Large-Scale Bandit Problems and KWIK Learning

Jacob Abernethy JABER@SEAS.UPENN.EDU
Kareem Amin AKAREEM@SEAS.UPENN.EDU
Computer and Information Science, University of Pennsylvania

Moez Draief M.DRAIEF@IMPERIAL.AC.UK
Electrical and Electronic Engineering, Imperial College, London

Michael Kearns MKEARNS@CIS.UPENN.EDU
Computer and Information Science, University of Pennsylvania

Abstract

We show that parametric multi-armed bandit (MAB) problems with large state and action spaces can be algorithmically reduced to the supervised learning model known as "Knows What It Knows" or *KWIK* learning. We give matching impossibility results showing that the KWIK-learnability requirement cannot be replaced by weaker supervised learning assumptions. We provide such results in both the standard parametric MAB setting, as well as for a new model in which the action space is finite but growing with time.

1. Introduction

We examine multi-armed bandit (MAB) problems in which both the state (sometimes also called context) and action spaces are very large, but learning is possible due to parametric or similarity structure in the payoff function. Motivated by settings such as web search, where the states might be all possible user queries, and the actions are all possible documents or advertisements to display in response, such large-scale MAB problems have received a great deal of recent attention (Li et al., 2010; Langford & Zhang, 2007; Lu et al., 2010; Slivkins, 2011; Beygelzimer et al., 2011; Wang et al., 2008; Auer et al., 2007; Bubeck et al., 2008; Kleinberg et al., 2008; Amin et al., 2011a;b).

Our main contribution is a new algorithm and reduction showing a strong connection between large-scale MAB problems and the *Knows What It Knows* or *KWIK* model of supervised learning (Li et al., 2011; Li & Littman, 2010; Sayedi et al., 2010; Strehl & Littman, 2007; Walsh et al., 2009). KWIK learning is an online model of learning a class of functions that is strictly more demanding than standard no-regret online learning, in that the learning algorithm *must* either make an accurate prediction on each trial or output "don't know". The performance of a KWIK algorithm is measured by the number of such don't-know trials.

Our first results show that the large-scale MAB problem given by a parametric class of payoff functions can be efficiently reduced to the supervised KWIK learning of the same class. Armed with existing algorithms for KWIK learning, e.g. for noisy linear regression (Strehl & Littman, 2007; Walsh et al., 2009), we obtain new algorithms for large-scale MAB problems. We also give a matching intractability result showing that the demand for KWIK learnability is necessary, in that it cannot be replaced with standard online no-regret supervised learning, or weaker models such as PAC learning, while still implying a solution to the MAB problem. Our reduction is thus tight with respect to the necessity of the KWIK learning assumption.

We then consider an alternative model in which the action space remains large, but in which only a subset is available to the algorithm at any time, and this subset is growing with time. This even better models settings such as sponsored search, where the space of *possible* ads is very large, but at any moment the search engine can only display those ads that have actually been placed by advertisers. We again show that such MAB problems can be reduced to KWIK learning, provided the arrival rate of new actions is sublinear in the num-

Proceedings of the 30^th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. Copyright 2013 by the author(s).

Moez Draief is supported by FP7 EU project SMART (FP7-287583).

ber of trials. We also give information-theoretic impossibility results showing that this reduction is tight, in that weakening its assumptions no longer implies solution to the MAB problem. We conclude with a brief experimental illustration of this arriving-action model.

While much of the prior work on KWIK learning has studied the model for its own sake, our results demonstrate that the strong demands of the KWIK model provide benefits for large-scale MAB problems that are provably not provided by weaker models of supervised learning. We hope this might actually motivate the search for more powerful KWIK algorithms. Our results also fall into the line of research showing reductions and relationships between bandit-style learning problems and traditional supervised learning models (Langford & Zhang, 2007; Beygelzimer et al., 2011; Beygelzimer & Langford, 2009).

2. Large-Scale Multi-Armed Bandits

The Setting. We consider a sequential decision problem in which a learner, on each round t, is presented with a *state* $\mathbf{x}^t$, chosen by Nature from a large state space $\mathcal{X}$. The learner responds by choosing an *action* $\mathbf{a}^t$ from a large action space $\mathcal{A}$. We assume that the learner's (noisy) payoff is $f_\theta(\mathbf{x}^t, \mathbf{a}^t) + \eta^t$, where η^t is an i.i.d. random variable with $\mathbb{E}[\eta^t] = 0$. The function f_θ is unknown to the learner, but is chosen from a (parameterized) family of functions $\mathcal{F}_\Theta = \{f_\theta : \mathcal{X} \times \mathcal{A} \to \mathbb{R}^+ \mid \theta \in \Theta\}$ that is known to the learner. We assume that every $f_\theta \in \mathcal{F}_\Theta$ returns values bounded in $[0, 1]$. In general we make no assumptions on the sequence of states $\mathbf{x}^t$, stochastic or otherwise. An instance of such a MAB problem is fully specified by $(\mathcal{X}, \mathcal{A}, \mathcal{F}_\Theta)$.

We will informally use the term "large-scale MAB problem" to indicate that both $|\mathcal{X}|$ and $|\mathcal{A}|$ are large or infinite, and that we seek algorithms whose resource requirements are greatly sublinear or independent of both. This is in contrast to works in which either only $|\mathcal{X}|$ was assumed to be large (Langford & Zhang, 2007; Beygelzimer et al., 2011) (which we shall term "large-state"; it is also commonly called *contextual bandits* in the literature), or only $|\mathcal{A}|$ is large (Kleinberg et al., 2008) (which we shall term "large-action"). We now define our notion of *regret*, which permits arbitrary sequences of states.

Definition 1. *An algorithm for the large-scale MAB problem* $(\mathcal{X}, \mathcal{A}, \mathcal{F}_\Theta)$ *is said to have* **no regret** *if, for any* $f_\theta \in \mathcal{F}_\Theta$ *and any sequence* $\mathbf{x}^1, \mathbf{x}^2, \ldots \mathbf{x}^T \in \mathcal{X}$*, the algorithm's action sequence* $\mathbf{a}^1, \mathbf{a}^2, \ldots \mathbf{a}^T \in \mathcal{A}$ *satisfies* $R_A(T)/T \to 0$ *as* $T \to \infty$*, where we define* $R(T) \triangleq$ $E\left[\sum_{t=1}^{T} \max_{\mathbf{a}_*^t \in \mathcal{A}} f_\theta(\mathbf{x}^t, \mathbf{a}_*^t) - f_\theta(\mathbf{x}^t, \mathbf{a}^t)\right]$.

We shall be particularly interested in algorithms for which we can provide fast *rates* of convergence to no regret.

Example: Pairwise Interaction Models. We introduce a running example we shall use to illustrate our assumptions and results; other examples are discussed later. Let the state $\mathbf{x}$ and action $\mathbf{a}$ both be (bounded norm) d-dimensional vectors of reals. Let θ be a (bounded) d^2-dimensional parameter vector, and let $f_\theta(\mathbf{x}, \mathbf{a}) = \sum_{1 \le i,j \le d} \theta_{i,j} x_i a_j$; we then define $\mathcal{F}_\Theta$ to be the class of all such models f_θ. In such models, the payoffs are determined by pairwise interactions between the variables, and both the sign and magnitude of the contribution of $x_i a_j$ is determined by the parameter $\theta_{i,j}$. For example, imagine an application in which each state $\mathbf{x}$ represents demographic and behavioral features of an individual web user, and each action $\mathbf{a}$ encodes properties of an advertisement that could be presented to the user. A zipcode feature in $\mathbf{x}$ indicating the user lives in an affluent neighborhood and a language feature in $\mathbf{a}$ indicating that the ad is for a premium housecleaning service might have a large positive coefficient, while the same zipcode feature might have a large negative coefficient with a feature in $\mathbf{a}$ indicating that the service is not yet offered in the user's city.

3. Assumptions: KWIK Learnability and Fixed-State Optimization

We next articulate the two assumptions we require on the class $\mathcal{F}_\Theta$ in order to obtain resource-efficient no-regret MAB algorithms. The first is KWIK learnabilty of $\mathcal{F}_\Theta$, a strong notion of *supervised* learning, introduced by Li et al. in 2008 (Li et al., 2008; 2011). The second is the ability to find an approximately optimal action for a *fixed* state. Either one of these conditions in isolation is clearly insufficient for solving the large-scale MAB problem: KWIK learning of $\mathcal{F}_\Theta$ has no notion of choosing actions, but instead assumes input-output pairs $\langle \mathbf{x}, \mathbf{a} \rangle, f_\theta(\mathbf{x}, \mathbf{a})$ are simply given; whereas the ability to optimize actions for fixed states is of no obvious value in our changing-state MAB model. We will show, however, that together these assumptions can exactly compensate for each other's deficiencies and be combined to solve the large-scale MAB problem.

