Tighter Bounds for Multi-Armed Bandits with Expert Advice

H. Brendan McMahan and **Matthew Streeter**
Google, Inc.
Pittsburgh, PA 15213, USA

Abstract

Bandit problems are a classic way of formulating exploration versus exploitation tradeoffs. Auer *et al.* [ACBFS02] introduced the EXP4 algorithm, which explicitly decouples the set of A actions which can be taken in the world from the set of M experts (general strategies for selecting actions) with which we wish to be competitive. Auer *et al.* show that EXP4 has expected cumulative regret bounded by $\mathcal{O}(\sqrt{TA\log M})$, where T is the total number of rounds. This bound is attractive when the number of actions is small compared to the number of experts, but poor when the situation is reversed. In this paper we introduce a new algorithm, similar in spirit to EXP4, which has a bound of $\mathcal{O}(\sqrt{TS\log M})$. The S parameter measures the extent to which expert recommendations agree; we always have $S \leq \min\{A, M\}$. We discuss practical applications that arise in the contextual bandits setting, including sponsored search keyword advertising. In these problems, common context means many actions are irrelevant on any given round, and so $S << \min\{A, M\}$, implying our bounds offer a significant improvement. The key to our new algorithm is a linear-programing-based exploration strategy that is optimal in a certain sense. In addition to proving tighter bounds, we run experiments on real-world data from an online advertising problem, and demonstrate that our refined exploration strategy leads to significant improvements over known approaches.

1 Introduction

The various formulations of the k-armed bandit problem provide clean frameworks for analyzing tradeoffs between exploration and exploitation, and hence have seen extensive attention from researchers in a variety of fields.

A bandit problem takes place over a series of rounds. On each round t, the algorithm selects some action $\acute{a}_t \in \mathcal{A}$ to be executed in the world. After $\acute{a}_t$ is chosen, a reward $r_t(\acute{a}_t)$ is obtained and observed by the algorithm. The goal is to maximize the sum of rewards $\sum_t r_t(\acute{a}_t)$. In this paper we adopt the nonstochastic viewpoint: we make no assumptions about the source of rewards, and so seek bounds that hold for arbitrary sequences of reward vectors (we assume each reward is in $[0, 1]$).[1] It is not possible to make any guarantees about cumulative reward (for example, we might face a sequence of vectors where every action on every round gets reward 0). Instead, algorithms for this problem bound performance in terms of regret, the difference between the algorithm's cumulative reward and the reward achieved by the best fixed action. Such nonstochastic assumptions are justified in changing worlds, where the past performance of actions may not be indicative of their future rewards.

In many real-world problems, however, it is not appropriate to compare ourselves to the performance of the best fixed action: for example, suppose the actions are advertisements that could be shown in response to queries on a search engine. Any single ad will have terrible performance if shown for all queries, and so treating this as a single multi-armed bandit problem would provide extremely weak guarantees. Instead, our approach decouples the actions $\mathcal{A}$ that can be taken in the world from the set of strategies (experts) $\mathcal{M}$ with which we wish to be competitive. This approach is not new to this paper: the EXP4 algorithm of [ACBFS02] addresses this problem. However, the bounds for that algorithm are only useful when the set of strategies $\mathcal{M}$ is larger than $\mathcal{A}$. We propose and analyze a new algorithm for this problem which addresses this issue.

For an algorithm to perform as well as the best expert from $\mathcal{M}$, it must implicitly estimate the cumulative reward obtained by each expert. If experts often agree on the actions they recommend, intuitively this estimation problem should become easier; however, current bounds for the problem do not reflect this. We propose a new algorithm, NEXP (the N is for Nonuniform exploration), which solves a linear program to select a distribution on actions that offers a locally optimal (with respect to our analysis) balance of exploration and exploitation. This algorithm has a bound of $\mathcal{O}(\sqrt{GS\log M})$ on total regret, where $M = |\mathcal{M}|$, $A = |\mathcal{A}|$, G is a bound on the best expert's cumulative reward (for example, $G = T$), and S is a parameter that measures the

[1] The stochastic version, where the reward of each action is drawn i.i.d. from some distribution (unknown to the algorithm) on each round, has also been extensively studied. Lai and Robbins [LR85] is the foundational paper.

Alg	Bound	Example	Reference
EXP3	$2.63\sqrt{GM \log M}/T$	0.689	[ACBFS02]
EXP4	$2.63\sqrt{GA \log M}/T$	2.178	[ACBFS02]
NEXP	$2.63\sqrt{GS \log M}/T$	0.097	this paper

Table 1: Bounds for the bandit problem with expert advice with A actions and M experts. G is a bound on the reward of the best expert. The parameter S, introduced in this paper, satisfies $S \leq \min\{A, M\}$, and is often much less. To make these bounds concrete, the "Example" column shows the bound on expected regret per round for $A = 10000$, $M = 1000$, $S = 20$, and $T = G = 100,000$. Note that the bound for EXP4 is vacuous. EXP3 directly selects experts, without using the structure induced by the actions.

extent to which the expert's recommendations agree. Importantly, $S \leq \min\{A, M\}$, and for some problems (such as the sponsored search advertising problem mentioned above) it can be many orders of magnitude smaller.

The paper is organized as follows: Section 2 completes the formal statement of the problem, defines notation, and compares the bounds for our algorithm to previously published results. Section 3 introduces several real-world instances of this setting where the tighter bounds proved can have significant practical impact. Section 4 summarizes related work. In Section 5 we introduce our algorithm and present and prove bounds. Section 6 presents experiments.

2 Preliminaries

An instance $\mathcal{I} = (\mathcal{A}, r, e)$ of the bandit problem with expert advice is defined by a sequence of reward vectors that is fixed in advance, with rewards bounded in $[0, 1]$, that is $r_t : \mathcal{A} \to [0, 1]$. The bandit algorithm has access to the recommendations of M experts from a set $\mathcal{M}$. Each expert i suggests a probability distribution $e_{i,t}$ over actions on each round t. These recommendations must be fixed (though not necessarily known to the algorithm) in advance, to the extent that they do not depend on the actions selected on earlier rounds. We discuss the ramifications of this assumption in the next section.

Our goal is to construct a randomized algorithm that on each round proposes a distribution p over the actions in such a way that our cumulative regret is small. In order to formalize the notion of regret, let G_i be a random variable (with respect to the distributions $e_{i,t}$) giving the performance of the i-th expert on a fixed problem instance $\mathcal{I}$. Then, taking expectation with respect to the draws from $e_{i,t}$,

$$E[G_i] = \sum_{t=1}^{T} \sum_{a \in \mathcal{A}} e_{i,t}(a) r_t(a) = \sum_{t=1}^{T} e_{i,t} \cdot r_t$$

is the expected performance of the i-th expert on $\mathcal{I}$. We then define $G_{\text{OPT}} = \max_i \{E[G_i]\}$ and $G_{\text{ALG}} = \sum_t r_t(\acute{a}_t)$, where $\acute{a}_t$ is the action chosen by the algorithm on round t. Then,

$$\text{Regret} = G_{\text{OPT}} - G_{\text{ALG}}.$$

The cumulative reward G_{ALG} of the algorithm is a random variable (with a distribution dependent on the randomization used by the algorithm in choosing the $\acute{a}_t$), and so Regret is also a random variable.

