Journal for General Philosophy of Science manuscript No.
(will be inserted by the editor)

Adaptive algorithms for meta-induction

N.N.

Received: date / Accepted: date

Abstract Work in online learning traditionally considered induction-friendly (e.g. stochastic with a fixed distribution) and induction-hostile (adversarial) settings separately. While algorithms like Exp3 that have been developed for the adversarial setting are applicable to the stochastic setting as well, the guarantees that can be obtained are usually worse than those that are available for algorithms that are specifically designed for stochastic settings. Only recently, there is an increasing interest in algorithms that give (near-) optimal guarantees with respect to the underlying setting, even in case its nature is unknown to the learner.

In this paper, we review various online learning algorithms that are able to adapt to the hardness of the underlying problem setting. While our focus lies on the application of adaptive algorithms as meta-inductive methods that combine given base methods, concerning theoretical properties we are also interested in guarantees that go beyond a comparison to the best fixed base learner.

Keywords Online learning · Regret · Prediction with expert advice · Multi-armed bandit problem

1 Introduction

The program of meta-induction as introduced by Schurz (2019) is built upon formal results from online learning theory. The goal of this paper is to discuss some of the relevant literature from this field under the perspective of the meta-inductive program. In particular, we want to give pointers to alternative as well as complementary settings, algorithms, and performance measures that we think are worth considering.

Address(es) of author(s) should be given

Most importantly, we want to focus on learning (meta-)algorithms that are particularly adaptive in the sense that they are able to combine (near-) optimal behavior in different settings. These algorithms do not just satisfy a uniform worst case performance guarantee, but are able to adapt to the difficulty of the underlying problem, which is obviously of importance for the program of meta-induction.

2 Which Setting of Induction?

2.1 Schurz' setting of meta-induction

Schurz (2019) considers the following setting of sequence prediction. The goal is to predict for discrete time steps $t = 1, 2, \ldots$ the event e_t of an unknown sequence $e_1, e_2, \ldots$, where it is assumed that events are encoded as real numbers in the closed interval $[0, 1]$. At each time step t the learner selects a prediction p_t taken from $[0, 1]$ and suffers a loss of $\ell(p_t, e_t)$ for a *loss function* $\ell : [0, 1] \times [0, 1] \to [0, 1]$ that intuitively measures the distance between the prediction p_t and the true event e_t.

A straightforward choice to measure the performance of a learner choosing predictions p_t would be the *total loss*

$$L_T := \sum_{t=1}^{T} \ell(p_t, e_t)$$

after T steps. More precisely, one would consider a learner to be successful if the average loss $\frac{1}{T} L_T$ vanishes for $T \to \infty$, that is, $\lim_{T\to\infty} \frac{1}{T} L_T = 0$. Obviously, the considered setting is very general and it is easy to see that for each learning algorithm there will be sequences e_t so that its total loss with respect to a reasonable loss function will be large, that is, it will suffer total loss linear in T (i.e., $L_T \approx c \cdot T$ for a constant c).

There are two ways to make learning possible in the considered setting. The first is to make some assumptions on the generation of the underlying sequence to be predicted. For example one may assume that the sequence is generated by a fixed probability distribution unknown to the learner. This would presume an induction-friendly setting and hence is not suitable if one aims at a general justification of induction.

The alternative is to restrict the hypothesis space by assuming that the learner has access to a set of experts indexed by $1, 2, \ldots, K$. At each step t each expert i produces her own prediction $p_t^{(i)}$. The learner can observe all the experts' predictions as well as the respective losses $\ell_t^{(i)} := \ell(p_t^{(i)}, e_t)$. Moreover, the goal of the learner is more modest. Instead of insisting on achieving sublinear total loss, she is happy to compete with the best fixed expert in hindsight. That is, if $L_T^{(i)} := \sum_{t=1}^{T} \ell_t^{(i)}$ denotes the total loss of expert i after T steps, we are interested in learning algorithms that are able to keep the difference to $\min_i L_T^{(i)}$ small. Accordingly, we aim at algorithms for which

$\lim_{T\to\infty}\frac{1}{T}(L_T - \min_i L_T^{(i)}) = 0$. More generally, we are also interested in the rate with which the loss difference approaches 0 and define the *regret* of a learning algorithm $\mathcal{A}$ after T steps as

$$R_T^{\mathcal{A}} := L_T - \min_i L_T^{(i)}. \tag{1}$$

2.2 Prediction with Expert Advice

The setting introduced in the previous section is a special case of what is known as *prediction with expert advice* (short PEA in the following) in the online learning community. More generally, the events e_t are taken from an outcome space $\mathcal{O}$, and the learner's prediction is more generally coined as a *decision* taken from a decision space $\mathcal{D}$. Accordingly, the loss function $\ell : \mathcal{D} \times \mathcal{O} \to \mathbb{R}$ provides non-negative feedback for each decision-outcome-pair. In some cases this can be interpreted as a distance (as before), or it can be e.g. the cost (or reward) for taking a decision in some state of nature. Apart from these slight generalizations the setting remains the same as before.

2.3 Learning Reductions to PEA

First note that while the considered online prediction setting may seem very special, it can represent seemingly different or more general scenarios as well. Consider e.g. the following online classification problem. The learner receives a sequence of observations $x_1, x_2, \ldots$ one after another. For each observation x_t taken from some space $\mathcal{X}$ the task is to provide a label $y_t \in \mathcal{Y}$. In the simplest case the label is binary, i.e., $\mathcal{Y} = \{0, 1\}$, e.g. indicating some particular property of the respective observation. For example, the x_t could be image data and the label would be 1 if the image shows a horseshoe crab and 0 otherwise. Transferred to the sequence prediction setting the goal of the learner is to predict the next entry in the sequence $x_1, y_1, x_2, y_2, \ldots, x_t$, that is, $\mathcal{D} = \mathcal{Y}$ and $\mathcal{O} = \mathcal{Y} \times \mathcal{X}$.