3.1. KWIK Learning

In the KWIK learning protocol (Li et al., 2008), we assume we have an input space $\mathcal{Z}$ and an output space

$\mathcal{Y} \subset \mathbb{R}$. The learning problem is specified by a function $f : \mathcal{Z} \to \mathcal{Y}$, drawn from a specified function class $\mathcal{F}$. The set $\mathcal{Z}$ can generally be arbitrary but, looking ahead, our reduction from a large-scale MAB problem $(\mathcal{X}, \mathcal{A}, \mathcal{F}_\Theta)$ to a KWIK problem will set the function class as $\mathcal{F} = \mathcal{F}_\Theta$ and the input space as $\mathcal{Z} = \mathcal{X} \times \mathcal{A}$, the joint state and action spaces.

The learner is presented with a sequence of observations $\mathbf{z}^1, \mathbf{z}^2, \ldots \in \mathcal{Z}$ and, immediately after observing $\mathbf{z}^t$, is asked to make a prediction of the value $f(\mathbf{z}^t)$, but is allowed to predict the value $\perp$ meaning "don't know".

Thus in KWIK model the learner may confess ignorance on any trial. Upon a report of "don't know", where $y^t = \perp$, the learner is given feedback, receiving a noisy estimate of $f(\mathbf{z}^t)$. However, if the learner chooses to make a prediction of $f(\mathbf{z}^t)$, *no feedback* is received [1], and this prediction *must* be ϵ-accurate, or else the learner fails entirely. In the KWIK model the aim is to make only a *bounded number* of $\perp$ predictions, and thus make ϵ-accurate predictions on almost every trial. Specifically:

```
1: Nature selects $f \in \mathcal{F}$
2: **for** $t = 1, 2, 3, \ldots$ **do**
3:     Nature selects $\mathbf{z}^t \in \mathcal{Z}$ and presents to learner
4:     Learner predicts $y^t \in \mathcal{Y} \cup \{\perp\}$
5:     **if** $y^t = \perp$ **then**
6:         Learner observes value $f(\mathbf{z}^t) + \eta^t$,
7:         where $\eta^t$ is a bounded 0-mean noise term
8:     **else if** $y^t \neq \perp$ and $|y^t - f(\mathbf{z}^t)| > \epsilon$ **then**
9:         FAIL and **exit**
10:    **end if**
11:    // Continue if $y^t$ is $\epsilon$-accurate
12: **end for**
```

Definition 2. *Let the error parameter be $\epsilon > 0$ and the failure parameter be $\delta > 0$. Then $\mathcal{F}$ is said to be* KWIK-learnable *with don't-know bound $\mathbf{B} = \mathbf{B}(\epsilon, \delta)$ if there exists an algorithm such that for any sequence $\mathbf{z}^1, \mathbf{z}^2, \mathbf{z}^3, \ldots \in \mathcal{Z}$, the sequence of predictions $y^1, y^2, \ldots \in \mathcal{Y} \cup \{\perp\}$ satisfies $\sum_{t=1}^{\infty} \mathbf{1}[y^t = \perp] \leq \mathbf{B}$, and the probability of* `FAIL` *is at most δ. Any class $\mathcal{F}$ is said to be* efficiently *KWIK-learnable if there exists an algorithm that satisfies the above condition and on every round runs in time* $\text{poly}(\epsilon^{-1}, \delta^{-1})$.

Example Revisited: Pairwise Interactions. We show that KWIK learnability holds here. Recalling that $f_\theta(\mathbf{x}, \mathbf{a}) = \sum_{1 \leq i,j \leq d} \theta_{i,j} x_i a_j$, we can linearize the model by viewing the KWIK inputs as having d^2 components $z_{i,j} = x_i a_j$, with coefficients $\theta_{i,j}$, and the KWIK learnability of $\mathcal{F}_\Theta$ simply reduces to KWIK noisy linear regression, which has an efficient algorithm (Li et al., 2011; Strehl & Littman, 2007; Walsh et al., 2009).

[1]This aspect of KWIK learning is crucial for our reduction.

3.2. Fixed-State Optimization

We next describe the aforementioned fixed-state optimization problem for $\mathcal{F}_\Theta$. Assume we have a fixed function $f_\theta \in \mathcal{F}_\Theta$, a fixed state $\mathbf{x} \in \mathcal{X}$, and some $\epsilon > 0$. Then an algorithm shall be referred to as a *fixed-state optimization* algorithm for $\mathcal{F}_\Theta$ if the algorithm makes a series of (action) queries $\mathbf{a}^1, \mathbf{a}^2, \ldots \in \mathcal{A}$, and in response to $\mathbf{a}^i$ receives approximate feedback y^i satisfying $|y^i - f_\theta(\mathbf{x}, \mathbf{a}^i)| \leq \epsilon$; and then outputs a final action $\hat{\mathbf{a}} \in \mathcal{A}$ satisfying $\arg\max_{\mathbf{a} \in \mathcal{A}} \{f_\theta(\mathbf{x}, \mathbf{a})\} - f_\theta(\mathbf{x}, \hat{\mathbf{a}}) \leq \epsilon$. In other words, for any fixed state $\mathbf{x}$, given access only to (approximate) input-output queries to $f_\theta(\mathbf{x}, \cdot)$, the algorithm finds an (approximately) optimal action under f_θ and $\mathbf{x}$. It is not hard to show that if we define $\mathcal{F}_\Theta(\mathcal{X}, \cdot) = \{f_\theta(\mathbf{x}, \cdot) : \theta \in \Theta, \mathbf{x} \in \mathcal{X}\}$ — which defines a class of *large-action* MAB problems *induced* by the class $\mathcal{F}_\Theta$ of large-scale MAB problems, each one corresponding to a *fixed* state — then the assumption of fixed-state optimization for $\mathcal{F}_\Theta$ is in fact equivalent to having a no-regret algorithm for $\mathcal{F}_\Theta(\mathcal{X}, \cdot)$. In this sense, the reduction we will provide shortly can be viewed as showing that KWIK learnability bridges the gap between the large-scale problem $\mathcal{F}_\Theta$ and its induced large-action problem $\mathcal{F}_\Theta(\mathcal{X}, \cdot)$.

Example Revisited: Pairwise Interactions. We show that fixed-state optimization holds here. For any fixed state $\mathbf{x}$ we wish to approximately maximize the output of $f_\theta(\mathbf{x}, \mathbf{a}) = \sum_{i,j} \theta_{i,j} x_i a_j$ from approximate queries. Since $\mathbf{x}$ is fixed, we can view the coefficient on a_j as $\tau_j = \sum_i \theta_{i,j} x_i$. While there is no hope of distinguishing θ and $\mathbf{x}$, there is no need to: querying on the jth standard basis vector returns (an approximation to) the value of τ_j. After doing so for each dimension j, we can output whichever basis vector yielded the highest payoff.

4. A Reduction of MAB to KWIK

We now give a reduction and algorithm showing that the assumptions of both KWIK-learnability and fixed-state optimization of $\mathcal{F}_\Theta$ suffice to obtain an efficient no-regret algorithm for the MAB problem for $\mathcal{F}_\Theta$. The high-level idea of the algorithm is as follows. Upon receiving the state $\mathbf{x}_t$, we attempt to simulate the assumed fixed-state optimization algorithm `FixedStateOpt` on $f_\theta(\mathbf{x}^t, \cdot)$. Unfortunately, we do not have the required oracle access to $f_\theta(\mathbf{x}^t, \cdot)$, due to the fact that the state changes with each action that we take. Therefore, we will instead make use of the assumed KWIK learning algorithm as a surrogate. So long as KWIK never outputs $\perp$, the optimization subroutine terminates with an approximate optimizer for $f_\theta(\mathbf{x}^t, \cdot)$. If KWIK returns $\perp$ sometime during the simulation of FixedStateOpt, we halt that optimization but increase the don't-know count of KWIK, which can only happen finitely often. The precise algorithm follows.