Since only $r_t(\acute{a}_t)$ is observed, even post-hoc we will not be able to exactly calculate our regret as in the experts setting where r_t is fully observable;[2] instead we can bound $E[\text{Regret}]$, the algorithm's expected regret. Unless otherwise stated, expectations are with respect to any internal randomness of the bandit algorithm. In the case where experts make probabilistic recommendations e, this expectation may implicitly include draws from these distributions, depending on whether the given algorithm internally samples from these distributions. The notation used in this paper is summarized in Table 2. We use subscripts for time, but sometimes omit them when referring to time t, so w_i is $w_{i,t}$ implicitly.

Our Main Result We can now state the main theoretical result of this paper. Define

$$s_t = \sum_a \max_i \{e_{i,t}(a)\} \ .$$

Observe that $s_t \leq A$, and also $s_t \leq \sum_a \sum_i e_i(a) = M$. If all the experts recommend the same distribution, then $s_t = 1$. If all experts are deterministic, then s_t is the number of distinct actions recommended by the experts.

Our main theorem (stated fully as Theorem 3) shows that the NEXP algorithm we introduce, when run with appropriate parameters, satisfies

$$E[\text{Regret}] \leq 2.63\sqrt{SG \ln M} \tag{1}$$

where $S = \max_t \{s_t\} \leq \min\{A, M\}$.

The dependence on the maximum of s_t over all rounds is perhaps unsatisfying: suppose on a single round all the expert are "confused" and each puts probability 1.0 on a separate action: suddenly $S = M$, and our bound is no better than EXP3, even if on every other round the experts agree entirely—clearly a tighter bound should be possible in this case. Under mild assumptions on the rewards of the problem, we strengthen our main theorem to handle this situation, replacing S in Equation (1) with $\tilde{S}$, the harmonic mean of the n_s largest s_t; the precise statement is given as Theorem 6.

Comparison to Previous Bounds The first algorithm proposed for the bandit problem with expert advice was EXP4 [ACBFS02], which bounds the expected regret by $\mathcal{O}(\sqrt{GA \log M})$, where G is an upper bound on G_{OPT}. Since rewards are in $[0, 1]$, one can always use $G = T$. Dividing the bound by T shows that the per-round regret goes to zero as $T \to \infty$, and so this is a no-regret algorithm. This bound is good when the number of actions is small and the number of experts is large.

What if the number of actions is much larger than the number of experts? If we have deterministic experts that recommend a single action a_i (e.g., $e_i(a_i) = 1$), then we can construct a reward vector r' directly on experts, where $r'(i) = e_{i,t} \cdot r_t = r_t(a_{i,t})$. Now we apply a standard nonstochastic multi-armed bandit algorithm (say, EXP3, also from [ACBFS02]) where the arms of the bandit problem are

[2] [FS95] describes the fully-observable experts setting in detail; for a comparison of the fully and partially observable cases, see [DHK07].

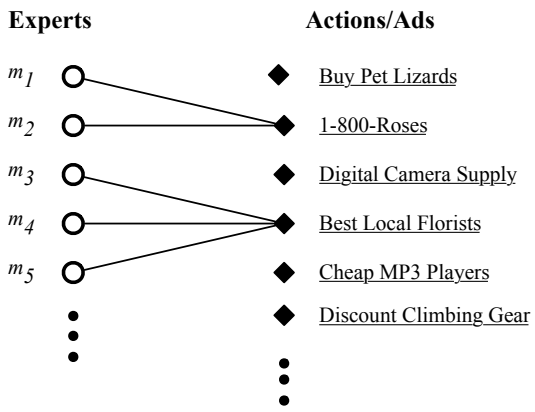

Figure 1: The ad selection bandit problem with expert advice. Each expert i corresponds to a deterministic function m_i mapping queries to a recommend ad to show for the query. While the set $\mathcal{A}$ of all ads can be very large, given knowledge of the query many of the schemes m_i are likely to suggest the same action/ad. Our algorithm leverages this fact to achieve sharper regret bounds. In this example round (considering only the experts and actions shown), we have $M = 5$, $A = 6$, and $s_t = 2$.

the deterministic experts from the original bandit problem with expert advice. This gives a bound of $\mathcal{O}(\sqrt{GM \log M})$ on expected regret, which is better than the bound of EXP4 when $A > M$. If the recommendations e_i are stochastic, $e_{i,t} \cdot r_t$ is not fully observable, but a method based on sampling deterministic experts from the stochastic ones can be applied to obtain bounds on expected regret.

Intuitively, the extra information provided by observing the recommendations of each expert should only make the problem easier, but in the case of large numbers of actions EXP4 actually has worse bounds than EXP3 (and significantly worse performance in our experiments as well). Our new algorithm resolves this deficiency. In fact, our bounds remain unchanged even if an entirely different set of actions are recommended on each round or if actions are arbitrarily re-indexed on each round. This makes it clear that the *only* value in knowing the experts' recommendations (rather than, say, just being able to blindly follow the experts' advice) comes from their use in correlating the performance of the experts, making estimating each expert's performance easier. In the full-information case (i.e., the full reward vector r_t is observed on each round), the actions can be effectively ignored, as the exact expected performance of each expert can be computed directly, and so a standard full-information algorithm like Hedge can be applied directly to the experts and performs essentially optimally.

These bounds are summarized in Table 1, along with some concrete average per-round regret numbers; the parameters for the example were chosen to be reasonable both in terms of computation and data, but show that both EXP3 and EXP4 might perform poorly. As the gap between S and $\min\{A, M\}$ grows large, there are problems where both EXP3 and EXP4 will provide vacuous guarantees, but NEXP's bounds will be quite tight.

	Problem Statement
$\mathcal{A}$	set of actions, $\lvert\mathcal{A}\rvert = A$
$\mathcal{M}$	set of experts, $\lvert\mathcal{M}\rvert = M$
a	generic action
$\acute{a}_t$	action played by the algorithm on round t
T	total number of rounds
t	index of a round
r	reward vector on actions
$e_{i,t}(a)$	expert i's recommended probability on a on round t
s_t	$\sum_a \max_i \{e_{i,t}(a)\}$
	Algorithm
p	distribution on actions executed in world
$\tilde{p}$	"ideal" (non-exploration) distribution on actions
q	distribution on experts
w	weights on experts
W	sum of weights
	Bounds and Analysis Variables
G_{OPT}	cumulative reward of best expert in hindsight
G_i	cumulative reward of the i-th expert
G_{ALG}	cumulative reward of a bandit algorithm
S	bound on s_t, $S \geq \max_t \{s_t\}$
z_t	bound on $\tilde{p}_t(a)/p_t(a)$ for all a
Z	bound on z_t, $Z \geq \max_t \{z_t\}$
G	bound on G_{OPT}

Table 2: Summary of notation.

3 Applications

While the improvement in bounds for bandit problems with expert advice is interesting from a purely theoretical point of view, we also believe that many (perhaps even most) real-world bandit problems are better framed as bandit problems with expert advice. To support this claim, we consider several motivating problem domains where expert advice is particularly useful and the new algorithms introduced in this paper are particularly advantageous.

Search Engine Keyword Advertising The problem of selecting and pricing ads to be shown alongside Internet search engine queries has received a great deal of attention lately, for example [RCKU08, Var07, GP07, WVLL07, PO07].

