Improved Strongly Adaptive Online Learning using Coin Betting

Kwang-Sung Jun
UW-Madison

Francesco Orabona
Stony Brook University

Stephen Wright
UW-Madison

Rebecca Willett
UW-Madison

Abstract

This paper describes a new parameter-free online learning algorithm for changing environments. In comparing against algorithms with the same time complexity as ours, we obtain a strongly adaptive regret bound that is a factor of at least $\sqrt{\log(T)}$ better, where T is the time horizon. Empirical results show that our algorithm outperforms state-of-the-art methods in learning with expert advice and metric learning scenarios.

1 Introduction

Machine learning has made heavy use of the i.i.d. assumption, but this assumption does not hold in many applications. For example, in online portfolio management, stock price trends can vary unexpectedly, and the ability to track changing trends and adapt to them are crucial in maximizing one's profit. Another example is seen in product reviews, where words describing product quality may change over time as products and customer's taste evolve. Keeping track of the changes in the metric describing the relationship between review text and rating is crucial for improving analysis and quality of recommendations.

We consider the problem of adapting to a changing environment in the online learning context. Let $\mathcal{D}$ be the decision space, $\mathcal{L}$ be loss functions that map $\mathcal{D}$ to $\mathbb{R}$, and T be the target time horizon. Let $\mathcal{A}$ be an online learning algorithm and $\mathcal{W} \subseteq \mathcal{D}$ be the set of comparator decisions. (Often, $\mathcal{W} = \mathcal{D}$.) We define the online learning problem in Figure 1. The usual goal of online learning is to find a strategy that compares favorably with the best fixed comparator in $\mathcal{W}$, in hindsight. Specifically, we seek a low value of the following (cumulative) static regret objective: $\text{Regret}_T^{\mathcal{A}} := \sum_{t=1}^T f_t(\mathbf{x}_t^{\mathcal{A}}) - \min_{\mathbf{w}\in\mathcal{W}} \sum_{t=1}^T f_t(\mathbf{w})$.

```
At each time $t = 1, 2, \ldots, T$,
  • The learner $\mathcal{A}$ picks a decision $\mathbf{x}_t^{\mathcal{A}} \in \mathcal{D}$.
  • The environment reveals a loss function $f_t \in \mathcal{L}$.
  • The learner $\mathcal{A}$ suffers loss $f_t(\mathbf{x}_t^{\mathcal{A}})$.
```

Figure 1: Online learning protocol

When the environment is changing, static regret is not a suitable measure, since it compares the learning strategy against a decision that is fixed. We need to make use of stronger notions of regret that allow for comparators to change over time. To define such notions, we introduce the notation $[T] := \{1, \ldots, T\}$ and $[A..B] = \{A, A+1, \ldots, B\}$. Daniely et al. [5] defined *strongly adaptive regret (SA-Regret)*, which requires an algorithm to have low (cumulative) regret over any contiguous time interval $I = [I_1..I_2] \subseteq [T]$.[1] Another notion, m-shift regret [10], measures instead the regret w.r.t. a comparator that changes at most m times in T time steps. Note that the SA-Regret is a stronger notion than the m-shift regret since the latter can be derived directly from the former [12, 5], as we show in our supplementary material. We define SA-Regret and m-shift regret precisely in Section 1.1.

Several generic online algorithms that adapt to a changing environment have been proposed recently. Rather than being designed for a specific learning problem, these are "meta algorithms" that take *any* online learning algorithm as a black-box and turn it into an adaptive one. We summarize the SA-Regret of existing meta algorithms in Table 2. In particular, the pioneering work of Hazan et al. [9] introduced the *adaptive regret*, that is a slightly weaker notion than the SA-Regret, and proposed two meta algorithms called FLH and AFLH. However, their SA-Regret depends on T rather than $|I|$. The SAOL approach of [5] improves the SA-Regret to $O\left(\sqrt{(I_2 - I_1)\log^2(I_2)}\right)$.

In this paper, we propose a new meta algorithm called *Coin Betting for Changing Environment* (CBCE) that

Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54. Copyright 2017 by the author(s).

[1] Strongly adaptive regret is similar to the notion of adaptive regret introduced by [9], but emphasizes the dependency on the interval length $|I|$.

Algorithm	m-shift regret	Time	Agnostic to m
Fixed Share [10, 3]	$\sqrt{mT(\log N + \log T)}$	NT	✗
	$\sqrt{m^2T(\log N + \log T)}$	NT	✓
GeneralTracking⟨EXP⟩ [8]	$\sqrt{mT(\log N + m\log^2 T)}$	$NT\log T$	✓
	$\sqrt{mT(\log N + \log^2 T)}$	$NT\log T$	✗
$(\gamma \in (0,1))$	$\sqrt{\frac{1}{\gamma}mT(\log N + m\log T)}$	$NT^{1+\gamma}\log T$	✓
	$\sqrt{\frac{1}{\gamma}mT(\log N + \log T)}$	$NT^{1+\gamma}\log T$	✗
ATV [12]	$\sqrt{mT(\log N + \log T)}$	NT^2	✓
SAOL⟨MW⟩ [5]	$\sqrt{mT(\log N + \log^2 T)}$	$NT\log T$	✓
CBCE⟨CB⟩ (ours)	$\sqrt{mT(\log N + \log T)}$	$NT\log T$	✓

Table 1: m-shift regret bounds of LEA algorithms. Our proposed algorithm achieves the best regret among those with the same time complexity and does not need to know m. Each quantity omits constant factors. Agnostic to m means that an algorithm does not need to know the number m of switches in the best expert.

Algorithm	SA-Regret order	Time factor
FLH [9]	$\sqrt{T}\log T$	T
AFLH [9]	$\sqrt{T\log T\log(I_2-I_1)}$	$\log T$
SAOL [5]	$\sqrt{(I_2-I_1)\log^2(I_2)}$	$\log T$
CBCE (ours)	$\sqrt{(I_2-I_1)\log(I_2)}$	$\log T$

Table 2: SA-Regret bounds of meta algorithms on $I \subseteq [T]$. Our proposed algorithm achieves the best SA-Regret. We show the part of the regret due to the meta algorithm only, not the black-box. The last column is the multiplicative factor in the time complexity introduced by the meta algorithm.

combines the sleeping bandits idea [2, 6] with the Coin Betting (CB) algorithm [13]. The SA-Regret of CBCE is better by a factor $\sqrt{\log(I_2)}$ than that of SAOL, as shown in Table 2. We present our extension of CB to sleeping bandits and prove its regret bound in Section 3. This result leads to the improved SA-Regret bound of CBCE in Section 4.