This latter setting obviously corresponds to scenarios usually associated with induction where one e.g. "concludes" from uniform observations $x_1, x_2, \ldots, x_{t-1}$ all having the same property (i.e., label) y to observation x_t (or more generally all further observations $x_t, x_{t+1}, \ldots$) also having property y.

We note however, that formally it is not straightforward to apply the results and algorithms usually given for the PEA setting to classification problem as specified before. For a more detailed account see e.g. Section 5 of Cesa-Bianchi et al. (1997) or the application of the Hedge algorithm of Freund and Schapire (1997) to classification problems in form of the influential *Boosting* algorithm (see also Section 3.3.2 below). For more on reductions of different learning settings see e.g. (Langford and Zadrozny, 2005).

In the course of this paper we will also consider different feedback models like the multi-armed bandit problem as well as performance measures that are alternatives to the notion of regret defined in (1). In the remainder of this section however we want to discuss some more technical choices to be made in the meta-inductive PEA setting.

2.4 Encoding of events

For technical reasons the decision space in the PEA setting is often assumed to be a convex subset of some vector space. In particular, many algorithms such as the exponential forecasting algorithm EAW suggested by Schurz (2019) predict a convex combination of the predictions of the single experts. As in the setting of Schurz (2019) predictions as well as events are assumed to be in $[0, 1]$ that way one always obtains a valid prediction in $[0, 1]$.

However, if the events are encodings of instances taken from a different domain it is by no means guaranteed that the convex combination of the experts' predictions will result in a number that corresponds to an instance of the original instance set. This would need additional assumptions like that the encoding is surjective, i.e., any number in $[0, 1]$ is an encoding of an element of the original instance set. In many natural cases such an assumption will not hold, for example, if the the original instance set is countable. We refer to Section 3.2 for a more detailed discussion of such discrete settings.

2.5 The choice of the loss function

Concerning the loss function, also depending on the specific setting some choices may appear more natural than others. In general, the loss function will have an effect on the performance of a prediction algorithm and hence also on the choice of the preferred algorithm. For example, the main result[1] for the EAW algorithm suggested by Schurz (2019) rests on the assumption that the used loss function is convex in the first argument. For particular loss functions improved performance guarantees can be given, cf. Chapter 3 of (Cesa-Bianchi and Lugosi, 2006) and the respective Proposition 6.7 for exp-concave loss functions in (Schurz, 2019).

In particular for the application to meta-induction, the choice of the loss function may be not just a convention. Rather a whole spectrum of induction problems should be covered, so that one has to put some effort in the translation of the particular problem instances into the proposed setting of meta-induction. Thus, beside the problem of encoding these instances by numbers in $[0, 1]$, finding a suitable loss function that defines a natural distance between the original instances is an additional challenge. This is basically a (metric) multi-dimensional scaling problem for which several techniques have been developed (Borg and Groenen, 2005).

[1] Theorem 6.9 of (Schurz, 2019) on p. 145, cf. also Theorem 1 below.

An interesting aspect of the PEA setting is that the underlying sequence actually vanishes behind the loss sequences produced by each expert. Accordingly, it may be natural to eliminate this aspect of the setting by assuming that one directly observes loss sequences that can be interpreted as real rewards obtained by the learner, for which a comparison is more straightforward. In this setting the learner does not predict an underlying sequence any more, but only chooses an expert (or more generally a distribution over experts that can be interpreted as randomization). Such a setting is e.g. considered by Cesa-Bianchi et al. (2007) (cf. Section 3.3.3 below and also the bandit setting discussed in Section 4).

2.6 Translation and scaling of events and losses

Closely related to the choice of the loss function is the question of translation or scaling of losses or events. In general, losses can be arbitrary real numbers. However, the range of the loss function will obviously have an effect on the achievable regret, so that it may seem preferable to normalize losses on some closed interval like $[0,1]$. Indeed, in the general PEA setting the sets $\mathcal{O}$ and $\mathcal{D}$ are known, so that it is easy to define a loss function that maps to $[0,1]$. However, as discussed in detail by Cesa-Bianchi et al. (2007) (cf. also Cesa-Bianchi and Lugosi, 2006, Section 2.6) the respective transformation may harm the respective regret bounds that can be achieved. For the discussion of algorithms that are invariant with respect to natural transformations such as translation and scalings we refer to the discussion in Section 3.3.3.

Note that in many practical settings the range of the sequence to be predicted will not be known in advance, so that it is difficult to specify a transformation of the actual events to a bounded interval like $[0,1]$. However, clever techniques can deal also with unknown ranges, see e.g. variants of the Prod algorithm of Cesa-Bianchi et al. (2007) discussed in Section 3.3.3 below.

3 Adaptive Algorithms

3.1 The exponentially weighted average forecaster

The algorithm favored by Schurz (2019) is the *exponentially weighted average forecaster* of Cesa-Bianchi and Lugosi (2006) (called EAW for *exponential attractivity weighted* forecaster by Schurz, 2019). As several other algorithms some of which we will discuss below this algorithm works with weights assigned to the experts that are used to choose a prediction and that change depending on the expert's success. For the algorithm at hand the initial weights $w_1^{(i)}$ at time $t=1$ are set to 1 for each expert i. The update at any step t is done according to

$$w_t^{(i)} := w_t^{(i)} \exp(-\eta \ell_t^{(i)})/N \tag{2}$$

for a parameter η and a factor N that normalizes weights to sum up to 1. The prediction for step t is then given by $\sum_i w_t^{(i)} p_t^{(i)}$. One can show that the regret after any T steps is[2] $O(\sqrt{T} \log K)$. The following result of Cesa-Bianchi and Lugosi (2006) basically corresponds to (6.7) of Schurz (2019).