Algorithm 1 KWIKBandit: MAB Reduction to KWIK + FixedStateOpt

```
1: Initialize KWIK to learn unknown $f_\theta \in \mathcal{F}_\Theta$.
2: for $t = 1, 2, \ldots$ do
3:     $\mathbf{x}^t \xleftarrow{\text{receive}}$ MAB
4:     $i \xleftarrow{\text{set}} 0$
5:     feedbackflag $\xleftarrow{\text{set}}$ FALSE
6:     Init FixedStateOpt$^t$ to optimize $f_\theta(\mathbf{x}^t, \cdot)$
7:     while $i \xleftarrow{\text{set}} i + 1$ do
8:         $\mathbf{a}_i^t \xleftarrow{\text{query}}$ FixedStateOpt$^t$
9:         if FixedStateOpt$^t$ terminates then
10:            $\mathbf{a}^t \xleftarrow{\text{set}} \mathbf{a}_i^t$
11:            break while
12:        end if
13:        $\mathbf{z}^t = (\mathbf{x}^t, \mathbf{a}_i^t) \xrightarrow{\text{input}}$ KWIK
14:        $\hat{y}_i^t \xleftarrow{\text{output}}$ KWIK
15:        if $\hat{y}_i^t = \perp$ then
16:            $\mathbf{a}^t \xleftarrow{\text{set}} \mathbf{a}_i^t$
17:            feedbackflag $\xleftarrow{\text{set}}$ TRUE
18:            break while
19:        else
20:            $\hat{y}_i^t \xrightarrow{\text{feedback}}$ FixedStateOpt$^t$
21:        end if
22:    end while
23:    $\mathbf{a}^t \xrightarrow{\text{action}}$ MAB
24:    $f_\theta(\mathbf{x}^t, \mathbf{a}^t) + \eta^t = y^t \xleftarrow{\text{observe}}$ MAB
25:    if feedbackflag = TRUE then
26:        $y^t \xrightarrow{\text{feedback}}$ KWIK
27:    end if
28: end for
```

Theorem 1. *Assume we have a family of functions $\mathcal{F}_\Theta$, a KWIK-learning algorithm KWIK for $\mathcal{F}_\Theta$, and a fixed-state optimization algorithm FixedStateOpt. Then the* average *regret of Algorithm 1, $R_A(T)/T$, will be arbitrarily small for appropriately-chosen ϵ and δ, and large enough T. Moreover, the running time is polynomial in the running time of KWIK and FixedStateOpt.*

Proof. We first bound the cost of Algorithm 1. Let us consider the result of one round of the outermost loop, i.e. for some fixed t. First, consider the event that KWIK does not FAIL on any trial, so we are guaranteed that $\hat{y}_i^t$ is an ϵ-accurate estimate of $f_\theta(\mathbf{x}^t, \mathbf{a}_i^t)$. In this case the **while** loop can be broken in one of two ways: (1) KWIK returns $\perp$ on the pair $(\mathbf{x}^t, \mathbf{a}_i^t)$. In this case, because we have assumed a bounded range for f_θ, we can say that $\max_{\mathbf{a}_*^t} f_\theta(\mathbf{x}^t, \mathbf{a}_*^t) - f_\theta(\mathbf{x}^t, \mathbf{a}^t) \leq 1$. (2) FixedStateOpt terminates and returns $\mathbf{a}^t$. But this $\mathbf{a}^t$ is ϵ-optimal per our definition, hence we have that $\max_{\mathbf{a}_*^t} f_\theta(\mathbf{x}^t, \mathbf{a}_*^t) - f_\theta(\mathbf{x}^t, \mathbf{a}^t) \leq \epsilon$.

Therefore, on a trial t, we can bound $\max_{\mathbf{a}_*^t} f_\theta(\mathbf{x}^t, \mathbf{a}_*^t) - f_\theta(\mathbf{x}^t, \mathbf{a}^t) \leq \mathbf{1}[\texttt{KWIK outputs } \perp \text{ on round } t] + \epsilon$.

Taking the average over $t = 1, \ldots, T$ we have

$$\frac{1}{T}\sum_{t=1}^{T} \max_{\mathbf{a}_*^t} f_\theta(\mathbf{x}^t, \mathbf{a}_*^t) - f_\theta(\mathbf{x}^t, \mathbf{a}^t) \leq \frac{\mathbf{B}(\epsilon, \delta)}{T} + \epsilon \quad (1)$$

where $\mathbf{B}(\epsilon, \delta)$ is the don't-know bound of KWIK. Inequality (1) holds on the event that KWIK does not FAIL. By definition, the probability that it does FAIL is at most δ, and in that case all we can say is that $(1/T)\sum_{t=1}^{T} \max_{\mathbf{a}_*^t} f_\theta(\mathbf{x}^t, \mathbf{a}_*^t) - f_\theta(\mathbf{x}^t, \mathbf{a}^t) \leq 1$ Therefore:

$$\frac{R(T)}{T} \leq \frac{\mathbf{B}(\epsilon, \delta)}{T} + \epsilon + \delta. \quad (2)$$

We must now show that the quantity on the right hand side of the equation 2 vanishes with correctly chosen ϵ and δ. But this is achieved trivially: for any small $\gamma > 0$ if we select $\delta = \epsilon < \gamma/3$ and for $T > \frac{3\mathbf{B}(\epsilon,\delta)}{\gamma}$ we have that $\frac{\mathbf{B}(\epsilon,\delta)}{T} + \epsilon + \delta < \gamma$ as desired. □

Algorithm 1 is not exactly a no-regret MAB algorithm, since it requires parameter choices to obtain small regret. But this is easily remedied.

Corollary 1. *Under the assumptions of Theorem 1, there exists a no-regret algorithm for the MAB problem on $\mathcal{F}_\Theta$.*

Proof sketch. This follows as a direct consequence of Theorem 1 and a standard use of the "doubling trick" for selecting the input parameters in an online fashion. The simple construction runs a sequence of versions of Algorithm 1 with decaying choices of ϵ, δ. A detailed proof is provided in the Appendix. □

The interesting case occurs when $\mathcal{F}_\Theta$ is efficiently KWIK-learnable with a polynomial don't-know bound. In that case, we can obtain fast rates of convergence to no-regret. For all known KWIK algorithms $\mathbf{B}(\epsilon, \delta)$ is polynomial in ϵ^{-1} and poly-logarithmic in δ^{-1}. The following corollary is left as a straightforward exercise, following from equation (2).

Corollary 2. *If the don't-know bound of* KWIK *is* $\mathbf{B}(\epsilon, \delta) = O(\epsilon^{-d} \log^k \delta^{-1})$ *for some* $d > 0, k \geq 0$ *then we have* $R(T)/T = O\left(\left(\frac{1}{T}\right)^{\frac{1}{d+1}} \log^k T\right)$.

Example Revisited: Pairwise Interactions. As we have previously argued, the assumptions of KWIK learning and fixed-state optimization are met for the class of pairwise interaction models, so Theorem 1 can be applied directly, yielding a no-regret algorithm. More generally, a no-regret result can be obtained for any $\mathcal{F}_\Theta$ that can be similarly "linearized"; this includes a rather rich class of graphical models for bandit problems studied in (Amin et al., 2011a) (whose main result can be viewed as a special case of Theorem 1). Other applications of Theorem 1 include $\mathcal{F}_\Theta$ that obey a Lipschitz condition, where we can apply covering techniques to obtain the KWIK subroutine (details omitted), and various function classes in the boolean setting (Li et al., 2011).

4.1. No Weaker General Reduction

While Theorem 1 provides general conditions under which large-scale MAB problems can be solved efficiently, the assumption of KWIK learnability of $\mathcal{F}_\Theta$ is still a strong one, with noisy linear regression being the richest problem for which there is a known KWIK algorithm. For this reason, it would be nice to replace the KWIK learning assumption with a weaker learning assumption [2]. However, in the following theorem, we prove (under standard cryptographic assumptions) that there is in fact *no* general reduction of the MAB problem for $\mathcal{F}_\Theta$ to a weaker model of supervised learning. More precisely, we show that the "next strongest" standard model of supervised learning after KWIK, which is no-regret on arbitrary sequences of trials, does not imply no-regret MAB. This immediately implies that even weaker learning models (such as PAC learnability) also cannot suffice for no-regret MAB.