We can effectively apply our algorithm to this problem as follows. Consider a set of M different schemes for determining which ads to show on a particular query. Let $\mathcal{A}$ be the total set of advertisements available to be shown across all possible queries, and let X be the set of possible queries. Each (deterministic) expert i is associated with a function $m_i : X \to \mathcal{A}$ (for simplicity, we assume we show one ad per query). Clearly, the set $\mathcal{A}$ may be extremely large, and the EXP4 regret bound will likely be vacuous. Further, the set $\mathcal{M}$ may also be very large—suppose, for example, we have a family of schemes for showing ads that are parameterized by some vector $\theta \in \Theta$; we might construct the set $\mathcal{M}$ by discretizing Θ. If Θ has even moderately high dimension, the square-root dependence on M from EXP3 will be prohibitive. However, here we see that the structure induced by the context can be used to our advantage: for many queries, only a small number of ads will be relevant (see Figure 1).

If all of the schemes m_i are basically in agreement for most queries, then our job of selecting the best one should become much easier. This is exactly the intuition that our algorithm captures. We apply our algorithm to a problem from this setting in Section 6, and demonstrate substantial empirical improvements.

Online Choice of Active Learning Algorithms Baram *et al.* proposed using EXP4 to dynamically choose among several active learning algorithms in the pool-based active learning setting [BEYL04]. They empirically evaluate their approach using EXP4 with $M = 3$ active learning algorithms; on each round each algorithm suggests an example from a pool $\mathcal{A}$ of unlabeled examples which it would like to have labeled (the size of this pool ranged from 215 to 8300 in their experiments). Ideally, the reward associated with labelling a particular example should be the differential improvement in generalization error gained by having access to the label. This is not generally available, so [BEYL04] introduced the Classification Entropy Maximization (CEM) heuristic, and used it to assign a reward for the example labelled. They show empirically that this is quite effective, and further that their approach does a remarkably good job of tracking the best expert (individual active learning algorithm) with almost no regret. In fact, on some problems their combined approach outperforms any of the individual experts (that is, achieves negative regret).[3]

The Contextual Bandits Problem The above applications can be viewed as examples of contextual multi-armed bandit problems [LZ07] (also known as bandits with side information). In this setting, on each round the algorithm has access to a vector $x_t \in X$ that provides context (side information) for the current round, which may be used in determining which action to take. Formally, an instance of the non-stochastic contextual bandits problem $\mathcal{I} = (\mathcal{A}, r, x)$ is given by a sequence (x_t, r_t), where the side information $x_t \in X$ is observed by the algorithm before a_t is chosen and reward $r_t(a_t)$ is obtained and observed.

Rather than try to leverage the context information x in a general-purpose way that is applicable to arbitrary instances $\mathcal{I} = (\mathcal{A}, r, x)$, we consider introducing domain-specific experts that know how to make use of the side information, but then ignore the side information in the master algorithm: this transforms the contextual bandit problem to a bandit problem with expert advice. A domain-specific scheme for incorporating the context can be viewed as an expert template $e'_i : X \to \Delta(\mathcal{A})$, where $\Delta(\mathcal{A})$ is the set of probability distributions over actions. Using these templates, given an instance $\mathcal{I} = (\mathcal{A}, r, x)$ of the contextual bandits problem we construct an instance $\mathcal{I} = (\mathcal{A}, r, e)$ of the bandit problem with expert advice by setting $e_{i,t} = e'_i(x_t)$ for all i and t. This is the approach implicitly taken in the applications just discussed.

When this transformation is applied and the context is actually highly relevant (as is the query in determining which ads to show), it is likely that the experts access to the common x_t will cause S to be much smaller than A, and so the approach taken in this paper will be particularly beneficial.

History-Dependent Experts There is a subtle issue with the application of EXP4 (or our algorithms) to many practical problems, including the online choice of active learning algorithms. In this domain, the recommendations e_i naturally depend on the observations and context from the previous rounds. In particular, the recommendations of the active learning algorithms depend on which examples have previously been labelled. In a contextual bandits problem, some experts might be online learning algorithms that continue to train on the newly labelled examples $(x_t, r_t(a_t))$ (where, $r_t(a_t)$ can be associated with a label).

In a pure contextual bandits problem, as just defined, the x_t and $e_{i,t}$ are jointly fixed in advance, and so dependence of e_t on $x_1, \ldots, x_t$ (or even r_t) is implicitly allowed by the bounds. However, if $e_{i,t}$ is actually a function of $a_1, \ldots, a_{t-1}$, we cannot bound regret with respect to the expected gain of following expert i on every time step. Hence, even though [BEYL04] show that their combined approach does as well or better than each individual learning algorithm, the bound from EXP4 provides no such guarantee. Regret bounds still hold, but they bound regret with respect to the post-hoc sequence of recommendations that each expert i actually made. It is straightforward to construct pathological examples where this quantity may be very different than the expected performance of expert i had it been played on every round (for example, consider an expert that makes perfect recommendations, but only if its advice is followed exactly on the first k rounds).

If the experts are history-dependent, our algorithm is really trying to solve a reinforcement learning problem where the state is the history of past actions. If it is reasonable to make assumptions about the transition probabilities and rewards, or if we are allowed to reset to previously visited states, then standard reinforcement learning techniques can be applied, for example [KMN00]. If one believes that only a limited window of history matters to the performance of experts, approaches like those of [dFM06] can be adapted. In this work we are unwilling to make such strong assumptions, but it means our bounds can only hold with respect to the post-hoc recommendations of the experts. However, for many practical applications (including the online choice of active learning algorithms setting) it is reasonable to believe that pathological cases like the above will not occur, or even that the experts will do better based on the obtained shared history than if they had been run independently. In these cases, algorithms for the bandit problem with expert advice are an appropriate (and easy to implement) option.

[3]Interestingly, the small M and large A imply that EXP3 gives better bounds for this problem than EXP4. However, because the individual algorithms use all of the labels observed so far, the uniform-random exploration done by EXP4 may in fact be a benefit: it provides labelled training data to each active learning algorithm that might not have been available to any of the active learning algorithms if they had been run individually. This may explain both why EXP4 is so effective despite its poor bounds, and also why the authors observe that their combined approach can actually outperform the individual algorithms. (We will have more to say on the exact nature of these bounds in a few paragraphs).

4 Related Work

We have already mentioned several important pieces of related work. For an excellent summary of bounds for standard bandit problems, comparisons to the full information setting, and generalizations, see [DHK07].

Langford and Zhang [LZ07] formalize a general contextual multi-armed bandit problem under stochastic assumptions. In particular, they assume that each (x_t, r_t) is drawn i.i.d. from a fixed (i.e., independent of t) distribution. Their focus is on the case where $\mathcal{M}$ is an infinite hypothesis space (but with finite VC dimension), and so their work can be seen as extending supervised learning techniques to the contextual bandits problem. When their algorithm is applied to a finite hypothesis space $\mathcal{M}$ of size M, they get bounds of the form

$$\mathcal{O}(G^{2/3}A^{1/3}(\ln M)^{1/3}).$$

Thus, compared to EXP4, they get a better dependence on M and A, but worse dependence on G. However, this is really an apples-to-oranges comparison, as their work makes a strong probabilistic assumption on (x_t, r_t) in order to be able to handle infinite hypothesis spaces. In contrast, we make no distributional assumptions and so get bounds that hold for arbitrary sequences (x_t, r_t). We can also obtain much tighter bounds in terms of the number of actions A (in some cases removing the dependence entirely), and we can combine entirely arbitrary and unrelated ways of incorporating the side information.