Theorem 1 (Theorem 2.2 of Cesa-Bianchi and Lugosi, 2006) *After any T steps the regret of the exponentially weighted average forecaster choosing $\eta = \sqrt{8(\log K)/T}$ is bounded by $\sqrt{T(\log K)/2}$, provided that the loss function ℓ is convex in its first argument.*

It can be shown that this bound is in general not improvable, cf. Section 3.7 of Cesa-Bianchi and Lugosi (2006). Choosing time-dependent learning rates $\eta_t := \sqrt{8(\log T)/t}$ one can obtain similar results without knowing the horizon T in advance, cf. Theorem 2.3 of Cesa-Bianchi and Lugosi (2006) and the corresponding Theorem 6.9 of Schurz (2019).

3.2 The weighted majority algorithm and mistake bounds

As discussed in Section 6.7 of (Schurz, 2019), the original PEA setting has its limitations. Most importantly, for the very natural settings in which the outcome space $\mathcal{O}$ and the decision space $\mathcal{D}$ coincide and are discrete, a (convex) combination of expert predictions as usually suggested by weighted forecasting algorithms does not give a valid prediction in general. Still the situation is not completely hopeless when interpreting the weights of the experts as probabilities of a randomized approach.

While randomization is a powerful tool, having access to independent random variables may be considered as a (too) strong assumption. Schurz (2019) suggests to simulate randomization via a collective of experts approximating the probability distribution over the possible predictions (cf. Section 6.7.2 of Schurz, 2019). However, in some simple cases one can avoid randomization using a simple algorithm also employing exponential weights to decide according to the *weighted majority*.

The most basic form of the *weighted majority* algorithm has been suggested by Littlestone and Warmuth (1994) for binary prediction problems. That is, outcome space $\mathcal{O}$ and decision space $\mathcal{D}$ are binary, i.e., $\mathcal{O} = \mathcal{D} = \{0, 1\}$, and the loss function is defined by $\ell(d, o) = |d - o|$ for $d \in \mathcal{D}$ and $o \in \mathcal{O}$.

The weighted majority algorithm works as follows. Initializing weights $w_1^{(i)} = 1$ for each expert i, in each time step one sums up over the experts that predict 0 and compares this total weight to the respective sum over all experts predicting 1. The group of experts with larger total weight decides on the prediction at the current step. After observing the true outcome all experts' weights that predicted wrongly are discounted by a factor η with $0 < \eta < 1$.

[2] Here and in the following we use the Landau notation $O(\cdot)$ to denote asymptotic upper bounds on the regret.

This can be interpreted as an update rule of the form

$$w_t^{(i)} := w_{t-1}^{(i)} \eta^{\ell_t^{(i)}} \tag{3}$$

for ℓ as defined before.

Note that this algorithm is not only very simple, it also does not need any randomization. The analysis of the algorithm is as simple as the algorithm itself and yields the following upper bound on the number of wrong predictions of the algorithm.

Theorem 2 (Littlestone and Warmuth, 1994) *The number m of mistakes made by the weighted majority algorithm is bounded according to*

$$m \leq \frac{\log(K) + m_i \log\left(\frac{1}{\eta}\right)}{\log\left(\frac{2}{1+\eta}\right)}, \tag{4}$$

where m_i is the number of mistakes made by expert i.

Note that unlike the regret bounds we have seen so far this *mistake bound* is independent of the horizon. Actually, setting $\eta := \sqrt{(\log K)/T}$ it is easy to obtain from (4) a regret bound of optimal order $O(\sqrt{T \log K})$ after T steps similar to Theorem 1.

The weighted majority algorithm can be generalized to non-binary prediction settings resulting in an algorithm similar to the exponentially weighted average forecaster algorithm. We will discuss the arising Hedge algorithm and its variants in more detail in the following section. It will turn out that the weighted majority algorithm and its relatives can be used directly or with slight modifications to obtain adaptivity results that are stronger than the regret bounds we have seen so far.

3.3 Adaptive algorithms based on weighted majority

In some sense, one could say that regret bounds for algorithms like the exponentially weighted average forecaster or the weighted majority algorithm actually are not so much about learning or inductive inference as about keeping track of the current best expert. Indeed, as already mentioned, the underlying sequence to be predicted in the PEA setting actually vanishes and what matters in the end are the loss sequences of the experts. Also with respect to the framework of meta-induction it would be desirable to have algorithms that are not only adaptive in the sense that they minimize the loss with respect to the best expert, but are able to perform better on sequences that are *easy* to predict. As in the PEA setting the experts serve as a basis for prediction, this is obviously a challenging task.

3.3.1 I.i.d. losses

As just noted, what matters in the PEA setting are the loss sequences of the experts. So when talking about *easy* sequences we not necessarily mean the underlying sequence to be predicted but rather the loss sequences of the experts. Then a particularly easy setting is one where these loss sequences are stochastic and i.i.d. We emphasize once more that this not necessarily means that the underlying observations are stochastic and i.i.d. It is only guaranteed that the interplay between observations and experts results in stochastic and i.i.d. losses. There are several ways how this can happen, for example, when the experts predict the observations perfectly only with some i.i.d. noise added.

In this particular setting a simple algorithm like *Follow the Leader* (FTL, called *Imitate the Best* by Schurz, 2019) that chooses the best expert so far works perfectly and indeed achieves constant regret. That is, the regret after any T steps will be bounded by a constant that is independent of T, cf. (de Rooij et al., 2014) for details. Importantly, a lot of common PEA algorithms with known regret guarantees perform worse than FTL on i.i.d. loss sequences. On the other hand, for certain loss sequences FTL is known suffer linear regret, see de Rooij et al. (2014).