Our improved bound yields a number of improvements in various online learning problems. In describing these improvements, we denote by $\mathcal{M}\langle\mathcal{B}\rangle$ a complete algorithm assembled from meta algorithm $\mathcal{M}$ and black-box $\mathcal{B}$.

Consider the learning with expert advice (LEA) problem with N experts. CBCE with black-box CB (CBCE⟨CB⟩, in our notation) has the m-shift regret

$$O\sqrt{mT(\log N + \log T)}$$

and time complexity $O(NT\log T)$. This regret is a factor $\sqrt{\log T}$ better than those algorithms with the same time complexity. Although AdaNormalHedge.TV (ATV) and Fixed Share achieve the same regret, the former has larger time complexity, and the latter requires prior knowledge of the number of shifts m. We summarize the m-shift regret bounds of various algorithms in Table 1.

In Online Convex Optimization (OCO) with G-Lipschitz loss functions over a convex set $D \in \mathbb{R}^d$ of diameter B, online gradient descent has regret $O(BG\sqrt{T})$. CBCE with black-box OGD (CBCE⟨OGD⟩) then has the following SA-Regret:

$$O((BG + \sqrt{\log(I_2)})\sqrt{|I|})\ ,$$

which improves by a factor $\sqrt{\log(I_2)}$ over SAOL⟨OGD⟩.

In Section 5, we compare CBCE empirically to a number of meta algorithms within a changing environment in two online learning problems: (*i*) LEA and (*ii*) Mahalanobis metric learning. We observe that CBCE outperforms the state-of-the-art methods in both tasks, thus confirming our theoretical findings.

1.1 Preliminaries

In this section we define some concepts that will be used in the rest of the paper.

A learner's SA-Regret is obtained by evaluating static regret on all (contiguous) time intervals $I = [I_1..I_2] \subseteq [T]$ of a given length τ. Specifically, the SA-Regret of an algorithm $\mathcal{A}$ at time T for length τ is

$$\text{SA-Regret}_T^{\mathcal{A}}(\tau) := \max_{I\subseteq[T]:|I|=\tau}\left(\sum_{t\in I} f_t(\mathbf{x}_t^{\mathcal{A}}) - \min_{\mathbf{w}\in\mathcal{W}}\sum_{t\in I} f_t(\mathbf{w})\right)\ . \quad (1)$$

We call an algorithm *strongly adaptive* if it has a low value of SA-Regret. We call $\mathbf{w}_{1:T} := \{\mathbf{w}_1,\ldots,\mathbf{w}_T\}$ an *m-shift sequence* if it changes at most m times, that is, $\sum_{j=1}^{T-1}\mathbb{1}\{\mathbf{w}_j \neq \mathbf{w}_{j+1}\} \le m$. We define

$$m\text{-Shift-Regret}_T^{\mathcal{A}} := \sum_{t=1}^{T} f_t(\mathbf{x}_t^{\mathcal{A}}) - \min_{\mathbf{w}_{1:T}\in\mathcal{W}^T\,:\,m\text{-shift seq.}}\sum_{t=1}^{T} f_t(\mathbf{w}_t)\ .$$

2 A Meta Algorithm for Changing Environments

Let $\mathcal{B}$ be a black-box online learning algorithm following the protocol in Figure 1. A trick commonly used in designing a meta algorithm $\mathcal{M}$ for changing environments is to initiate a new instance of $\mathcal{B}$ at every time step [9, 8, 1]. That is, we run $\mathcal{B}$ independently for each interval J in $\{[t..\infty] \mid t = 1, 2, \ldots\}$. Denote by $\mathcal{B}_J$ the run of black-box $\mathcal{B}$ on interval J. A meta algorithm at time t combines the decisions from the runs $\{\mathcal{B}_J\}_{J \ni t}$ by weighted average. The key idea is that at time t, some of the outputs $\mathcal{B} \in \{\mathcal{B}_J\}_{J \ni t}$ are not based on any data prior to time $t' < t$, so that if the environment changes at time t', those outputs may be given a larger weight by the meta algorithm, allowing it to adapt more quickly to the change. This trick requires updating of t instances of the black-box algorithm at each time step t, leading to a factor-of-t increase in the time complexity. This factor can be reduced to $O(\log t)$ by restarting black-box algorithms on a carefully designed set of intervals such as the geometric covering intervals [5] (GC) or the data streaming technique [9, 8] (DS) that is a special case of a more general set of intervals considered in [14]. While both GC and DS achieve the same goal as we show in our supplementary material,[2] we use the former as our starting point for ease of exposition.

Geometric Covering Intervals. Define $\mathcal{J}_k := \{[(i \cdot 2^k) .. ((i+1) \cdot 2^k - 1)] : i \in \mathbb{N}\}$, $\forall k \in \{0, 1, \ldots\}$ to be the collection of intervals of length 2^k. The geometric covering intervals [5] are

$$\mathcal{J} := \bigcup_{k \in \{0,1,\ldots\}} \mathcal{J}_k \ .$$

That is, $\mathcal{J}$ is the set of intervals of doubling length, with intervals of size 2^k exactly partitioning the set $\mathbb{N} \backslash \{1, \ldots, 2^k - 1\}$, see Figure 2.

Define the set of intervals that includes time t as $\text{Active}(t) := \{J \in \mathcal{J} : t \in J\}$. One can easily show that $|\text{Active}(t)| = \lfloor \log_2(t) \rfloor + 1$. Since at most $O(\log(t))$ intervals contain any given time point t, the time complexity of the meta algorithm is a factor $O(\log(t))$ larger than that of the black-box $\mathcal{B}$.

The key result of the geometric covering intervals strategy is the following Lemma from [5], which shows that an arbitrary interval I can be partitioned into a sequence of smaller blocks whose lengths successively double, then successively halve.

Lemma 1. ([5, Lemma 5]) *Any interval $I \subseteq \mathbb{N}$ can be partitioned into two finite sequences of disjoint and consecutive intervals, denoted $\{J^{(-a)}, J^{(-a+1)}, \ldots, J^{(0)}\}$ and $\{J^{(1)}, J^{(2)}, \ldots, J^{(b)}\}$ where $\forall i \in [(-a)..b]$, we have $J^{(i)} \in \mathcal{J}$ and $J^{(i)} \subset I$, such that*

$$|J^{(-i)}|/|J^{(-i+1)}| \le 1/2, \quad i = 1, 2, \ldots, a;$$

[2]Except for a subtle case, which we also discuss in our supplementary material.

