sum_{i=1}^n (f(c_i, a_i) - \ell_i)^2$.

Access to an offline least-square regression oracle is also a commonly used assumption in stochastic CMAB literature (see, e.g., [51, 53, 4]). The reason behind the choice of least-squares regression for the loss approximation is that least-squares regression is both compatible with approximating an expectation given stochastic examples and can be implemented efficiently for many function classes, with the clear example of a linear functions, for which the solution is given by a closed form.

Lemma 5 in [53] presents a uniform convergence guarantee with respect to any function sequence. The lemma is given in Lemma B.4. An immediate corollary of it implies a uniform convergence guarantee of an offline least squares regression oracle. The corollary is below, for proof see Corollary B.5.

Corollary 2.3 (uniform convergence of offline least-squares regression)**.** *Let $\hat{f}_2, \ldots \in \mathcal{F}$ denote the sequence of least squares minimizers and let $\pi_1, \pi_2, \ldots$ denote the sequence of contextual played policies. The following holds for any $\delta \in (0,1)$ and $t \geq 2$ with probability at least $1 - \delta/4$.*

$$\sum_{i=1}^{t-1} \mathbb{E}_{c_i}\left[\mathbb{E}_{a_i \sim \pi_i(c_i,\cdot)}\left[\left(\hat{f}_t(c_i,a_i) - f_\star(c_i,a_i)\right)^2\right]\right] \leq 68 \log(4|\mathcal{F}|t^3/\delta).$$

3 Algorithm and Main Result

Algorithm 1 presents an optimistic policy optimization algorithm for stochastic CMABs.

An integral ingredient of our algorithm is the generalization of the *counterfactual exploration bonuses* [53] to stochastic policies. We define the exploration bonus for arm a at round k and context c as

$$b_k^\beta(c,a) = \min\left\{1, \frac{\beta/2}{1 + \sum_{i=1}^{k-1} \pi_i(c,a)}\right\}, \tag{1}$$

where $\beta = O(\sqrt{K})$ is a tunable parameter controlling the overall exploration level.

The intuition behind these bonuses is that they quantify how well explored each arm is for the current context. To see this, consider a counterfactual scenario in which the same context c had appeared in all previous rounds. At each past round $i = 1, \ldots, k-1$, the agent would have sampled arms according to the past policies $\pi_1(c,\cdot), \pi_2(c,\cdot), \ldots, \pi_{k-1}(c,\cdot)$. Hence, the imaginary expected number of times arm a would have been played for this context is $\mathbb{E}\left[N_{k-1}(c,a) \mid c\right] = \mathbb{E}\left[\sum_{i=1}^{k-1} \pi_i(c,a) \,\middle|\, c\right]$. This counterfactual quantity serves as a realization of the expected number of times arm a would have been chosen for c in past rounds. When this quantity is large, the algorithm has implicitly gathered substantial information about arm a for the context c, and thus assigns it a small bonus. Conversely, when it is small, the bonus remains large, encouraging further exploration. The behavior of this bonus is analogous to the standard exploration bonus in stochastic MABs [6].

[2]The assumption is standard as the extension to infinite function classes is also straightforward, see [47] for offline regression with infinite function classes. We consider finite function classes for the sake of mathematical convenience and readability.

Based on this optimistic exploration principle, our algorithm performs policy optimization by applying exponential policy improvements, as is standard in RL literature (see, e.g., [48]). During initialization, the agent plays the uniform distribution over actions for any context that may be sampled. Given the first observed context, the agent samples an action uniformly at random, plays it, and updates the oracle with the resulting observation. Then, for rounds $t = 2, 3, \ldots, K$, the agent computes the policies as described next. She observes the current context c_t and counterfactually evaluates all past policies $\pi_2(c_t, \cdot), \ldots, \pi_{t-1}(c_t, \cdot)$ in order to compute the exploration bonus and subsequently derive $\pi_t(c_t, \cdot)$. For each $k = 1, 2, \ldots, t-1$, to compute the next policy $\pi_{k+1}(c_t, \cdot)$, the agent uses the approximated loss returned by the oracle at round k (computed from the data collected in rounds $\{1, \ldots, k-1\}$) denoted by $\hat{f}_k$. To this approximation, she adds the counterfactual exploration bonus from Equation (1), which depends on the probabilities of all past policies $\pi_1, \ldots, \pi_{k-1}$ for playing each specific action a under the current context c_t. The next policy is then defined by an exponential improvement of the current policy with respect to its optimistic loss approximation.

Algorithm 1 Optimistic Policy Optimization for CMAB (OPO-CMAB)

Inputs: learning rate η , number of episodes K, tuning parameter β.
Initialization:
- π_1 is the uniform distribution over actions for all $c \in \mathcal{C}$.
- $f_1 \in \mathcal{F}$ is chosen arbitrarily.

for $t = 1, 2, \ldots, K$ **do**
  Observe fresh context c_t.
  if $t \geq 2$ **then**
    for $k = 1, \ldots, t-1$ **do**
      Compute for all $a \in \mathcal{A}$:
      $\hat{\ell}_k(c_t, a) = \max\{0, \hat{f}_k(c_t, a) - b_k^\beta(c_t, a)\}$ where $b_k^\beta(c, a) = \min\left\{1, \frac{\beta/2}{1+\sum_{i=1}^{k-1} \pi_i(c,a)}\right\}$.
      {Policy Evaluation}
      Compute for all $a \in \mathcal{A}$:
      $\pi_{k+1}(c_t, a) = \frac{\pi_k(c_t,a) \exp\left(-\eta \hat{\ell}_k(c_t,a)\right)}{\sum_{a' \in \mathcal{A}} \pi_k(c_t,a') \exp\left(-\eta \hat{\ell}_k(c_t,a')\right)}$ {Policy Improvement}
    end for
  end if
  Sample action $a_t \sim \pi_t(c_t, \cdot)$, play it and observe ℓ_t.
  Update the loss approximation using the optimization oracle $\mathcal{O}_{\text{sq}}^{\mathcal{F}}$:

$$\hat{f}_{t+1} \in \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{t} (f(c_i, a_i) - \ell_i)^2$$

end for

Theorem 3.1 (Regret bound)**.** *For an appropriate choice of β, η, we have with probability at least $1 - \delta$ that*

$$\mathcal{R}_K \leq \widetilde{O}\Big(\sqrt{K|\mathcal{A}| \log(|\mathcal{F}|/\delta)}\Big).$$

Discussion. Our algorithm integrates a policy optimization update rule with offline function approximation for loss prediction. We employ an optimistic approximation by incorporating exploration bonuses tailored for both offline function approximation and stochastic policies. Intuitively, the bonus assigned to each action reflects the counterfactual expected number of samples that would have been collected for that action if the current context had been observed throughout all previous rounds. The loss estimators are clipped to $[0, 1]$ to ensure that it is bounded. The algorithm is computationally efficient, assuming the availability of an efficient oracle implementation.

4 Regret Analysis

In this section, we analyze the regret of Algorithm 1, proving Theorem 3.1. We follow the regret decomposition of Shani et al. [48] but adapt it to function approximation in the model of stochastic CMAB. We start our analysis by noting that with probability at least $1-\delta/2$, $\mathcal{R}_K \leq \mathbb{E}\mathcal{R}_K + 2\sqrt{2K\log(4/\delta)}$. As the contexts are iid, and the policies $\{\pi_k\}_{k=1}^K$ are determined completely by the history, the above is a direct implication of Azuma-Hoeffding's inequality. (Full proof is given in Corollary B.3).

Hence, we focus on bounding the expected regret, which will imply a high probability regret bound, by the above. To obtain the desired bound, we decompose the expected regret as

$$\mathbb{E}\mathcal{R}_K = \sum_{k=1}^{K} \mathbb{E}_{c_k}\Big[\Big\langle \pi_k(c_k,\cdot), \ell(c_k,\cdot) - \hat{\ell}_k(c_k,\cdot)\Big\rangle\Big] \tag{2}$$

$$+ \sum_{k=1}^{K} \mathbb{E}_{c_k}\Big[\Big\langle \pi_k(c_k,\cdot) - \pi_\star(c_k,\cdot), \hat{\ell}_k(c_k,\cdot)\Big\rangle\Big] \tag{3}$$

$$+ \sum_{k=1}^{K} \mathbb{E}_{c_k}\Big[\Big\langle \pi_\star(c_k,\cdot), \hat{\ell}_k(c_k,\cdot) - \ell(c_k,\cdot)\Big\rangle\Big] \tag{4}$$

where term (2) stands for the expected approximation error of the loss with respect to the played policies $\{\pi_k\}_{k=1}^K$, term (3) is the sub-optimality of the true optimal policy $\pi_\star$ on the approximated loss function in each round, and term (4) is the expected approximation error of the loss with respect to the optimal policy. In what follows, we bound each of the terms separately. We start our analysis by stating that the sum of bonuses is upper bounded, in expectation over the context for the played policies. This lemma will be used to bound (2).

Lemma 4.1 (Bonuses bound)**.** *The following holds true.*

$$\sum_{k=1}^{K} \mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}} \pi_k(c_k,a) b_k^\beta(c_k,a)\right] \leq \beta|\mathcal{A}|\log(K+1).$$

We derived the above using an algebraic calculation of logarithmic sums (see Lemma A.1). Using this lemma, we obtain the following upper bound of term (2), stated in the next lemma. For full proof of the lemma, see Lemma A.2 in Appendix.

Lemma 4.2 (Term (2) bound)**.** *For any $\delta \in (0,1)$, let $\beta = \sqrt{\frac{K34\log(4|\mathcal{F}|K^3/\delta)}{|\mathcal{A}|}}$. Then, with probability at least $1-\delta/4$, it holds that*

$$\sum_{k=1}^{K} \mathbb{E}_{c_k}\Big[\Big\langle \pi_k(c_k,\cdot), \ell(c_k,\cdot) - \hat{\ell}_k(c_k,\cdot)\Big\rangle\Big] \leq \widetilde{O}\Big(\sqrt{K|\mathcal{A}|\log(|\mathcal{F}|/\delta)}\Big).$$

The next upper bound of term (3) follows directly from Online Mirror Descent (OMD) analysis, for proof see Lemma A.3.

Lemma 4.3 (Term (3) bound)**.** *For the choice in $\eta = \sqrt{\log|\mathcal{A}|/K}$, the following holds true.*

$$\sum_{k=1}^{K} \mathbb{E}_{c_k}\Big[\Big\langle \pi_k(c_k,\cdot) - \pi_\star(c_k,\cdot), \hat{\ell}_k(c_k,\cdot)\Big\rangle\Big] \leq O(\sqrt{K\log|\mathcal{A}|}).$$

Lastly, we bound term (4) using a similar proof technique to that of term (2), see Lemma A.4.

Lemma 4.4 (Term (4) bound)**.** *For any $\delta \in (0,1)$ let $\beta = \sqrt{\frac{K34\log(4|\mathcal{F}|K^3/\delta)}{|\mathcal{A}|}}$. Then, the following holds with probability at least $1-\delta/4$.*

$$\sum_{k=1}^{K} \mathbb{E}_{c_k}\Big[\Big\langle \pi_\star(c_k,\cdot), \hat{\ell}_k(c_k,\cdot) - \ell(c_k,\cdot)\Big\rangle\Big] \leq \widetilde{O}\Big(\sqrt{K|\mathcal{A}|\log(|\mathcal{F}|/\delta)}\Big).$$

Using all the above, we prove Theorem 3.1.

