Learning with contextual information in non-stationary environments

Sean Anderson SEANANDERSON@UCSB.EDU and **João P. Hespanha** HESPANHA@ECE.UCSB.EDU
University of California Santa Barbara, Santa Barbara, CA 93106

Editors: N. Ozay, L. Balzano, D. Panagou, A. Abate

Abstract

We consider a repeated decision-making setting in which the decision maker has access to contextual information and lacks a model or a priori knowledge of the relationship between the actions, context, and costs that they aim to minimize. Moreover, we assume that the environment may be non-stationary due to the presence of other agents that may be reacting to our decisions. We propose an algorithm inspired by log-linear learning that uses Boltzmann distributions to generate stochastic policies. We consider two general notions of context and provide regret bounds for each: 1) a finite number of possible measurements and 2) a continuum of measurements that weight a set of finite classes. In the non-stationary setting, we incur some regret but can make it arbitrarily small. We illustrate the operation of the algorithm through two examples: one that uses synthetic data (based on the rock-paper-scissors game) and another that uses real data for malware classification. Both examples exhibit (by construction or naturally) significant lack of stationarity.

Keywords: no regret dynamics, non-stationary learning, adversarial learning, contextual information, multiplicative weights update

1. Introduction

We address learning in repeated decision making with contextual (or side) information. Specifically, a decision maker needs to select an action to minimize a prescribed cost with feedback from measurements that provide context. Importantly, we consider scenarios in which the cost depends on latent random variables whose distributions change with time without a stationary model. This prevents a Bayesian optimal decision, based on past data.

Repeated decision making arises in numerous domains, including clinical trials, portfolio selection, pricing, recommender systems, anomaly detection, cyber security, among many others (Bouneffouf et al., 2020). In essentially all these examples, first principle models for the relationship between actions, contextual information, and costs are not available and must be learned from data. The need to consider non-stationary processes arises when the mechanisms that generate the contextual information and/or the rewards/costs involve strategic agents that observe and react to past decisions. For example, decisions made by a cybersecurity system may inform future attackers about the system's defenses and vulnerabilities and thus affect their behavior.

Learning without stationarity may seem an impossible task, but Robbins (1951); Hannan (1957) showed that it can be accomplished by abandoning the goal of optimal Bayesian decision-making and, instead, accept the weaker goal of no-regret or low-regret with respect to some reasonable baseline performance. We follow this approach and focus our attention on algorithms that result in low average regret against any static decision rules, either deterministic or stochastic.

For non-stationary problems, using stationary policies as a baseline to determine regret may seems restrictive, as we would like to consider baselines that can "track" the current state of the process. To accomplish this, we deviate from the classic notion of average regret over the whole set of

past decisions, and consider instead average regret just over a window of recent measurements. We then ask for low-regret against any stationary policy that is optimal over *every past window of recent measurements*. In this paper, we work with "soft windows," in the sense that we actually compute regret over the full set of past decisions, but weight the costs in distant past with an exponentially decaying term. The use of soft windows turns out to be computationally much more efficient than fixed windows and, as we shall see, provides good performance in non-stationary environments.

We propose an algorithm we refer to as Boltzmann learning inspired by log-linear learning (Blume, 1993) that when using unweighted average regret and in the absence of contextual information essentially becomes multiplicative weights update (Littlestone and Warmuth, 1994; Auer et al., 1995; Arora et al., 2012). However, we express the algorithm in terms of Boltzmann distributions, analogous to what is done in log-linear learning (Blume, 1993; Marden and Shamma, 2012), to somewhat facilitate the extension to the exponentially weighted case.

This paper makes several key contributions:
1.) We consider decision making with contextual information, which raises the problem from selecting actions to selecting feedback policies. We initially consider a finite set of measurements/contexts corresponding to all possible measurements we could observe. We then extend this to stochastic classifier-based policies that are motivated by scenarios where "raw measurements" are processed by a classification system that attempts to classify each raw measurement into a finite set of classes and produces a probability distribution that quantifies the uncertainty in this classification.

Both for the case of a finite number of measurements and for measurements consisting of probability distributions (classifier-based policies), we provide explicit bounds on the regret. These bounds show the regret can be made arbitrarily small and establish regret scaling laws.
2.) We then consider a new form of exponentially weighted regret that discounts the effect of decisions made in the distant past. This form of regret is especially important in non-stationary environments, as confirmed by synthetically generated and real data examples. Our regret bounds show that regret scales with $\mathcal{O}(\sqrt{\mu_\infty \lambda})$ where λ is the forgetting factor used for the exponential weighting and μ_∞ the average number of decisions per unit time–essentially showing that the more decisions made per unit of time, the more we can afford to forget. Over K decisions, in the absence of forgetting ($\lambda = 0$) our regret bounds exhibit the familiar scaling $\mathcal{O}(\sqrt{K})$, which is well known in the absence of contextual information.
3.) We illustrate our algorithm through two examples. The first example uses synthetic data obtained from simulating a player of rock-paper-scissors that faces an adversary that simulates an optimal player, but this adversary uses a "bad random number generator" to make its selection. Non-stationarity arises from the opponent occasionally changing its random number generator. With synthetic data, we have ground truth and can control the degree of non-stationarity, which enables us to examine how quickly and how well different algorithms adapt. The second example is about malware detection and the problem here is to decide whether or not a specific piece of code should be classified as malware. This is a prototypical example of a problem that lacks stationarity because malware producers constantly react to the software that detects malware.