Several authors have considered applying bandit-style algorithms to sponsored search auctions. Explore-exploit tradeoffs may arise at two different levels in this domain. Most prior work (including [PO07] and [WVLL07]) has addressed the tradeoff between showing ads that are known to have a good click-through-rate (CTR) versus the need to show ads with unknown CTRs in order to estimate their relevance. These algorithms directly propose a set of ads to show on each query. In particular, Pandey and Olston [PO07] consider a bandit-based algorithm that directly tries to learn click-through rates as well as correctly allocate ads to queries in the budget-limited case. Gonen and Pavlov [GP07] study a similar problem, but also consider advertiser incentives. Our approach is orthogonal to this work, as we address the exploration/exploitation tradeoff at the meta-level: given a selection $m_1, \ldots, m_M$ of possible algorithms (possibly including those from the above references), how do we trade off evaluating these different algorithms versus using the algorithm currently estimated to be best?

5 Algorithms and Analysis

The algorithms we analyze have the general form given in Figure 2; the key distinction between the algorithms in this family is the choice of the exploration policy F_{mix}. Our recommended approach, LP-Mix, is given by the linear program in the figure. We refer to this algorithm as NEXP(LP-Mix) or just NEXP.

The distribution $\tilde{p}$ can be viewed as the ideal distribution to follow if all of our estimates were perfect; it corresponds to the exponential weighting scheme used by algorithms like Hedge [FS95]. The key algorithmic choice is how to modify $\tilde{p}$ to ensure sufficient exploration. It will become clear from

Algorithm NEXP

Choose parameter α and subroutine F_{mix}

Add the expert $e_0(a) = \frac{1}{s_t}\max_i\{e_{i,t}(a)\}$ to $\mathcal{M}$

$(\forall i \in \mathcal{M})\ w_{i,1} \leftarrow 1$

for $t = 1, 2, \ldots, T$ **do**

Observe expert distributions $e_1, \ldots, e_M$

$W_t \leftarrow \sum_{i=1}^M w_{i,t}$

$q_i \leftarrow w_{i,t}/W_t$

$\tilde{p}(a) \leftarrow \sum_{i=1}^M q_i e_i(a)$

$p \leftarrow F_{\text{mix}}(\tilde{p}, q, e)$ // For example, LP-Mix$_\alpha$

Draw $\acute{a}$ randomly according to p.

Take action $\acute{a}$, observe reward $r(\acute{a})$

$(\forall i)\ \hat{y}_i \leftarrow \frac{e_i(\acute{a})}{p(\acute{a})} r(\acute{a})$

$(\forall i)\ w_{i,(t+1)} \leftarrow w_{i,t}\exp(\alpha\hat{y}_i)$

end for

Subroutine LP-Mix$_\alpha$ $(\tilde{p}, q, e)$ solves for p // Use for F_{mix}

Solve the linear program below, and return p

$$\begin{aligned} \max_{p,c} \quad & c \\ \text{subject to} \quad & \forall a \quad p(a) \geq \alpha \max_i\{e_i(a)\} \\ & \forall a \quad p(a) \geq c\tilde{p}(a) \\ & \sum_a p(a) = 1. \end{aligned}$$

Figure 2: Algorithm NEXP. Variables used only in a single iteration of the **for** loop have subscripts t omitted. The function F_{mix} takes an ideal exploitation distribution $\tilde{p}$ and modifies it to ensure sufficient exploration. The solution p to LP-Mix is the recommended choice for F_{mix}; other choices are discussed in the text. Algorithm LP-Mix-Solve (Figure 3) can be used to solve LP-Mix efficiently.

Lemma 2 that we will want a p that satisfies the following properties for an appropriate choice of α and as small a z_t as possible:

$$(\boldsymbol{\alpha}):\ \forall t, i, a \quad \frac{e_{i,t}(a)}{p_t(a)} \leq \frac{1}{\alpha}, \qquad (\boldsymbol{Z}):\ \forall a, t \quad \frac{\tilde{p}_t(a)}{p_t(a)} \leq z_t. \tag{2}$$

The bound $(\boldsymbol{\alpha})$ ensures sufficient exploration, in particular that our importance-weighted estimates $\hat{y}_i$ of the true reward of each expert remain bounded; $(\boldsymbol{Z})$ bounds the componentwise ratio of the exploitation distribution $\tilde{p}$ we would like to play to the exploration-modified distribution p we actually play. The need for this componentwise-ratio definition of "distance" will become clear in the proof of Lemma 2.

For our analysis, we assume our set of experts contains an expert that recommends the distribution $e_{0,t}(a) = \frac{1}{s_t}\max_i\{e_{i,t}(a)\}$ on each round. If this is not the case, M becomes $M + 1$ in the bounds, and the $e_{0,t}$ expert can be

added in the algorithm implementation on a per-round basis.

Exploration Strategies We consider several exploration strategies (subroutines F_{mix}), all of which we will be able to analyze using Lemma 2.

UA-Mix$_\gamma$: uniform distribution on actions. For all $a \in \mathcal{A}$, use

$$p(a) = (1-\gamma)\tilde{p}(a) + \gamma\frac{1}{A}. \tag{3}$$

This produces an algorithm that is almost identical to the original EXP4, and for which we prove identical bounds. The only difference is that NEXP(UA-Mix) adds an additional expert e_0, while EXP4 adds an additional expert that plays the uniform distribution over all actions.

UE-Mix$_\gamma$: uniform distribution on experts. Let $p_u(a) = \frac{1}{M}\sum_i e_i(a)$, and for all $a \in \mathcal{A}$, use

$$p(a) = (1-\gamma)\tilde{p}(a) + \gamma p_u(a). \tag{4}$$

This produces an algorithm similar to running EXP3 on the experts, but it works immediately for experts that recommend general probability distributions, and it takes advantage of importance weighting to update the estimates for all experts that recommended the action $\acute{a}$ actually played.

LP-Mix$_\alpha$: "optimal" exploration. Given a constant α, use the p derived by solving the linear program given in Figure 2. Theorem 7 gives an efficient, easy-to-implement algorithm for solving this LP.

We now begin the analysis of these three algorithms in a unified framework. The next lemma shows that even though many per-round variables are not independently distributed due to dependence on the prior actions chosen, in an important case we can still treat their expectations independently:

Lemma 1 *Let X_t be a random variable associated with the t-th round of NEXP whose value depends on the outcome of previous randomness (i.e., the history $a_1, \ldots, a_{t-1}$). Then, if for all possible histories $a_1, \ldots, a_{t-1}$*

$$E[X_t \mid a_1, \ldots, a_{t-1}] = \bar{x}_t$$

for a fixed value $\bar{x}_t$ (independent of the previous a's, and hence independent of the distribution p), we have

$$E\left[\sum_{t=1}^{T} X_t\right] = \sum_{t=1}^{T} \bar{x}_t.$$

The proof follows from linearity of expectation.

The above lemma will typically be applied where the random variable X_t depends on the distribution p_t; observe that p_t is a fixed distribution given $a_1, \ldots, a_{t-1}$. The next lemma gives a general purpose bound in terms of the bounds α and z_t of Equation (2). The results for specific algorithms will follow by plugging in suitable constants based on the different exploration strategies.