Theorem 2. *There exists a class of models* $\mathcal{F}_\Theta$ *such that*

- $\mathcal{F}_\Theta$ *is fixed-state optimizable.*
- *There is an efficient algorithm* A *such that on an arbitrary sequence of* T *trials* $\mathbf{z}^t$, A *makes a prediction* $\hat{y}^t$ *of* $y^t = f_\theta(\mathbf{z}^t)$ *and receives* y^t *as feedback; and the total regret* $err(T) \triangleq \sum_{t=1}^T |y^t - \hat{y}^t|$ *is sublinear in* T. *Thus we have only no-regret supervised learning instead of the stronger KWIK learning.*
- *Under standard cryptographic assumptions, there is no polynomial-time algorithm for the no-regret MAB problem for* $\mathcal{F}_\Theta$, *even if the state sequence is generated randomly from the uniform distribution.*

We leave this proof for the Appendix.

5. A Model for Arriving Actions

In the model examined so far, we have been assuming that the action space $\mathcal{A}$ is large — exponentially large or perhaps infinite — but also that the *entire* action space is available on every trial. In many natural settings, however, this property may be violated. For instance, in sponsored search, while the space of all possible ads is indeed very large, at any given moment the search engine can choose to display only those ads that have actually been created by extant advertisers. Furthermore these advertisers arrive gradually over time, creating a growing action space. In this setting, the algorithm of Theorem 1 cannot be applied, as it assumes the ability to optimize over *all* of $\mathcal{A}$ at each step. In this section we introduce a new model and algorithm to capture such scenarios.

Setting. As before, the learner is presented with a sequence of arriving states $\mathbf{x}^1, \mathbf{x}^2, \mathbf{x}^3, \ldots \in \mathcal{X}$. The set of available actions, however, shall not be fixed in advance but instead will grow with time. Let $\mathcal{F}$ be the set of all possible actions where, formally, we shall imagine that each $f \in \mathcal{F}$ is a function $f : \mathcal{X} \rightarrow [0, 1]$; $f(\mathbf{x})$ represents the payoff of action f on $\mathbf{x} \in \mathcal{X}$ [3]. Initially the action pool is $\mathcal{F}^0 \subset \mathcal{F}$, and on each round t a (possibly empty) set of new actions $S^t \subset \mathcal{F}$ arrives and is added to the pool, hence the available action pool on round t is $\mathcal{F}^t := \mathcal{F}^{t-1} \cup S^t$. We emphasize that when we say a new set of actions "arrives", we do *not* mean that the learner is given the actual *identity* of the corresponding functions, which it must learn to approximate, but rather that the learner is given (noisy) black-box input-output access to them. Let $N(t) = |\mathcal{F}^t|$ denote the size of the action pool at time t. Our results will depend crucially on this growth rate $N(t)$, in particular on it being *sublinear* [4]. One

[2]Note that we should not expect to replace or weaken the assumption of fixed-state optimization, since we have already noted that this is already implied by a no-regret algorithm for the MAB problem.

[3]Note that now each action is represented by its own payoff function, in contrast to the earlier model in which actions were inputs $\mathbf{a}$ into the $f_\theta(\mathbf{x}, \mathbf{a})$. The models coincide if we choose $\mathcal{F} = \{f_\theta(\cdot, \mathbf{a}) : \mathbf{a} \in \mathcal{A}, \theta \in \Theta\}$.

[4]Sublinearity of $N(t)$ seems a mild and natural assumption in many settings; certainly in sponsored search we expect user queries to vastly outnumber new advertisers. Another example is crowdsourcing systems, where the arriving actions are workers that can be assigned tasks, and $f(\mathbf{x})$ is the quality of work that worker f does on task $\mathbf{x}$. If

interpretation of this requirement, and our theorem that exploits it, is as a form of Occam's Razor: since new functions arriving means more parameters for the MAB algorithm to learn, it turns out to be necessary and sufficient that they arrive at a strictly slower rate than the data (trials).

We now precisely state the *arriving action* learning protocol:

```
1: Learner given an initial action pool $\mathcal{F}^0 \subset \mathcal{F}$
2: for $t = 1, 2, 3, \ldots$ do
3:     Learner receives new actions $S^t \subset \mathcal{F}$ and updates pool $\mathcal{F}^t \leftarrow \mathcal{F}^{t-1} \cup S^t$
4:     Nature selects $\mathbf{x}^t \in \mathcal{Z}$, presents to learner
5:     Learner selects some $f^t \in \mathcal{F}^t$, and receives payoff $f^t(\mathbf{x}^t) + \eta^t$; $\eta^t$ is i.i.d. with $\mathbb{E}[\eta^t] = 0$
6: end for
```

We now define our notion of regret for the arriving action protocol.

Definition 3. *Let A be an algorithm for making a sequence of decisions $f^1, f^2, \ldots$ according to the arriving action protocol. Then we say that A has* no regret *if on any sequence of pairs $(S^1, \mathbf{x}^1), (S^2, \mathbf{x}^2), \ldots, (S^t, \mathbf{x}^T)$, $R_A(T)/T \to 0$ as $T \to \infty$, where we re-define $R_A(T) \triangleq E\left[\sum_{t=1}^T \max_{f_*^t \in \mathcal{F}^t} f_*^t(\mathbf{x}^t) - \sum_{t=1}^T f^t(\mathbf{x}^t)\right]$.*

Reduction to KWIK Learning. Similar to Section 4, we now show how to use the KWIK learnability assumption on $\mathcal{F}$ to construct a no-regret algorithm in the arriving action model. The key idea, described in the reduction below, is to endow each action f in the current action pool with its own $\texttt{KWIK}_f$ subroutine. On every round, after observing the task $\mathbf{x}^t$, we shall query $\texttt{KWIK}_f$ for a prediction of $f(\mathbf{x}^t)$ for each $f \in W^t$. If *any* subroutine $\texttt{KWIK}_f$ returns $\perp$, we immediately stop and play action $f^t \leftarrow f$. This can be thought of as an *exploration step* of the algorithm. If every $\texttt{KWIK}_f$ returns a value, we simply choose the arg max as our selected action.

Theorem 3. *Let A denote Algorithm 2. For any $\epsilon > 0$ and any choice of $\{\mathbf{x}^t, S^t\}$,*

$$R_A(T) \leq N(T)\mathbf{B}(\epsilon, \delta) + 2T\epsilon + \delta N(T)T.$$

where $\mathbf{B}(\epsilon, \delta)$ is a bound on the number of $\perp$ returned by the KWIK-Subroutine used in A.

Proof. The probability that at least one of the $N(T)$ KWIK algorithms will FAIL is at most $\delta N(T)$. In that

Algorithm 2 No-Regret Learning in the Arriving Action Model

```
1: for $t = 1, 2, 3, \ldots$ do
2:     Learner receives new actions $S^t$
3:     Learner observes task $\mathbf{x}^t$
4:     for $f \in S^t$ do
5:         Initialize a subroutine $\texttt{KWIK}_f$ for learning $f$
6:     end for
7:     for $f \in \mathcal{F}^t$ do
8:         Query $\texttt{KWIK}_f$ for prediction $\hat{y}_f^t$
9:         if $\hat{y}_f^t = \perp$ then
10:            Take action $f^t = f$
11:            Observe $y^t \leftarrow f^t(\mathbf{x}^t)$
12:            Input $y^t$ into $\texttt{KWIK}_f$, and break
13:        end if
14:    end for
15:    // If no KWIK subroutine
16:    // returns ⊥, simply choose best!
17:    Take action $f^t = \arg\max_{f \in \mathcal{F}^t} \hat{y}_f^t$
18: end for
```

case, we suffer the maximum possible T regret, accounting for the $\delta N(T)T$ term. Otherwise, on each round t we query every $f \in \mathcal{F}^t$ for a prediction, and either one of two things can occur: (a) $\texttt{KWIK}_f$ reports $\perp$ in which case we can suffer regret at most 1; or (b) each $\texttt{KWIK}_f$ returns a real prediction $\hat{y}_f^t \neq \perp$ that is ϵ-accurate, in which case we are guaranteed that the regret of f^t is no more than 2ϵ. More precisely, we can bound the regret on round t as

$$\max_{f_*^t \in \mathcal{F}^t} f_*^t(\mathbf{x}^t) - f^t(\mathbf{x}^t) \tag{3}$$
$$\leq \mathbf{1}[\texttt{KWIK}_f \text{ outputs } \hat{y}_f^t = \perp \text{ for some } f] + 2\epsilon.$$