Proof of Theorem 3.1. By taking a union bound over the events specified in Lemmas 4.2 and 4.4, and combine it with the result of Lemma 4.3, we obtain that with probability at least $1-\delta/2$, $\mathbb{E}\mathcal{R}_k \leq \widetilde{O}\Big(\sqrt{K|\mathcal{A}|\log(|\mathcal{F}|/\delta)}\Big)$, which implies that with probability at least $1-\delta$ it holds that $\mathcal{R}_k \leq \widetilde{O}\Big(\sqrt{K|\mathcal{A}|\log(|\mathcal{F}|/\delta)}\Big)$, as desired. □

5 Experiments

As is standard in the CMAB literature, we evaluate our algorithm using the Vowpal Wabbit (VW)[3] benchmark suite [8], which implements the practical CMAB algorithms. The state-of-the-art empirical comparison on this benchmark is by Foster and Krishnamurthy [19], who extensively evaluated the efficient, regression oracle based CMAB algorithms (`SquareCB`, `AdaCB`, `RegCB`) and a supervised learning baseline `Supervised` against their proposed algorithm `FastCB`, across various hyper-parameters and both square and logistic losses. They found `FastCB` with logistic loss performed best, followed by `SquareCB` (logistic and squared loss), while `AdaCB` and `RegCB` lagged behind. Following this setup, we compare our method `OPO-CMAB` to `FastCB`, `SquareCB`, `RegCB`, `AdaCB`, and the supervised baseline `Supervised`[4].

Implementation Details. Following prior work, all algorithms are implemented in VW using an online regression oracle for both linear regression and logistic regression. We integrated `OPO-CMAB` into VW and implemented it as in Algorithm 1, with small changes: (i) the exploration bonus is set adaptively as $\beta_k = \gamma\sqrt{k/|\mathcal{A}|}$ with γ being a tuned hyper-parameter; (ii) η is now also a tuned hyper-parameter and (iii) support also logistic regression for better accommodating to multi-class and multi-label tasks as in the tested datasets. Further details about implementation are provided in Appendix C.2.

Experimental setup and evaluation. Previous works [19, 8] evaluate all algorithms on more than 500 multi-class and multi-label classification datasets from OpenML. Due to computational limitations, we evaluated all algorithms on 18 relatively small datasets arbitrarily selected among those 515 datasets. As these are multiclass classification datasets, we simulate bandit feedback as the agent receives loss 0 for a correct prediction and 1 otherwise, similarly to the prior work.

Performance is measured by the *Progressive Validation (PV)* loss [9], which for algorithm `A` and a dataset of K examples defined as $L_{PV}(\mathtt{A}, K) = \frac{1}{K}\sum_{k=1}^{K}\ell_k(a_k)$. By definition, the PV-loss directly reflects regret differences between algorithms.

Following previous work, we tuned the hyper-parameters of each algorithm by choosing the configuration that achieves the lowest final PV-loss for each dataset. The set of tested values for each parameter can be found in Table 1, Appendix C. The parameters values for the known algorithms were chosen as described in previous work. For our `OPO-CMAB`, we select γ and η values from a small set of relevant values on a discretized 10-scale grid. The final parameter values chosen for each algorithm and dataset can be found in Table 2, Appendix C. We then run the selected configuration on 10 random permutations of the related dataset. In the presented plots, we compare the averaged PV-loss decay curve of each algorithm as well as Standard deviation (Std) on three selected datasets, in which the supervised baseline converged to non-trivial final PV-loss.

In Appendix C, we support those plots by presenting an approximate Z-test, to evaluate the statistical significance of the results (Figure 3) and a table summarizes the mean differences in the final PV-loss of each algorithm from the supervised baseline (Figure 2).

Discussion of results. In Figure 1a, all algorithms appear to reach a plateau after 100 examples, which is expected for a 3-armed CMAB instance. As anticipated, `Supervised` achieves the lowest PV-losses throughout the run, closely followed by `OPO-CMAB`, with `RegCB`

[3] `https://vowpalwabbit.org/`

[4] Our code is publicly available at https://github.com/orinL/OPO_CMAB_Experiments_code

slightly higher. In fourth place is `FastCB`. `SquareCB` ranks fifth, and `AdaCB` is last, both with a notable gap from the other algorithms. (See Figure 2 in Appendix C).

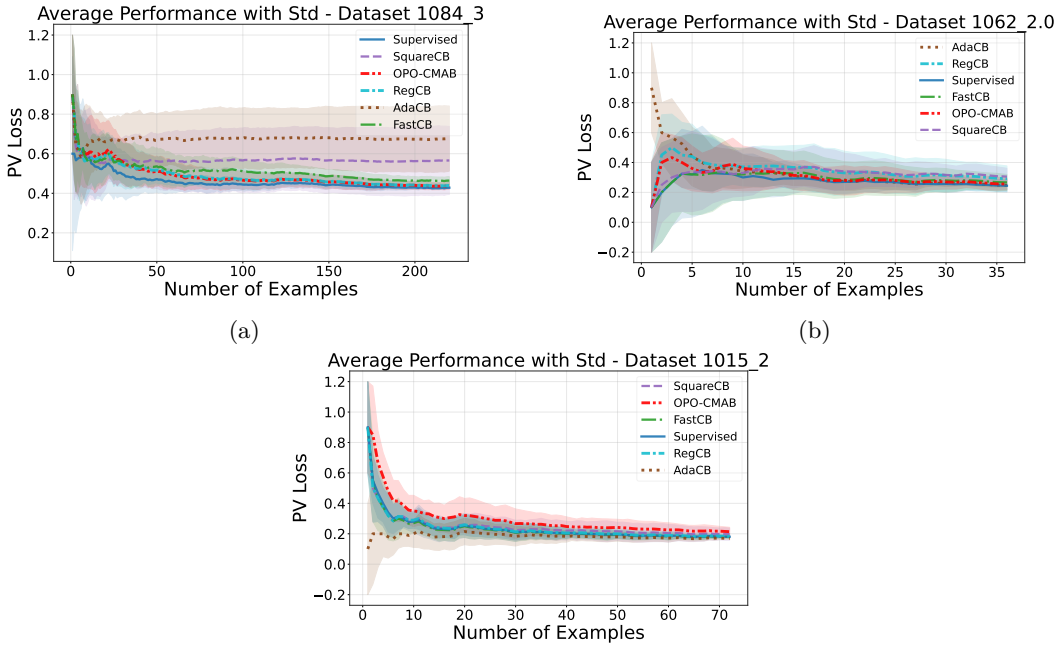

Figure 1: Average performance comparison on datasets 1084, 1062, 1015.

In this case, approximate Z-tests on the averaged PV-losses across permutations reveal that a significant fraction (9/15) of the wins, when comparing each pair of algorithms on this dataset, are statistically significant. (See Figure 3 in Appendix C).

In Figure 1b, it appears that all algorithms nearly reach a plateau, as expected for 2-armed CMAB. `Supervised` achieves the lowest PV-losses throughout the run, followed by `AdaCB` and `OPO-CMAB`, which have very similar PV loss values. In fourth place is `FastCB`, then `RegCB` with higher PV-losses, and finally `SquareCB`.

The final mean PV-losses of `AdaCB` and `OPO-CMAB` are the closest to `Supervised`, with relatively small differences (as shown in Figure 2, Appendix C). In addition, the approximate Z-tests on the averaged PV losses across permutations show that none of the pairwise differences on this dataset are statistically significant. (See Figure 3 in Appendix C).

In Figure 1c, all algorithms appear to reach a plateau after 50 examples, as expected for a 2-armed CMAB. `AdaCB` achieves the lowest PV-loss throughout the run, followed closely by `FastCB`, `Supervised`, and `RegCB`, which have very similar PV-loss values; `SquareCB` performs slightly worse. For this dataset, our `OPO-CMAB` shows slightly higher PV-loss values than the other algorithms, but its performance remains competitive.

The final mean PV-loss differences between all algorithms to the supervised baseline are relatively small, further indicating similar performance (as shown in Figure 2 in Appendix C). Additionally, by the results of approximate Z-tests on the mean PV-loss across permutations, for every pair of algorithms, none of the wins are statistically significant (see Figure 3 in Appendix C).

Overall, the experiments demonstrate that all algorithms exhibit similar performance on the tested datasets, with no significantly evidence that any one algorithm outperforms the others. This supports our claim that our algorithm is competitive with previously known CMAB algorithms. For more details regarding experiments, see Appendix C.

6 Conclusions and Discussion

In this work we aim to establish a fundamental principle: policy optimization updates that use function approximation-based loss estimators can achieve optimal regret bounds for stochastic contextual multi-armed bandits (CMABs).

Our main contribution is a regret minimization algorithm for stochastic CMABs with general offline function approximation, based on policy optimization updates. The algorithm is computationally efficient (assuming access to an efficient oracle) and achieves an optimal regret bound of $\widetilde{O}(\sqrt{K|\mathcal{A}|\log|\mathcal{F}|})$ with high probability.

A natural direction for future work is to extend our theoretical results beyond CMABs to more general settings such as contextual RL, RL with large state spaces, and other rich RL problems. We hope our results will inspire further research in these areas.

Empirically, our method demonstrates competitive performance across a range of multiclass datasets, with no statistically significant evidence that any algorithm consistently outperforms the others. However, our implementation of `OPO-CMAB` in VW is limited by its $O(T^2)$ runtime and $O(T)$ space complexity. Since our primary contribution is theoretical, we focused on demonstrating applicability and competitiveness, rather than developing an efficient practical implementation. Thus, we did not explore optimizations for `OPO-CMAB` in this work.

Nevertheless, we believe our technique could inspire many efficient heuristics involving policy optimization updates with function approximation-based loss predictors. Potential directions include using noisy batching instead of exhaustive computation of exploration bonuses, heuristic look-back rather than recovering all history, and more. Developing such practical and scalable heuristics of our technique is a promising direction for future research and could have a significant impact in many applications.

Acknowledgments

OL thanks Alon Peled-Cohen for fruitful discussions and Idan Schwartz for his assistance with running the experiments. We also thank the reviewers for their thoughtful comments.

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 882396), by the Israel Science Foundation, the Yandex Initiative for Machine Learning at Tel Aviv University and a grant from the Tel Aviv University Center for AI and Data Science (TAD).

References

[1] Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári. Improved algorithms for linear stochastic bandits. *Advances in neural information processing systems*, 24, 2011.

[2] N. Abe and P. M. Long. Associative reinforcement learning using linear probabilistic concepts. In *ICML*, pages 3–11. Citeseer, 1999.

[3] A. Agarwal, M. Dudík, S. Kale, J. Langford, and R. Schapire. Contextual bandit learning with predictable rewards. In *Artificial Intelligence and Statistics*, pages 19–26. PMLR, 2012.

[4] A. Agarwal, M. Dudik, S. Kale, J. Langford, and R. Schapire. Contextual bandit learning with predictable rewards. In N. D. Lawrence and M. Girolami, editors, *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, volume 22 of *Proceedings of Machine Learning Research*, pages 19–26, La Palma, Canary Islands, 21–23 Apr 2012. PMLR.

[5] A. Agarwal, D. Hsu, S. Kale, J. Langford, L. Li, and R. Schapire. Taming the monster: A fast and simple algorithm for contextual bandits. In E. P. Xing and T. Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32, pages 1638–1646, 2014.

[6] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002.

[7] P. L. Bartlett, E. Hazan, and A. Rakhlin. Adaptive online gradient descent. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pages 65–72. Curran Associates, Inc., 2007. URL https://proceedings.neurips.cc/paper/2007/hash/afd4836712c5e77550897e25711e1d96-Abstract.html.

[8] A. Bietti, A. Agarwal, and J. Langford. A contextual bandit bake-off. *J. Mach. Learn. Res.*, 22:133:1–133:49, 2021. URL https://jmlr.org/papers/v22/18-863.html.

[9] A. Blum, A. Kalai, and J. Langford. Beating the hold-out: Bounds for k-fold and progressive cross-validation. In S. Ben-David and P. M. Long, editors, *Proceedings of the Twelfth Annual Conference on Computational Learning Theory, COLT 1999, Santa Cruz, CA, USA, July 7-9, 1999*, pages 203–208. ACM, 1999. doi: 10.1145/307400.307439. URL https://doi.org/10.1145/307400.307439.