Lemma 2 *If conditions $(\boldsymbol{\alpha})$ and $(\boldsymbol{Z})$ are satisfied by the p distributions selected by NEXP(F_{mix}), then*

$$E[G_{ALG}] \geq \left(\frac{1}{Z} - (e-2)\alpha S\right) G_{OPT} - \frac{1}{\alpha Z}\ln M \tag{5}$$

where $Z \geq \max_t \{z_t\}$.

Proof: Unless otherwise stated, variables are defined as in Table 2, though in some cases subscript t's have been added. All expectations are with respect to the draws $\acute{a}_t \sim p_t$. The basic proof technique follows the lines of those for EXP3 and EXP4 (see [ACBFS02] and [CBL06]). The key is relating W_t, the sum of our weights on the last round, to both our performance and the performance of the best expert. To do this, we will use the inequality $\exp(x) \leq 1 + x + (e-2)x^2$ for $x \in [0,1]$. For compactness, we write $\kappa = e - 2 \approx 0.72$.

$$\begin{aligned}\frac{W_{t+1}}{W_t} &= \sum_i \frac{w_{i,t}}{W_t}\exp(\alpha\hat{y}_{i,t}) \\ &\leq \sum_i q_{i,t}[1 + \alpha\hat{y}_{i,t} + \kappa(\alpha\hat{y}_{i,t})^2] \\ &= 1 + \alpha\sum_i q_{i,t}\hat{y}_{i,t} + \kappa\alpha^2\sum_i q_{i,t}\hat{y}_{i,t}^2,\end{aligned}$$

noting that because $\hat{y}_i \leq e_i(\acute{a})/p(\acute{a}) \leq \frac{1}{\alpha}$, we have $\alpha\hat{y}_i \in [0,1]$. Now, taking logs and summing t from 1 to T, we have for the left-hand side

$$\sum_{t=1}^{T} \ln\frac{W_{t+1}}{W_t} = \sum_{t=1}^{T}(\ln W_{t+1} - \ln W_t) = \ln W_{T+1} - \ln M,$$

and using $\ln(1+x) \leq x$ for the right-hand side, we have

$$\underbrace{\ln W_{T+1} - \ln M}_{\text{(I). } G_{\text{OPT}}} \leq \sum_{t=1}^{T}\Big[\underbrace{\alpha\sum_i q_{i,t}\hat{y}_{i,t}}_{\text{(II). } G_{\text{ALG}}} + \underbrace{\kappa\alpha^2\sum_i q_{i,t}\hat{y}_{i,t}^2}_{\text{(III). Regret}}\Big], \tag{6}$$

where the underbraces indicate the quantities to which we relate each term. First we relate term (I) to the gain of the best expert. Note that $\hat{y}_{i,t}$ is an unbiased estimate of the reward we would have received on the t-th round if we had chosen expert i, and

$$w_{i,T+1} = \prod_{t=1}^{T}\exp\left(\alpha\hat{y}_{i,t}\right) = \exp\left(\alpha\sum_{t=1}^{T}\hat{y}_{i,t}\right).$$

Thus, $w_{i,T+1}$ is the exponentiated scaled estimated total reward of expert i. Using the fact that $\ln\sum_a \exp(x_a)$ is a good approximation for $\max_a\{x_a\}$, we can show $\ln W_{T+1}$ must be close to the total reward of the best expert. In particular, for any expert k, we have

$$\ln W_{T+1} \geq \ln w_{k,T+1} = \alpha\sum_{t=1}^{T}\hat{y}_{k,t}. \tag{7}$$

We can relate term (II) in Equation (6) to our algorithm's actual gain on each round (dropping t subscripts):

$$\sum_i q_i\hat{y}_i = \sum_i \frac{q_i e_i(\acute{a})}{p(\acute{a})}r(\acute{a}) = \frac{\tilde{p}(\acute{a})}{p(\acute{a})}r(\acute{a}) \leq Zr(\acute{a}). \tag{8}$$

Combining the main inequality (6) with the bounds of (7) and (8), we have

$$\alpha\sum_{t=1}^{T}\hat{y}_{k,t} - \ln M \leq \alpha\sum_{t=1}^{T}Zr_t(\acute{a}_t) + \kappa\alpha^2\sum_{t=1}^{T}\sum_i q_{i,t}\hat{y}_{i,t}^2\,. \tag{9}$$

Then, dividing by αZ and rearranging, we have

$$\sum_{t=1}^{T} r_t(\acute{a}_t) \geq \frac{1}{Z}\sum_{t=1}^{T}\hat{y}_{k,t} - \frac{1}{\alpha Z}\ln M - \frac{\kappa\alpha}{Z}\sum_{t=1}^{T}\sum_i q_{i,t}\hat{y}_{i,t}^2. \tag{10}$$

We now bound term (III), which contributes to our regret. It is here that our analysis diverges from the analysis of EXP4. Consider some particular t and define $\bar{e}(a) = \max_i \{e_i(a)\}$. Again omitting t subscripts, we have

$$\begin{aligned}\sum_i q_i\hat{y}_i^2 &= \sum_i q_i \frac{e_i(\acute{a})^2 r(\acute{a})^2}{p(\acute{a})^2} \\ &\leq \sum_i q_i \frac{e_i(\acute{a})\bar{e}(\acute{a}) r(\acute{a})^2}{p(\acute{a})^2} \\ &= \frac{\tilde{p}(\acute{a})\bar{e}(\acute{a})}{p(\acute{a})^2} r(\acute{a})^2 \leq Z\bar{e}(\acute{a})\frac{r(\acute{a})}{p(\acute{a})}, \end{aligned} \tag{11}$$

recalling $r(a) \in [0,1]$ and so $r(a)^2 \leq r(a)$. Define

$$\hat{r}(a) = \begin{cases} r(\acute{a})/p(\acute{a}) & \text{if } a = \acute{a} \\ 0 & \text{otherwise.}\end{cases}$$

Summing the bound of Equation (11) over t, and using $S \geq s_t$, we have

$$\begin{aligned}\sum_t\sum_i q_{i,t}\hat{y}_{i,t}^2 &\leq Z\sum_{t=1}^{T}\bar{e}_t(\acute{a}_t)\frac{r_t(\acute{a}_t)}{p(\acute{a}_t)} \\ &\leq ZS\sum_{t=1}^{T}\sum_a \frac{\bar{e}_t(a)}{s_t}\hat{r}_t(a) \\ &= ZS\sum_{t=1}^{T}\sum_a e_{0,t}(a)\hat{r}_t(a)\,. \end{aligned}$$

Note that for any a where $p(a) > 0$, $E_{\acute{a}\sim p}[\hat{r}(a)] = r(a)$. We have $q_{i,t} > 0$ for all i, and so condition ($\boldsymbol{Z}$) implies $p(a) > 0$ whenever $e_{0,t}(a) > 0$. Thus, applying Lemma 1 to $\hat{r}(a)$,

$$E\left[\sum_{t=1}^{T}\sum_a e_{0,t}(a)\hat{r}_t(a)\right] = G_0 \leq G_{\text{OPT}}, \tag{12}$$

where G_0 is the reward for always following the advice of the expert e_0. The distribution p_t is fixed given $a_1, \ldots a_{t-1}$, so for any k,

$$E_{\acute{a}_t}[\hat{y}_{k,t} \mid a_1, \ldots a_{t-1}] = \sum_a p_t(a)\frac{e_{k,t}(a)}{p_t(a)} r_t(a) = e_{k,t}\cdot r_t,$$

and so again using Lemma 1,

$$E\left[\sum_{t=1}^{T}\hat{y}_{k,t}\right] = G_k.$$

By definition, $E[\sum_{t=1}^{T} r(\acute{a}_t)] = E[G_{\text{ALG}}]$, and so combining these expectations with Equation (10) and taking the max over k,

$$E[G_{\text{ALG}}] \geq \frac{1}{Z}G_{\text{OPT}} - \frac{1}{\alpha Z}\ln M - \kappa\alpha S G_{\text{OPT}}$$

which proves the theorem. ■

We now consider bounds for specific versions of the algorithm, parameterized by different choices of the F_{mix} function. We begin with our main theorem for NEXP(LP-Mix).