Of course, the total number of times that any $\texttt{KWIK}_f$ subroutine returns $\perp$ is no more than $\mathbf{B}(\epsilon, \delta)$, hence the total number of $\perp$'s after T rounds is no more than $N(T)\mathbf{B}(\epsilon, \delta)$. Summing (3) over $t = 1, \ldots, T$ gives the desired bound and we are done. □

As a consequence of the previous theorem, we achieve a simple corollary:

Corollary 3. *Assume that $\mathbf{B}(\epsilon, \delta) = O(\epsilon^{-d} \log^k \delta^{-1})$ for some $d > 0$, and $k \geq 0$. Then $\frac{R_A(T)}{T} = O\left(\left(\frac{N(T)}{T}\right)^{1/(d+1)} \log^k T\right)$. This tends to 0 as long as $N(T)$ is "slightly" sublinear in T; $T = o(T/\log^{k(d+1)}(T))$.*

Proof. Without loss of generality we can assume $\mathbf{B}(\epsilon, \delta) \leq \frac{c}{\epsilon}^{-d} \log \delta^{-1}$ for all ϵ, δ and some constant

the workers also contribute tasks (as in services like stackoverflow.com), and do so at some constant rate, it is easily verified that $N(t) = \sqrt{t}$.

$c > 0$. Applying Theorem 3 gives $\frac{R_A(T)}{T} \leq \frac{N(T)}{T}\frac{c}{\epsilon^d} + 2\epsilon + \delta N(T)$

Choosing $\delta = 1/T$ and $\epsilon = \left(\frac{N(T)}{T}\right)^{1/(d+1)}$ allows us to conclude that $R_A(T)/T \leq (c+2)\left(\frac{N(T)}{T}\right)^{1/(d+1)} \log^k T$ and hence we are done. □

Impossibility Results. The following two theorems show that our assumptions of the KWIK learnability of $\mathcal{F}$ and sublinearity of $N(t)$ are both necessary, in the sense that relaxing either is not sufficient to imply a no-regret algorithm for the arriving action MAB problem. Unlike the corresponding result of Theorem 2, those below do not rely on complexity-theoretic assumptions, but are information-theoretic. The full proof of Theorem 5 is provided in the Appendix.

Theorem 4. *(Relaxing sublinearity of $N(t)$ insufficient to imply no-regret on MAB) There exists a class $\mathcal{F}$ that is KWIK-learnable with a don't-know bound of 1 such that if $N(t) = t$, for any learning algorithm A and any T, there is a sequence of trials in the arriving action model such that $R_A(T)/T > c$ for some constant $c > 0$.*

Proof. Let $\mathcal{A} = \mathbb{N}$ and $e : \mathbb{N} \to \mathbb{R}^+$ be a fixed encoding function satisfying $e(n) \leq \gamma$ for any n, and let d be a corresponding decoding function satisfying $(d \circ e)(n) = n$.

Consider $\mathcal{F} = \{f_n \mid n \in \mathbb{N}\}$, where $f_n(n) = 1$ and $f_n(n') = e(n)$ for all other n'. The class $\mathbb{N}$ is KWIK-learnable with at most a single $\perp$ in the noise-free case. Observing $f_n(n')$ for an unknown f_n and arbitrary $n' \in \mathbb{N}$ immediately reveals the identity of f_n. Either $f_n(n') = 1$, in which case $n = n'$, or else $n = d(f_n(n'))$.

Let $\mathcal{A}$ and $\mathcal{F}$ be as just described. There exists an absolute constant $c > 0$ such that for any $T \geq 4$, there exists a sequence $\{n_t, S^t\}$ satisfying $N(T) = T$, and $R_A(T)/T > c$ for any A.

Let σ be a random permutation of $\{1, ..., T\}$, and S^1 be the ordered set $\{f_{\sigma(1)}, f_{\sigma(2)}, ..., f_{\sigma(T)}\}$. In other words, the actions $f_1, ..., f_T$ are shuffled, and immediately presented to the algorithm on the first round. $S^t = \emptyset$ for $t > 1$. Let n_t be drawn uniformly at random from $\{1, ..., T\}$ on each round t.

Immediately, we have that $E\left[\sum_{t=1}^{T} \max_{f \in \mathcal{F}^t} f(n_t)\right] = T$ since $F^t = \{1, ..., T\}$ for all t.

Now consider the actions $\{\hat{f}_t\}$ selected by an arbitrary algorithm A. Define $\hat{F}(\tau) = \{\hat{f}_t \in \mathcal{F} \mid t < \tau\}$, the actions that have been selected by A before time τ. Let $U(\tau) = \{n \in \mathbb{N} \mid f_n \in \hat{F}(\tau)\}$ be the states n, such the corresponding best action f_n has been used in the past, before round τ. Also let $\bar{\mathcal{F}}(\tau) = \{1, ..., T\} \setminus \hat{\mathcal{F}}(\tau)$.

Let R_τ be the reward earned by the algorithm at time τ. If $n_\tau \in U(\tau)$, then the algorithm has played action f_{n_τ} in the past, and knows its identity. Therefore, it may achieve $R_\tau = 1$. Since n_τ is drawn uniformly at random from $\{1, ..., T\}$, $P(n_\tau \in U(\tau) \mid U(\tau)) = \frac{|U(\tau)|}{T}$. Otherwise, in order to achieve $R_\tau = 1$, any algorithm must select $\hat{f}_\tau$ from amongst $\bar{F}(\tau)$. But since the actions are presented as a random permutation, and no action in $\bar{F}(\tau)$ has been selected on a previous round, any such assignment satisfies $P(\hat{f}_\tau = f_{n_\tau} \mid n_\tau \notin U(\tau)) = \frac{1}{T - |U(\tau)|}$.

Therefore for any algorithm we have:

$$\begin{aligned}
&P(R_\tau = 1 \mid U(\tau)) \\
&\leq P(n_\tau \in U(\tau) \mid U(\tau)) + P(n_\tau \notin U(\tau), \hat{f}_\tau = f_{n_\tau} \mid U(\tau)) \\
&\leq \frac{|U(\tau)|}{T} + \left(1 - \frac{|U(\tau)|}{T}\right)\left(\frac{1}{T - |U(\tau)|}\right)
\end{aligned}$$

Note that the right hand side of the last expression is a convex combination of 1 and $\left(\frac{1}{T - |U(\tau)|}\right) \leq 1$, and is therefore increasing as $\frac{|U(\tau)|}{T}$ increases. Since $|U(\tau)| < \tau$ with probability 1, we have:

$$P(R_\tau = 1) \leq \frac{\tau}{T} + \left(1 - \frac{\tau}{T}\right)\left(\frac{1}{T - \tau}\right) \qquad (4)$$

Let $Z(T) = \sum_{\tau=1}^{T} I(R_\tau = 1)$, count the number of rounds on which $R_\tau = 1$. This gives us:

$$\begin{aligned}
E[Z(T)] &= \sum_{\tau=1}^{T} P(R_\tau = 1) \\
&\leq \frac{T}{2} + \sum_{\tau=1}^{T/2} P(R_\tau = 1) \\
&\leq \frac{T}{2} + \frac{T}{2}\left[\frac{T}{2T} + \left(1 - \frac{T}{2T}\right)\left(\frac{1}{T - T/2}\right)\right]
\end{aligned}$$

Where the last inequality follows from the fact that equation 4 is increasing in τ.