[10] Q. Cai, Z. Yang, C. Jin, and Z. Wang. Provably efficient exploration in policy optimization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1283–1294. PMLR, 2020. URL http://proceedings.mlr.press/v119/cai20d.html.

[11] A. Cassel and A. Rosenberg. Warm-up free policy optimization: Improved regret in linear markov decision processes. In A. Globersons, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/05d42b7bd130ecbc57fdf02cbdcd370e-Abstract-Conference.html.

[12] N. Cesa-Bianchi and G. Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006. ISBN 978-0-521-84108-5. doi: 10.1017/CBO9780511546921. URL https://doi.org/10.1017/CBO9780511546921.

[13] W. Chu, L. Li, L. Reyzin, and R. Schapire. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 208–214. JMLR Workshop and Conference Proceedings, 2011.

[14] Y. Dai, H. Luo, C. Wei, and J. Zimmert. Refined regret for adversarial mdps with linear function approximation. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 6726–6759. PMLR, 2023. URL https://proceedings.mlr.press/v202/dai23b.html.

[15] J. C. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, 2011. doi: 10.5555/1953048.2021068. URL https://dl.acm.org/doi/10.5555/1953048.2021068.

[16] M. Dudik, D. Hsu, S. Kale, N. Karampatziakis, J. Langford, L. Reyzin, and T. Zhang. Efficient optimal learning for contextual bandits. *arXiv preprint arXiv:1106.2369*, 2011.

[17] E. Even-Dar, S. M. Kakade, and Y. Mansour. Online markov decision processes. *Math. Oper. Res.*, 34(3):726–736, 2009. doi: 10.1287/MOOR.1090.0396. URL https://doi.org/10.1287/moor.1090.0396.

[18] D. Foster and A. Rakhlin. Beyond ucb: Optimal and efficient contextual bandits with regression oracles. In *International Conference on Machine Learning*, pages 3199–3210. PMLR, 2020.

[19] D. J. Foster and A. Krishnamurthy. Efficient first-order contextual bandits: Prediction, allocation, and triangular discrimination. *Advances in Neural Information Processing Systems*, 34:18907–18919, 2021.

[20] D. J. Foster, A. Agarwal, M. Dudík, H. Luo, and R. E. Schapire. Practical contextual bandits with regression oracles. In J. G. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1534–1543. PMLR, 2018. URL http://proceedings.mlr.press/v80/foster18a.html.

[21] D. J. Foster, S. Kale, H. Luo, M. Mohri, and K. Sridharan. Logistic regression: The importance of being improper. *CoRR*, abs/1803.09349, 2018. URL http://arxiv.org/abs/1803.09349.

[22] D. J. Foster, S. M. Kakade, J. Qian, and A. Rakhlin. The statistical complexity of interactive decision making. *arXiv preprint arXiv:2112.13487*, 2021.

[23] D. J. Foster, A. Rakhlin, D. Simchi-Levi, and Y. Xu. Instance-dependent complexity of contextual bandits and reinforcement learning: A disagreement-based perspective. In M. Belkin and S. Kpotufe, editors, *Conference on Learning Theory, COLT 2021, 15-19 August 2021, Boulder, Colorado, USA*, volume 134 of *Proceedings of Machine Learning Research*, page 2059. PMLR, 2021. URL http://proceedings.mlr.press/v134/foster21a.html.

[24] A. Glaese, N. McAleese, M. Trebacz, J. Aslanides, V. Firoiu, T. Ewalds, M. Rauh, L. Weidinger, M. J. Chadwick, P. Thacker, L. Campbell-Gillingham, J. Uesato, P. Huang, R. Comanescu, F. Yang, A. See, S. Dathathri, R. Greig, C. Chen, D. Fritz, J. S. Elias, R. Green, S. Mokrá, N. Fernando, B. Wu, R. Foley, S. Young, I. Gabriel, W. Isaac, J. Mellor, D. Hassabis, K. Kavukcuoglu, L. A. Hendricks, and G. Irving. Improving alignment of dialogue agents via targeted human judgements. *CoRR*, abs/2209.14375, 2022. doi: 10.48550/ARXIV.2209.14375. URL https://doi.org/10.48550/arXiv.2209.14375.

[25] S. Gu, E. Holly, T. P. Lillicrap, and S. Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE International Conference on Robotics and Automation, ICRA 2017, Singapore, Singapore, May 29 - June 3, 2017*, pages 3389–3396. IEEE, 2017. doi: 10.1109/ICRA.2017.7989385. URL https://doi.org/10.1109/ICRA.2017.7989385.

[26] J. Jain, Y. Ramaswamy, L. Gudala, R. R. Hossein, and G. S. Satya. Personalized movie recommendation system based on proximal policy optimization. In *2025 3rd International Conference on Data Science and Information System (ICDSIS)*, pages 1–6. IEEE, 2025.

[27] N. Karampatziakis and J. Langford. Online importance weight aware updates. In F. G. Cozman and A. Pfeffer, editors, *UAI 2011, Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence, Barcelona, Spain, July 14-17, 2011*, pages 392–399. AUAI Press, 2011. URL https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=2234&proceeding_id=27.

[28] T. Lancewicki and Y. Mansour. Near-optimal regret using policy optimization in online mdps with aggregate bandit feedback. *CoRR*, abs/2502.04004, 2025. doi: 10.48550/ARXIV.2502.04004. URL https://doi.org/10.48550/arXiv.2502.04004.

[29] T. Lancewicki, A. Rosenberg, and Y. Mansour. Learning adversarial markov decision processes with delayed feedback. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 7281–7289. AAAI Press, 2022. doi: 10.1609/AAAI.V36I7.20690. URL https://doi.org/10.1609/aaai.v36i7.20690.

[30] T. Lancewicki, A. Rosenberg, and D. Sotnikov. Delay-adapted policy optimization and improved regret for adversarial MDP with delayed bandit feedback. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 18482–18534. PMLR, 2023. URL https://proceedings.mlr.press/v202/lancewicki23a.html.

[31] J. Langford and T. Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007. URL https://proceedings.neurips.cc/paper_files/paper/2007/file/4b04a686b0ad13dce35fa99fa4161c65-Paper.pdf.

[32] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.*, 17:39:1–39:40, 2016. URL https://jmlr.org/papers/v17/15-522.html.

[33] O. Levy, A. B. Cassel, A. Cohen, and Y. Mansour. Eluder-based regret for stochastic contextual mdps. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=47jMS97wJX.

[34] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. In Y. Bengio and Y. LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1509.02971.

[35] H. Liu, C. Wei, and J. Zimmert. Towards optimal regret in adversarial linear mdps with bandit feedback. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=6yv8UHVJn4.

[36] H. Luo, C. Wei, and C. Lee. Policy optimization in adversarial mdps: Improved exploration via dilated bonuses. In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 22931–22942, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/c1b8bf9e071c0dabb899e7a27f353762-Abstract.html.

[37] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. A. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nat.*, 518(7540):529–533, 2015. doi: 10.1038/NATURE14236. URL https://doi.org/10.1038/nature14236.

[38] G. Neu, A. György, C. Szepesvári, and A. Antos. Online markov decision processes under bandit feedback. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada*, pages 1804–1812. Curran Associates, Inc., 2010. URL https://proceedings.neurips.cc/paper/2010/hash/7bb060764a818184ebb1cc0d43d382aa-Abstract.html.

[39] F. Orabona. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2019.

[40] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. F. Christiano, J. Leike, and R. Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed,

A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html.

[41] J. Peters and S. Schaal. Policy gradient methods for robotics. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2006, October 9-15, 2006, Beijing, China*, pages 2219–2225. IEEE, 2006. doi: 10.1109/IROS.2006.282564. URL https://doi.org/10.1109/IROS.2006.282564.

[42] J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008. doi: 10.1016/J.NEUNET.2008.02.003. URL https://doi.org/10.1016/j.neunet.2008.02.003.

[43] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html.

[44] S. Ross, P. Mineiro, and J. Langford. Normalized online learning. In A. E. Nicholson and P. Smyth, editors, *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, UAI 2013, Bellevue, WA, USA, August 11-15, 2013*. AUAI Press, 2013. URL https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=2415&proceeding_id=29.

[45] J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. In Y. Bengio and Y. LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1506.02438.

[46] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

[47] S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, 2014. ISBN 978-1-10-705713-5. URL http://www.cambridge.org/de/academic/subjects/computer-science/pattern-recognition-and-machine-learning/understanding-machine-learning-theory-algorithms.

[48] L. Shani, Y. Efroni, A. Rosenberg, and S. Mannor. Optimistic policy optimization with bandit feedback. In *International Conference on Machine Learning*, pages 8604–8613. PMLR, 2020.

[49] U. Sherman, T. Koren, and Y. Mansour. Improved regret for efficient online reinforcement learning with linear function approximation. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 31117–31150. PMLR, 2023. URL https://proceedings.mlr.press/v202/sherman23a.html.

[50] U. Sherman, A. Cohen, T. Koren, and Y. Mansour. Rate-optimal policy optimization for linear markov decision processes. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=VJwsDwuiuH.

[51] D. Simchi-Levi and Y. Xu. Bypassing the monster: A faster and simpler optimal algorithm for contextual bandits under realizability. *Mathematics of Operations Research*, 47(3):1904–1931, 2022.

[52] N. Stiennon, L. Ouyang, J. Wu, D. M. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. F. Christiano. Learning to summarize from human feedback. *CoRR*, abs/2009.01325, 2020. URL `https://arxiv.org/abs/2009.01325`.

[53] Y. Xu and A. Zeevi. Upper counterfactual confidence bounds: a new optimism principle for contextual bandits. *arXiv preprint arXiv:2007.07876*, 2020.

[54] Y. Zhu, D. J. Foster, J. Langford, and P. Mineiro. Contextual bandits with large action spaces: Made practical. In *International Conference on Machine Learning*, pages 27428–27453. PMLR, 2022.

A Proofs: Regret Analysis

In this section, we provide the full analysis of the regret of Algorithm 1, proving Theorem 3.1. We follow the regret analysis of [48] but adapt it to function approximation for stochastic CMAB. We start our analysis by noting that with probability at least $1-\delta/2$,

$$\mathcal{R}_K \leq \mathbb{E}\mathcal{R}_K + 2\sqrt{2K\log(4/\delta)}.$$

As the contexts are iid, and the policies $\{\pi_k\}_{k=1}^K$ are determined by the history, the above is a direct implication of Azuma-Hoeffding's inequality. (Full proof is given in Corollary B.3).

Hence, we focus on bounding the expected regret, which will imply a high probability regret bound, by the above. To obtain the desired bound, we decompose the expected regret as follows.

$$\begin{aligned}
\mathbb{E}\mathcal{R}_K &= \sum_{k=1}^K \mathbb{E}_{c_k}[\langle \pi_k(c_k,\cdot) - \pi_\star(c_k,\cdot), \ell(c_k,\cdot)\rangle] \\
&= \underbrace{\sum_{k=1}^K \mathbb{E}_{c_k}\left[\langle \pi_k(c_k,\cdot), \ell(c_k,\cdot)\rangle - \left\langle \pi_k(c_k,\cdot), \hat{\ell}_k(c_k,\cdot)\right\rangle\right]}_{(2)} \\
&\quad + \underbrace{\sum_{k=1}^K \mathbb{E}_{c_k}\left[\left\langle \pi_k(c_k,\cdot), \hat{\ell}_k(c_k,\cdot)\right\rangle - \left\langle \pi_\star(c_k,\cdot), \hat{\ell}_k(c_k,\cdot)\right\rangle\right]}_{(3)} \\
&\quad + \underbrace{\sum_{k=1}^K \mathbb{E}_{c_k}\left[\left\langle \pi_\star(c_k,), \hat{\ell}_k(c_k,\cdot)\right\rangle - \langle \pi_\star(c_k,\cdot), \ell(c_k,\cdot)\rangle\right]}_{(4)}
\end{aligned}$$

We bound each term separately, but first, we upper-bound the sum of bonuses.