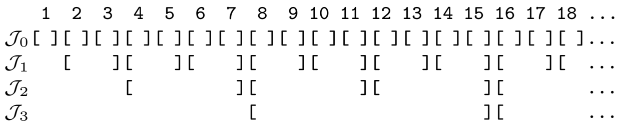


Figure 2: Geometric covering intervals. Each interval is denoted by `[ ]`.

$$|J^{(i+1)}|/|J^{(i)}| \le 1/2, \quad i = 1, 2, \ldots, b-1 \ .$$

Regret Decomposition. We show now how to use the geometric covering intervals to decompose the SA-Regret of a complete algorithm $\mathcal{M}\langle\mathcal{B}\rangle$. We use the notation

$$R_I^{\mathcal{A}}(\mathbf{w}) := \sum_{t \in I} f_t(\mathbf{x}_t^{\mathcal{A}}) - \sum_{t \in I} f_t(\mathbf{w}) \ ,$$

and from (1) we see that $\text{SA-Regret}_T^{\mathcal{A}}(\tau) = \max_{I \subseteq [T] : |I| = \tau} \max_{\mathbf{w} \in \mathcal{W}} R_I^{\mathcal{A}}(\mathbf{w})$.

Denote by $\mathbf{x}_t^{\mathcal{B}_J}$ the decision from black-box $\mathcal{B}_J$ at time t and by $\mathbf{x}_t^{\mathcal{M}\langle\mathcal{B}\rangle}$ the combined decision of the meta algorithm. Since $\mathcal{M}\langle\mathcal{B}\rangle$ is a combination of a meta $\mathcal{M}$ and a black-box $\mathcal{B}$, its regret depends on both $\mathcal{M}$ and $\mathcal{B}$. Perhaps surprisingly, we can decompose the two sources of regret additively through the geometric covering $\mathcal{J}$, as we now describe. Choose some $I \subseteq [T]$, and let $\bigcup_{i=-a}^{b} J^{(i)}$ be the partition of I obtained from Lemma 1. Then, the regret of $\mathcal{M}\langle\mathcal{B}\rangle$ on I can be decomposed as follows:

$$\begin{aligned}
R_I^{\mathcal{M}\langle\mathcal{B}\rangle}(\mathbf{w}) &= \sum_{t \in I} \left(f_t(\mathbf{x}_t^{\mathcal{M}\langle\mathcal{B}\rangle}) - f_t(\mathbf{w}) \right) \\
&= \sum_{i=-a}^{b} \left(\sum_{t \in J^{(i)}} f_t(\mathbf{x}_t^{\mathcal{M}\langle\mathcal{B}\rangle}) - f_t(\mathbf{x}_t^{\mathcal{B}_{J^{(i)}}}) + \right. \\
&\qquad \left. f_t(\mathbf{x}_t^{\mathcal{B}_{J^{(i)}}}) - f_t(\mathbf{w}) \right) \\
&= \underbrace{\sum_{i=-a}^{b} \underbrace{\sum_{t \in J^{(i)}} \left(f_t(\mathbf{x}_t^{\mathcal{M}\langle\mathcal{B}\rangle}) - f_t(\mathbf{x}_t^{\mathcal{B}_{J^{(i)}}}) \right)}_{=:(\text{meta regret on } J^{(i)})}}_{=:(\text{meta regret on } I)} + \\
&\qquad \sum_{i=-a}^{b} \underbrace{\sum_{t \in J^{(i)}} \left(f_t(\mathbf{x}_t^{\mathcal{B}_{J^{(i)}}}) - f_t(\mathbf{w}) \right)}_{=:(\text{black-box regret on } J^{(i)})} \ . \qquad (2)
\end{aligned}$$

(We purposely use symbol J for intervals in $\mathcal{J}$ and I for a generic interval that is not necessarily in $\mathcal{J}$.) The black-box regret on $J = [J_1..J_2] \in \mathcal{J}$ is exactly the standard regret for $T = |J|$, since the black-box run $\mathcal{B}_J$ was started from time J_1. Thus, in order to prove that a meta algorithm $\mathcal{M}$ suffers low SA-Regret, we must show two things.

1. $\mathcal{M}$ has low regret on interval $J \in \mathcal{J}$.
2. The outer sum over i in (2) is small for both the meta and the black-box.

Daniely et al. [5] address the second issue above efficiently in their analysis. They show that if the black-box regret on $J^{(i)}$ scales like $\tilde{O}(\sqrt{|J^{(i)}|})$ (where $\tilde{O}$ ignores logarithmic factors), then the second double summation of (2) is[3] $\tilde{O}(\sqrt{|I|})$, which is perhaps the best one can hope for. The same holds true for the meta algorithm. Thus, it remains to focus on the first issue above, which is our main contribution.

In the next two sections, we show how to design our meta algorithm. In Section 3 we propose a novel method that incorporates sleeping bandits and the coin betting framework. Section 4 describes how our method can be used as a meta algorithm with strongly adaptive regret guarantees.

3 Coin Betting Meets Sleeping Experts

Our meta algorithm is an extension of the coin-betting framework [13] based on sleeping experts [2, 6]. It is parameter-free (there is no explicit learning rate) and has near-optimal regret. Our construction, described below, might also be of independent interest.

Sleeping Experts. In the learning with expert advice (LEA) framework, the decision set is $\mathcal{D} = \Delta^N$, an N-dimensional probability simplex of weights assigned to the experts. To distinguish LEA from the general online learning problem, we use notation $\mathbf{p}_t$ in place of $\mathbf{x}_t$ and h_t in place of f_t. Let $\boldsymbol{\ell}_t := (\ell_{t,1}, \ldots, \ell_{t,N})^\top \in [0,1]^N$ be the vector of loss values of experts at time t that is provided by the environment. The learner's loss function is $h_t(\mathbf{p}) := \mathbf{p}^\top \boldsymbol{\ell}_t$.

Since $\mathbf{p} \in \mathcal{D}$ is a probability vector, the learner's decision can be viewed as hedging between the N alternatives. Let $\mathbf{e}_i$ be an indicator vector for dimension i; e.g., $\mathbf{e}_2 = (0,1,0,\ldots,0)^\top$. In this notation, the comparator set $\mathcal{W}$ is $\{\mathbf{e}_1, \ldots, \mathbf{e}_N\}$. Thus, the learner aims to compete with a strategy that commits to a single expert for the entire time $[1..T]$.

The decision set may be nonconvex, for example, when $\mathcal{D} = \{\mathbf{e}_1, \ldots, \mathbf{e}_N\}$ [4, Section 3]. In this case, no hedging is allowed; the learner must pick an expert. To choose an element of this set, one could first choose an element p_t from Δ^N, then make a decision $\mathbf{e}_i \in \mathcal{D}$ with probability $p_{t,i}$. For such a scheme, the regret guarantee is the same as in the standard LEA, but with *expected* regret.