For both examples we compare our algorithm against two well known approaches: a support vector machine classifier and a neural network-based classifier that are periodically retrained. Either of these would stop working almost immediately if trained solely on "old data", so we train both algorithms on a rolling window of past data. In practice, this makes either of the competing algorithms computationally much more expensive than Boltzmann learning and yet their performance is, at best, comparable with the proposed algorithms. The rest of the paper is organized as follows:

we provide a brief literature review in Section 2 and then introduce the general problem of an agent trying to minimize a cost function given some measurements in Section 3. In Section 4 we present the non-stationary setting and definitions of regret. We also present the decision-making algorithm and regret bound for the finite measurement settings. We then extend the finite measurements to the continuum measurement setting in Section 5. Section 6 compares the proposed algorithm with competing approaches in the context of two numerical examples.

2. Related Work

Robbins (1951) inspired Hannan (1957); Gilliland (1969) to conceptualize regret as the difference between the average cost and the optimal Bayesian decision for the observed empirical distribution of the other player. Extensive literature in this area include (Gilliland, 1969; Bubeck and Cesa-Bianchi, 2012; Shalev-Shwartz, 2012; Roughgarden, 2016). Closely related to our work is the *randomized weighted majority algorithm* introduced by Littlestone and Warmuth (1994), which guarantees an average regret over T decisions among N options $O(\sqrt{(\log N)/T})$ (Blum and Mansour, 2007). The variation of this algorithm, also known as the multiplicative weights update, that appears in (Roughgarden, 2016) can be viewed as selecting actions according to Boltzmann distributions with energies constructed from past costs, and leads to an (asymptotically) similar regret. Hedge (Freund and Schapire, 1997) uses similar updates but is usually used in expert advice problems. We also highlight a class of algorithms known as *follow the regularized leader* (Shalev-Shwartz and Singer, 2006; Shalev-Shwartz, 2012), which provides an umbrella for multiple algorithms, including the multiplicative weights update (Littlestone and Warmuth, 1994; Auer et al., 1995; Arora et al., 2012). Online convex optimization classically tries to minimize a static regret, but Zinkevich (2003) introduced the idea of dynamic regret whereby comparisons are made against a sequence of comparators, further developed in Zhang et al. (2018); Zhao et al. (2022).

The *multi-armed bandit* problem considers decision-making scenarios where at each round an agent selects an action and only observes the corresponding reward (Robbins, 1952). Traditionally considered in stationary environments, this has been extended to specific non-stationary settings, e.g., Besbes et al. (2014); Cheung et al. (2022); Russac et al. (2019). *Contextual bandits* such as those proposed by Chu et al. (2011); Agrawal and Goyal (2013); Abbasi-Yadkori et al. (2011) introduce "context" or "side information" that aids decision-making. Auer et al. (2002) introduced the well-known Exp3 algorithm in *adversarial bandits* and differs from our work in that it lacks contextual information or the use of exponentially decaying weights. *Adversarial contextual bandits* have been studied for linear objectives by Neu and Olkhovskaya (2020).

3. Problem Formulation

We consider a scenario where an agent seeks to minimize a cost by making a sequence of decisions in response to a stream of measurements. The kth decision made by the agent at some time t_k consists of selecting an action a_k among a set $\mathcal{A}$ of possible actions. Each action a_k results in a cost $J(a_k, \omega_k)$ that also depends on an unknown stochastic *latent variable* ω_k, which captures changes in the reward due to exogenous factors such as an adversary or seasonal trends; the agent seeks to minimize the expected value $\mathrm{E}\left[J(a_k, \omega_k)\right]$.

To decide on the value of a_k, the agent has available a *context measurement* y_k, taking values in a set $\mathcal{Y}$ of all possible measurements. A *deterministic feedback policy* is a function $\rho : \mathcal{Y} \to \mathcal{A}$ that

selects the action a_k according to $a_k = \rho(y_k) \in \mathcal{A}, \quad \forall k \geqslant 1$. Similarly, a *stochastic decision policy* is a function $\rho : \mathcal{Y} \to \Pi_{\mathcal{A}}$ that maps measurements to the simplex $\Pi_{\mathcal{A}}$ of probability distributions over the finite set $\mathcal{A}$, with the understanding that the action a_k will be selected with the probability distribution $\pi_k = \rho(y_k) \in \Pi_{\mathcal{A}}$.

4. Boltzmann Learning

When the statistics of the cost $J(a, \omega_k)$, $k \geqslant 1$ are unknown and possibly vary across time, it is unreasonable to expect that we will ever be able to use a finite set of data to estimate a policy that selects each individual a_k optimally to minimize the corresponding $J(a, \omega_k)$. Instead, we will seek to learn policies that lead to small "regret" with respect to any policy ρ within a given class of policies P, when the agent looks back at the actual "average cost" incurred over a past time horizon. The following definition formalizes the meaning of "regret" and "average cost."