Lemma A.1 (Bonuses bound, restatement of Lemma 4.1)**.** *The following holds.*

$$\sum_{k=1}^K \mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}} \pi_k(c_k,a) b_k^\beta(c_k,a)\right] \leq \beta|\mathcal{A}|\log(K+1).$$

Proof. We first note that

$$\begin{aligned}
\sum_{k=1}^K \mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}} \pi_k(c_k,a) b_k^\beta(c_k,a)\right] &= \sum_{k=1}^K \mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}} \pi_k(c_k,a)\min\left\{1, \frac{\beta/2}{1+\sum_{i=1}^{k-1}\pi_i(c_k,a)}\right\}\right] \\
&\leq \frac{\beta}{2}\sum_{k=1}^K \mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}} \frac{\pi_k(c_k,a)}{1+\sum_{i=1}^{k-1}\pi_i(c_k,a)}\right] \\
&= \frac{\beta}{2}\mathbb{E}_c\left[\sum_{a\in\mathcal{A}}\sum_{k=1}^K \frac{\pi_k(c,a)}{1+\sum_{i=1}^{k-1}\pi_i(c,a)}\right].
\end{aligned}$$

Next, we fix a context $c\in\mathcal{C}$ and bound

$$\sum_{a\in\mathcal{A}}\sum_{k=1}^K \frac{\pi_k(c,a)}{1+\sum_{i=1}^{k-1}\pi_i(c,a)}$$

Using Lemma C.1 from Levy et al. [33], given in Lemma B.6, we obtain that for any fixed context it holds that

$$\sum_{a\in\mathcal{A}}\sum_{k=1}^K \frac{\pi_k(c,a)}{1+\sum_{i=1}^{k-1}\pi_i(c,a)} \leq 2|\mathcal{A}|\log(K+1).$$

By taking an expectation over the context on both sides, we obtain

$$\mathbb{E}_c\left[\sum_{a\in\mathcal{A}}\sum_{k=1}^{K}\frac{\pi_k(c,a)}{1+\sum_{i=1}^{k-1}\pi_i(c,a)}\right]\leq 2|\mathcal{A}|\log(K+1).$$

Lastly, we plug the latter into our first inequality and conclude the lemma. □

We then move to bound each one of the three terms.

Lemma A.2 (Term (2) bound, restatment of Lemma 4.2)**.** *For any $\delta\in(0,1)$, let $\beta=\sqrt{\frac{K34\log(4|\mathcal{F}|K^3/\delta)}{|\mathcal{A}|}}$. Then, with probability at least $1-\delta/4$ it holds that*

$$\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\langle\pi_k(c_k,\cdot),\ell(c_k,\cdot)\rangle-\left\langle\pi_k(c_k,\cdot),\hat{\ell}_k(c_k,\cdot)\right\rangle\right]\leq\widetilde{O}\Big(\sqrt{K|\mathcal{A}|\log(|\mathcal{F}|/\delta)}\Big).$$

Proof. The following holds with probability at least $1-\delta/4$.

$$\begin{aligned}
&\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\langle\pi_k(c_k,\cdot),\ell(c_k,\cdot)\rangle-\left\langle\pi_k(c_k,\cdot),\hat{\ell}_k(c_k,\cdot)\right\rangle\right]\\
&=\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)\Big(\ell(c_k,a)-\max\Big\{0,\hat{f}_k(c_k,a)-b_k^{\beta}(c_k,a)\Big\}\Big)\right]\\
&\leq\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)\Big(\ell(c_k,a)-\Big(\hat{f}_k(c_k,a)-b_k^{\beta}(c_k,a)\Big)\Big)\right]\\
&=\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)\Big(f_\star(c_k,a)-\hat{f}_k(c_k,a)\Big)\right]+\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)b_k^{\beta}(c_k,a)\right]\\
&\leq\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)|f_\star(c_k,a)-\hat{f}_k(c_k,a)|\right]+\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)b_k^{\beta}(c_k,a)\right]\\
&=\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)\min\Big\{1,|f_\star(c_k,a)-\hat{f}_k(c_k,a)|\Big\}\right]+\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)b_k^{\beta}(c_k,a)\right]\\
&\qquad\qquad\text{(Since both function are bounded in } [0,1]\text{, so is the absolute value)}\\
&\leq\sum_{k=2}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)\min\Big\{1,|f_\star(c_k,a)-\hat{f}_k(c_k,a)|\Big\}\right]+\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)b_k^{\beta}(c_k,a)\right]+1\\
&=\sum_{k=2}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)\min\left\{1,\sqrt{\frac{\beta}{\beta}\frac{1+\sum_{i=1}^{k-1}\pi_i(c_k,a)}{1+\sum_{i=1}^{k-1}\pi_i(c_k,a)}}|f_\star(c_k,a)-\hat{f}_k(c_k,a)|\right\}\right]\\
&+\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)b_k^{\beta}(c_k,a)\right]+1\\
&\leq\sum_{k=2}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)\min\left\{1,\ \frac{1}{2}\left(\frac{\beta}{1+\sum_{i=1}^{k-1}\pi_i(c_k,a)}+\frac{1}{\beta}\left(1+\sum_{i=1}^{k-1}\pi_i(c_k,a)\right)\Big(f_\star(c_k,a)-\hat{f}_k(c_k,a)\Big)^2\right)\right\}\right]\\
&\qquad\qquad\text{(By AM-GM inequality)}\\
&+\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)b_k^{\beta}(c_k,a)\right]+1\\
&\leq\sum_{k=2}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)\min\left\{1,\frac{\beta/2}{1+\sum_{i=1}^{k-1}\pi_i(c_k,a)}\right\}\right]+\frac{K}{2\beta}\\
&\qquad\text{(Since all terms in the min are positive for } \beta>0\text{, holds by } \min\{\cdot,\cdot\}\text{ properties)}
\end{aligned}$$

$$+\frac{1}{2\beta}\sum_{k=2}^{K}\mathbb{E}_{c_k}\left[\sum_{i=1}^{k-1}\mathbb{E}_{a\sim\pi_i(c_k,\cdot)}\left[\left(f_\star(c_k,a)-\hat{f}_k(c_k,a)\right)^2\right]\right]+\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)b_k^\beta(c_k,a)\right]+1$$

$$\leq 2\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)b_k^\beta(c_k,a)\right]+\frac{K}{2\beta}+\frac{1}{2\beta}\sum_{k=2}^{K}\sum_{i=1}^{k-1}\mathbb{E}_{c_i}\left[\mathbb{E}_{a\sim\pi_i(c_i,\cdot)}\left[\left(f_\star(c_i,a)-\hat{f}_k(c_i,a)\right)^2\right]\right]+1$$

(By the bonus definition in Algorithm 1, and linearity of expectation, as the contexts are iid.)

$$\leq 2\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\sum_{a\in\mathcal{A}}\pi_k(c_k,a)b_k^\beta(c_k,a)\right]+\frac{K}{2\beta}+\frac{68\log(4|\mathcal{F}|K^3/\delta)K}{2\beta}+1$$

(Holds with probability at least $1-\delta/4$, by Corollary 2.3)

$$\leq 2\beta|\mathcal{A}|\log(K+1)+\frac{K}{2\beta}+\frac{68\log(4|\mathcal{F}|K^3/\delta)K}{2\beta}+1. \qquad \text{(By Lemma A.1)}$$

Finally, by setting $\beta=\sqrt{\frac{K34\log(4|\mathcal{F}|K^3/\delta)}{|\mathcal{A}|}}$ we obtain that term (2) is bounded as

$$\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\langle\pi_k(c_k,\cdot),\ell(c_k,\cdot)\rangle-\left\langle\pi_k(c_k,\cdot),\hat{\ell}_k(c_k,\cdot)\right\rangle\right]\leq\widetilde{O}\Big(\sqrt{K|\mathcal{A}|\log(|\mathcal{F}|/\delta)}\Big).$$

□

We continue to bound term (3).

Lemma A.3 (Term (3) bound, restatment of Lemma 4.3)**.** *For the choice in $\eta=\sqrt{\log|\mathcal{A}|/K}$, the following holds true.*

$$\sum_{k=1}^{K}\mathbb{E}_{c_k}\left[\left\langle\pi_k(c_k,\cdot)-\pi_\star(c_k,\cdot),\hat{\ell}_k(c_k,\cdot)\right\rangle\right]\leq O(\sqrt{K\log|\mathcal{A}|}).$$

Proof. Observe that the policy we compute in Algorithm 1 for each observed context $c\in\mathcal{C}$ is the solution of the same optimization problem as in Online Mirror Descent (OMD) algorithm (see Appendix B.1 for more information), using the KL-divergence for the Bregman divergence term.

Formally, our policy satisfies the following for every context $c\in\mathcal{C}$ and round $k\in[K]$.

$$\pi_{k+1}(c,\cdot)\in\arg\min_{\pi\in\Delta(\mathcal{A})}\eta\left\langle\hat{\ell}_k(c,\cdot),\pi-\pi_k(c,\cdot)\right\rangle+d_{KL}(\pi||\pi_k(c,\cdot)). \tag{5}$$

Hence, term (3) is the linear approximation term of the policy computed by the OMD algorithm, in expectation over the contexts.

As our loss estimators are bounded in $[0,1]$ and $\pi_1(c,\cdot)$ is uniform over the actions for every context $c\in\mathcal{C}$, we can apply the fundamental inequality of online mirror descent for the KL divergence (Theorem 10.4 in [39], V1), to obtain the following for each fixed context $c\in\mathcal{C}$ separately.

$$\sum_{k=1}^{K}\left\langle\hat{\ell}_k(c,\cdot),\pi_k(c,\cdot)-\pi_\star(c,\cdot)\right\rangle\leq\frac{\log|\mathcal{A}|}{\eta}+\frac{\eta}{2}\sum_{k=1}^{K}\pi_k(c,a)\underbrace{\hat{\ell}_k(c,a)^2}_{\leq 1}\leq\frac{\log|\mathcal{A}|}{\eta}+\frac{\eta K}{2}.$$

For $\eta=\sqrt{2\log|\mathcal{A}|/K}$ we obtain for each context $c\in\mathcal{C}$ that

$$\sum_{k=1}^{K}\left\langle\hat{\ell}_k(c,\cdot),\pi_k(c,\cdot)-\pi_\star(c,\cdot)\right\rangle\leq O(\sqrt{K\log|\mathcal{A}|}).$$

Since inner product of real vectors is symmetric,

$$\sum_{k=1}^{K}\left\langle\pi_k(c,\cdot)-\pi_\star(c,\cdot),\hat{\ell}_k(c,\cdot)\right\rangle\leq O(\sqrt{K\log|\mathcal{A}|}).$$

By taking an expectation over the contexts on both sides of the inequality, using linearity of expectation we obtain

$$\sum_{k=1}^{K} \mathbb{E}_{c_k}\Big[\Big\langle \pi_k(c_k,\cdot) - \pi_\star(c_k,\cdot), \hat{\ell}_k(c_k,\cdot)\Big\rangle\Big] \leq O(\sqrt{K\log|\mathcal{A}|}),$$

as desired. □

Lastly, we bound term (4).