[3] This is essentially the same argument as the "doubling trick" described in [4, Section 2.3]

Recall that each black-box run $\mathcal{B}_J$ is on a different interval J. The meta algorithm's role is to hedge bets over multiple black-box runs. Thus, it is natural to treat each run $\mathcal{B}_J$ as an *expert* and use an existing LEA algorithm to combine decisions from each expert $\mathcal{B}_J$. The loss incurred on run $\mathcal{B}_J$ is $\ell_{t,\mathcal{B}_J} := f_t(\mathbf{x}_t^{\mathcal{B}_J})$.

The challenge is that each expert $\mathcal{B}_J$ may not output decisions at time steps outside the interval J. This problem can be reduced to the *sleeping experts* problem studied in [2, 6], where experts are not required to provide decisions at every time step; see [12] for detail. We introduce a binary indicator variable $\mathcal{I}_{t,i} \in \{0,1\}$, which is set to 1 if expert i is awake (that is, outputting a decision) at time t, and zero otherwise. Define $\boldsymbol{\mathcal{I}}_t := [\mathcal{I}_{t,1}, \mathcal{I}_{t,2}, \ldots, \mathcal{I}_{t,N}]^\top$ where N can be countably infinite. Note that the algorithm is aware of $\boldsymbol{\mathcal{I}}_t$ and must assign zero weight to the experts that are sleeping: $\mathcal{I}_{t,i} = 0 \implies p_{t,i} = 0$. We would like to have a guarantee on the regret w.r.t. expert i, but only for the time steps where expert i is awake. Following [12], we aim to have a regret bound w.r.t. $\mathbf{u} \in \Delta^N$ as follows:

$$\text{Regret}_T(\mathbf{u}) := \sum_{t=1}^{T} \sum_{i=1}^{N} \mathcal{I}_{t,i} u_i (\langle \boldsymbol{\ell}_t, \mathbf{p}_t \rangle - \ell_{t,i}) \,. \qquad (3)$$

If we set $\mathbf{u} = \mathbf{e}_j$ for some j, the above is simply regret w.r.t. expert j while that expert is awake. Furthermore, if $\mathcal{I}_{t,j} = 1$ for all $t \in [T]$, then it recovers the standard static regret in LEA.

Coin Betting for LEA. We consider the coin betting framework of Orabona and Pál [13], where one can construct an LEA algorithm based on the so-called *coin betting potential* function F_t. A player starts from the initial endowment 1. At each time step, the adversary tosses a coin arbitrarily, with the player deciding upon which side to bet (heads or tails). Then the outcome is revealed. The adversary can manipulate the weight of the coin in $[0,1]$ as well, in a manner not known to the player before betting.

We encode a coin flip at iteration t as $\widetilde{g}_t \in [-1,1]$ where positive (negative) means heads (tails), and $|\widetilde{g}_t|$ indicates the weight. Let Wealth_{t-1} be the total money the player possesses after time step $t-1$. The player decides which side and how much money to bet. We encode the player's decision as the signed betting fraction $\beta_t \in (-1,1)$, where the positive (negative) sign indicates head (tail) and the absolute value $|\beta_t| \in [0,1)$ indicates the fraction of his money to bet. Thus, the actual amount of betting is $w_t := \beta_t \text{Wealth}_{t-1}$. Once the weighted coin flip $\widetilde{g}_t$ is revealed, the player's wealth changes: $\text{Wealth}_t = \text{Wealth}_{t-1} + \widetilde{g}_t \beta_t \text{Wealth}_{t-1}$. The player makes (loses) money when the betted side is correct (wrong), and the amount of wealth change depends on both the flip weight $|\widetilde{g}_t|$ and his betting amount $|\beta_t|$.

In the coin betting framework, the betting fraction β_t is determined by a potential function F_t, and we can simplify w_t as follows:

$$\begin{aligned} z_t &:= \sum_{\tau=1}^{t-1} \widetilde{g}_\tau \\ \beta_t(z_t) &:= \frac{F_t(z_t+1) - F_t(z_t-1)}{F_t(z_t+1) + F_t(z_t-1)} \qquad (4) \\ w_t &= \beta_t(z_t) \cdot \left(1 + \sum_{\tau=1}^{t-1} \widetilde{g}_\tau w_\tau\right) . \end{aligned}$$

We use β_t in place of $\beta_t(\sum_{\tau=1}^{t-1} \widetilde{g}_\tau)$ when it is clear from the context. A sequence of coin betting potentials $F_1, F_2, \ldots$ satisfies the following key condition (the complete list of conditions can be found in [13]): F_t must lower-bound the wealth of a player who bets by (4):

$$\forall t,\ F_t\left(\sum_{\tau=1}^{t} \widetilde{g}_\tau\right) \le 1 + \sum_{\tau=1}^{t} \widetilde{g}_\tau w_\tau . \qquad (5)$$

This bound becomes useful in regret analysis. We emphasize that the term w_t is decided before $\widetilde{g}_t$ is revealed, yet the inequality (5) holds for any $\widetilde{g}_t \in [-1, 1]$.

Orabona and Pál [13] have devised a reduction of LEA to the simple coin betting problem described above. The idea is to instantiate a coin betting problem for each expert i where the signed coin flip $\widetilde{g}_{t,i}$ is set as a conditionally truncated regret w.r.t. expert i, rather than being set by an adversary. We denote by $\beta_{t,i}$ the betting fraction for expert i and by $w_{t,i}$ the amount of betting for expert i, $\forall i \in [N]$.

We apply the same treatment under the sleeping experts setting and propose a new algorithm **Sleeping CB**. Since some experts may not output a decision at time t, Sleeping CB requires a different definition of β_t. We define $S_{t,i} := 1 + \sum_{\tau=1}^{t-1} \mathcal{I}_{\tau,i}$ and define the following modifications of (4)

$$\begin{aligned} z_{t,i} &:= \sum_{\tau=1}^{t-1} \mathcal{I}_{t,i} \widetilde{g}_{\tau,i} \\ \beta_{t,i}(z_{t,i}) &:= \frac{F_{S_{t,i}}(z_{t,i}+1) - F_{S_{t,i}}(z_{t,i}-1)}{F_{S_{t,i}}(z_{t,i}+1) + F_{S_{t,i}}(z_{t,i}-1)} . \end{aligned}$$

Further, we denote by $\boldsymbol{\pi}_{\mathcal{I}_t}$ the prior $\boldsymbol{\pi}$ restricted to experts that are awake ($\mathcal{I}_{t,i} = 1$), and define $[x]_+ := \max\{x, 0\}$. Algorithm 1 specifies the Sleeping CB algorithm.