Definition 1 (Unweighted average regret) *A rule used to select the a_k has α_1-α_0* (unweighted) average regret over a horizon of length K for a class of policies P *if*

$$\frac{1}{K}\sum_{k=1}^{K} \mathrm{E}\big[J(a_k, \omega_k)\big] \leqslant \alpha_1\Big(\frac{1}{K}\sum_{k=1}^{K} \mathrm{E}\big[J(\rho(y_k), \omega_k)\big] + \alpha_0\Big), \quad \forall \rho \in P. \tag{1}$$

For $\alpha_1 = 1$ and $\alpha_0 = 0$ the inequality in (1) indicates that the average cost incurred for the first K decisions (left-hand side) is no worse than the average cost that would have been incurred by any policy $\rho \in P$ (right-hand side). When $\alpha_1 > 1$ and/or $\alpha_0 > 0$ some regret may arise when we compare our selection of the a_k with what the policy ρ would have selected, but this regret remains small as long as α_1 and α_0 do not grow much above 1 and 0, respectively. We distinguish between "policies" (memory-less maps from measurements to action) and "learning rules" to select the actions a_k, which typically involve some form of learning and thus have memory.

Especially suitable for non-stationary processes, we also consider a notion of regret that includes an exponentially decaying factor that is not restricted to a fixed horizon length and discounts the weight of costs incurred in the distant past.

Definition 2 (Discounted regret) *A rule used to select the a_k has α_1-α_0* discounted regret with a discount factor $\lambda > 0$ for a class of policies P *if*

$$\frac{1}{W_k}\sum_{\ell=1}^{k} e^{-\lambda(t_k - t_\ell)}\, \mathrm{E}\big[J(a_\ell, \omega_\ell)\big] \leqslant \alpha_1\Big(\frac{1}{W_k}\sum_{\ell=1}^{k} e^{-\lambda(t_k - t_\ell)}\, \mathrm{E}\big[J(\rho(y_\ell), \omega_\ell)\big] + \alpha_0\Big), \quad \forall k \geqslant 1, \rho \in P, \tag{2}$$

where $W_k \coloneqq \sum_{\ell=1}^{k} e^{-\lambda(t_k - t_\ell)}$ is a normalizing factor.

Both definitions (1) and (2) involve expected values for the regret, and therefore allow individual realizations to exhibit large regret as long as they have low probability. The following definitions are more demanding in that they require low regret with probability one, when we restrict our attention to the information available at the time each decision is made.

The *information available to decide on the action a_k* includes all past measurements, up to and including y_k; as well as the past outcomes $J(a, \omega_\ell)$, $a \in \mathcal{A}$ for decision $\ell < k$. Formally, this can be

represented by a filtration $\{\mathcal{F}_k : k \geqslant 1\}$, with each σ-algebra $\mathcal{F}_k$ generated by the random variables that encode all available information available to select a_k:

$$\mathcal{F}_k \coloneqq \big\{y_1, \dots, y_k, \quad J(a, \omega_1), \dots, J(a, \omega_{k-1}) : \forall a \in \mathcal{A}\big\}. \tag{3}$$

Definition 3 (Discounted regret with filtration) *A rule used to select the a_k has α_1-α_0* discounted regret with probability one for a discount factor $\lambda > 0$ *if instead,*

$$\frac{1}{W_k}\sum_{\ell=1}^{k} e^{-\lambda(t_k - t_\ell)}\,\mathrm{E}\big[J(a_\ell, \omega_\ell) \mid \mathcal{F}_\ell\big] \overset{\text{wpo}}{\leqslant} \alpha_1\Big(\frac{1}{W_k}\sum_{\ell=1}^{k} e^{-\lambda(t_k - t_\ell)} J(\rho(y_\ell), \omega_\ell) + \alpha_0\Big),\ \forall k \geqslant 1, \rho \in P, \tag{4}$$

where the left-hand sides of these inequalities includes an expectation conditioned to all the information that was available prior to the selection of a_k, and should be understood as the "best guess" at $J(a_k, \omega_k)$, given all the information that was available prior to the selection of a_k.

By the monotonicity of the expected value, taking expectations on both sides of (4), we obtain (2), which shows that discounted regret with probability one implies discounted regret. For $\lambda = 0$ and $k = K$, we recover the α_1-α_0 (unweighted) average regret over a horizon of length K.

4.1. Main Result: Finite Measurement Sets

We consider decision rules for which actions are selected according to Boltzmann distributions with energies constructed using the values of $J(a, \omega_k)$ that became known "after-the-fact". Specifically, we define the energy to be

$$E_a(k+1; y) \coloneqq \begin{cases} 0 & k = 0, \\ \sum_{\ell=1}^{k} e^{-\lambda(t_{k+1} - t_\ell)} J(a, \omega_\ell) I_{y_\ell = y}, & k \geqslant 1, \end{cases} \qquad \forall a \in \mathcal{A},\ y \in \mathcal{Y}, \tag{5}$$

where $I_{y_\ell = y} \in \{0, 1\}$ denotes the indicator function of the event $y_\ell = y$ and makes sure that the energy $E_a(k, y)$ is constructed solely from rewards $J(a, \omega_\ell)$ associated with the measurement y.