Theorem 3 *Algorithm NEXP (LP-Mix), run with parameter* $\alpha = \min\left\{\frac{1}{S}, \sqrt{\ln M}/\sqrt{(e-1)SG}\right\}$*, has expected regret bounded by*

$$E[\textit{Regret}] \leq 2\sqrt{(e-1)SG\ln M}.$$

Proof: For the case when $\alpha = 1/S$, solving

$$\frac{1}{S} \leq \sqrt{\ln M}/\sqrt{(e-1)SG}$$

for G shows that the gain of the best expert must be less than $\sqrt{SG\ln M}$ and so the result follows immediately. In the other case, we first show that, for this choice of α, the optimum z_t of the linear program is at most $\frac{1}{1-\gamma}$, where $\gamma = S\alpha$. To see this, let $p(a) = (1-\gamma)\tilde{p}(a) + \gamma e_{0,t}(a)$. Because $e_i(a) \leq \bar{e}(a)$ and $p(a) \geq \gamma e_{0,t}(a) \geq \alpha\bar{e}(a)$, we have

$$\frac{e_i(a)}{p(a)} \leq \frac{\bar{e}(a)}{\alpha\bar{e}(a)} = \frac{1}{\alpha}.$$

Thus, p is a feasible solution to the linear program. Furthermore,

$$\frac{\tilde{p}(a)}{p(a)} = \frac{p(a) - \gamma e_{0,t}(a)}{(1-\gamma)p(a)} \leq \frac{1}{(1-\gamma)}$$

which implies $z_t \leq \frac{1}{1-\gamma}$.

Applying Theorem 2 and substituting into Equation (5) with $Z = \frac{1}{1-\gamma}$, we have

$$E[G_{\text{ALG}}] \geq ((1-\gamma) - \kappa\alpha S)\, G_{\text{OPT}} - \frac{(1-\gamma)}{\alpha}\ln M$$

where $\kappa \equiv e-2$. Dropping the $(1-\gamma)$ on the $\ln M$ term, plugging in $\gamma = S\alpha$, re-arranging, and substituting G for G_{OPT} gives

$$E[\text{Regret}] = G_{\text{OPT}} - E[G_{\text{ALG}}] \leq (e-1)S\alpha G + \frac{1}{\alpha}\ln M.$$

Plugging in our choice of α proves the theorem. ■

Note that the optimal choice of α depends on S and G; if good estimates of these are not available in advance, then one can make conservative guesses initially. If the current estimate is ever exceeded, then one simply restarts the algorithm after re-setting γ based on doubling the exceeded estimate. This only inflates the bounds by a constant factor. Such approaches are standard, for details see [CBL06].

For completeness, we also derive regret bounds for the EXP3 and EXP4 like algorithms NEXP(UA-Mix) and NEXP(UE-Mix):

Theorem 4 *Algorithm NEXP (UA-Mix), run with parameter* $\gamma = \min\left\{1, \sqrt{M\ln M}/\sqrt{(e-1)G}\right\}$ *has*

$$E[\textit{Regret}] \leq 2\sqrt{(e-1)GM\ln M},$$

and NEXP(UE-Mix), using

$$\gamma = \min\left\{1, \sqrt{A\ln M}/\sqrt{(e-1)G}\right\},$$

has

$$E[\textit{Regret}] \leq 2\sqrt{(e-1)GA\ln M}.$$

Proof (sketch): For NEXP(UE-Mix), the result follows along the lines of the previous theorem after showing $\alpha = \gamma/M$ satisfies condition ($\boldsymbol{\alpha}$), $Z = 1/(1-\gamma)$ satisfies ($\boldsymbol{Z}$), and $S \leq M$. For NEXP(UA-Mix), the proof uses $\alpha = \gamma/A$, $Z = 1/(1-\gamma)$, and $S \leq A$. ■

Bounds In Terms of the Average s_t. We now prove a bound that depends on the per-round s_t, rather than the max over all rounds. We will need the following lemma about weighted sums:

Lemma 5 *Fix constants $\bar{A} \geq \bar{a} \geq 0$. Let $w_1, \ldots, w_T \in \mathbb{R}^+$ be a sequence of non-negative real numbers, let $a_1, \ldots, a_T \in [0, \bar{a}]$, with the additional constraint that $\sum_t a_t \geq \bar{A}$. Let $n = \lfloor \bar{A}/\bar{a} \rfloor$. Then,*

$$\sum_t w_t a_t \geq \mathit{MB}(w, n) \sum_t a_t,$$

where $\mathit{MB}(w, n)$ is the mean of the n smallest w's.

Proof: Assume without loss of generality that w is sorted in non-decreasing order. Let $A = \sum_{t=1}^{T} a_t$ and $m = \lfloor A/\bar{a} \rfloor$. The sum $\sum_t w_t a_t$ is minimized by setting $a_t = \bar{a}$ for $1 \leq t \leq m$ and $a_{m+1} = A - m\bar{a}$. Thus,

$$\begin{aligned}
\sum_{t=1}^{T} w_t a_t &\geq \bar{a} \sum_{t=1}^{m} w_t + a_{m+1} w_{m+1} \\
&= m\bar{a}\,\mathrm{MB}(w, m) + a_{m+1} w_{m+1} \\
&\geq (m\bar{a} + a_{m+1})\,\mathrm{MB}(w, m) \qquad (13) \\
&\geq \mathrm{MB}(w, n) \sum_{t=1}^{T} a_t \qquad (14)
\end{aligned}$$

where line (13) follows because $\mathrm{MB}(w, m) \leq w_{m+1}$ and line (14) uses $A = m\bar{a} + a_{m+1}$ and $\mathrm{MB}(w, m) \geq \mathrm{MB}(w, n)$ because $m \geq n$. ■

Now we can prove the following Theorem, strengthening Theorem 2. The key additional assumption is that we can bound G_0 away from zero, as this lets us show that a few "bad" s_t can't hurt us too much.

Theorem 6 *Suppose $G_0 \geq S$, and let*

$$\tilde{S} = \frac{1}{\mathit{MB}(1/s_t, n_s)}$$

be the harmonic mean of the n_s largest s_t, where $n_s = \lfloor G_0/S \rfloor$.