The regret of Sleeping CB is bounded in Theorem 1. (All proofs appear as supplementary material.) Unlike the standard CB, in which all the experts use F_t at time t, expert i in Sleeping CB uses $F_{S_{t,i}}$, which is different for each expert. For this reason, the proof of the CB regret in [13] does not transfer easily to the regret (3) of Sleeping CB, and a solution to it is the cornerstone of an improved strongly adaptive regret bound.

Algorithm 1 Sleeping CB

```
Input: Number of experts $N$, prior distribution $\boldsymbol{\pi} \in \Delta^N$
for $t = 1, 2, \ldots$ do
  For each $i \in \text{Active}(t)$, set
    $w_{t,i} \leftarrow \beta_{t,i}(z_{t,i}) \cdot (1 + \sum_{\tau=1}^{t-1} z_{\tau,i} w_{\tau,i})$.
  For each $i \in \text{Active}(t)$, set $\widehat{p}_{t,i} \leftarrow \pi_i \mathcal{I}_{t,i} [w_{t,i}]_+$.
  Predict with $\mathbf{p}_t \leftarrow \begin{cases} \widehat{\mathbf{p}}_t / ||\widehat{\mathbf{p}}_t||_1 & \text{if } ||\widehat{\mathbf{p}}_t||_1 > 0 \\ \boldsymbol{\pi}_{\mathcal{I}_t} & \text{if } ||\widehat{\mathbf{p}}_t||_1 = 0. \end{cases}$
  Receive loss vector $\boldsymbol{\ell}_t \in [0,1]^N$.
  The learner suffers loss $h_t(\mathbf{p}_t) = \langle \boldsymbol{\ell}_t, \mathbf{p}_t \rangle_{\mathcal{I}_t}$.
  For each $i \in \text{Active}(t)$, set
    $\widetilde{g}_{t,i} \leftarrow \begin{cases} h_t(\mathbf{p}_t) - \ell_{t,i} & \text{if } w_{t,i} > 0 \\ [h_t(\mathbf{p}_t) - \ell_{t,i}]_+ & \text{if } w_{t,i} \le 0. \end{cases}$
end for
```

Theorem 1. (Regret of Sleeping CB) *Let $\{F_t\}_{t \ge 1}$ be a sequence of potential functions that satisfies (5). Assume that F_t is even (symmetric around zero) $\forall t \ge 1$. Suppose $\log F_{S_{T,i}}(x) \ge h_{S_{T,i}}(x) := c_1 \frac{x^2}{S_{T,i}} + c_{2,i}$ for some $c_1 > 0$ and $c_{2,i} \in \mathbb{R}$ for all $i \in [N]$. Then, Algorithm 1 satisfies*

$$\text{Regret}_T(\mathbf{u}) \le \sqrt{c_1^{-1} \cdot \left(\sum_{i=1}^{N} u_i S_{T,i}\right) \cdot \left(\text{KL}(\mathbf{u}||\boldsymbol{\pi}) - \sum_{i=1}^{N} u_i c_{2,i}\right)} .$$

Note that if $\mathbf{u} = \mathbf{e}_j$, then the regret scales with $S_{T,j}$, which is essentially the number of time steps at which expert j is awake.

Among multiple choices for the potential, we use the Krichevsky-Trofimov (KT) potential [13] that satisfies (5) (see [13] for the proof):

$$F_t(x) = \frac{2^t \cdot \Gamma(\delta+1) \cdot \Gamma(\frac{t+\delta+1}{2} + \frac{x}{2}) \Gamma(\frac{t+\delta+1}{2} - \frac{x}{2})}{\Gamma(\frac{\delta+1}{2})^2 \cdot \Gamma(t+\delta+1)},$$

where $\delta \ge 0$ is a time shift parameter that we set to 0 in this work. One can show that the betting fraction β_t defined in (4) for KT potential exhibits a simpler form: $\beta_t = \frac{\sum_{\tau=1}^{t-1} \widetilde{g}_\tau}{t+\delta}$ [13] and, for Sleeping CB, $\beta_t = \frac{\sum_{\tau=1}^{t-1} \mathcal{I}_{\tau,i} \widetilde{g}_{\tau,i}}{S_{t,i}+\delta}$. We present the regret of Algorithm 1 with the KT potential in Corollary 1.

Corollary 1. *Let $\delta = 0$. The regret of Algorithm 1 with the KT potential is*

$$\text{Regret}_T(\mathbf{u})$$

Algorithm 2 Coin Betting for Changing Environment (CBCE)

```
**Input**: A black-box algorithm $\mathcal{B}$ and a prior distribution $\boldsymbol{\pi} \in \Delta^{|\mathcal{J}|}$ over $\{\mathcal{B}_J \mid J \in \mathcal{J}\}$.
**for** $t = 1$ **to** $T$ **do**
    For each $J \in \text{Active}(t)$, set
        $w_{t,\mathcal{B}_J} \leftarrow \beta_{t,\mathcal{B}_J}(z_{t,\mathcal{B}_J}) \cdot (1 + \sum_{\tau=1}^{t-1} z_{\tau,\mathcal{B}_J} w_{\tau,\mathcal{B}_J})$
    Set $\widehat{p}_{t,\mathcal{B}_J} \leftarrow \pi_{\mathcal{B}_J} \mathcal{I}_{t,\mathcal{B}_J}[w_{t,\mathcal{B}_J}]_+$ for $J \in \text{Active}(t)$ and 0 for $J \notin \text{Active}(t)$.
    Compute $\mathbf{p}_t \leftarrow \begin{cases} \widehat{\mathbf{p}}_t/||\widehat{\mathbf{p}}_t||_1 & \text{if } ||\widehat{\mathbf{p}}_t||_1 > 0 \\ [\pi_{\mathcal{B}_J}]_{J \in \text{Active}(t)} & \text{if } ||\widehat{\mathbf{p}}_t||_1 = 0. \end{cases}$
    The black-box run $\mathcal{B}_J$ picks a decision $\mathbf{x}_t^{\mathcal{B}_J} \in \mathcal{D}$, $\forall J \in \text{Active}(t)$.
    The learner picks a decision $\mathbf{x}_t = \sum_{J \in \mathcal{J}} p_{t,\mathcal{B}_J} \mathbf{x}_t^{\mathcal{B}_J}$.
    Each black-box run $\mathcal{B}_J$ that is awake ($J \in \text{Active}(t)$) suffers loss $\ell_{t,\mathcal{B}_J} := f_t(\mathbf{x}_t^{\mathcal{B}_J})$.
    The learner suffers loss $f_t(\mathbf{x}_t)$.
    For each $J \in \text{Active}(t)$, set
        $\widetilde{g}_{t,\mathcal{B}_J} \leftarrow \begin{cases} f_t(\mathbf{x}_t) - \ell_{t,\mathcal{B}_J} & \text{if } w_{t,\mathcal{B}_J} > 0 \\ [f_t(\mathbf{x}_t) - \ell_{t,\mathcal{B}_J}]_+ & \text{if } w_{t,\mathcal{B}_J} \le 0. \end{cases}$
**end for**
```

$$\le \sqrt{2\left(\sum_{i=1}^N u_i S_{T,i}\right) \cdot \left(\text{KL}(\mathbf{u}||\boldsymbol{\pi}) + \frac{1}{2}\ln(T) + 2\right)} .$$