The action a_k selected at time t_k is then selected stochastically using the following (conditional) Boltzmann distribution

$$\mathrm{P}\left(a_k = a \mid \mathcal{F}_k\right) = \frac{e^{-\beta E_a(k; y_k)}}{\Gamma_k(y_k)}, \qquad \Gamma_k(y_k) \coloneqq \sum_{\bar{a} \in \mathcal{A}} e^{-\beta E_{\bar{a}}(k; y_k)}, \tag{6}$$

for some constant $\beta \geqslant 0$ that plays the role of the inverse of a thermodynamic temperature. Note that for $\lambda > 0$, the computational requirements are equivalent to the $\lambda = 0$ case.

Theorem 1 (Finite measurement sets) *Assume that $J(a, \omega_k) \overset{\text{wpo}}{\in} [J_{\min}, J_{\max}],\ \forall a \in \mathcal{A},\ k \geqslant 1$. For every deterministic stationary policy ρ,*

$$\frac{1}{W_k}\sum_{\ell=1}^{k} e^{-\lambda(t_k - t_\ell)}\,\mathrm{E}\big[J(a_\ell, \omega_\ell) \mid \mathcal{F}_\ell\big] \overset{\text{wpo}}{\leqslant} \frac{1}{\delta}\Big(\frac{1}{W_k}\sum_{\ell=1}^{k} e^{-\lambda(t_k - t_\ell)} J(\rho(y_\ell), \omega_\ell) + \frac{e^{\lambda(t_{k+1} - t_k)} |\mathcal{Y}| \log |\mathcal{A}|}{\beta W_k} - (1 - \delta) J_{\min}\Big), \tag{7}$$

with $W_k \coloneqq \sum_{\ell=1}^{k} e^{-\lambda(t_k - t_\ell)}$, $\delta \coloneqq \frac{1 - e^{-\beta(J_{\max} - J_{\min})}}{\beta(J_{\max} - J_{\min})} \in [0, 1)$. (Proof in (Anderson and Hespanha, 2024).) □

4.2. No Regret

The following result shows that for unweighted average regret ($\lambda = 0$), it is always possible to get the constants α_1 and α_0 in (1) arbitrarily close to 1 and 0, respectively, for a sufficiently large K. For discounted regret, it is also possible to get the constants α_1 and α_0 in (2) arbitrarily close to 1 and 0, respectively, for a sufficiently small λ. However, in this case we need "persistent feedback."

Assumption 1 (Persistent Feedback) *There are finite constants $\mu_0 \geqslant \mu_\infty \geqslant 0$ such that the time interval $t_{k_2} - t_{k_1}$ between receiving feedback on the costs $J(a, \omega_{k_1})$, $J(a, \omega_{k_2})$, $\forall a \in \mathcal{A}$ satisfy*

$$t_{k_2} - t_{k_1} \leqslant \mu_\infty(k_2 - k_1 - 1) + \mu_0, \quad \forall 1 \leqslant k_1 < k_2. \qquad \square$$

This assumption essentially requires that the average time interval between decisions remain bounded, since $\frac{t_{k_2}-t_{k_1}}{k_2-k_1} \leqslant \mu_\infty + \frac{\mu_0-\mu_\infty}{k_2-k_1} \in [\mu_\infty, \mu_0], \quad \forall 1 \leqslant k_1 < k_2$. We can thus view μ_∞ as an asymptotic upper bound on the average time interval between decisions (as $k_2 - k_1 \to \infty$) and μ_0 as a short-term upper bound (when $k_2 = k_1 + 1$).

Corollary 1 (No regret) *For α_1-α_0 unweighted average regret ($\lambda = 0$), for every K, it is always possible to select β in (6) to get $\alpha_1 = 1 + O(1/\sqrt{K})$ and $\alpha_0 = O(1/\sqrt{K})$ in (1). Under the Persistent Feedback Assumption, for α_1-α_0 discounted regret, for every $\lambda > 0$, it is always possible to select β in (6) to get $\alpha_1 = 1 + O(\sqrt{\mu_\infty \lambda})$ and $\alpha_0 = O(\sqrt{\mu_\infty \lambda})$ in (2), for a sufficiently large k. (See proof in (Anderson and Hespanha, 2024)).* $\square$

For the case $\lambda > 0$, the quantity $\tau_{1/2} \coloneqq \log(2)/\lambda$ can be viewed as the time it takes for the weights in (4) to decrease to $1/2$ (called the half-life) and Corollary 1 shows a scaling of the regret with 1 over the square root of $\frac{\tau_{1/2}}{\mu_\infty}$, which is essentially the average number of decisions per half-life.