Then algorithm NEXP (LP-Mix), run with parameter $\alpha = \sqrt{\ln M}/\sqrt{(e-1)\tilde{S}G}$, has regret bounded by

$$E[\mathit{Regret}] \leq 2\sqrt{(e-1)\tilde{S}G \ln M}\,.$$

Proof (sketch): Building on the proof of Lemma 2 and Theorem 3, it suffices to show that

$$E\left[\sum_t \sum_i q_{i,t} \hat{y}_{i,t}^2\right] \leq Z\tilde{S}G_0\,.$$

Algorithm LP-Mix-Solve

Define $p_{\min}(a) = \alpha \max_i \{e_i(a)\}$

Initialize $\bar{c} \leftarrow 1$

repeat

Let $\mathcal{A}_0 = \{a : p_{\min}(a) \geq \bar{c}\tilde{p}(a)\}$, and set

$$\bar{c} \leftarrow \frac{1 - \sum_{a \in \mathcal{A}_0} p_{\min}(a)}{\sum_{a \in \mathcal{A} \setminus \mathcal{A}_0} \tilde{p}(a)} \qquad (15)$$

until the update (15) produces no change in $\bar{c}$

Return the distribution $p(a) = \max\{p_{\min}(a), \bar{c}\tilde{p}(a)\}$

Figure 3: Algorithm LP-Mix-Solve.

Using Equation (11), we have

$$\sum_t \sum_i q_{i,t} \hat{y}_{i,t}^2 \leq Z \sum_t r_t(\acute{a}_t) \frac{\bar{e}_t(\acute{a}_t)}{p(\acute{a}_t)}\,.$$

Recall that $\bar{e}_t(a) = \max_i \{e_{i,t}(a)\}$. Let $\mathcal{A}_p = \{a \,|\, p(a) > 0\}$, and taking expectations, we get

$$E\left[\sum_t \sum_i q_{i,t} \hat{y}_{i,t}^2\right] \leq Z \sum_t \sum_{a \in \mathcal{A}_p} r_t(a) \bar{e}_t(a) = Z \sum_t g_t$$

where $g_t = \sum_{a \in \mathcal{A}_p} r_t(a) \bar{e}_t(a)$. It remains to show that $\sum_t g_t \leq \tilde{S}G_0$. To see this, note that $g_t \leq S$ and $\sum_t g_t \geq G_0 \geq S$. Thus, by Lemma 5,

$$G_0 = \sum_t \frac{1}{s_t} g_t \geq \mathrm{MB}(1/s_t, n_s) \sum_t g_t\,.$$

Rearranging this inequality gives $\sum_t g_t \leq \tilde{S}G_0$. ■

A Fast Algorithm for LP-Mix In this section, we present an efficient and easy-to-implement algorithm for solving LP-Mix. The algorithm, given in Figure 3, iteratively refines an upper bound $\bar{c}$ on the optimal objective function value until it reaches a feasible (and optimal) solution. Its performance is summarized in Theorem 7.

Theorem 7 *Assuming the linear program is feasible, algorithm LP-Mix-Solve runs for at most $1 + |\{a \,|\, \tilde{p}(a) > 0\}|$ iterations before returning an optimal p for the linear program*

$$\begin{aligned}
\max_{p,c} \quad & c \\
\textit{subject to} \quad & \forall a \quad p(a) \geq \alpha \max_i \{e_i(a)\} \\
& \forall a \quad p(a) \geq c\tilde{p}(a) \\
& \sum_a p(a) = 1.
\end{aligned}$$

Proof: Consider an arbitrary feasible solution (p, c). We first show that our algorithm maintains the invariant $\bar{c} \geq c$. First, note that

$$c = c \sum_a \tilde{p}(a) \leq \sum_a p(a) = 1\,.$$

so the invariant is true initially. For any set $\mathcal{A}_0 \subseteq \mathcal{A}$, we have

$$\sum_{a \in \mathcal{A}_0} p_{\min}(a) + \sum_{a \in \mathcal{A} \setminus \mathcal{A}_0} c\tilde{p}(a) \leq \sum_{a \in \mathcal{A}} p(a) = 1 \ .$$

Rearranging this inequality shows that (15) maintains the invariant.

Let c^* be the optimal value of the objective function. We next show that if $\bar{c} > c^*$, then applying (15) will reduce $\bar{c}$. To see this, consider the point $(\bar{p}, \bar{c})$, where

$$\bar{p}(a) = \max \{p_{\min}(a), \bar{c}\tilde{p}(a)\} \ .$$

Because $\bar{c} > c^*$, the point $(\bar{p}, \bar{c})$ cannot be feasible, which implies $\sum_a \bar{p}(a) \neq 1$ (the other two constraints are satisfied by construction). Assume $\sum_a \bar{p}(a) > 1$. Then, for the $\mathcal{A}_0$ defined from $\bar{c}$,

$$1 < \sum_a \bar{p}(a) = \sum_{a \in \mathcal{A}_0} p_{\min}(a) + \sum_{a \in \mathcal{A} \setminus \mathcal{A}_0} \bar{c}\tilde{p}(a) \ .$$

Rearranging this inequality implies that (15) will decrease $\bar{c}$. On the other hand, if $\sum_a \bar{p}(a) < 1$ then we could increase the components of $\bar{p}$ arbitrarily to obtain a feasible solution $(\bar{p}', \bar{c})$, contradicting $\bar{c} > c^*$.

Thus, $\bar{c}$ keeps decreasing until $\bar{c} = c^*$, at which point $\bar{c}$ no longer changes. This shows that our algorithm is correct assuming it terminates.

We now consider the time the algorithm requires. Because $\bar{c}$ is non-increasing, the set $\mathcal{A}_0$ can only grow across iterations. Furthermore, by inspection of (15) we see that if $\bar{c}$ decreases, $|\mathcal{A}_0|$ must have increased. Thus there can be at most $|\mathcal{A}|$ iterations before $\bar{c}$ does not change (at which point the algorithm terminates). To tighten this bound, note that every action a with $\tilde{p}(a) = 0$ is added to $\mathcal{A}_0$ on the first iteration, so in fact the number of iterations can be at most $1 + |\{a \,|\, \tilde{p}(a) > 0\}|$.

■

6 Experiments

We compare our new algorithm to EXP3 and EXP4 on a large, real-world problem: predicting ad clicks on a search engine. EXP3 is run directly on the experts, as discussed in Section 2. In the "real" ad-selection bandit problem, the search engine chooses a few (say 10) ads to show from a presumably much larger set of ads targeted at a particular query; from this, we construct a smaller bandit problem where we pretend only the 10 ads actually shown are relevant, and from this set select a single ad to "show." This simplification is necessary because we only observe rewards (click vs. no-click) for the ads that were shown to users. This sidesteps a typical challenge in evaluating bandit algorithms on real datasets: if the bandit problem was "real" then only a single reward is observed each round, but for low-variance evaluation of different bandit algorithms, the experimenter needs access to the full reward vector.

Our datasets are based on anonymized query information from google.com.[4] From a 12 month period, we collected queries for a particular phrase (e.g., "canon 40d") where at least two ads were shown and at least one ad was clicked by a user. We then transformed this to a prediction problem with a feature vector x for each (query, ad) pair that was shown, using features based on the text of the ad and the query; the target label is 1 if the ad was clicked, and 0 otherwise.

[4]No user-specific data was used in these experiments.

	Avg. Regret	
	Actual	Bound
Exp4	0.580	1.905
Exp3	0.143	0.303
NEXP	**0.047**	0.106

Table 3: Average experimental per-round regret and theoretical bounds, for a problem with $T = 19,713$, $A = 3567$, $M = 90$, $S = 11$, and $G_{\text{OPT}} = 0.649$. Regrets are averaged over 100 runs, and 95% confidence intervals are all tighter than ± 0.004. Parameters were set and bounds computed using the true S and G; note the bound on EXP4 is vacuous.