4 Coping with a Changing Environment by Sleeping CB

In this section, we synthesize the results in Sections 2 and 3 to specify and analyze our algorithm. Recall that a meta algorithm must efficiently aggregate decisions from multiple black-box runs that are active at time t. We treat each black-box run as an expert. Since we run a black-box instance for each interval $J \in \mathcal{J}$, there are a countably infinite number of experts. Thus, one can use Sleeping CB (Algorithm 1) as the meta algorithm, with geometric covering intervals. The complete algorithm, which we call **Coin Betting for Changing Environment (CBCE)**, is shown in Algorithm 2.

We make use of the following assumption.

Assumption A1. *The loss function f_t is convex and maps to $[0,1]$, $\forall t \in \mathbb{N}$.*

Nonconvex loss functions can be accommodated by randomized decisions: We choose the decision $\mathbf{x}_t^{\mathcal{B}_J}$ from black-box $\mathcal{B}_J$ with probability $p_{t,\mathcal{B}_J}$. It is not difficult to show that the same regret bound holds, but now in expectation. When loss functions are unbounded, they can be scaled and restricted to $[0,1]$. Although this leads to possible nonconvexity, we can still obtain an expected regret bound from the randomized decision process just described.

We define our choice of prior $\bar{\boldsymbol{\pi}} \in \Delta^{|\mathcal{J}|}$ as follows:

$$\bar{\pi}_{\mathcal{B}_J} := Z^{-1}\left(\frac{1}{J_1^2(1 + \lfloor \log_2 J_1 \rfloor)}\right), \ \forall J \in \mathcal{J} \ , \quad (6)$$

where Z is a normalization factor. Note that $Z < \pi^2/6$ since there exist at most $1 + \lfloor \log_2 J_1 \rfloor$ distinct intervals starting at time J_1, so Z is less than $\sum_{t=1}^\infty t^{-2} = \pi^2/6$.

We bound the meta regret w.r.t. a black-box run $\mathcal{B}_J$ as follows.

Lemma 2. (Meta regret of CBCE) *Assume A1. Suppose we run CBCE (Algorithm 2) with a black-box algorithm $\mathcal{B}$, prior $\bar{\boldsymbol{\pi}}$, and $\delta = 0$. The meta regret of CBCE$\langle\mathcal{B}\rangle$ on interval $J = [J_1..J_2] \in \mathcal{J}$ is*

$$\sum_{t \in J} f_t(\mathbf{x}_t^{\text{CBCE}\langle\mathcal{B}\rangle}) - f_t(\mathbf{x}_t^{\mathcal{B}_J})$$
$$\le \sqrt{|J|\,(7\ln(J_2) + 5)} = O(\sqrt{|J|\log J_2}) \ .$$

We now present the bound on the SA-Regret $R_I^{\text{CBCE}\langle\mathcal{B}\rangle}(\mathbf{w})$ w.r.t. $\mathbf{w} \in \mathcal{W}$ on intervals $I \subseteq [T]$ that are not necessarily in $\mathcal{J}$.

Theorem 2. (SA-Regret of CBCE$\langle\mathcal{B}\rangle$) *Assume A1 and that the black-box algorithm $\mathcal{B}$ has regret $R_T^{\mathcal{B}}$ bounded by cT^α, where $\alpha \in (0,1)$. Let $I = [I_1..I_2]$. The SA-Regret of CBCE with black-box $\mathcal{B}$ on the interval I w.r.t. any $\mathbf{w} \in \mathcal{W}$ is bounded as follows:*

$$R_I^{\text{CBCE}\langle\mathcal{B}\rangle}(\mathbf{w}) \le \frac{4}{2^\alpha - 1}c|I|^\alpha + 8\sqrt{|I|(7\ln(I_2) + 5)}$$
$$= O(c|I|^\alpha + \sqrt{|I|\ln I_2}) \ .$$

For the standard LEA problem, one can run the algorithm CB with KT potential (equivalent to Sleeping CB with $\mathcal{I}_{t,i} = 1, \forall t, i$), which achieves static regret $O(\sqrt{T\log(NT)})$ [13]. Using CB as the black-box algorithm, the regret of CBCE$\langle\mathcal{B}\rangle$ on I is $R_I^{\text{CBCE}\langle\text{CB}\rangle}(\mathbf{w}) = O(\sqrt{|I|\log(NI_2)})$, and so SA-Regret$_T^{\text{CBCE}\langle\text{CB}\rangle}(|I|) = O(\sqrt{|I|\log(NT)})$. It follows that the m-shift regret of CBCE$\langle$CB$\rangle$ is $O(\sqrt{mT\log(NT)})$ using the technique presented our supplementary material.

As said above, our bound improves over the best known result with the same time complexity in [5]. The key ingredient that allows us to get a better bound is the Sleeping CB Algorithm 1, that achieves a better SA-Regret than the one of [5]. In the next section, we will show that the empirical results also confirm the theoretical gap of these two algorithms.

Discussion. Note that one can obtain the same result using the data streaming intervals (DS) [9, 8] in place of the geometric covering intervals (GC). Section F of our supplementary material elaborates on this with a Lemma stating that DS induces a partition of an interval I in a very similar way to GC (a sequence of intervals of doubling lengths).