5. Continuum Measurement Spaces with Classifier-Based Policies

We now consider rules that make decisions based on context measurements y that belong to a measurement space $\mathcal{Y} \coloneqq \Pi_{\mathcal{C}} \subset \mathbb{R}^{|\mathcal{C}|}$ consisting of probability distributions over some discrete set $\mathcal{C}$. In this case, we define $\alpha_1 - \alpha_0$ regret over the class P of *stochastic classifier-based policies*, which select action $a_k = a$ with a probability that is linear on the measurement y_k:

$$\mathrm{P}(a_k = a \mid y_k) = \langle z_a, y_k \rangle, \quad \forall a \in \mathcal{A}, \tag{8}$$

where the vectors $\{z_a \in \mathbb{R}^{|\mathcal{C}|} : a \in \mathcal{A}\}$ characterize a specific linear stochastic policy and must be normalized so that the right hand side results in a valid probability distribution for every $y \in \mathcal{Y} \coloneqq \Pi_{\mathcal{C}}$. Specifically, these vectors must satisfy

$$\langle z_a, y \rangle \geqslant 0, \quad \sum_a \langle z_a, y \rangle = 1, \quad \forall y \in \mathcal{Y} \coloneqq \Pi_{\mathcal{C}}. \tag{9}$$

In this setting a classification system attempts to classify "raw measurements" into a finite set of classes $\mathcal{C}$ and produces a probability distribution for class assignment. Our kth context measurement $y_k \in \Pi_{\mathcal{C}}$ is then a vector containing the probabilities that the kth raw measurement belongs to each class in $\mathcal{C}$, *as reported by the classifier.* Denoting by $\eta_k \in \mathcal{C}$ the class of the kth raw measurement,

the entries of z_a can then be viewed as conditional probabilities of selecting the action $a_k = a$ given that $\eta_k = c$, for each $c \in \mathcal{C}$. In this case, (8) can be viewed as computing the probability of selecting $a_k = a$ using the law of total probability:

$$\mathrm{P}(a_k = a \mid y_k) = \sum_{c \in \mathcal{C}} \underbrace{\mathrm{P}(a_k = a \mid \eta_k = c)}_{\substack{\text{entries of } z_a, \\ \text{defined by the policy}}} \underbrace{\mathrm{P}(\eta_k = c \mid y_k)}_{\substack{\text{entries of } y_k, \text{ reported} \\ \text{by the classifier}}} = \langle z_a, y_k \rangle. \tag{10}$$

While (10) motivates the formula (8) used by stochastic classifier-based policies, none of the results that follow assume that the classifier is accurate in computing class probabilities. In fact, we do not assume that the context measurements are produced by an actual classifier nor that the algorithm that generates the context measurements y_k is stationary. Additionally, we can use multiple classifiers as we will demonstrate in the numerical results. Nevertheless, our goal is to design a learning rule that will lead to low regret with respect to every policy of the general form (8).

5.1. Boltzmann Learning

We now select actions according to a mixture of Boltzmann distributions, each corresponding to one pair $(a, c) \in \mathcal{A} \times \mathcal{C}$. To this effect, we define energies:

$$E_{a,c}(k+1) \coloneqq \begin{cases} 0 & k = 0, \\ \sum_{\ell=1}^{k} e^{-\lambda(t_{k+1} - t_\ell)} J(a, \omega_\ell) \langle e_c, y_\ell \rangle, & k \geqslant 1, \end{cases} \qquad \forall a \in \mathcal{A},\ c \in \mathcal{C}, \tag{11}$$

where e_c denotes the cth vector of the canonical basis of $\mathbb{R}^{|\mathcal{C}|}$ and therefore $\langle e_c, y_\ell \rangle$ extracts from y_ℓ the probability corresponding to the class $c \in \mathcal{C}$. We then select action $a_k = a$ with probability

$$\mathrm{P}\left(a_k = a \mid \mathcal{F}_k\right) = \frac{\sum_c \langle e_c, y_k \rangle \mathrm{P}\left(a_k = a, \eta_k = c \mid \mathcal{F}_k\right)}{\sum_c \langle e_c, y_k \rangle \sum_{\bar{a}} \mathrm{P}\left(a_k = \bar{a}, \eta_k = c \mid \mathcal{F}_k\right)}, \quad \forall a \in \mathcal{A},$$

where

$$\mathrm{P}\left(a_k = a, \eta_k = c \mid \mathcal{F}_k\right) = \frac{e^{-\beta E_{a,c}(k)}}{\Gamma_k(y_k)} \qquad \Gamma_k(y_k) \coloneqq \sum_{\bar{c} \in \mathcal{C}, \bar{a} \in \mathcal{A}} e^{-\beta E_{\bar{a},\bar{c}}(k)}.$$

Theorem 2 (Classifier-based measurement sets) *Assume that* $J(a, \omega_k) \overset{\text{wpo}}{\in} [J_{\min}, J_{\max}]$, $\forall a \in \mathcal{A},\ k \geqslant 1$. *For every stochastic classifier-based policy defined by* (8),

$$\frac{1}{W_k} \sum_{\ell=1}^{k} e^{-\lambda(t_k - t_\ell)} \mathrm{E}\left[J(a_\ell, \omega_\ell) \mid \mathcal{F}_\ell\right] \overset{\text{wpo}}{\leqslant} \frac{1}{\delta}\Bigg(\frac{1}{W_k} \sum_{\ell=1}^{k} e^{-\lambda(t_k - t_\ell)} \Big(\sum_{a \in \mathcal{A}} \langle z_a, y_\ell \rangle J(a, \omega_\ell) \Big) + \frac{e^{\lambda(t_{k+1} - t_k)} |\mathcal{C}| \log |\mathcal{A}||\mathcal{C}|}{\beta W_k} - (1 - \delta) J_{\min} \Bigg). \tag{12}$$