3.3.2 Hedge

The question whether it is possible to combine FTL with other algorithms to guarantee good performance on easy sequences but still overall regret bounds of order $O(\sqrt{T \log K})$ has sparked some interesting research, starting with the *Hedge* strategy of Freund and Schapire (1997, 1999).

The *Hedge* algorithm of Freund and Schapire (1997, 1999) is a generalization of the weighted majority algorithm using the same weight updates as given in (3) but for arbitrary loss functions ℓ. More precisely, the learner at each step t chooses a probability distribution $(\bar{w}_t^{(1)}, \bar{w}_t^{(2)}, \ldots, \bar{w}_t^{(K)})$ over the experts[3] and suffers a loss of $\sum_i \bar{w}_t^{(i)} \ell_t^{(i)}$. The regret of this generalized algorithm can be shown to be of order $O(\sqrt{T \log K})$ just as for the weighted majority algorithm. Freund and Schapire (1997) also present an application to classification problems in form of the seminal *Boosting* algorithm that enables to combine several weak learners (that perform just a bit better than random) to achieve high overall performance.

3.3.3 Prod

The Prod algorithm of Cesa-Bianchi et al. (2007) has been proposed for a more general setting in that the sign of the losses can be arbitrary. As before, the learner chooses at each step t a probability distribution $(\bar{w}_t^{(1)}, \bar{w}_t^{(2)}, \ldots, \bar{w}_t^{(K)})$ over the experts and receives a payoff of $\sum_i \bar{w}_t^{(i)} x_t^{(i)}$, where $x_t^{(i)}$ is the payoff of expert i. Here payoffs are taken from $[-M, M]$, that

[3] Hedge simply normalizes the computed weights $w_t^{(i)}$ to obtain the probabilities $\bar{w}_t^{(i)}$.

is, they can be positive (payoffs) or negative (losses). The PROD algorithm updates the weights for each expert i according to

$$w_t^{(i)} := w_{t-1}^{(i)}(1 + \eta x_t^{(i)}) \tag{5}$$

for a a parameter η, initializing weights to 1 as usual.

Whereas the regret bounds we have seen so far depend on the horizon T as well as on the range of the payoffs,[4] Cesa-Bianchi et al. (2007) provide what they call *higher order* bounds that depend on the sum of the payoffs (resulting in *first order* bounds) or the squared sums of the payoffs (giving *second order* bounds). For example, defining $Q_T^* := \sum_{t=1}^T (x_t^{(i^*)})^2$ to be the sum of the squares of the best[5] arm's payoff, they show the following bound on the regret of PROD.

Theorem 3 (Cesa-Bianchi et al., 2007) *The regret of* PROD *with parameter* $\eta := \min\{1/(2M), \sqrt{(\log K)/Q^*}\}$ *is upper bounded by*

$$\max\left\{2\sqrt{Q_T^* \log K}, 4M \log K\right\}. \tag{6}$$

Obviously, Q_T^* is upper bounded by TM^2, so that the second order bound of Theorem 3 improves over the bounds we have seen so far whenever Q_T^* is smaller than TM^2. While Theorem 3 assumes knowledge of the quantities Q_T^* and M, Cesa-Bianchi et al. (2007) also provide adaptations of the basic PROD algorithm to deal with the usual case when neither Q_T^* nor M are known. Further, second-order bounds for the weighted majority algorithm using a time-dependent learning rate are given as well.

Another important contribution of Cesa-Bianchi et al. (2007) is that they also provide bounds that are *stable* under certain natural transformations like additive translations and rescalings of the payoffs. It is also discussed how given (unstable) regret bounds can be improved by a meta-algorithm using artificial translations of the payoffs, see Section 5 of (Cesa-Bianchi et al., 2007).

This is taken a step further by de Rooij et al. (2014) whose algorithms *AdaHedge* and *FlipFlop* are themselves translation invariant. That is, not only the regret bounds that hold for these algorithms are invariant under translation, the algorithms do not change their behavior if payoffs (or losses) are shifted or rescaled. As these algorithms are a bit more involved than the simple weight update algorithms we consider here, we refer to (de Rooij et al., 2014) for details.

[4] The latter dependence has not been made explicit so far, as we have assumed that losses are taken from $[0, 1]$. Obviously, when taking losses from a larger interval $[0, M]$, the respective regret guarantees scale with an additional factor of M.

[5] The best arm i^* here is the best arm after T steps, which in general depends on T. For the sake of readability we skipped this dependence from the notation.

3.3.4 D-Prod

Based on Prod, Even-Dar et al. (2008) suggested the D-Prod algorithm that aims to compete not only against the best expert but also against an arbitrary given distribution D over the experts. The update of the weights is defined by

$$w_t^{(i)} := w_{t-1}^{(i)}\big(1 + \eta(x_t^{(i)} - x_t^{(D)})\big), \tag{7}$$

where $x_t^{(D)}$ denotes the payoff of the distribution D over the experts. This distribution is also assigned a constant weight according to (7). Using a proper initialization of weights one can show that the regret of D-Prod with respect to the distribution D over the experts is constant, while the regret with respect to the best expert in hindsight is not much worse than for the algorithms we have seen so far.

Theorem 4 (Even-Dar et al., 2008) *The regret of D-Prod with parameter $\eta := \sqrt{(\log K)/T}$ and initial weights $w_1^{(i)} := \eta/K$ and $w_1^{(D)} := 1 - \eta$ is upper bounded by*

$$O\left(\sqrt{T\log K} + \sqrt{\frac{T}{\log K}}\log T\right). \tag{8}$$

Further, the regret with respect to the distribution D over the experts is constant.

This result is also applicable to the case where one of the experts performs well in some particular settings (e.g., i.i.d. payoffs). If D concentrates all probability mass to this particular expert, D-Prod is guaranteed to perform basically as good as this expert in the specified settings.