Lemma A.4 (Term (4) bound, restatment of Lemma 4.4)**.** *For any $\delta \in (0,1)$ let $\beta = \sqrt{\frac{K34\log(4|\mathcal{F}|K^3/\delta)}{|\mathcal{A}|}}$. Then, the following holds with probability at least $1-\delta/4$.*

$$\sum_{k=1}^{K} \mathbb{E}_{c_k}\Big[\Big\langle \pi_\star(c_k,\cdot), \hat{\ell}_k(c_k,\cdot) - \ell(c_k,\cdot)\Big\rangle\Big] \leq \widetilde{O}\Big(\sqrt{K|\mathcal{A}|\log(|\mathcal{F}|/\delta)}\Big).$$

Proof. The following holds with probability at least $1-\delta/4$.

$$\begin{aligned}
&\sum_{k=1}^{K} \mathbb{E}_{c_k}\Big[\Big\langle \pi_\star(c_k,\cdot), \hat{\ell}_k(c_k,\cdot) - \ell(c_k,\cdot)\Big\rangle\Big] \\
&\leq \sum_{k=2}^{K} \mathbb{E}_{c_k}\Big[\Big\langle \pi_\star(c_k,\cdot), \hat{\ell}_k(c_k,\cdot) - \ell(c_k,\cdot)\Big\rangle\Big] + 1 \\
&= \sum_{k=2}^{K} \mathbb{E}_{c_k}\left[\sum_{a\in A} \pi_\star(c_k,a)\Big(\max\{0, \hat{f}_k(c_k,a) - b_k^\beta(c_k,a)\} - \ell(c_k,a)\Big)\right] + 1 \\
&= \sum_{k=2}^{K} \mathbb{E}_{c_k}\left[\sum_{a\in A} \pi_\star(c_k,a)\max\{0-\ell(c_k,a), \hat{f}_k(c_k,a) - \ell(c_k,a) - b_k^\beta(c_k,a)\}\right] + 1 \\
&= \sum_{k=2}^{K} \mathbb{E}_{c_k}\left[\sum_{a\in A} \pi_\star(c_k,a)\max\left\{\underbrace{-f_\star(c_k,a)}_{=:\alpha_{k,a}}, \underbrace{\hat{f}_k(c_k,a) - f_\star(c_k,a) - b_k^\beta(c_k,a)}_{=:\rho_{k,a}}\right\}\right] + 1 = (\star).
\end{aligned}$$

We now note that for all $k \geq 2, a \in \mathcal{A}$ it holds that $\alpha_{k,a} \leq 0$, as $f_\star \in [0,1]$. We next upper bound $\rho_{k,a}$ and then use the following fact to obtain the final upper bound.

Fact A.5. For any $x, y, z \in \mathbb{R}$ such that $z \geq y$ it holds that $\max\{x,y\} \leq \max\{x,z\}$.

Hence, we continue by upper bounding $\rho_{k,a}$, for any fixed $k \geq 2$ and $a \in \mathcal{A}$.

$$\begin{aligned}
\rho_{k,a} &= \Big(\hat{f}_k(c_k,a) - f_\star(c_k,a)\Big) - b_k^\beta(c_k,a) \\
&\leq |\hat{f}_k(c_k,a) - f_\star(c_k,a)| - b_k^\beta(c_k,a) \\
&= \min\Big\{1, |\hat{f}_k(c_k,a) - f_\star(c_k,a)|\Big\} - b_k^\beta(c_k,a) \\
&\qquad\qquad \text{(Since the functions are bounded in } [0,1] \text{ so is the absolute value)} \\
&= \min\left\{1, \sqrt{\frac{\beta}{\beta}\frac{1+\sum_{i=1}^{k-1}\pi_i(c_k,a)}{1+\sum_{i=1}^{k-1}\pi_i(c_k,a)}}|\hat{f}_k(c_k,a) - f_\star(c_k,a)|\right\} - b_k^\beta(c_k,a) \\
&\leq \min\left\{1, \frac{\beta/2}{1+\sum_{i=1}^{k-1}\pi_i(c_k,a)} + \frac{1}{2\beta}\left(1+\sum_{i=1}^{k-1}\pi_i(c_k,a)\right)\Big(\hat{f}_k(c_k,a) - f_\star(c_k,a)\Big)^2\right\} - b_k^\beta(c_k,a) \\
&\qquad\qquad \text{(By AM-GM inequality)} \\
&\leq \underbrace{\min\left\{1, \frac{\beta/2}{1+\sum_{i=1}^{k-1}\pi_i(c_k,a)}\right\}}_{=b_k^\beta(c_k,a)} + \frac{1}{2\beta}\left(1+\sum_{i=1}^{k-1}\pi_i(c_k,a)\right)\Big(\hat{f}_k(c_k,a) - f_\star(c_k,a)\Big)^2 - b_k^\beta(c_k,a) \\
&\qquad \text{(Since all terms in the min are positive for } \beta > 0, \text{ holds by } \min\{\cdot,\cdot\} \text{ properties)}
\end{aligned}$$

$$= \frac{1}{2\beta}\left(1 + \sum_{i=1}^{k-1} \pi_i(c_k, a)\right)\left(\hat{f}_k(c_k, a) - f_\star(c_k, a)\right)^2.$$

Hence, we choose $\xi_{k,a} := \frac{1}{2\beta}\left(1 + \sum_{i=1}^{k-1} \pi_i(c_k, a)\right)\left(\hat{f}_k(c_k, a) - f_\star(c_k, a)\right)^2$, and note that for $\beta > 0$ we have $\xi_{k,a} \geq 0$. We then apply Fact A.5 to upper bound $(\star)$. We obtain

$$\begin{aligned}
(\star) &= \sum_{k=2}^{K} \mathbb{E}_{c_k}\left[\sum_{a\in A} \pi_\star(c_k, a) \max\left\{\underbrace{-f_\star(c_k, a)}_{\alpha_{k,a}}, \underbrace{\hat{f}_k(c_k, a) - f_\star(c_k, a) - b_k^\beta(c_k, a)}_{\rho_{k,a}}\right\}\right] + 1 \\
&\leq \sum_{k=2}^{K} \mathbb{E}_{c_k}\left[\sum_{a\in A} \pi_\star(c_k, a) \max\left\{\underbrace{-f_\star(c_k, a)}_{\leq 0}, \underbrace{\frac{1}{2\beta}\left(1 + \sum_{i=1}^{k-1} \pi_i(c_k, a)\right)\left(\hat{f}_k(c_k, a) - f_\star(c_k, a)\right)^2}_{\xi_{k,a}\geq 0}\right\}\right] + 1 \\
&\qquad\qquad \text{(Since } \xi_{k,a} \geq \rho_{k,a}\text{)} \\
&= \sum_{k=2}^{K} \mathbb{E}_{c_k}\left[\sum_{a\in A} \pi_\star(c_k, a) \frac{1}{2\beta}\left(1 + \sum_{i=1}^{k-1} \pi_i(c_k, a)\right)\left(\hat{f}_k(c_k, a) - f_\star(c_k, a)\right)^2\right] + 1 \\
&\leq \frac{K}{2\beta} + \frac{1}{2\beta}\sum_{k=2}^{K} \mathbb{E}_{c_k}\left[\sum_{a\in A} \underbrace{\pi_\star(c_k, a)}_{\leq 1} \sum_{i=1}^{k-1} \pi_i(c_k, a)\left(\hat{f}_k(c_k, a) - f_\star(c_k, a)\right)^2\right] + 1 \\
&\leq \frac{K}{2\beta} + \frac{1}{2\beta}\sum_{k=2}^{K} \mathbb{E}_{c_k}\left[\sum_{i=1}^{k-1}\sum_{a\in A} \pi_i(c_k, a)\left(\hat{f}_k(c_k, a) - f_\star(c_k, a)\right)^2\right] + 1 \\
&= \frac{K}{2\beta} + \frac{1}{2\beta}\sum_{k=2}^{K}\sum_{i=1}^{k-1} \mathbb{E}_{c_i}\left[\mathbb{E}_{a\sim\pi_i(c_i,\cdot)}\left(\hat{f}_k(c_i, a) - f_\star(c_i, a)\right)^2\right] + 1 \\
&\qquad\qquad \text{(By linearity of expectation, as the contexts are iid)} \\
&\leq \frac{K}{2\beta} + \frac{68\log(4|\mathcal{F}|K^3/\delta)K}{2\beta} + 1. \\
&\qquad\qquad \text{(Holds with probability at least } 1 - \delta/4 \text{ by Corollary 2.3)}
\end{aligned}$$

Finally, by setting $\beta = \sqrt{\frac{K34\log(4|\mathcal{F}|K^3/\delta)}{|\mathcal{A}|}}$ we obtain that term (4) is bounded as $\widetilde{O}\left(\sqrt{K|\mathcal{A}|\log(|\mathcal{F}|/\delta)}\right)$. □

B Auxiliary Lemmas

B.1 Online Mirror Descent

In the Online Mirror Descent (OMD) algorithm [39], in each round of the game, the agent solves the following optimization problem to compute the next policy.

$$x_{k+1} \in \arg\min_{x\in\Delta(d)} \eta\langle g_k, x - x_k\rangle + B_\psi(x, x_k), \tag{6}$$

where ψ is used for the Bregman's divergence term, which in our case, is the KL divergence. The following lemma states the fundamental inequality of OMD over the simplex with the KL divergence (or, the variants of it, e.g., Hedge and EXP3), which will be used for our analysis.

Theorem B.1 (Lemma 16 in [48], originally Theorem 10.4 in V1 of [39], Fundamental inequality of Online Mirror Descent)**.** *Assume for $g_{k,i} \geq 0$ for $k = 1, \ldots, K$ and $i = 1, \ldots, d$. Let $C = \Delta_d$ and $\eta > 0$. Using OMD with the KL-divergence, learning rate η, and with*

uniform initialization $x_1 = (1/d, \ldots, 1/d)$*, the following holds for any* $u \in \Delta_d$*,*

$$\sum_{k=1}^{K} \langle g_k, x_k - u \rangle \leq \frac{\log d}{\eta} + \frac{\eta}{2} \sum_{k=1}^{K} \sum_{i=1}^{d} x_{k,i} g_{k,i}^2.$$

B.2 Concentration Inequalities

Theorem B.2 (Azuma-Hoeffding's inequality)**.** *Let* $(X_i)_{i=1}^N$ *be a martingale difference sequence with respect to the filtration* $(\mathcal{F}_i)_{i=0}^N$ *such that* $|X_i| \leq B$ *almost surely for all* $i \in [N]$*. Then with probability at least* $1 - \delta$*,*

$$\left| \sum_{i=1}^{N} X_i \right| \leq B \sqrt{2N \log \frac{2}{\delta}}.$$

Corollary B.3 (High probability regret)**.** *With probability at least* $1 - \delta/2$*, it holds that*

$$\mathcal{R}_K \leq \mathbb{E}\mathcal{R}_K + 2\sqrt{2K \log(4/\delta)}.$$

Proof. The proof follows directly from Theorem B.2, as the contexts are stochastic and sampled iid in each round, and the policies $\{\pi_k\}_{k=2}^K$ are determined completely by the history, where π_1 is set to be the random policy. Thus, we have that.

$$X_k := \langle \pi_k(c_k, \cdot) - \pi_\star(c_k, \cdot), \ell(c_k, \cdot) \rangle - \mathbb{E}_{c_k}[\langle \pi_k(c_k, \cdot) - \pi_\star(c_k, \cdot), \ell(c_k, \cdot) \rangle]$$

defines a martingale difference sequence $(X_k)_{k=1}^K$ with respect to the filtration $H_k = \{(c_i, a_i, \ell_i)\}_{i=1}^k$ which is the history up to (including) time k, for all $k \in [1, K]$ and $H_0 = \emptyset$ is the empty history. This holds true since X_k is determined by H_k for all k, and

$$\begin{aligned}
\mathbb{E}[X_k | H_{k-1}] =& \mathbb{E}[\langle \pi_k(c_k, \cdot) - \pi_\star(c_k, \cdot), \ell(c_k, \cdot) \rangle - \mathbb{E}_{c_k}[\langle \pi_k(c_k, \cdot) - \pi_\star(c_k, \cdot), \ell(c_k, \cdot) \rangle] | H_{k-1}] \\
=& \mathbb{E}_{c_k}[\langle \pi_k(c_k, \cdot) - \pi_\star(c_k, \cdot), \ell(c_k, \cdot) \rangle | H_{k-1}] - \mathbb{E}_{c_k}[\langle \pi_k(c_k, \cdot) - \pi_\star(c_k, \cdot), \ell(c_k, \cdot) \rangle | H_{k-1}] \\
=& 0.
\end{aligned}$$