Using the first 9 months of data, we trained a family of logistic regression models $m(\lambda, [a, b])$, where λ gives the amount of L_1-regularization and $[a, b]$ indicates which months of data this particular model trained on; for example, $[a, b] = [1, 9]$ trains on all the data, while $[9, 9]$ trains on only the most recent month.

These models were fixed, and used to produce experts for a hypothetical bandit problem played on the data from months 10–12. Each timestep t in the bandit problem maps to a real query that occured on google.com. On each round, the bandit algorithm faces the problem of choosing a single ad to show. The full set $\mathcal{A}$ of actions corresponds to the set of all ads shown on the included queries over the 3 months. On a given round, an ad/action a has reward 1 if it was shown by the search engine and was clicked by the user, and 0 otherwise.

For each model m, the bandit algorithm has access to a deterministic expert $E(m)$. The expert $E(m)$ receives side-information, namely, the query phrase and the set of ads google showed when the query originally occured (only these ads can have positive reward). The expert/model then predicts the probability of a click on each ad in this set, and recommends deterministically the action which received the highest prediction. For example, if ads (a_1, a_2, a_3) were shown on the query for round t, and model m predicts $(0.05, 0.02, 0.03)$ respectively, then the expert $E(m)$ recommends the distribution $(1, 0, 0)$.

Our goal is not to fully capture the complexity of deciding which ads to show alongside search results. In particular, we ignore the effect of the position in which ads are shown, the auction typically used to rank and price the ads, the fact that multiple ads are usually shown, and the fact that the set of available actions would typically be a larger set (e.g., all ads targeted at the query from advertisers with remaining budgets). However, we believe our setup captures enough of the essence of the problem to be useful for evaluating how well different bandit algorithms might apply to such real-world problems.

We report results for a representative dataset, based on queries for "canon 40d". About 200,000 training examples were selected from the 9-month training period, on which we trained 90 models based on different combinations of

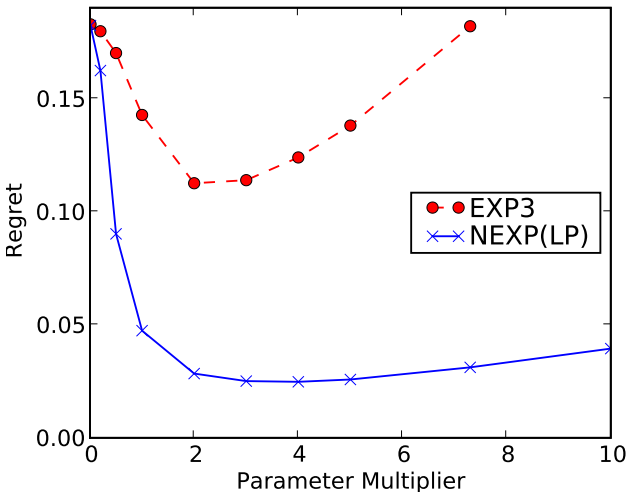

Figure 4: Effect on regret of varying γ for EXP3 and α for NEXP. The Y-axis is scaled so that the top of the plot corresponds to the performance of the *worst* expert; NEXP outperforms EXP3 for all parameters. The parameter values are given as multiples of the parameters used in Table 3.

the regularization and date-range parameters. From the following 3-months of data, we formed a bandit problem with 19,713 timesteps and 90 experts.

Table 3 shows the experimental average per round regret along with the corresponding bound. The parameters were chosen based on Theorem 3 for NEXP and the corresponding results from [ACBFS02] for EXP3 and EXP4, using the true values for S and G—in practice good estimates of these are likely to be available in advance; for example $S \leq 11$ follows immediately from the side information present in our example domain, since this is the maximum number of ads google.com shows on a single query. Figure 4 shows the effect of using different parameters than those recommended by the regret bounds. We explored the parameter space by multiplying the parameter settings used for Table 3 by multipliers $m \in [0, 10]$, discarding values that produced infeasible parameter settings ($\gamma > 1, \alpha > 1/S$). The total number of actions available in this problem is so large that EXP4 performs hopelessly badly, as indicated in Table 3, and so it is omitted from Figure 4. In fact, the theory suggests EXP4 should get $\gamma = 1$ for this problem (always play uniformly random actions); we experimented with different parameter settings, and the best results were effectively for $\gamma = 0$, which essential plays a random expert. [5]

These experiments demonstrate that the optimized exploration strategy used by NEXP is not only a theoretical improvement useful in deriving tighter regret bounds, but also an important algorithmic improvement that can produce significantly lower regret in real-world applications.

[5] It is possible to run EXP4 with uniform exploration on the actions that some expert recommends with positive probability. The standard analysis of EXP4 does not apply to this modified algorithm, however. And, while this algorithm can easily be analyzed along the lines of Theorem 3, it fails immediately if one introduces a single "unsure" expert which puts some small probability on each action.

7 Conclusions

We have introduced NEXP, a new algorithm for the bandit problem with expert advice. NEXP provides a bound of $\sqrt{GS \log M}$ on expected cumulative regret, where $S \leq \min\{A, M\}$ (in practice, S can be much smaller). A refined bound shows that a certain average $\tilde{S}$ can be used in place of S. Experiments demonstrated that on a realistic problem of significant real-world importance, our improved algorithms dramatically outperform previously published approaches.

Acknowledgements The authors would like to thank Gary Holt, Arkady Epshteyn, Brent Bryan, Mike Meyer, and Andrew Moore for interesting discussions and feedback.

References

[ACBFS02] Peter Auer, Nicol Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.

[BEYL04] Yoram Baram, Ran El-Yaniv, and Kobi Luz. Online choice of active learning algorithms. *J. Mach. Learn. Res.*, 5:255–291, 2004.

[CBL06] Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA, 2006.

[dFM06] Daniela Pucci de Farias and Nimrod Megiddo. Combining expert advice in reactive environments. *J. ACM*, 53(5):762–799, 2006.

[DHK07] Varsha Dani, Thomas Hayes, and Sham M. Kakade. The price of bandit information for online optimization. In *NIPS*, 2007.

[FS95] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37, 1995.

[GP07] Rica Gonen and Elan Pavlov. An incentive-compatible multi-armed bandit mechanism. In *PODC*, pages 362–363, 2007.

[KMN00] Michael Kearns, Yishay Mansour, and Andrew Y. Ng. Approximate planning in large POMDPs via reusable trajectories. In *NIPS*, 2000.

[LR85] T. L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Adv. in Appl. Math.*, 6:4–22, 1985.

[LZ07] John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. *NIPS*, 2007.

[PO07] Sandeep Pandey and Christopher Olston. Handling advertisements of unknown quality in search advertising. In *NIPS*, pages 1065–1072, 2007.

[RCKU08] F. Radlinski, D. Chakrabarti, R. Kumar, and E. Upfal. Mortal multi-armed bandits. In *NIPS*, 2008.

[Var07] Hal R. Varian. Position auctions. *International Journal of Industrial Organization*, 25(6):1163–1178, December 2007.

[WVLL07] Jennifer Wortman, Yevgeniy Vorobeychik, Lihong Li, and John Langford. Maintaining equilibria during exploration in sponsored search auctions. In *WINE*, pages 119–130, 2007.