(See proof in (Anderson and Hespanha, 2024).) □

Comparing (12) with (7) in Theorem 1, we see that growing $\mathcal{Y}$ from a finite set to the (infinite) set of probability distributions led to the term $|\mathcal{Y}| \log |\mathcal{A}|$ to be replaced by $|\mathcal{C}| \log |\mathcal{A}||\mathcal{C}|$, thus incurring an additional logarithmic penalty in the number of measurement classes. The no regret results in Corollary 1 essentially remain the same for this case.

6. Numerical Experiments

We illustrate the behavior of Boltzmann learning in the finite measurement and continuum measurement settings by comparing against alternative approaches to action selection, which include machine learning models. For the latter, in Scikit-learn (Pedregosa et al., 2011) we implement a 1) support vector machine (SVM) with radial basis function kernel and 2) multi-layer perceptron neural network (MLP) that are periodically retrained to handle non-stationarities. We do not discount the existence of potentially more accurate models than the ones presented here but rather use these to highlight the relatively high computational burden associated with these compared to Boltzmann learning. Code for this paper is available at `www.github.com/seanjkanderson/boltzmann-learning`.

We train SVMs and MLPs to construct estimates of the cost J associated with measurement-action pairs $(y_k, a),\ \forall a \in \mathcal{A}$ and then select the action that minimizes the estimated cost. We also trained the SVMs and MLPs as "classifiers" that predict the opponent's action, but the former approach led to the best results, so we only present those results here. However, the results from the classifier-based approach are included in (Anderson and Hespanha, 2024).

Training the SVMs and MLPs solely on an initial dataset results in decisions that are unable to handle a non-stationary environment. In view of that, we continuously retrain these decision makers on a sliding window of data of length H. In contrast, Boltzmann learning is designed to handle a time-varying environment, so we simply ran the algorithms described in Sections 4.1 and 5.

6.1. Finite Measurements: Rock-Paper-Scissors

We first consider the game rock-paper-scissors (RPS), a.k.a. Roshambo, with two players. In every round both players simultaneously choose to play rock, paper, or scissors such that $\mathcal{A} = \{R, P, S\}$, with rock beating scissors, scissors beating paper, and paper beating rock.

In this scenario, Player 1 uses Boltzmann learning while Player 2 uses a "bad randomizer" to select actions. More specifically, Player 2 plays a fixed sequence of 150 randomly selected actions in a round-robin fashion. The "contextual information" used by player 1 consists of the 5 most recent actions, leading to a total of $3^5 = 243$ distinct measurements in $\mathcal{Y}$. We introduce non-stationarity by changing the 150 action sequence that player 2 uses at iteration 20k. The SVM and MLP are trained on a sequence that includes the outcomes of the last 1k episodes, which enables both algorithms to observe at least 6 full sequences of the actions that player 2 keeps repeating. In this case, this horizon provides the MLP enough data to achieve optimal performance under stationary operation.
Results Under stationary conditions, the MLP attains the best performance, followed closely by Boltzmann learning with $\lambda = 1\mathrm{e}-3, \beta = 1$. This can be seen in Figure 1(a) from iterations 5k to 20k and then from 25k to 100k. When the opponent's policy changes at iteration 20k, all policies go through a period of poor performance but eventually recover. The same two policies mentioned above recover the fastest, reaching a low cost after around 25k iterations, whereas the Bayesian estimator (which assumes stationarity) recovers the slowest at about 62k.

The performance of Boltzmann learning depends on the selection of the parameters λ and β, with a small value for λ resulting in a slow recovery from changes in the environment (dark blue curve closely follows the Bayesian estimator) and a very large value for λ unable to produce a sufficiently low regret (light green curve above -0.02). The poor performance for large λ should be expected in view of the results in Section 4.2 that show small regret generally requires a small λ.