In addition, $|X_k| \leq 2$ for all k. Hence, we can apply Theorem B.2 and obtain for any fixed $K \in \mathbb{N}$, when summing over $k = 1, 2, \ldots, K$, that the following holds with probability at least $1 - \delta/2$,

$$\left| \sum_{k=1}^{K} X_k \right| \leq 2\sqrt{2K \log(4/\delta)}.$$

In addition,

$$\begin{aligned}
\left| \sum_{k=1}^{K} X_k \right| =& \left| \sum_{k=1}^{K} (\langle \pi_k(c_k, \cdot) - \pi_\star(c_k, \cdot), \ell(c_k, \cdot) \rangle - \mathbb{E}_{c_k}[\langle \pi_k(c_k, \cdot) - \pi_\star(c_k, \cdot), \ell(c_k, \cdot) \rangle]) \right| \\
=& \left| \sum_{k=1}^{K} \langle \pi_k(c_k, \cdot) - \pi_\star(c_k, \cdot), \ell(c_k, \cdot) \rangle - \sum_{k=1}^{K} \mathbb{E}_{c_k}[\langle \pi_k(c_k, \cdot) - \pi_\star(c_k, \cdot), \ell(c_k, \cdot) \rangle] \right| \\
=& |\mathcal{R}_K - \mathbb{E}\mathcal{R}_K|.
\end{aligned}$$

Hence, with probability at least $1 - \delta/2$,

$$|\mathcal{R}_K - \mathbb{E}\mathcal{R}_K| \leq 2\sqrt{2K \log(4/\delta)},$$

which implies the corollary. □

B.3 Oracle Convergence

Lemma 5 in [53] presents a uniform convergence guarantee for offline lease-square regression.

Lemma B.4 (uniform convergence over all sequences of estimators, Lemma 5 in [53]). *For an arbitrary contextual bandit algorithm, for all $\delta \in (0,1)$, with probability at least $1-\delta/2$,*

$$\sum_{i=1}^{t-1} \mathbb{E}_{c_i,a_i}\Big[\big(f_t(c_i,a_i) - f_\star(c_i,a_i)\big)^2 \mid H_{i-1}\Big] \\ \leq 68\log(2|\mathcal{F}|t^3/\delta) + 2\sum_{i=1}^{t-1}\big(f_t(c_i,a_i)-\ell_i\big)^2 - \big(f_\star(c_i,a_i)-\ell_i\big)^2,$$

uniformly over all $t \geq 2$ and all fixed sequence $f_2, f_3, \ldots \in \mathcal{F}$.

In this lemma, the filtration defined by $H_i := \{(c_k, a_k, \ell_k)\}_{k=1}^i$, which is the history of observations up to time i.

The following is a direct corollary of Lemma B.4, which applies to the sequence of least-squares minimizers.

Corollary B.5 (uniform convergence of offline least-squares regression, restatement of Corollary 2.3). *Let $f_2, f_3, \ldots \in \mathcal{F}$ denote the sequence of least squares minimizers and let $\pi_1, \pi_2, \ldots$ denote the sequence of played contextual policies. The following holds for any $\delta \in (0,1)$ and $t \geq 2$ with probability at least $1-\delta/4$.*

$$\sum_{i=1}^{t-1} \mathbb{E}_{c_i}\Big[\mathbb{E}_{a_i\sim\pi_i(c_i,\cdot)}\Big[\big(f_t(c_i,a_i) - f_\star(c_i,a_i)\big)^2\Big]\Big] \leq 68\log(4|\mathcal{F}|t^3/\delta).$$

Proof. For the sequence of least squares minimizers, for all $t \geq 2$, it holds that

$$\sum_{i=1}^{t-1}\big(f_t(c_i,a_i)-\ell_i\big)^2 - \big(f_\star(c_i,a_i)-\ell_i\big)^2 \leq 0.$$

In addition, in each round k, given the history $H_{k-1} = \{(c_i, a_i, \ell_i)\}_{i=1}^{k-1}$, we have that π_k is determined completely. Hence, the corollary follows from Lemma B.4 for the choice in $\delta' = \delta/2$. □

B.4 Additional Algebraic Lemmas

Lemma B.6 (Lemma C.1 in [33]). *Let $S_t = \lambda + \sum_{k=1}^{t-1} x_k$ and $x_t \in [0,\lambda]$ for all t. Then*

$$\sum_{t=1}^{T} \frac{x_t}{S_t} \leq 2\log(T+1).$$

C Experiments

As is standard in the CMAB literature, we evaluate the performance of our algorithm using the Vowpal Wabbit[5] benchmark suite [8], which implements all known feasibly-implementable CMAB algorithms or their practical variants.

Due to the diversity of algorithm implementations in the library, some methods require a cost-sensitive classification oracle (e.g., `OnlineCover`-the heuristics of `ILTCB` [5]), while others rely on an online regression oracle (`SquareCB` [18], `AdaCB` [23], `RegCB` [20] and `FastCB` [19]). We emphasize that even algorithms originally designed for offline regression oracles can be effectively executed using an online regression oracle, which is a stronger oracle capable of handling adversarial examples [12, 18]. In the Vowpal Wabbit implementation, all algorithms that are using a regression oracle are implemented using its online variant, which is the base learner of this library. Following this convention, we implement `OPO-CMAB` using the base online regression oracle provided by the library.

The state-of-the-art empirical evaluation of CMAB algorithms on the Vowpal Wabbit benchmark is presented in Foster and Krishnamurthy [19]. They conduct extensive experiments comparing all known efficient CMAB algorithms based on regression oracles (e.g., `SquareCB, AdaCB, RegCB`) as well as a supervised learning algorithm as a mild baseline, against their proposed algorithm, `FastCB`. The evaluation spans a wide range of hyperparameter settings and considers both square loss and logistic loss for the regression oracle. Their findings indicate that `FastCB` with logistic loss achieves the best performance, followed by `SquareCB` with logistic and squared loss in second and third place, respectively. `AdaCB` and `RegCB` perform significantly worse. Based on these results, we adopt the similar experimental setup and focus our comparison on the regression-bases candidates: `FastCB`, `SquareCB`, `RegCB` and `AdaCB` as well as `Supervised` which is the supervised learning benchmark. More details about the used implementations and hyperparameter are given in the sequel.

C.1 Function classes and Oracles

For the function class and oracle, we follow the same setup as in [19] that is also described in detail in [8]. We give the details below for completeness.

Oracle. All evaluated algorithms, including our `OPO-CMAB`, are implemented in Vowpal Wabbit (VW), using its base online learning procedure, regardless of whether they are designed for online or offline regression oracles. This procedure implements Importance Weighted Regression (IWR) for both square loss and logistic loss, as described in [8]. Specifically, the oracle performs Online Gradient Descent (OGD) [7] updates with an adaptive step size (controlled via the learning-rate hyperparameter) following [15], normalization as in [44], and importance weighting as in [27]. In addition, VW applies a Cost-Sensitive One-Against-All (CSOAA) reduction for multiclass classification, which ultimately yields label predictions for each action. All tested algorithms are constrained to use this IWR-based oracle exclusively, which in the code reflected by the 'mtr' cb-type parameter.

Losses and Function Class. Regarding the losses and function class used, we again adopt the same setup tested by [19, 8]. Hence, we test all the algorithms when applying the online logistic regression oracle using the log loss. We also test all of the algorithms, excluding `FastCB`, using the square loss. The two mathematically defined bellow, for a true label y and predicted label $\hat{y}$.

$$\ell_{\text{sq}}(\hat{y}, y) = (\hat{y} - y)^2.$$
$$\ell_{\log}(\hat{y}, y) = y \log(1/\hat{y}) + (1 - y) \log(1/1 - \hat{y}).$$

We note that the log-loss (or cross-entropy) in this notation is defined for binary classification, i.e., $y, \hat{y} \in \{0, 1\}$, and we next extend it to multiclass classification using the sigmoid link-function, which formally defined as

$$\sigma(x) = \frac{1}{1 + \exp(-x)}.$$

[5] `https://vowpalwabbit.org/`

Then,

$$\ell_{\text{logistic}}(\hat{y}, y) = \ell_{\log}(\sigma(\hat{y}), y).$$

See [21] for more information regarding this equivalence. When applying logistic regression, we define the function class $\mathcal{F}$ as the class of generalized linear models, formally given by

$$\mathcal{F} = \left\{(c, a) \to \sigma(\langle w, \phi(c, a)\rangle) | w \in \mathbb{R}^d\right\},$$

where $\sigma(\cdot)$ is the logistic link function defined above, and $\phi : \mathcal{C} \times \mathcal{A} \to \mathbb{R}^d$ is a dataset-related feature map.

When using the squared loss, we instead choose $\mathcal{F}$ to be the class of linear functions

$$\mathcal{F} = \left\{(c, a) \to \langle w, \phi(c, a)\rangle | w \in \mathbb{R}^d\right\}.$$

All of these implementation choices are already supported by the Vowpal Wabbit library, and also performed by [19][6].

C.2 Implementation Details

Bellow we elaborate on the implementation and hyper-parameters tested for each algorithm.

Implementation of known algorithms. For the implementations and experimental setup of `SquareCB`, `FastCB`, `Supervised` we use the source code provided by [19][7]. We use the hyperparameter values tested in this work, among them $\rho \in \{0.25, 0.5\}, \gamma_0 \in \{1000, 700, 400, 100, 50, 10\}$. Due to limited computational resources, we use as learning rates a subset of the learning rates tested by Foster and Krishnamurthy [19] and mentioned in Table 1.
For `RegCB` we use the standard implementation of it in Vowpal Wabbit, with the hyper-parameters choice of $c_0 \in 10^{\{-1,-2,-3\}}$ suggested by Bietti et al. [8].
For `AdaCB`, we use the `SquareCB` implementation of Vowpal Wabbit, which includes elimination option that uses confidence bounds computed as for `RegCB` to eliminate sub-optimal actions, and then apply Inverse Gap Weightening (IGW)-based policy for the action selection itself. We test the algorithm on the set of hyperparameter suggested by Foster et al. [23]: $c_0 \in 10^{\{-1,-2,-3\}}, \rho \in \{0.25, 0.5\}, \gamma_0 \in \{1000, 700, 400, 100, 50, 10\}$. For a summary of all hyper-parameter values used for tuning, see Table 1.

As for the implementation of `OPO-CMAB`, we integrate it into the same source code ourself, and the implementation details are given next.

OPO-CMAB implementation. We integrate `OPO-CMAB` into the Vowpal Wabbit library, implementing the exact pseudo-code provided in Algorithm 1, with the following practical modifications:
We define the bonus factor at each round k as $\beta_k = \gamma\sqrt{\frac{k}{|\mathcal{A}|}}$, where γ is a tuned hyperparameter. This differs from the constant $\beta = \sqrt{\frac{34K\log(4|\mathcal{F}|K^3/\delta)}{|\mathcal{A}|}}$ used in our theoretical analysis. This change is motivated by the following:

1. *Adaptivity to unknown horizon.* The new form of β_k allows the algorithm to operate without prior knowledge of the total number of rounds K.
2. *Practical tuning and computational efficiency.* The constant term in the original β expression, $\sqrt{34\log(4|\mathcal{F}|K^3/\delta)}$, may be difficult to compute when $\mathcal{F}$ is infinite or extremely large. We replace it with a tunable parameter γ to allow for more practical and flexible implementation of the confidence bounds width factor.