The background of the development of D-Prod is a trade-off between the performance with respect to the *best* expert and the regret with respect to the *average* over all experts. Indeed, so-called *difference algorithms* whose experts' weights in the two experts case (i.e., $K = 2$) only depend on the difference of the experts' losses can be shown to cannot have small regret in both cases. More precisely, the product of the two regret bounds is at least linear in T as shown in Section 3 of Even-Dar et al. (2008). For example, a difference algorithm having regret of order $O(\log T)$ in one setting cannot have anything better than $O(T/\log T)$ regret in the other setting. We note that all algorithms we have considered so far are difference algorithms. Unlike that, D-Prod is no difference algorithm (as it considers the performance of the special D-expert in the weight updates) and can escape this lower bound. Even-Dar et al. (2008) also provide some general lower bounds that hold for all algorithms showing that any algorithm with regret of order $O(\sqrt{T})$ with respect to the best expert must have regret of at least order $\sqrt{T}$ with respect to a fixed distribution D over experts. Accordingly, it is not possible to improve the regret bound with respect to the best expert of Theorem 4 to $O(\sqrt{T})$ without losing the constant regret with respect to the distribution D over the experts.

3.3.5 $(\mathcal{A}, \mathcal{B})$-Prod

The $(\mathcal{A}, \mathcal{B})$-Prod algorithm suggested by Sani et al. (2014) is a meta-algorithm that combines two learning algorithms $\mathcal{A}$ and $\mathcal{B}$, where $\mathcal{A}$ is supposed to be a general purpose algorithm with regret guarantees in every scenario, while $\mathcal{B}$ is a baseline algorithm, similar to the distribution over experts for D-Prod. Working on the meta-level, $(\mathcal{A}, \mathcal{B})$-Prod attributes weights $w_t^{(\mathcal{A})}$, $w_t^{(\mathcal{B})}$ to the base algorithms using an update as for D-Prod, that is, weights for $\mathcal{B}$ remain constant, while weights for $\mathcal{A}$ are updated according to

$$w_t^{(\mathcal{A})} := w_{t-1}^{(\mathcal{A})}\big(1 + \eta(\ell_t^{(\mathcal{B})} - \ell_t^{(\mathcal{A})})\big), \tag{9}$$

where $\ell_t^{(\mathcal{A})}$ and $\ell_t^{(\mathcal{B})}$ denote the loss of algorithm $\mathcal{A}$ and $\mathcal{B}$, respectively.

Similar to the guarantees for D-Prod, the regret of $(\mathcal{A}, \mathcal{B})$-Prod with respect to $\mathcal{B}$ is constant, while the regret with respect to $\mathcal{A}$ is of order $O(\sqrt{T \log T})$.

Theorem 5 (Sani et al., 2014) $(\mathcal{A}, \mathcal{B})$-Prod *applied with parameter* $\eta := \sqrt{(\log T)/(4T)}$ *and initial weights* $w_1^{(\mathcal{A})} := \eta$ *and* $w_1^{(\mathcal{B})} := 1 - \eta$ *has regret upper bounded by* $2\sqrt{T \log T}$ *with respect to algorithm* $\mathcal{A}$*, while the regret with respect to* $\mathcal{B}$ *is bounded by the constant* $2 \log 2$*.*

Using e.g. an algorithm like FTL for $\mathcal{B}$ one obtains constant regret with respect to FTL, which means that for any easy case where FTL works well, $(\mathcal{A}, \mathcal{B})$-Prod will have similarly good performance. On the other hand, choosing for $\mathcal{A}$ a safe algorithm with $O(\sqrt{T \log K})$ regret guarantees will safeguard the algorithm against sequences where FTL performs poorly. The additional regret of order $O(\sqrt{T \log T})$ is the price to pay for the good performance in easy settings.

Once more, we refer to the algorithm FlipFlop of de Rooij et al. (2014) that combines FTL with AdaHedge (a version of Hedge with time dependent learning rate) and whose regret can be shown to be upper bounded (apart from a constant factor) both by the regret of FTL and AdaHedge. For a more detailed discussion of the subtle differences between the respective performance guarantees for FlipFlop, D-Prod, and $(\mathcal{A}, \mathcal{B})$-Prod we refer to Section 3.1 of Sani et al. (2014).

4 Best of Both Worlds Algorithms

4.1 Limited feedback – the bandit setting

The standard PEA setting assumes that the learner observes feedback for all the experts' predictions. However, there are several alternative settings with limited feedback, the most well-known of which is the multi-armed bandit problem that is also briefly discussed in Section 7.5 of (Schurz, 2019). The bandit setting is usually described rather in the form of a decision problem.

That is, the learner at time steps $t = 1, 2, \ldots$ chooses one element from a given set of arms indexed by $1, 2, \ldots, K$ (corresponding to the experts in the PEA setting). After each step she obtains (and observes) a reward for the chosen arm, and there is no information available about the rewards of the arms not chosen.

Note that the multi-armed bandit problem can be translated back in a prediction setting by considering that the learner at each step chooses to follow (the prediction of) an expert (or arm respectively) and observes the success of this expert's prediction. We have already mentioned that in the original PEA setting the underlying sequence to be predicted does not matter anymore as soon as a loss function is fixed and the goal is to minimize total regret. Accordingly, while in the bandit setting there is usually no underlying ground truth (as a sequence to be predicted) assumed, apart from the limited feedback this does not make any principled difference to the PEA setting.

Note that while in the full feedback setting of PEA the FTL algorithm is a reasonable choice, it is easy to see that in the bandit setting it will not work well in general. Here the so-called *exploration-exploration dilemma* becomes a challenge for the learner. She has to decide at each step whether to exploit her current knowledge and keep with the best arm so far, or to explore other arms that behaved suboptimally in past samples.