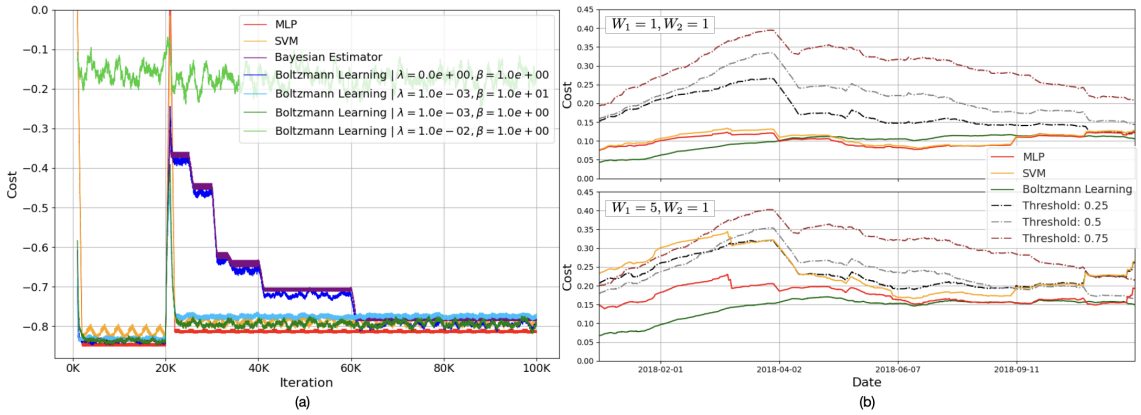

Figure 1: (a) RPS is played over 100k episodes (x-axis) and the cost of each game is encoded by +1 if player 1 loses, -1 if it wins, and 0 for a draw. The y-axis shows a rolling average of this cost over the last 1k episodes. (b) Malware detection is conducted on the streamed EMBER dataset where the x-axis has the iterations uniformly spaced but with the corresponding date. The y-axis shows a rolling average of the cost over the previous 100k points. The optimal policy in binary classification is typically a threshold policy, so we include the results of using a threshold policy based on the output from a simple average of the three stationary classifiers with thresholds $0.25, 0.5$, and 0.75. The results in the top and bottom differ by the false positives penalty W_1. In both cases for Boltzmann learning $\lambda = 1\text{e} - 1$ and $\beta = 1\text{e}2$.

6.2. Continuum Measurements: Malware Classification

We consider a malware classification problem where the adversary is sharing binary files, and we aim to classify the file as malicious or benign based on available file features. We aim to minimize the weighted sum of false positive and false negative classification errors: $J = W_1 P(\text{false pos.}) + W_2 P(\text{false neg.})$. We use the Elastic Malware Benchmark for Empowering Researchers (EMBER) 2018 dataset (Anderson and Roth, 2018), which contains 2,831 features extracted from binary files and corresponding ground-truth labels. We omit any unlabeled samples and, in order to capture the "dynamics" of malware production over time, we sort the data by credible header timestamps: we include samples starting from 2016 whose header timestamps are less than or equal to the approximate month in which they were first detected by malware researchers. Training classifiers on static datasets is common in malware classification (e.g., Anderson and Roth (2018)), thus motivating the use of models such as MLPs and SVMs, with periodic retraining to improve performance.

The contextual information used in this example are the classification results generated by three ensemble classifiers implemented in LightGBM (Ke et al., 2017). Two of the classifiers are gradient boosting machines, one with the hyperparameters from (Anderson and Roth, 2018) and the other with shallower depths and fewer leaves to (potentially) reduce overfitting. The third classifier is a random forest, which often exhibits less overfitting than gradient boosting machines.

This example uses the continuum measurement space introduced in Section 5. To this effect, we regard the output of the ith classifier as the probability p_i of a Bernoulli random variable c_i, and we take the context measurement y_k to be the probability distribution of the triple (c_0, c_1, c_2), where the

c_i are assumed independent. Since the triple includes 3 binary random variables, y_k is a distribution of a set with $2^3 = 8$ distinct measurements. It is important to emphasize that the results in Section 5 do not rely on the measurements y_k satisfying any specific probabilistic model so that the validity of the regret bounds do not rely on the "correctness" of the procedure to construct y_k. We compare our performance against an SVM and MLP with window H=10k, retraining every 10k steps, and using (y_k, a) as inputs (we consider raw measurements in (Anderson and Hespanha, 2024)).

Results Boltzmann learning performs comparably or better than the MLP and SVM, and at a fraction of the computational load. In Figure 1(b) in the top subplot with weights $W_1 = W_2 = 1$, we observe that Boltzmann learning (green) initially outperforms the MLP (red) and SVM (orange) until 2018-04 before performing slightly worse until 2018-09, and the three methods perform similarly for the remainder. However, when $W_1 = 5$ such that false positives are weighted more heavily, Boltzmann learning consistently has the lowest cost despite all three methods aiming to minimize the same objective. Intuitively, Boltzmann learning adaptively updates the policy only when alternative actions could lead to a lower cost. Instead of a fixed retraining schedule for the MLP and SVM, one could imagine multiple heuristics to trigger retraining, but using Boltzmann learning comes with strong theoretical performance guarantees. The SVM, MLP, and Boltzmann learning typically outperform static threshold policies, suggesting that dynamic classifier combinations are superior to static thresholds.

Compute comparison The SVM and MLP require significantly more compute than Boltzmann learning because these methods involve periodic retraining, whereas Boltzmann learning simply involves updating the energies in (5) and (11). The computation time results in Table 1 indicate Boltzmann learning is around an order of magnitude faster than alternatives, but this could be much improved through further code optimization that the SVM and MLP already benefit from. Boltzmann learning requires significantly less memory than alternatives because it does not store a window H of past data (measurement, action, and cost per action) but rather just current energy values for each action and measurement.