Similarly, we treat the parameter η as a tuned hyperparameter rather than constant that depends on $|\mathcal{A}|, |\mathcal{F}|$, for the same reasons as above. We note that our algorithm, similarly to

[6]For additional details, see Appendix D.2 in [19].
[7]The code is publicly available at `https://openreview.net/forum?id=3qYgdGj9Svt`

the others, have an additional learning rate parameter, that uses the base-learner oracle as an initial step size. We, similarly to the other algorithm, refer the oracle's learning rate as a tuned hyper-parameter.

An important difference between our algorithm and the others is that `OPO-CMAB` requires evaluating each new context using all past predictors, in order to compute exploration bonuses based on past counterfactual policies. To enable this functionality without interfering with the internal implementation of the base learner, we cache the learned weights at each round and use them to generate past predictions for the current context. This memory and run time-consuming implementation is due to the constraints implied by the oracle implementation of Vowpal Wabbit library.

Lastly, we remark that although `OPO-CMAB` was originally designed for the squared loss, the dataset used in this experimental setup involves multi-class and multi-label classification. In such settings, the logistic loss is often more appropriate, as it better reflects the probabilistic nature of the labels and can yield more accurate loss estimates in practice compared to the squared loss. Hence, we apply our algorithm using the logistic-loss as well, to provide it with more accurate predictors.

Apart from these changes, the rest of the implementation adheres closely to the description in Algorithm 1. The hyper-parameter values used for tuning `OPO-CMAB` also appear in Table 1.

Remark C.1. We note that the adaptive $\beta_k = O\left(\gamma\sqrt{\frac{k}{|\mathcal{A}|}}\right)$ choice can be proven to yield optimal regret bound for stochastic CMAB (see [53] for more information), however, for clean representation of the algorithm and main theoretical result, we chose in a static β parameter.

C.3 Experiments Description and Evaluation

In the following, we describe our experimental setup. Due to computational limitations, we restrict our evaluation to the following experimental setup.

Tested Datasets. We evaluated all the considered algorithms on 18 relatively-small size and arbitrarily selected out of the large set of 515 multiclass classification datasets defined in the Vowpal Wabbit suit.[8]. The tested datasets are specified in Table 2. In each dataset, every context is associated with a true label. Following prior works, we simulate bandit feedback by defining the agent's loss to be 0 if it predicts the correct label, and 1 otherwise.

Performance Evaluation. We again follow [19, 8] and evaluate the performance of each algorithm on a given dataset using the final *Progressive Validation (PV)* loss [9], which, for an algorithm `A` and a given dataset containing K examples, formally defined as

$$L_{PV}(\mathtt{A}, K) = \frac{1}{K}\sum_{k=1}^{K}\ell_k(a_k).$$

In our experiments, we measure the decrease in the PV-loss as a function of the number of examples.

Remark C.2. For any two algorithms `A`, `B`, let $\mathcal{R}_K^{\mathtt{A}}$ and $\mathcal{R}_K^{\mathtt{B}}$ denote their regret computed using the binary loss defined above on a given dataset. Then, it holds true that

$$\frac{1}{K}\left(\mathcal{R}_K^{\mathtt{A}} - \mathcal{R}_K^{\mathtt{B}}\right) = L_{PV}(\mathtt{A}, K) - L_{PV}(\mathtt{B}, K).$$

Hence, in this setting, the PV-loss is a representative quantity for the regret differences between any pair of algorithms.

In addition, we also consider the difference of the final PV-loss of each algorithm compare to that of the supervised learning mild baseline.

Lastly, similarly to prior works, we also perform *approximate Z-test* to examine the statistical significance of the results, which is formally defined as follows.

[8]All datasets are publicly available to download from OpenML collection `https://www.openml.org`

Definition C.3 (p-Statistically significant win)**.** For two algorithms a, b let p_a, p_b denote their PV-losses. We say that a p-significantly wins b if $1 - \Phi\left(\frac{p_a - p_b}{\sqrt{\frac{p_a(1-p_a)}{n} + \frac{p_b(1-p_b)}{n}}}\right) < p$, where n is the examples number and Φ is Gauss error function.

For each dataset and pair of algorithms, we compute the significance of the win. As common practice, results that are p-statistically significant for $p \leq 0.05$ considered as meaningful.

Hyper-parameters Tuning. We tuned all algorithms by running all the combinations of the the parameters specified in Table 1. For each algorithm, we run each combination once on each dataset, and choose the combination that provides the lowest final PV-loss. Then, we use only this best combination to run our experiments. The chosen parameter configurations are specified in Table 1.

Algorithm	γ/γ_0	ρ	c_0	η	**Loss**	**LRs**
Supervised	–	–	–	–	ℓ_{logistic}	$10^{\{1,0,-1,-2,-3\}}$
SquareCB	1000, 700, 400, 100, 50, 10	0.5, 0.25	–	–	$\ell_{\text{sq}}, \ell_{\text{logistic}}$	$10^{\{1,0,-1,-2,-3\}}$
FastCB	1000, 700, 400, 100, 50, 10	0.5, 0.25	–	–	ℓ_{logistic}	$10^{\{1,0,-1,-2,-3\}}$
AdaCB	1000, 700, 400, 100, 50, 10	0.5, 0.25	$10^{\{-1,-2,-3\}}$	–	$\ell_{\text{sq}}, \ell_{\text{logistic}}$	$10^{\{1,0,-1,-2,-3\}}$
RegCB	–	–	$10^{\{-1,-2,-3\}}$	–	$\ell_{\text{sq}}, \ell_{\text{logistic}}$	$10^{\{1,0,-1,-2,-3\}}$
OPO-CMAB	$10^{\{0,-1,-2\}}$	–	–	100, 10, 1, 0.2, 0.1, 0.01	$\ell_{\text{sq}}, \ell_{\text{logistic}}$	$10^{\{1,0,-1,-2,-3\}}$

Table 1: All tested hyperparameter values per algorithm.

Dataset	**Supervised**	**SquareCB**	**FastCB**	**AdaCB**	**RegCB**	**OPOCMAB**
1006	ℓ_{logistic}, lr: 10	ρ: 0.5, γ_0: 10, ℓ_{logistic}, lr: 10	ρ: 0.25, γ_0: 50, ℓ_{logistic}, lr: 10	ρ: 0.5, γ_0: 100, c_0: 0.001, ℓ_{logistic}, lr: 0.01	c_0: 0.001, ℓ_{sq}, lr: 1	η: 100, γ: 1, ℓ_{logistic}, lr: 0.1
1012	ℓ_{logistic}, lr: 1	ρ: 0.5, γ_0: 50, ℓ_{logistic}, lr: 0.1	ρ: 0.5, γ_0: 1000, ℓ_{logistic}, lr: 0.001	ρ: 0.5, γ_0: 50, c_0: 0.01, ℓ_{logistic}, lr: 0.1	c_0: 0.1, ℓ_{logistic}, lr: 0.1	η: 1, γ: 0.01, ℓ_{sq}, lr: 0.1
1015	ℓ_{logistic}, lr: 0.1	ρ: 0.25, γ_0: 1000, ℓ_{sq}, lr: 10	ρ: 0.25, γ_0: 100, ℓ_{logistic}, lr: 0.1	ρ: 0.5, γ_0: 100, c_0: 0.1, ℓ_{logistic}, lr: 0.1	c_0: 0.001, ℓ_{sq}, lr: 1	η: 100, γ: 0.1, ℓ_{sq}, lr: 0.1
1062	ℓ_{logistic}, lr: 1	ρ: 0.25, γ_0: 10, ℓ_{logistic}, lr: 1	ρ: 0.25, γ_0: 400, ℓ_{logistic}, lr: 0.1	ρ: 0.5, γ_0: 100, c_0: 0.01, ℓ_{logistic}, lr: 0.1	c_0: 0.001, ℓ_{logistic}, lr: 0.01	η: 100, γ: 0.01, ℓ_{sq}, lr: 1
1073	ℓ_{logistic}, lr: 0.01	ρ: 0.25, γ_0: 10, ℓ_{logistic}, lr: 1	ρ: 0.5, γ_0: 10, ℓ_{logistic}, lr: 10	ρ: 0.25, γ_0: 1000, c_0: 0.1, ℓ_{logistic}, lr: 0.001	c_0: 0.001, ℓ_{sq}, lr: 10	η: 1, γ: 0.01, ℓ_{logistic}, lr: 1
1084	ℓ_{logistic}, lr: 0.001	ρ: 0.5, γ_0: 100, ℓ_{sq}, lr: 0.1	ρ: 0.5, γ_0: 10, ℓ_{logistic}, lr: 0.001	ρ: 0.5, γ_0: 50, c_0: 0.1, ℓ_{logistic}, lr: 0.1	c_0: 0.1, ℓ_{logistic}, lr: 0.001	η: 10, γ: 0.01, ℓ_{logistic}, lr: 0.001
339	ℓ_{logistic}, lr: 1	ρ: 0.5, γ_0: 700, ℓ_{sq}, lr: 0.001	ρ: 0.5, γ_0: 700, ℓ_{logistic}, lr: 10	ρ: 0.5, γ_0: 100, c_0: 0.1, ℓ_{logistic}, lr: 0.01	c_0: 0.1, ℓ_{logistic}, lr: 10	η: 100, γ: 0.01, ℓ_{sq}, lr: 10
346	ℓ_{logistic}, lr: 1	ρ: 0.5, γ_0: 1000, ℓ_{logistic}, lr: 0.001	ρ: 0.5, γ_0: 1000, ℓ_{logistic}, lr: 0.001	ρ: 0.5, γ_0: 50, c_0: 0.1, ℓ_{logistic}, lr: 0.001	c_0: 0.01, ℓ_{logistic}, lr: 0.001	η: 10, γ: 1, ℓ_{logistic}, lr: 10
457	ℓ_{logistic}, lr: 1	ρ: 0.5, γ_0: 100, ℓ_{logistic}, lr: 0.01	ρ: 0.25, γ_0: 700, ℓ_{logistic}, lr: 0.001	ρ: 0.5, γ_0: 1000, c_0: 0.01, ℓ_{logistic}, lr: 0.001	c_0: 0.1, ℓ_{sq}, lr: 10	η: 100, γ: 0.1, ℓ_{logistic}, lr: 10
476	ℓ_{logistic}, lr: 0.1	ρ: 0.5, γ_0: 700, ℓ_{sq}, lr: 0.1	ρ: 0.25, γ_0: 100, ℓ_{logistic}, lr: 0.1	ρ: 0.25, γ_0: 10, c_0: 0.001, ℓ_{logistic}, lr: 1	c_0: 0.001, ℓ_{sq}, lr: 0.1	η: 100, γ: 0.01, ℓ_{sq}, lr: 10
729	ℓ_{logistic}, lr: 0.001	ρ: 0.5, γ_0: 1000, ℓ_{logistic}, lr: 0.1	ρ: 0.5, γ_0: 1000, ℓ_{logistic}, lr: 0.01	ρ: 0.25, γ_0: 10, c_0: 0.01, ℓ_{logistic}, lr: 0.001	c_0: 0.1, ℓ_{sq}, lr: 0.001	η: 0.1, γ: 0.1, ℓ_{sq}, lr: 1
785	ℓ_{logistic}, lr: 0.1	ρ: 0.5, γ_0: 10, ℓ_{sq}, lr: 0.1	ρ: 0.5, γ_0: 10, ℓ_{logistic}, lr: 0.1	ρ: 0.5, γ_0: 700, c_0: 0.001, ℓ_{logistic}, lr: 0.001	c_0: 0.01, ℓ_{sq}, lr: 0.1	η: 1, γ: 1, ℓ_{logistic}, lr: 10
835	ℓ_{logistic}, lr: 1	ρ: 0.5, γ_0: 400, ℓ_{sq}, lr: 0.001	ρ: 0.5, γ_0: 1000, ℓ_{logistic}, lr: 0.01	ρ: 0.5, γ_0: 1000, c_0: 0.001, ℓ_{logistic}, lr: 1	c_0: 0.01, ℓ_{logistic}, lr: 1	η: 100, γ: 1, ℓ_{sq}, lr: 1
848	ℓ_{logistic}, lr: 0.001	ρ: 0.5, γ_0: 700, ℓ_{logistic}, lr: 0.1	ρ: 0.5, γ_0: 700, ℓ_{logistic}, lr: 0.1	ρ: 0.5, γ_0: 700, c_0: 0.1, ℓ_{logistic}, lr: 0.01	c_0: 0.1, ℓ_{logistic}, lr: 1	η: 100, γ: 0.01, ℓ_{sq}, lr: 0.01
874	ℓ_{logistic}, lr: 1	ρ: 0.5, γ_0: 400, ℓ_{logistic}, lr: 0.01	ρ: 0.5, γ_0: 1000, ℓ_{logistic}, lr: 0.001	ρ: 0.5, γ_0: 10, c_0: 0.01, ℓ_{logistic}, lr: 10	c_0: 0.1, ℓ_{sq}, lr: 1	η: 1, γ: 0.1, ℓ_{sq}, lr: 1
905	ℓ_{logistic}, lr: 10	ρ: 0.5, γ_0: 700, ℓ_{logistic}, lr: 0.1	ρ: 0.5, γ_0: 700, ℓ_{logistic}, lr: 0.1	ρ: 0.25, γ_0: 700, c_0: 0.001, ℓ_{logistic}, lr: 0.001	c_0: 0.01, ℓ_{sq}, lr: 10	η: 100, γ: 1, ℓ_{sq}, lr: 1
928	ℓ_{logistic}, lr: 1	ρ: 0.25, γ_0: 50, ℓ_{logistic}, lr: 0.01	ρ: 0.25, γ_0: 700, ℓ_{logistic}, lr: 0.001	ρ: 0.25, γ_0: 10, c_0: 0.01, ℓ_{logistic}, lr: 0.001	c_0: 0.1, ℓ_{logistic}, lr: 0.001	η: 0.1, γ: 0.1, ℓ_{logistic}, lr: 1
964	ℓ_{logistic}, lr: 0.001	ρ: 0.25, γ_0: 100, ℓ_{sq}, lr: 0.01	ρ: 0.5, γ_0: 700, ℓ_{logistic}, lr: 0.01	ρ: 0.5, γ_0: 50, c_0: 0.01, ℓ_{logistic}, lr: 0.001	c_0: 0.001, ℓ_{logistic}, lr: 0.01	η: 0.1, γ: 0.1, ℓ_{sq}, lr: 10