Thus $E[Z(T)] \leq \frac{3T}{4} + \frac{1}{2}$. On rounds where $R_\tau \neq 1$, R_τ is at most γ, giving:

$$R_A(T)/T \geq 1 - \left[\frac{3}{4} + \frac{1}{2T} + \frac{\gamma}{4}\right]$$

Taking $T \geq 4$, gives us:

$R_A(T)/T \geq \frac{1}{8} - \frac{\gamma}{4}$. Since γ is arbitrary we have the desired result. □

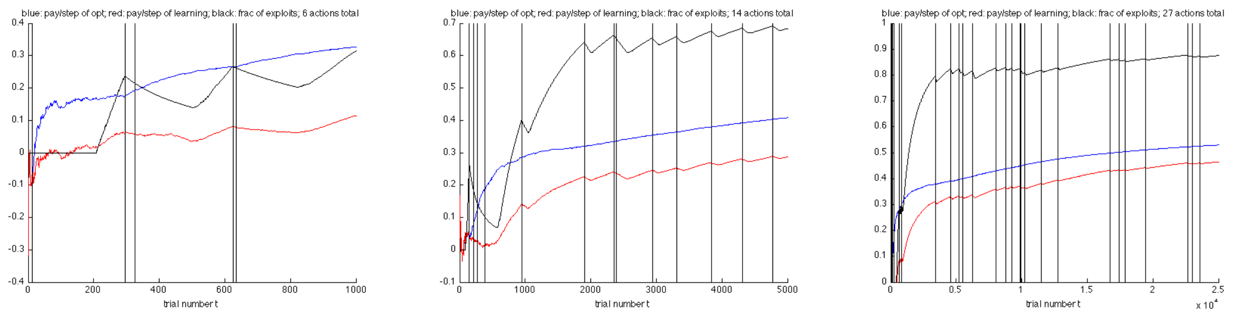

Figure 1. Simulations of Algorithm 2 at three timescales; see text for details.

Theorem 5. *(Relaxing KWIK to supervised no-regret insufficient to imply no-regret on MAB) There exists a class $\mathcal{F}$ that is supervised no-regret learnable such that if $N(t) = \sqrt{t}$, for any learning algorithm A and any T, there is a sequence of trials in the arriving action model such that $R_A(T)/T > c$ for some constant $c > 0$.*

6. Experiments

We now give a brief experimental illustration of our models and results. For the sake of brevity we examine only our algorithm in the arriving action model just discussed. We consider a setting in which both states $\mathbf{x}$ and the actions or functions $\mathbf{f}$ are described by unit-norm, 10-dimensional real vectors, and the value taking $\mathbf{f}$ in state $\mathbf{x}$ is simply the inner product $\mathbf{f}\cdot\mathbf{x}$. For this class of functions we thus implemented the KWIK linear regression algorithm (Walsh et al., 2009), which is given a fixed accuracy target or threshold of $\epsilon = 0.1$, and which is simulated with Gaussian noise added to payoffs with $\sigma = 0.1$. New actions/functions arrived stochastically, with the probability of a new $\mathbf{f}$ being added on trial t being $0.1/\sqrt{t}$; thus in expectation we have sublinear $N(t) = O(\sqrt{t})$. Both the $\mathbf{x}$ and the $\mathbf{f}$ are selected uniformly at random. On top of the KWIK subroutine, we implemented Algorithm 2.

In Figure 1 we show snapshots of simulations of this algorithm at three different timescales — after 1000, 5000, and 25,000 trials respectively. The snapshots are indeed from three independent simulations in order to illustrate the variety of behaviors induced by the exogenous stochastic arrivals of new actions/functions, but also to show typical performance for each timescale.

In each subplot, we plot three quantities. The blue curve show the average reward per step so far for the omniscient offline optimal that is *given* each weight $\mathbf{f}$ as it arrives, and thus always chooses the optimal available action on every trial. This curve is the best possible performance, and is the target of the learning algorithm. The red curve shows the average reward per step so far for Algorithm 2. The black curve shows the fraction of *exploitation steps* for the algorithm so far (the last line of Algorithm 2, where we are guaranteed to choose an approximately optimal action). The vertical lines indicate trials in which a new action/function was added.

First considering $T = 1000$ (left panel, in which a total of 6 actions are added), we see that very early (as soon as the second action arrives, and thus there is a choice over which the offline omniscient can optimize) the algorithm badly underperforms, and is never exploiting — new actions are arriving at rate at which the learning algorithm cannot keep up. At around 200 trials, the algorithm has learned all available actions well enough to start to exploit, and there is an attendant rise in performance; however, each time a new action arrives, both exploitation and performance drop temporarily as new learning must ensue.

At the $T = 5000$ timescale (middle panel, 14 actions added), exploitation rates are consistently higher (approaching 0.6 or 60% of the trials), and performance is beginning to converge to the optimal. New action arrivals still cause temporary dips, but overall upward progress is setting in.

At $T = 25,000$ (right panel, 27 actions added), the algorithm is exploiting over 80% of the time, and performance has converged to optimal up to the $\epsilon = 0.1$ accuracy set for the KWIK subroutine. If ϵ tends to 0 as T increases, as in the formal analysis, we eventually converge to 0 regret.

Acknowledgements

We give warm thanks to Sergiu Goschin, Michael Littman, Umar Syed and Jenn Wortman Vaughan for early discussions that led to many of the results presented here.

References

Amin, K., Kearns, M., and Syed, U. Graphical models for bandit problems. In *Proceedings of the 27th Annual Conference Uncertainty in Artificial Intelligence (UAI)*, 2011a.

Amin, K., Kearns, M., and Syed, U. Bandits, query learning, and the haystack dimension. In *Proceedings of the 24th Annual Conference on Learning Theory (COLT)*, 2011b.

Auer, Peter, Ortner, Ronald, and Szepesvári, Csaba. Improved rates for the stochastic continuum-armed bandit problem. In *In 20th Conference on Learning Theory (COLT)*, pp. 454–468, 2007.

Beygelzimer, Alina and Langford, John. The offset tree for learning with partial labels. In *KDD*, pp. 129–138, 2009.

Beygelzimer, Alina, Langford, John, Li, Lihong, Reyzin, Lev, and Schapire, Robert E. Contextual bandit algorithms with supervised learning guarantees. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

Bubeck, Sébastien, Munos, Rémi, Stoltz, Gilles, and Szepesvári, Csaba. Online optimization in x-armed bandits. In *NIPS*, pp. 201–208, 2008.

Kleinberg, Robert, Slivkins, Aleksandrs, and Upfal, Eli. Multi-armed bandits in metric spaces. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 681–690, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-047-0. doi: http://doi.acm.org/10.1145/1374376.1374475.

Langford, John and Zhang, Tong. The epoch-greedy algorithm for contextual multi-armed bandits. In *Advances in Neural Information Processing Systems 20 (NIPS)*, 2007.

Li, L. and Littman, M.L. Reducing reinforcement learning to KWIK online regression. *Annals of Mathematics and Artificial Intelligence*, 58(3):217–237, 2010.

Li, L., Littman, M.L., and Walsh, T.J. Knows what it knows: a framework for self-aware learning. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, pp. 568–575, 2008.

Li, L., Littman, M.L., Walsh, T.J., and Strehl, A.L. Knows what it knows: a framework for self-aware learning. *Machine Learning*, 82(3):399–443, 2011.

Li, Lihong, Chu, Wei, Langford, John, and Schapire, Robert E. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International World Wide Web Conference*, 2010.

Lu, Tyler, Pal, David, and Pal, Martin. Contextual multi-armed bandits. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

Sayedi, A., Zadimoghaddam, M., and Blum, A. Trading off mistakes and don't-know predictions. In *NIPS*, 2010.

Slivkins, Aleksandrs. Contextual bandits with similarity information. In *Proceedings of the 24th Annual Conference on Learning Theory (COLT)*, 2011.

Strehl, Alexander and Littman, Michael L. Online linear regression and its application to model-based reinforcement learning. In *Advances in Neural Information Processing Systems 20 (NIPS)*, 2007.

Walsh, T.J., Szita, I., Diuk, C., and Littman, M.L. Exploring compact reinforcement-learning representations with linear regression. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 591–598. AUAI Press, 2009.

Wang, Yizao, Audibert, Jean-Yves, and Munos, Rémi. Algorithms for infinitely many-armed bandits. In *Advances in Neural Information Processing Systems 21 (NIPS)*, pp. 1729–1736, 2008.

A. Appendix

A.1. Proof of Corollary 1

Restatement of Corollary 1:
Assume we have a family of functions $\mathcal{F}_\Theta$, a KWIK-learning algorithm `KWIK` for $\mathcal{F}_\Theta$, and a fixed-state optimization algorithm `FixedStateOpt`. Then there exists a no-regret algorithm for the MAB problem on $\mathcal{F}_\Theta$.

Proof. Let $A(\epsilon, \delta)$ denote Algorithm 1 when parameterized by ϵ and δ. We construct a no-regret algorithm A^* for the MAB problem on $\mathcal{F}_\Theta$ that operates over a series of epochs. On the start of epoch i, A^* simply runs a fresh instance of $A(\epsilon_i, \delta_i)$, and does so for τ_i rounds. We will describe how $\epsilon_i, \delta_i, \tau_i$ are chosen.