Table 1: We perform computations in Python on the CPU of a 2021 M1 Max chip with 10 cores and 32GB RAM. We use M here to denote either $|\mathcal{Y}|$ or $|\mathcal{C}|$ depending on the example ($H > M$ usually). These memory requirements do not consider the model size, where Boltzmann learning only requires β and λ, but the MLP and SVM require model weights.

Metric	**MLP**	**SVM**	**BL**
RPS time (ms)	1.4	1.6	0.056
Malware (probabilities) time (ms)	1.2	1.4	0.27
Memory requirements (hard window of size H)	$H\big(M + 2\lvert\mathcal{A}\rvert\big)$	$H\big(M + 2\lvert\mathcal{A}\rvert\big)$	$M\lvert\mathcal{A}\rvert$

7. Conclusion

We presented a computationally efficient decision-making algorithm for settings in which contextual measurements are available and the environment is non-stationary and showed that regret can be made arbitrarily small. Numerical experiments illustrated how we can perform comparably or better than significantly more computationally intensive decision-making algorithms.

8. Acknowledgements

This material is based upon work supported by the U.S. Office of Naval Research MURI grant No. N00014-23-1-2708, National Science Foundation grant No. 2229876, and National Science Foundation Graduate Research Fellowship under Grant No. 2139319. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors(s) and do not necessarily reflect the views of the National Science Foundation or U.S. Office of Naval Research.

References

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. *Advances in neural information processing systems*, 24, 2011.

Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. In *Int. Conf. on Machine Learning*, pages 127–135. PMLR, 2013.

Hyrum S. Anderson and Phil Roth. EMBER: an open dataset for training static PE malware machine learning models. *CoRR*, abs/1804.04637, 2018.

Sean Anderson and João Pedro Hespanha. Learning with contextual information in nonstationary environments: technical report, Nov. 2024. URL https://web.ece.ucsb.edu/~hespanha/techrep.html.

Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of computing*, 8(1):121–164, 2012.

Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Proc. of IEEE 36th annual foundations of computer science*, pages 322–331. IEEE, 1995.

Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multi-armed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002.

Omar Besbes, Yonatan Gur, and Assaf Zeevi. Stochastic multi-armed-bandit problem with non-stationary rewards. *Advances in neural information processing systems*, 27, 2014.

Avrim Blum and Yishay Mansour. Learning, regret minimization, and equilibria. In Noam Nisan, Tim Roughgarden, Éva Tardos, and Vijay V. Vazirani, editors, *Algorithmic Game Theory*, chapter 4. Cambridge University Press, 2007.

Lawrence E Blume. The statistical mechanics of strategic interaction. *Games and economic behavior*, 5(3):387–424, 1993.

Djallel Bouneffouf, Irina Rish, and Charu Aggarwal. Survey on applications of multi-armed and contextual bandits. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2020.

Sébastien Bubeck and Nicolo Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

Wang Chi Cheung, David Simchi-Levi, and Ruihao Zhu. Hedging the drift: Learning to optimize under nonstationarity. *Management Science*, 68(3):1696–1713, 2022.

Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. Contextual bandits with linear payoff functions. In *Proc. of the Fourteenth International Conf. on Artificial Intelligence and Statistics*, pages 208–214. JMLR Workshop and Conf. Proceedings, 2011.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

Dennis C Gilliland. Approximation to bayes risk in sequences of non-finite games. *The Annals of Mathematical Statistics*, 40(2):467–474, 1969.

James Hannan. Approximation to bayes risk in repeated play. *Contributions to the Theory of Games*, 3:97–139, 1957.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

Nick Littlestone and Manfred K Warmuth. The weighted majority algorithm. *Information and computation*, 108(2):212–261, 1994.

Jason R Marden and Jeff S Shamma. Revisiting log-linear learning: Asynchrony, completeness and payoff-based implementation. *Games and Economic Behavior*, 75(2):788–808, 2012.

Gergely Neu and Julia Olkhovskaya. Efficient and robust algorithms for adversarial linear contextual bandits. In Jacob Abernethy and Shivani Agarwal, editors, *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pages 3049–3068. PMLR, 09–12 Jul 2020.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Herbert Robbins. Asymptotically subminimax solutions of compound statistical decision problems. In *Proc. of the second Berkeley symposium on mathematical statistics and probability*, volume 2, pages 131–149. University of California Press, 1951.

Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 55:527–535, 1952.

Tim Roughgarden. *Twenty lectures on algorithmic game theory*. Cambridge University Press, 2016.

Yoan Russac, Claire Vernade, and Olivier Cappé. Weighted linear bandits for non-stationary environments. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

Shai Shalev-Shwartz and Yoram Singer. Convex repeated games and fenchel duality. *Advances in neural information processing systems*, 19, 2006.

Lijun Zhang, Shiyin Lu, and Zhi-Hua Zhou. Adaptive online learning in dynamic environments. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Peng Zhao, Yan-Feng Xie, Lijun Zhang, and Zhi-Hua Zhou. Efficient methods for non-stationary online learning. *Advances in Neural Information Processing Systems*, 35:11573–11585, 2022.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (icml-03)*, pages 928–936, 2003.