Research on bandit problems for a long time considered two different and well-separated settings, the *stochastic* and the *nonstochastic* multi-armed bandit problem. In both settings it is assumed that rewards are bounded, and in the following we assume them to be contained in $[0, 1]$.

4.2 Stochastic bandits

In the *stochastic* multi-armed bandit problem it is assumed that the rewards of each arm i come from an unknown probability distribution over some bounded interval with mean μ_i. The learner does not know the reward distributions but is aware of operating in a stochastic setting. She competes against the optimal mean reward $\mu^* := \max_i \mu_i$ and the respective measure called *pseudo-regret* is defined as

$$T\mu^* - \sum_{t=1}^{T} \mu_{I_t}, \tag{10}$$

where T is the considered time horizon and I_t is the arm chosen at step t.

As already mentioned the greedy FTL algorithm, that after initially sampling each arm once chooses at each further step t the arm

$$I_t := \arg\max_i \hat{r}_i \tag{11}$$

that maximizes the observed average reward $\hat{r}$ so far, is a risky choice. Indeed, the optimal arm may underperform in the first few samples taken from it, and will then not be chosen again, leading to regret linear in T. However, slight modifications work well in practice and in theory. For example, adding a bit of randomization by choosing a random arm with some small probability ε and following (11) with probability $1 - \varepsilon$ is sufficient to obtain a well working algorithm. While for fixed ε the regret remains linear, if ε is chosen to decrease over time one can derive strong performance bounds (Auer et al., 2002a). Note that the added randomization is dealing with the exploration-exploitation dilemma in a very direct way with ε being the fraction of exploration the algorithm conducts.

A class of simple and popular algorithms that deal with the exploration-exploitation problem implicitly add a bonus term to the average reward $\hat{r}_i$ of each arm i and choose similar to (11)

$$I_t := \arg\max_i \hat{r}_i + b(i,t), \tag{12}$$

where the bonus term $b(i,t)$ usually depends on the respective arm i and the current time step t. Often the bonus term corresponds to a confidence interval for the respective empirical mean $\hat{r}_i$, so that with high probability each value $\hat{r}_i + b(i,t)$ is larger than the true mean μ_i. Consequently, for the chosen arm at step t the value $\hat{r}_i + b(i,t) > \mu^*$. As the confidence intervals shrink with the number of samples taken from each arm, after not too many steps with high probability the chosen arm will be the optimal one. However, the bonus term usually also increases slowly with t to guarantee the necessary amount of exploration, so that the algorithm keeps sampling suboptimal arms every now and then. The most well-known bandit algorithm in the stochastic setting is Ucb (Auer et al., 2002a) which chooses

$$I_t := \arg\max_i \hat{r}_i + \sqrt{\frac{2\log t}{n_i}}, \tag{13}$$

where n_i is the number of times arm i has been chosen so far. The bonus term corresponds to a confidence interval[6] that shrinks with the number of samples n_i but also slowly increases with t.

Theorem 6 (Auer et al., 2002a) *The pseudo-regret of* Ucb *is bounded by*

$$\sum_{i:\mu_i<\mu^*} \left(\frac{8\log T}{\mu^* - \mu_i} + 1 \right).$$

While this bound can still be slightly improved to match a corresponding lower bound (Lattimore, 2015), the logarithmic dependence on T cannot be avoided. Note that the bound of Theorem 6 is *problem dependent*, that is, it depends on the gaps $\mu^* - \mu_i$ between the average reward of an optimal and any suboptimal arm. The worst case bound applies to cases where these gaps are of order $\frac{1}{\sqrt{T}}$, when the bound basically becomes $O(\sqrt{KT\log T})$. This *problem independent* bound can be improved to $O(\sqrt{KT})$ as shown by Audibert and Bubeck (2010). This corresponds to the bounds that can be achieved in the nonstochastic setting considered below.

[6] More precisely to an *upper confidence bound*, therefore the name.

4.2.1 Excursion: Simple regret

An alternative performance measure for the stochastic bandit setting with different flavor (sometimes called *pure exploration*) is the following. Instead of maximizing the sum over all rewards, the learner after a certain number of time steps (potentially not known to the learner in advance) has to recommend an arm she considers to be best. So-called *simple regret* measures the distance between the average reward of this recommended and the true optimal arm (Bubeck et al., 2011).

A scientist will probably not be evaluated with respect to regret over all her hypotheses advanced in her career on a particular subject. Rather, one would be interested in her latest hypotheses. Accordingly, for this scenario simple regret appears to be a more suitable measure than the total regret considered so far. Note however that this measure is only sensible under the assumption of working in a stochastic setting.

4.3 Nonstochastic bandits

The second bandit setting considered in the literature is the *nonstochastic* (sometimes also called *adversarial*) setting. Here there are no assumptions on the rewards, except that they are bounded and fixed in advance (i.e., do not change with respect to the choice of the learner). Here the goal is to compete against the best arm in hindsight, that is, writing $r_t^{(i)}$ for the reward of arm i at step t the regret after T steps is defined as

$$\max_i \sum_{t=1}^{T} r_t^{(i)} - \sum_{t=1}^{T} r_t^{(I_t)}. \tag{14}$$

Several algorithms have been suggested in this setting. One of the first was Exp3 (Auer et al., 2002b) that uses exponential weights similar to the algorithms we have seen in the PEA setting. More precisely, initializing weights $w_1^{(i)}$ associated with each arm i to 1, EXP3 with parameter η computes probabilities

$$p_t^{(i)} := (1-\eta)\frac{w_t^{(i)}}{\sum_i w_t^{(i)}} + \frac{\eta}{K},$$

according to which the arm in step t is chosen. Afterwards, weights are updated for the chosen arm i taking into account the observed reward $r_t^{(i)}$ according to

$$w_{t+1}^{(i)} := w_t^{(i)} \exp(\eta r_t^{(i)}/p_t^{(i)}), \tag{15}$$

while for all other arms $w_{t+1}^{(i)} := w_t^{(i)}$.