First let $e(T)$ denote the number of epochs that A^* starts after T rounds. Let γ_i be the average regret suffered on the ith epoch. In other words, if $\mathbf{x}^{i,t}$ ($\mathbf{a}^{i,t}$) is the tth state (action) in the ith epoch, then $\gamma_i = E\left[\frac{1}{\tau_i}\sum_{t=1}^{\tau_i} \max_{\mathbf{a}_*^{i,t}\in\mathcal{A}} f_\theta(\mathbf{x}^{i,t}, \mathbf{a}_*^{i,t}) - f_\theta(\mathbf{x}^{i,t}, \mathbf{a}^{i,t})\right]$. We therefore can express the average regret of A^* as:

$$R_{A^*}(T)/T = \frac{1}{T}\sum_{i=1}^{e(T)} \tau_i\gamma_i \tag{5}$$

From Theorem 1, we know there exists a T_i and choices for ϵ_i and δ_i so that $\gamma_i < 2^{-i}$ so long as $\tau_i \geq T_i$. Let $\tau_1 = T_1$, and $\tau_i = \max\{2\tau_{i-1}, T_i\}$. These choices for τ_i, ϵ_i and δ_i guarantee that $\tau_{i-1} \leq \tau_i/2$, and also $\gamma_i < 2^{-i}$. Applying these facts respectively to Equation 5 allows us to conclude that:

$$\begin{aligned} R_{A^*}(T)/T &\leq \frac{1}{T}\sum_{i=1}^{e(T)} 2^{-(e(T)-i)}\tau_{e(T)}\gamma_i \\ &< \frac{1}{T}\sum_{t=1}^{e(T)} 2^{-e(T)}\tau_{e(T)} \leq e(T)2^{-e(T)} \end{aligned}$$

Theorem 1 also implies that $e(T) \to \infty$ as $T \to \infty$, and so A^* is indeed a no regret algorithm. □

A.2. Proof of Corollary 2

Restatement of Corollary 2:
If the don't-know bound of `KWIK` is $\mathbf{B}(\epsilon,\delta) = O(\epsilon^{-d}\log^k \delta^{-1})$ for some $d > 0, k \geq 0$ then there are choices of ϵ, δ so that the average regret of Algorithm 1 is

$$O\left(\left(\frac{1}{T}\right)^{\frac{1}{d+1}} \log^k T\right)$$

Proof. Taking $\epsilon = \left(\frac{1}{T}\right)^{\frac{1}{d+1}}$ and $\delta = \frac{1}{T}$ in Equation 2 in the proof of Theorem 1 suffices to prove the corollary. □

A.3. Proof of Theorem 2

We proceed to give a the proof of Theorem 2 in complete rigor. We will first give a more precise construction of the class of models $\mathcal{F}_\theta$ satisfying the conditions of the theorem.

Restatement of Theorem 2:
There exists a class of models $\mathcal{F}_\theta$ such that

- *$\mathcal{F}_\Theta$ is fixed-state optimizable;*
- *There is an efficient algorithm A such that on an arbitrary sequence of T trials $\mathbf{z}^t$, A makes a prediction $\hat{y}^t$ of $y^t = f_\theta(\mathbf{z}^t)$ and then receives y^t as feedback, and the total regret $\sum_{t=1}^T |y^t - \hat{y}^t|$ is sublinear in T (thus we have only no-regret supervised learning instead of the stronger KWIK);*
- *Under standard cryptographic assumptions, there is no polynomial-time algorithm for the no-regret MAB problem for $\mathcal{F}_\Theta$.*

Let $\mathbb{Z}_n = \{0, ..., n-1\}$. Suppose that Θ parameterizes a family of cryptographic trapdoor functions H_Θ (which we will use to construct $\mathcal{F}_\theta$). Specifically, each θ consists of a "public" and "private" part so that $\theta = (\theta_{\text{pub}}, \theta_{\text{pri}})$, and $H_\Theta = \{h_\theta : \mathbb{Z}_n \to \mathbb{Z}_n\}$. The cryptographic guarantee ensured by H_Θ is summarized in the following definition.

Definition 4. *Let $d = \lceil \log|\mathbb{Z}_n| \rceil$. Any family of cryptographic trapdoor functions H_Θ must satisfy the following conditions:*

- *(Efficiently Computable) For any θ, knowing just θ_{pub} gives an efficient (polynomial in d) algorithm for computing $h_\theta(\mathbf{a})$ for any $\mathbf{a} \in \mathbb{Z}_n$.*
- *(Not Invertible) Let k be chosen uniformly at random from $\mathbb{Z}_n$. Let A be an efficient (randomized) algorithm that takes θ_{pub} and $h_\theta(k)$ as input (but not θ_{pri}), and outputs an $\mathbf{a} \in \mathbb{Z}_n$. There is no polynomial q such that $P(h_\theta(k) = h_\theta(\mathbf{a})) \geq 1/q(d)$.*

Depending on the family of trapdoor functions, the second condition usually holds under an assumption that some problem is intractable (e.g. prime factorization).

We are now ready to describe $(\mathcal{F}_\Theta, \mathcal{A}, \mathcal{X})$. Fix n, and let $\mathcal{X} = \mathbb{Z}_n$ and $\mathcal{A} = \mathbb{Z}_n \cup \{\mathbf{a}^*\}$. For any $h_\theta \in H_\theta$, let h_θ^{-1} denote the inverse function to h_θ. Since h_θ may be many-to-one, for any y in the image of h_θ, arbitrarily define $h_\theta^{-1}(y)$ to be any x such that $h_\theta(x) = y$.

We will define the behavior of each $f_\theta \in \mathcal{F}_\Theta$ in what follows. First we will define a family of functions G_Θ. The behavior of each g_θ will be essentially identical to that of f_θ, and for the purposes of understanding the construction, it is useful to think of them as being exactly identical.

The behavior of g_θ on states $\mathbf{x} \in \mathbb{Z}_n$ is defined as follows. Given $\mathbf{x}$, to get the maximum payoff of 1, an algorithm must invert h_θ. In other words, $g_\theta(\mathbf{x}, \mathbf{a}) = 1$ only if $h_\theta(\mathbf{a}) = \mathbf{x}$ (for $\mathbf{a} \in \mathbb{Z}_n$, and not equal to the "special" action $\mathbf{a}^*$). For any other $\mathbf{a} \in \mathbb{Z}_n$, $g_\theta(\mathbf{x}, \mathbf{a}) = 0$.

On action $\mathbf{a}^*$, $g_\theta(\mathbf{x}, \mathbf{a}^*)$ reveals the location of $h_\theta^{-1}(\mathbf{x})$. Specifically $g_\theta(\mathbf{x}, \mathbf{a}^*) = \frac{0.5}{1+h_\theta^{-1}(\mathbf{x})}$ if $\mathbf{x}$ has an inverse and $g_\theta(\mathbf{x}, \mathbf{a}^*) = 0$ if $\mathbf{x}$ is not in the image of h_θ.

It's useful to pause here, and consider the purpose of the construction. Assume that θ_{pub} is known. Then if $\mathbf{x}$ and $\mathbf{a}$ ($\mathbf{a} \in \mathbb{Z}_n$) are presented simultaneously in the supervised learning setting, it's easy to simply check if $h_\theta(\mathbf{x}) = \mathbf{a}$, making accurate predictions. In the fixed-state optimization setting, querying $\mathbf{a}^*$ presents the algorithm with all the information it needs to find a maximizing action. However, in the bandit setting, if a new $\mathbf{x}$ is being drawn uniformly at random and presented to the algorithm, the algorithm is doomed to try to invert h_θ.

Now we want the identity of θ_{pub} to be revealed on any input to the function f_θ, but want the behavior of f_θ to be essentially that of g_θ. In order to achieve this, let $\lfloor \cdot \rfloor_*$ be the function which truncates a number to $p = 2d + 2$ bits of precision. This is sufficient precision to distinguish between the two smallest non-zero numbers used in the construction of g_θ, $\frac{1}{2}\frac{1}{n}$ and $\frac{1}{2}\frac{1}{n-1}$. Also fix an encoding scheme that maps each θ_{pub} to a unique number $[\theta_{\text{pub}}]$. We do this in a manner such that $2^{-2p} \leq [\theta_{\text{pub}}] < 2^{-p-1}$.