Our improved bound has another interesting implication. In designing strongly adaptive algorithms for LEA, there is a well known technique called "restarts" or "sleeping experts" that has time complexity $O(NT^2)$ [9, 12], and several studies used DS or GC to reduce the time complexity to $O(NT \log T)$ [9, 8, 5]. However, it was unclear whether it is possible to achieve both an m-shift regret of $O(\sqrt{mT(\log N + \log T)})$ and a time complexity of $O(NT \log T)$ without knowing m. Indeed, every study on m-shift regret with time $O(NT \log T)$ results in suboptimal m-shift regret bounds [5, 8, 9], to our knowledge. Furthermore, some studies (e.g., [12, Section 5]) speculated that perhaps applying the data streaming technique would increase its SA-Regret by a logarithmic factor. Our analysis implies that one can reduce the overall time complexity to $O(NT \log T)$ without sacrificing the order of SA-Regret and m-shift regret.

5 Experiments

We now turn to an empirical evaluation of algorithms for changing environments. We compare the performance of the meta algorithms under two online learning problems: (*i*) learning with expert advice (LEA) and (*ii*) metric learning (ML). We compare CBCE with SAOL [5] and AdaNormalHedge.TV (ATV) [12]. Although ATV was originally designed for LEA only, it is not hard to extend it to a meta algorithm and show that it has the same order of SA-Regret as CBCE using the same techniques.

For our empirical study, we replace the geometric covering intervals (GC) with the data streaming intervals (DS) [9, 8]. Let $u(t)$ be a number such that $2^{u(t)}$ is the largest power of 2 that divides t; e.g., $u(12) = 2$. The data streaming intervals are $\mathcal{J} = \{[t..(t+g\cdot 2^{u(t)}-1)] : t = 1, 2, \ldots\}$ for some $g \geq 1$. DS is an attractive alternative, unlike GC, (*i*) DS initiates one and only one black-box run at each time, and (*ii*) it is more flexible in that the parameter g can be increased to enjoy smaller regret in practice while increasing the time complexity by a constant factor.

For both ATV and CBCE, we set the prior $\boldsymbol{\pi}$ over the black-box runs as the uniform distribution. Note that this does not break the theoretical guarantees since the number of black-box runs are never actually infinite; we used $\bar{\boldsymbol{\pi}}$ (6) in Section 4 for ease of exposition.

5.1 Learning with Expert Advice (LEA)

We consider LEA with linear loss. That is, the loss function at time t is $h_t(\mathbf{p}) = \boldsymbol{\ell}_t^\top \mathbf{p}$. We draw linear loss $\boldsymbol{\ell}_t \in [0,1]^N, \forall t = 1, \ldots, 600$ for $N = 1000$ experts from Uniform$(0,1)$ distribution. Then, for time $t \in [1, 200]$, we reduce loss of expert 1 by subtracting 1/2 from its loss: $\ell_{t,1} \leftarrow [\ell_{t,1} - 1/2]_+$. For time $t \in [201, 400]$ and $t \in [401, 600]$, we perform the same for expert 2 and 3, respectively. Thus, the best expert is 1, 2, and 3 for time segment [1,200], [201,400], and [401,600], respectively. We use the data streaming intervals with $g = 2$. In all our experiments, DS with $g = 2$ outperforms GC while spending roughly the same time.

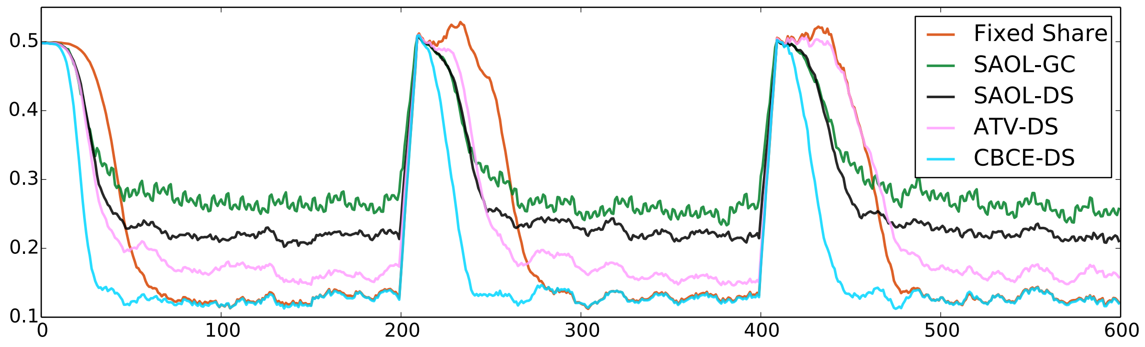
For each meta algorithm, we use the CB with KT potential [13] as the black-box algorithm. We warm-start each black-box run at time $t \geq 2$ by setting its prior to the decision $\mathbf{p}_{t-1}$ chosen by the meta algorithm at time step $t-1$. We repeat the experiment 50 times and plot their average loss by computing moving mean with window size 10 in Figure 3(a). Overall, we observe that CBCE (i) catches up with the environmental shift faster than any other algorithms and (ii) has the lowest loss when the shift has settled down. ATV is the second best, outperforming SAOL. Note that SAOL with GC (SAOL-GC) tends to incur larger loss than the SAOL with DS. We observe that this is true for every meta algorithm, so we omit the result here to avoid clutter. We also run Fixed Share using the parameters recommended by Corollary 5.1 of [4], which requires to know the target time horizon $T = 600$ and the true number of switches $m = 2$. Such a strong assumption is often unrealistic in practice. We observe that Fixed Share is the slowest in adapting to the environmental changes. Nevertheless, Fixed Share remains attractive since (i) after the switch has settled down its loss is competitive to CBCE, and (ii) its time complexity is lower than other algorithms ($O(NT)$ rather than $O(NT \log T)$).

5.2 Metric Learning

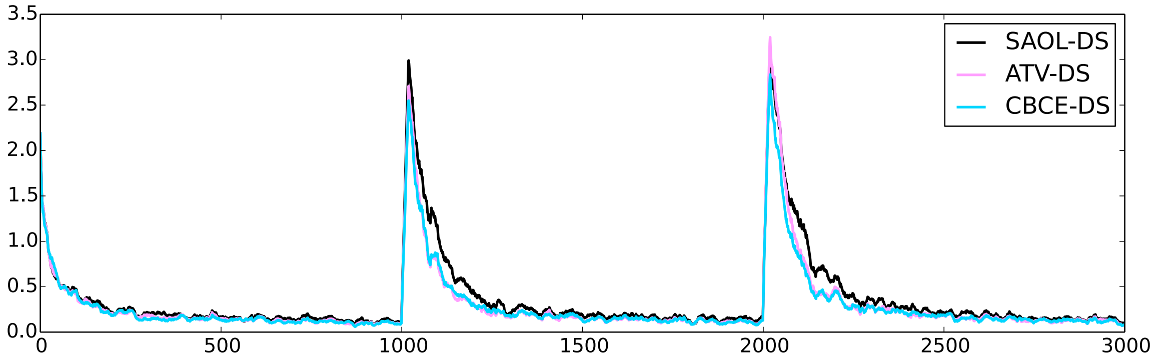
We consider the problem of learning squared Mahalanobis distance from pairwise comparisons using the mirror descent algorithm [11]. The data point at time t is $(\mathbf{z}_t^{(1)}, \mathbf{z}_t^{(2)}, y_t)$, where $y_t \in \{1, -1\}$ indicates whether or not $\mathbf{z}_t^{(1)} \in \mathbb{R}^d$ and $\mathbf{z}_t^{(2)} \in \mathbb{R}^d$ belongs to the same class. The goal is to learn a squared Mahalanobis distance parameterized by a positive semi-definite matrix $\mathbf{M}$ and a bias μ that have small loss $f_t([\mathbf{M}; \mu]) :=$