Although compared to the PEA setting the learner receives much less information, regret bounds with respect to the best fixed arm in hindsight are still of order $O(\sqrt{T})$, only the dependence on the number of arms K is slightly worse.

Theorem 7 (Auer et al., 2002b) *The expected regret of* Exp3 *using parameter* $\eta = \min\left\{1, \sqrt{\frac{K \log K}{(e-1)T}}\right\}$ *is bounded by*

$$2\sqrt{(e-1)KT \log K}.$$

While the parameter setting in Theorem 7 assumes knowledge of T the variant Exp3.1 uses time dependent parameters η_t and achieves the same regret bounds with only a slightly larger constant (Auer et al., 2002b). Auer et al. (2002b) also give lower bounds that show that any algorithm will suffer regret at least of order $\sqrt{KT}$. The gap between this lower bound and the upper bound of Theorem 7 can be closed. That is, there are algorithms that get rid of the logarithmic factor in the bound of Theorem 7, see (Audibert and Bubeck, 2010) for further details. Note that the regret of order $O(\sqrt{KT})$ corresponds to the regret bounds that are not problem dependent in the stochastic setting.

4.4 Best of both worlds algorithms

Only recently, algorithms have been proposed whose goal is achieve *the best of both worlds*, that is, the best possible bounds in the stochastic as well as the nonstochastic setting. This corresponds to the question of adaptive algorithms in the PEA setting that we have considered in Section 3. Note that any algorithm for the nonstochastic setting of course can be deployed in the stochastic setting as well, giving regret bounds of order $O(\sqrt{KT})$ specified in the previous section. However, if one wants to obtain logarithmic bounds in stochastic settings (cf. Section 4.2), one has to apply more sophisticated algorithms that e.g. constantly check whether they are in a stochastic domain. Such algorithms that are robust in an induction-hostile nonstochastic setting on the one hand but on the other hand manage to exploit the induction-friendly stochastic setting have been first proposed by Bubeck and Slivkins (2012). The regret bounds that hold for their Sao algorithm are summarized in the following theorem.

Theorem 8 (Bubeck and Slivkins, 2012) *The pseudo-regret of* Sao *in stochastic settings is upper bounded by*

$$O\left(K \frac{\log^2 T}{\Delta} \log K\right),$$

where $\Delta := \mu^* - \max_{i:\mu_i<\mu^*} \mu_i$ *is the gap between best and second best mean reward.*

The expected regret of Sao *in adversarial settings is upper bounded by*

$$O\left(\sqrt{KT \log^{\frac{3}{2}}(T) \log K}\right).$$

Comparing the bounds of Theorem 8 to those of Theorems 6 and 7, it can be seen that the former do not quite achieve the best possible bounds in both settings. Indeed while the bound on the expected regret in the adversarial setting has an additional polylogarithmic factor in T, the bound for the stochastic setting is also polylogarithmic in T.

These bounds have been complemented by Auer and Chiang (2016) whose Sapo algorithm can be shown to be best possible in the nonstochastic setting and have near-optimal *pseudo-regret*[7] in the adversarial setting.

Theorem 9 (Auer and Chiang, 2016) *The pseudo-regret of* Sapo *in stochastic settings is upper bounded by*

$$O\left(\sum_i \frac{\log T}{\mu^* - \mu_i}\right).$$

Its pseudo-regret in adversarial settings is upper bounded by

$$O(\sqrt{TK \log T}).$$

Auer and Chiang (2016) also show that no algorithm is able to combine the pseudo-regret guarantees of Theorem 6 in the stochastic setting with an upper bound on the expected regret of $\sqrt{KT}$ in the adversarial setting. We note that similar results are available for the pure exploration setting of Section 4.2.1 (Abbasi-Yadkori et al., 2018).

While the algorithms Sao and Sapo are explicitly defined to work well in stochastic and adversarial environments, similarly to some of the approaches we have seen for the PEA setting in Section 3, Agarwal et al. (2017) have proposed a meta-algorithm called *Corral* that combines various bandit algorithms. Obviously, due to the limited feedback making combinations of bandit algorithms work is a bigger challenge than in the PEA setting. Still, under some stability assumptions on the base algorithms one can show regret bounds for Corral that depend on the regret of the best base algorithm as well as on the number of base algorithms. It should be noted however that in order to take full advantage of the combination power of Corral it is necessary to know in advance the regret guarantee of the best base algorithm in the application scenario. For a detailed discussion and various applications to more general decision problems we refer to (Agarwal et al., 2017).

5 Changing Environments

The original PEA setting as well as the limited feedback nonstochastic bandit setting do not make any assumptions on the sequence to be predicted or the reward (or loss) sequences of the underlying arms, respectively. Accordingly, they already subsume changing environments. However, the regret usually

[7] This is a slightly weaker notion than that of expected regret, cf. e.g. Section 1 of Bubeck and Cesa-Bianchi (2012) for details.

is considered with respect to the best fixed arm in hindsight, whereas any adaptive algorithm would change between arms. As has already been observed by Bubeck and Cesa-Bianchi (2012) the notion of (pseudo-)regret in the case of nonstationary reward sequences can become practically vacuous e.g. if the total reward of any fixed arm is small. Accordingly, one can try to compete against the best way to follow an arm for a certain time period and is allowed to change arms at most S times in total. Auer et al. (2002b) have considered this problem and suggested a variant of Exp3 called Exp3.S that achieves regret of order $O(\sqrt{SKT\log(KT)}$ using S to tune the algorithm.