Table 2: Best hyperparameter configurations per algorithm and dataset.

C.4 Experiments and Results

Experimental setup and results. For each algorithm and dataset, we run the best hyperparameter configuration (as found in our tuning) on 10 random permutations of the dataset. The results are summarized in the following tables and plots.

First, we average the PV-loss at each time step across permutations to measure the average performance of each algorithm as a function of the number of observed examples. In Figures 1a to 1c, we show the averaged PV-loss curves as well as the Std on three representative datasets, chosen because in them the supervised baseline converges to a significantly better-than-random-policy final PV-loss.

Figure 2 presents a table summarizing, for each dataset, the difference in final average across permutations PV-loss between each algorithm and the supervised baseline. The algorithm with the best mean difference relative to the supervised baseline is highlighted. Negative numbers indicate the algorithm outperforms supervised; positive numbers show how close it comes to the supervised baseline.

Figure 3 reports the p-values for each dataset and pair of algorithms, computed using the mean final PV-loss over the 10 permutations of each dataset. Following [19], we use the symmetric error function for the p-values calculation, so the result for algorithms (a, b) is the same as for (b, a). Statistical significance is indicated for $p < 0.05$.

Discussion. Figure 2 shows that across all datasets, `FastCB` is closest to (and sometimes outperforms) the supervised baseline in 6 out of 18 datasets, followed by `AdaCB` (5/18), and our `OPO-CMAB` (4/18). `RegCB` and `SquareCB` lag behind, with 2 and 1 datasets out of the 18, respectively. As in most datasets the difference between the different algorithms are small, there is no clear evidence that one algorithm significantly outperform others.

Figure 3 shows a similar trend; except for dataset 1084, statistically significant wins are rare or nonexistent.

Overall, Figures 2 and 3 support the hypothesis that all algorithms perform comparably considering those datasets, with no statistically significant evidence that any one algorithm consistently outperforms the others.

Mean Differences from Supervised

Dataset	AdaCB	FastCB	OPOCMAB	RegCB	SquareCB
1006_2	0.179730	0.107432	0.143243	**0.106081**	0.114865
1012_2	**0.020103**	0.026804	0.037114	0.039175	0.029381
1015_2	**-0.011111**	-0.001389	0.033333	0.002778	0.016667
1062_2.0	**0.002778**	0.022222	0.011111	0.047222	0.061111
1073_2.0	0.002555	**-0.086496**	-0.048540	-0.081752	-0.063504
1084_3	0.248637	0.035909	**0.010000**	0.011364	0.137728
339_3	0.158333	**0.091666**	0.125000	0.094444	0.127778
346_2	**0.020000**	0.066000	0.044000	0.028000	0.054000
457_4	0.162963	0.085185	**0.051852**	0.125926	0.100000
476_2	**0.050000**	0.062000	0.082000	0.084000	0.064000
729_2	0.138636	**0.004546**	0.022727	0.086364	0.011364
785_2	0.008889	0.026667	**0.006667**	0.033333	0.031111
835_2	0.272917	0.260417	0.237500	0.233333	**0.214583**
848_2	0.084210	0.081579	0.152632	**0.028947**	0.081579
874_2	0.188000	**0.106000**	0.150000	0.130000	0.110000
905_2	0.100000	**0.038462**	0.076923	0.071795	0.038462
928_2	0.152174	0.171739	**0.136956**	0.169565	0.182609
964_2	0.094444	**0.027778**	0.100000	0.066667	0.127778

Winner (Best Performance)

Figure 2: Mean Difference from Supervised baseline.

C.5 Resources and Computation

Assets. We used the following assets to conduct our experiments.
The code for the Vowpal Wabbit library, which is publicly available at `https://vowpalwabbit.org/`, includes implementations of all tested CMAB algorithms, with the exception of `FastCB`.

The implementation of `FastCB`, as well as the full experimental setup and evaluation code, were obtained from the source code accompanying [19], which is publically available at `https://openreview.net/forum?id=3qYgdGj9Svt`.
Building on this foundation, we extended the codebase by adding our own implementations

Statistical Significance: Comprehensive P-Values Matrix

Dataset	Ada,Fast	Ada,Opo	Ada,Reg	Ada,Square	Ada,Sup	Fast,Opo	Fast,Reg	Fast,Square	Fast,Sup	Opo,Reg	Opo,Square	Opo,Sup	Reg,Square	Reg,Sup	Square,Sup
1006_2	0.0762	0.3733	0.0708	0.1121	0.0000	0.3786	0.9733	0.8544	0.0061	0.3607	0.4859	0.0003	0.8283	0.0068	0.0034
1012_2	0.8459	0.6227	0.5814	0.7880	0.5563	0.7660	0.7212	0.9406	0.4338	0.9527	0.8235	0.2799	0.7777	0.2543	0.3912
1015_2	0.8278	0.3372	0.7571	0.5422	0.8040	0.4580	0.9269	0.6950	0.9755	0.5155	0.7263	0.4768	0.7640	0.9513	0.7178
1062_2.0	0.7894	0.9082	0.5473	0.4329	0.9691	0.8794	0.7380	0.6053	0.7598	0.6267	0.5036	0.8776	0.8555	0.5218	0.4105
1073_2.0	0.0027	0.0877	0.0045	0.0268	0.9324	0.1976	0.8711	0.4337	0.0035	0.2602	0.6135	0.1046	0.5349	0.0059	0.0332
1084_3	0.0000	0.0000	0.0000	0.0006	0.0000	0.4398	0.4643	0.0024	0.2837	0.9675	0.0001	0.7647	0.0002	0.7338	0.0000
339_3	0.3827	0.6579	0.4024	0.6844	0.0432	0.6678	0.9718	0.6415	0.2555	0.6937	0.9710	0.1162	0.6670	0.2409	0.1079
346_2	0.5146	0.7342	0.9099	0.6303	0.7772	0.7552	0.5903	0.8649	0.3495	0.8209	0.8874	0.5334	0.7129	0.6919	0.4444
457_4	0.3600	0.1976	0.6556	0.4552	0.0632	0.7112	0.6395	0.8666	0.3515	0.4011	0.5902	0.5755	0.7640	0.1601	0.2709
476_2	0.8519	0.6219	0.6006	0.8278	0.4201	0.7593	0.7363	0.9754	0.3205	0.9758	0.7829	0.1932	0.7597	0.1830	0.3056
729_2	0.0726	0.1216	0.4865	0.0887	0.0632	0.8077	0.2750	0.9272	0.9513	0.3970	0.8792	0.7608	0.3175	0.2489	0.8788
785_2	0.8112	0.9762	0.7424	0.7652	0.9050	0.7881	0.9285	0.9523	0.7202	0.7200	0.7425	0.9287	0.9761	0.6541	0.6759
835_2	0.8624	0.6234	0.5831	0.4181	0.0001	0.7508	0.7073	0.5248	0.0001	0.9539	0.7504	0.0004	0.7947	0.0005	0.0014
848_2	0.9731	0.3876	0.4699	0.9731	0.2642	0.3692	0.4909	1.0000	0.2791	0.1113	0.3692	0.0464	0.4909	0.6950	0.2791
874_2	0.2446	0.5902	0.4110	0.2685	0.0065	0.5332	0.7339	0.9548	0.1254	0.7773	0.5712	0.0303	0.7770	0.0605	0.1119
905_2	0.4157	0.7636	0.7127	0.4157	0.1779	0.6083	0.6564	1.0000	0.5957	0.9462	0.6083	0.2963	0.6564	0.3287	0.5957
928_2	0.7906	0.8361	0.8134	0.6795	0.0330	0.6365	0.9765	0.8828	0.0162	0.6578	0.5351	0.0547	0.8596	0.0175	0.0105
964_2	0.4178	0.9467	0.7374	0.6887	0.2477	0.3802	0.6354	0.2249	0.7313	0.6875	0.7386	0.2212	0.4613	0.4131	0.1184

$p < 0.001$ $p < 0.01$ $p < 0.05$ $p < 0.1$ $p \geq 0.1$

Figure 3: p-values for each dataset and algorithms pair.
The p-values were computed using the mean PV-loss over the 10 permutations tested for each dataset.

and modifications. As noted by the authors of [19], their code builds upon the CMAB evaluation framework developed in [8], which is also publicly available at `https://github.com/vowpalwabbit/vowpal_wabbit/`.

Following previous works, we use in our experiments 18 datasets from the 515 multiclass and multi-label classification datasets used for CMAB evaluation from the OpenML collection. The datasets are publicly available for download at `https://www.openml.org`.

We also referring to each specific dataset's page on OpenML for additional information, including copyright. See dataset 1015: `https://www.openml.org/search?type=data&sort=runs&id=1015&status=active`, dataset 1062: `https://www.openml.org/search?type=data&sort=runs&id=1062&status=active`, and dataset 1084: `https://www.openml.org/search?type=data&sort=runs&id=1084&status=active`.

Computation resources. Experiments were conducted on a Linux CPU server with approximately 250 cores, from which we used 40-100 cores only. The total compute time required to run all the presented experiments was approximately 48 hours. The memory required to run the tests was about 1.5 TB.