We will define f_θ by letting $f_\theta(\mathbf{x}, \mathbf{a}) = \lfloor g_\theta(\mathbf{x}, \mathbf{a}) \rfloor_* + [\theta_{\text{pub}}]$. Intuitively, f_θ mimics the behavior of g_θ in its first p bits, then encodes the identity of θ_{pub} in its subsequent p bits. $[\theta_{\text{pub}}]$ is the smallest output of f_θ, and "acts as" zero.

The subsequent lemma establishes that the first two conditions of Theorem 2 are satisfies by F_Θ.

Lemma 1. *For any $f_\theta \in \mathcal{F}_\Theta$ and any fixed $\mathbf{x} \in \mathcal{X}$, $f(\mathbf{x}, \cdot)$ can be optimized from a constant number of queries, and poly(d) computation. Furthermore, there exists an efficient algorithm for the supervised no-regret problem on $\mathcal{F}_\Theta$ with err$(T) = O(\log T)$, requiring poly(d) computation per step.*

Proof. For any θ, the fixed-state optimization problem on $f_\theta(\mathbf{x}, \cdot)$ is solved by simply querying the special action $\mathbf{a}^*$. If $f_\theta(\mathbf{x}, \mathbf{a}^*) < 2^{-p-1}$, then $g_\theta(\mathbf{x}, \mathbf{a}^*) = 0$, and $\mathbf{x}$ is not in the image of h_θ. Therefore, $\mathbf{a}^*$ is a maximizing action, and we are done. Otherwise, $f_\theta(\mathbf{x}, \mathbf{a}^*)$ uniquely identifies the optimal action $h^{-1}(\mathbf{x})$, which we can subsequently query.

The supervised no-regret problem is similarly trivial. Consider the following algorithm. On the first state, it queries an arbitrary action, extracts its p lowest order bits, learning θ_{pub}. The algorithm can now compute the value of $f_\theta(\mathbf{x}, \mathbf{a})$ on any $(\mathbf{x}, \mathbf{a})$ pair where $\mathbf{a} \in \mathbb{Z}_n$. If $\mathbf{a} \in \mathbb{Z}_n$, the algorithm simply checks if $h_\theta(\mathbf{a}) = \mathbf{x}$. If so, it outputs $1 + [\theta_{\text{pub}}]$. Otherwise, it outputs $[\theta_{\text{pub}}]$.

The only inputs on which it might make a mistake take the form $(\mathbf{x}, \mathbf{a}^*)$. If the algorithm has seen the specific pair $(\mathbf{x}, \mathbf{a}^*)$, it can simply repeat the previously seen value of $f_\theta(\mathbf{x}, \mathbf{a}^*)$, resulting in zero error. Otherwise, if $(\mathbf{x}, \mathbf{a}^*)$ is a new input, the algorithm outputs $[\theta_{\text{pub}}]$, suffering $\lfloor \frac{0.5}{1+h^{-1}(\mathbf{x})} \rfloor_*$ error. Hence, after the first round, the algorithm cannot suffer error greater than $\sum_{t=1}^{T} \frac{0.5}{t} = O(\log T)$. $\square$

Finally, we argue that that an efficient no-regret algorithm for the large-scale bandit problem defined by $(\mathcal{F}_\Theta, \mathcal{A}, \mathcal{X})$ can be used as a black box to invert any $h_\theta \in H_\theta$.

Lemma 2. *Under standard cryptographic assumptions, there is no polynomial q and efficient algorithm `BANDIT` for the large-scale bandit problem on $\mathcal{F}_\Theta$ that guarantees $\sum_{t=1}^{T} \max_{\mathbf{a}_*^t} f_\theta(\mathbf{x}_t, \mathbf{a}_*^t) - f_\theta(\mathbf{x}_t, \mathbf{a}^t) < .5T$ with probability greater than $1/2$ when $T \leq q(d)$.*

Proof. Suppose that there were such a q, and algorithm `BANDIT`.

We can design an algorithm that takes θ_{pub} and $h_\theta(k^*)$ as input, for some unknown k^* chosen uniformly at random, and outputs an $\mathbf{a} \in \mathbb{Z}_n$ such that $P(h_\theta(k) = h_\theta(\mathbf{a})) \geq \frac{1}{2q(d)}$.

Consider simulating `BANDIT` for T rounds. On each round t, the state provided to `BANDIT` will be generated by selecting an action k_t from $\mathbb{Z}_n$ uniformly at random, and then providing BANDIT with the state $h_\theta(k_t)$. At which point, BANDIT will output an action and demand a reward. If the action selected by bandit is the special action $\mathbf{a}^*$, then its reward is simply $\lfloor 0.5/(1+k) \rfloor_* + [\theta_{\text{pub}}]$. If the action selected by bandit is $\mathbf{a}^t$ satisfying $h_\theta(\mathbf{a}^t) = h_\theta(k)$, its reward is $1 + [\theta_{\text{pub}}]$. Otherwise, it's reward is $[\theta_{\text{pub}}]$.

By hypothesis, with probability $1/2$, the actions $\mathbf{a}^t$ generated by BANDIT must satisfy $h(\mathbf{a}^t) = h_\theta(k_t)$ for at least one round $t \leq T$. Thus, if we choose a round τ uniformly at random from $\{1, ..., q(T)\}$, and give state $h_\theta(k^*)$ to BANDIT on that round, the action $\mathbf{a}^\tau$ returned by bandit will satisfy $P(h_\theta(\mathbf{a}^\tau) = h_\theta(k)) \geq \frac{1}{2q(d)}$. This inverts $h_\theta(k^*)$, and contradicts the assumption that h_θ belongs to a family of cryptographic trapdoor functions. □

A.4. Proof of Theorem 5

We now show that relaxing KWIK to supervised no-regret insufficient to imply no-regret on MAB.

Restatement of Theorem 5:
(Relaxing KWIK to supervised no-regret insufficient to imply no-regret on MAB) There exists a class $\mathcal{F}$ that is supervised no-regret learnable such that if $N(t) = \sqrt{t}$, for any learning algorithm A and any T, there is a sequence of trials in the arriving action model such that $R_A(T)/T > c$ for some constant $c > 0$.

Proof. First we describe the class $\mathcal{F}$. For any n-bit string x, let f_x be a function such that $f_x(x)$ is some large value, and for any $x' \neq x$, $f_x(x') = 0$. It's easy to see that $\mathcal{F}$ is not KWIK learnable with a polynomial number of don't-knows — we can keep feeding an algorithm different inputs $x' \neq x$, and as soon as the algorithm makes a prediction, we can re-select the target function to force a mistake. $\mathcal{F}$ is no-regret learnable, however: we just keep predicting 0. As soon as we make a mistake, we learn x, and we'll never err again, so our regret is at most $O(1/T)$.

Now in the arriving action model, suppose we initially start with r distinct functions/actions $f_i = f_{x_i} \in \mathcal{F}$, $i = 1, \ldots, r$. We will choose $N(T) = \sqrt{T}$, which is sublinear, and $r = \sqrt{T}$, and we can make T as large as we want. So we have a no-regret-learnable $\mathcal{F}$ and a sublinear arrival rate; now we argue that the arriving action MAB problem is hard.

Pick a random permutation of the f_i, and let i be the indices in that order for convenience. We start the task sequence with all x_1's. The MAB learner faces the problem of figuring out which of the unknown f_is has x_1 as its high-payoff input. Since the permutation was random, the expected number of assignments of x_1 to different f_i before this is learned is $r/2$. At that point, all the learner has learned is the identify of f_1 — the fact that it learned that other $f_i(x_1) = 0$ is subsumed by learning $f_1(x_1)$ is large, since the f_i are all distinct.

We then continue the sequence with x_2's until the MAB learner identifies f_2, which now takes $(r-1)/2$ assignments in expectation. Continuing in this vein, the expected number of assignments made before learning (say) half of the f_i is $\sum_{j=1}^{r/2}(r-j)/2 = \Omega(r^2) = \Omega(T)$. On this sequence of $\Omega(T)$ tasks, the MAB learner will have gotten non-zero payoff on only $r = \sqrt{T}$ rounds. The offline optimal, on the other hand, always knows the identity of the f_i and gets large payoff on every single task. So any learner's cumulative regret to offline grows linearly with T. □