$$[1 - y_t(\mu - (\mathbf{z}_t^{(1)} - \mathbf{z}_t^{(2)})^\top \mathbf{M}(\mathbf{z}_t^{(1)} - \mathbf{z}_t^{(2)}))]_+ + \rho||\mathbf{M}||_* \,,$$

where μ is the bias parameter and $||\cdot||_*$ is the trace norm. Such a formulation encourages predicting y_t with large margin and low rank in $\mathbf{M}$. A learned matrix $\mathbf{M}$ that has low rank can be useful in a number of machine learning tasks; e.g., distance-based classifications, clusterings, and low-dimensional embeddings. We refer to [11] for details.



(a) Learning with expert advice



(b) Metric learning

Figure 3: Experiment results: Our method CBCE outperforms several baseline methods.

We create a scenario that exhibits shifts in the metric, which is inspired by [7]. Specifically, we create a mixture of three Gaussians in $\mathbb{R}^3$ whose means are well-separated, and mixture weights are .5, .3, and .2. We draw 2000 points from it while keeping a record of their memberships. We repeat this three times independently and concatenate these three vectors to have 2000 9-dimensional vectors. Finally, we append to each point a 16-dimensional vector filled with Gaussian noise to have 25-dimensional vectors. Such a construction implies that for each point there are three independent cluster memberships. We run each algorithm for 1500 time steps. For time 1 to 500, we randomly pick a pair of points from the data pool and assign $y_t = 1$ ($y_t = -1$) if the pair belongs to the same (different) cluster under the first clustering. For time 501 to 1000 (1001 to 1500), we perform the same but under the second (third) clustering. In this way, a learner faces tracking the change in metric, especially the important low-dimensional subspaces for each time segment.

Since the loss of the metric learning is unbounded, we scale the loss by multiplying $1/5$ and then capping it above at 1 as in [7]. Although the randomized decision discussed in Section 4 can be used to maintain the theoretical guarantee, we stick to the weighted average since the event that the loss being capped at 1 is rare in our experiments. As in our LEA experiment, we use the data streaming intervals with $g = 2$ and initialize each black-box algorithm with the decision of the meta algorithm at the previous time step. We repeat the experiment 50 times and plot their average loss in Figure 3(b) by moving mean with window size 20. We observe that CBCE and ATV both outperforms SAOL. This confirms the improved regret bound of CBCE and ATV.

6 Future Work

Among a number of interesting directions, we are interested in reducing the time complexity in the online learning within a changing environment. For LEA, Fixed Share has the best time complexity. However, Fixed Share is inherently not parameter-free; especially, it requires the knowledge of the number of shifts m. Achieving the best m-shift regret bound without knowing m or the best SA-Regret bound in time $O(NT)$ would be an interesting future work. The same direction is interesting for the online convex optimization (OCO) problem. It would be interesting if an OCO algorithm such as online gradient descent can have the same SA-Regret as CBCE$\langle$OGD$\rangle$ without paying extra order of computation.

Acknowledgements

This work was supported by NSF Award IIS-1447449 and NIH Award 1 U54 AI117924-01. The authors thank András György for providing constructive feedback and Kristjan Greenewald for providing the metric learning code.

References

[1] D. Adamskiy, W. M. Koolen, A. Chernov, and V. Vovk, "A Closer Look at Adaptive Regret," in *Proceedings of the International Conference on Algorithmic Learning Theory (ALT)*, 2012, pp. 290–304.

[2] A. Blum and A. Blum, "Empirical Support for Winnow and Weighted-Majority Algorithms: Results on a Calendar Scheduling Domain," *Machine Learning*, vol. 26, no. 1, pp. 5–23, 1997.

[3] N. Cesa-Bianchi, P. Gaillard, G. Lugosi, and G. Stoltz, "Mirror descent meets fixed share (and feels no regret)," in *Advances in Neural Information Processing Systems (NIPS)*, 2012, pp. 980–988.

[4] N. Cesa-Bianchi and G. Lugosi, *Prediction, Learning, and Games.* Cambridge University Press, 2006.

[5] A. Daniely, A. Gonen, and S. Shalev-Shwartz, "Strongly Adaptive Online Learning," *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 1–18, 2015.

[6] Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth, "Using and combining predictors that specialize," *Proceedings of the ACM symposium on Theory of computing (STOC)*, vol. 37, no. 3, pp. 334–343, 1997.

[7] K. Greenewald, S. Kelley, and A. O. Hero, "Dynamic metric learning from pairwise comparisons," *54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2016.

[8] A. György, T. Linder, and G. Lugosi, "Efficient tracking of large classes of experts," *IEEE Transactions on Information Theory*, vol. 58, no. 11, pp. 6709–6725, 2012.

[9] E. Hazan and C. Seshadhri, "Adaptive Algorithms for Online Decision Problems," *IBM Research Report*, vol. 10418, pp. 1–19, 2007.

[10] M. Herbster and M. K. Warmuth, "Tracking the Best Expert," *Mach. Learn.*, vol. 32, no. 2, pp. 151–178, 1998.

[11] G. Kunapuli and J. Shavlik, "Mirror descent for metric learning: A unified approach," in *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Database (ECML/PKDD)*, 2012, pp. 859–874.

[12] H. Luo and R. E. Schapire, "Achieving All with No Parameters: AdaNormalHedge," in *Proceedings of the Conference on Learning Theory (COLT)*, 2015, pp. 1286–1304.

[13] F. Orabona and D. Pal, "Coin betting and parameter-free online learning," in *Advances in Neural Information Processing Systems (NIPS)*, 2016, pp. 577–585.

[14] J. Veness, M. White, M. Bowling, and A. György, "Partition tree weighting," in *Proceedings of the 2013 Data Compression Conference.* IEEE Computer Society, 2013, pp. 321–330.