Similarly, in the stochastic setting, it is easy to define a changing environment where the arms' distributions change at certain time steps. With the number of total changes S known in advance, one can simply apply the Exp3.S algorithm to achieve the mentioned regret guarantees. Only recently, an algorithm was suggested by Auer et al. (2019) that similar to the *best of both worlds* algorithms discussed in Section 4.4 tries to detect changes by itself and does not need to know the number of changes in advance. That way, the algorithm can be shown to achieve regret upper bounded by a term of order $O(\sqrt{SKT\log T})$. We refer to (Warmuth and Koolen, 2014) for more on so-called *shifting experts* and some related open problems.

We note that here we only mentioned settings with bandit feedback. For the respective problem in the PEA setting we refer to Chapter 5 of (Cesa-Bianchi and Lugosi, 2006) as well as Sections 7.3 and 9.2.5 of (Schurz, 2019).

6 Conclusion

While we naturally concentrated on sequence prediction models and the related problem settings like the multi-armed bandit problem, the mathematics literature is full of online settings where the goal is to make good decisions with having access to only part of the problem that becomes known only step by step. Typical settings include allocation, matching, scheduling, or packing to name a few. For an overview of techniques and typical results see e.g. Borodin and El-Yaniv (1998). It is notable that the weighted majority method is applicable to a wide range of some of these online problems and has been used and reinvented on several occasions, see (Arora et al., 2012) for an overview.

Acknowledgements This work has been supported by the Austrian Science Fund (FWF): I 3437-N33 in the framework of the CHIST-ERA ERA-NET (DELTA project).

References

Yasin Abbasi-Yadkori, Peter L. Bartlett, Victor Gabillon, Alan Malek, and Michal Valko. Best of both worlds: Stochastic & adversarial best-arm identification. In *Conference On Learning Theory, COLT 2018*, volume 75 of *Proceedings of Machine Learning Research*, pages 918–949. PMLR, 2018.

Alekh Agarwal, Haipeng Luo, Behnam Neyshabur, and Robert E. Schapire. Corralling a band of bandit algorithms. In *Proceedings of the 30th Conference on Learning Theory, COLT 2017*, volume 65 of *Proceedings of Machine Learning Research*, pages 12–38. PMLR, 2017.

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

Jean-Yves Audibert and Sébastien Bubeck. Regret bounds and minimax policies under partial monitoring. *J. Mach. Learn. Res.*, 11:2785–2836, 2010.

Peter Auer and Chao-Kai Chiang. An algorithm with nearly optimal pseudo-regret for both stochastic and adversarial bandits. In *Proceedings of the 29th Conference on Learning Theory, COLT 2016*, volume 49 of *JMLR Workshop and Conference Proceedings*, pages 116–120, 2016.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*, 47(2-3):235–256, 2002a.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, 2002b.

Peter Auer, Pratik Gajane, and Ronald Ortner. Adaptively tracking the best bandit arm with an unknown number of distribution changes. In *Conference on Learning Theory, COLT 2019, 25-28 June 2019*, volume 99 of *Proceedings of Machine Learning Research*, pages 138–158. PMLR, 2019.

Ingwer Borg and Patrick J.F. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer, 2005.

Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.

Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Found. Trends Mach. Learn.*, 5(1):1–122, 2012.

Sébastien Bubeck and Aleksandrs Slivkins. The best of both worlds: Stochastic and adversarial bandits. In *COLT 2012 - The 25th Annual Conference on Learning Theory*, volume 23 of *JMLR Proceedings*, pages 42.1–42.23, 2012.

Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. Pure exploration in finitely-armed and continuous-armed bandits. *Theor. Comput. Sci.*, 412(19):1832–1852, 2011.

Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

Nicolò Cesa-Bianchi, Yoav Freund, David Haussler, David P. Helmbold, Robert E. Schapire, and Manfred K. Warmuth. How to use expert advice. *J. ACM*, 44(3):427–485, 1997.

Nicolò Cesa-Bianchi, Yishay Mansour, and Gilles Stoltz. Improved second-order bounds for prediction with expert advice. *Mach. Learn.*, 66(2-3): 321–352, 2007.

Steven de Rooij, Tim van Erven, Peter D. Grünwald, and Wouter M. Koolen. Follow the leader if you can, hedge if you must. *J. Mach. Learn. Res.*, 15 (1):1281–1316, 2014.

Eyal Even-Dar, Michael J. Kearns, Yishay Mansour, and Jennifer Wortman. Regret to the best vs. regret to the average. *Mach. Learn.*, 72(1-2):21–37, 2008.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55 (1):119–139, 1997.

Yoav Freund and Robert E. Schapire. Adaptive game playing using multiplicative weights. *Games and Economic Behavior*, 29(1–2):79–103, 1999.

John Langford and Bianca Zadrozny. Relating reinforcement learning performance to classification performance. In *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005)*, volume 119 of *ACM International Conference Proceeding Series*, pages 473–480. ACM, 2005.

Tor Lattimore. Optimally confident UCB : Improved regret for finite-armed bandits. *CoRR*, abs/1507.07880, 2015. URL `http://arxiv.org/abs/1507.07880`.

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

Amir Sani, Gergely Neu, and Alessandro Lazaric. Exploiting easy data in online optimization. In *Advances in Neural Information Processing Systems 27, NIPS 2014*, pages 810–818, 2014.

Gerhard Schurz. *Hume's Problem Solved. The Optimality of Meta-Induction.* MT Press, 2019.

Manfred K. Warmuth and Wouter M. Koolen. Open problem: Shifting experts on easy data. In *Proceedings of The 27th Conference on Learning Theory, COLT 2014*, volume 35 of *JMLR Workshop and Conference Proceedings*, pages 1295–1298. JMLR.org, 2014.