Journal of Machine Learning Research 3 (2002) 363-396 Submitted 11/01; Published 11/02

Tracking a Small Set of Experts by Mixing Past Posteriors

Olivier Bousquet OLIVIER.BOUSQUET@TUEBINGEN.MPG.DE
Centre de Mathématiques Appliquées
Ecole Polytechnique
91128 Palaiseau, France

Manfred K. Warmuth MANFRED@CSE.UCSC.EDU
Computer Science Department
University of California, Santa Cruz
Santa Cruz, CA 95064, USA

Editor: Philip M. Long

Abstract

In this paper, we examine on-line learning problems in which the target concept is allowed to change over time. In each trial a master algorithm receives predictions from a large set of n experts. Its goal is to predict almost as well as the best sequence of such experts chosen off-line by partitioning the training sequence into $k+1$ sections and then choosing the best expert for each section. We build on methods developed by Herbster and Warmuth and consider an open problem posed by Freund where the experts in the best partition are from a small pool of size m. Since $k \gg m$, the best expert shifts back and forth between the experts of the small pool. We propose algorithms that solve this open problem by mixing the past posteriors maintained by the master algorithm. We relate the number of bits needed for encoding the best partition to the loss bounds of the algorithms. Instead of paying $\log n$ for choosing the best expert in each section we first pay $\log \binom{n}{m}$ bits in the bounds for identifying the pool of m experts and then $\log m$ bits per new section. In the bounds we also pay twice for encoding the boundaries of the sections.

Keywords: On-line learning, loss bounds, shifting experts, share updates.

1. Introduction

We consider the following standard on-line learning model in which a master algorithm has to combine the predictions from a set of experts (see e.g. Littlestone and Warmuth, 1994, Vovk, 1990, Cesa-Bianchi et al., 1997, Kivinen and Warmuth, 1999). Learning proceeds in trials. In each trial the master receives the predictions from n experts and uses them to form its own prediction. At the end of the trial both the master and the experts receive the true outcome and incur a loss measuring the discrepancy between their predictions and the outcome. The master maintains a weight for each of its experts. The weight of an expert is an estimate of the "quality" of this expert's predictions and the master forms its prediction based on a weighted combination of the experts' predictions. The master updates the experts' weights at the end of each trial based on the losses of the experts and master.

The goal is to design weight updates that guarantee that the loss of the master is never much larger than the loss of the best expert or the best convex combination of the losses of the experts. So here the best expert or convex combination serves as a comparator.

A more challenging goal is to learn well when the comparator changes over time. So now the sequence of trials is partitioned into sections. In each section the loss of the algorithm is compared to the loss of a particular expert and this expert changes at the beginning of a new section. The goal of the master now is to do almost as well as the best partition. Bounds of this type were first investigated by Littlestone and Warmuth (1994) and then studied in more detail by Herbster and Warmuth (1998) and Vovk (1999). Other work on learning in relation to a shifting comparator but not in the expert setting was done by Auer and Warmuth (1998), Herbster and Warmuth (2001) and Singer (1998).

In this paper we want to model situations where the comparators are from a small pool of m convex combinations of the n experts each represented by a probability vector $\tilde{u}_j$, $(1 \leq j \leq m)$. In the initial segment a convex combination $\tilde{u}_1$ might be the best comparator. Then at some point there is a shift and $\tilde{u}_2$ does well. In a third section, $\tilde{u}_1$ might again be best and so forth. The pool size is small ($m \ll n$) and the best comparator switches back and forth between the few convex combinations in the pool ($m \ll k$, where k is the number of shifts). Of course, the convex combinations of the pool are not known to the master algorithm.

This type of setting was popularized by an open problem posed by Freund (2000). In his version of the problem he focused on the special case where the pool consists of single experts (i.e. the convex combinations in the pool are unit vectors). Thus the goal is to develop bounds for the case when the comparator shifts back and forth within a pool of m out a much larger set of n experts.

Herbster and Warmuth (1998) developed bounds where the additional loss of the algorithm over the loss of the best comparator partition is proportional to the number of bits needed to encode the partition. Following this approach Freund suggests the following additional loss bound for his open problem: $\log \binom{n}{m} \approx m \log \frac{n}{m}$ bits for choosing the pool of m experts, $\log m$ bits per segment for choosing an expert from the pool, and $\log \binom{T-1}{k} \approx k \log \frac{T}{k}$ bits for specifying the k boundaries of the segments (where T is the total number of trials).

In this paper we solve Freund's open problem. Our methods build on those developed by Herbster and Warmuth (1998). There are two types of updates: a *Loss Update* followed by a *Mixing Update*. The Loss Update is the standard update used for the expert setting (see e.g. Littlestone and Warmuth, 1994, Vovk, 1990, Haussler et al., 1998, Kivinen and Warmuth, 1999) in which the weights of the experts decay exponentially with the loss. In the case of the log loss this becomes Bayes rule for computing the posterior weights for the experts. In the new Mixing Update the weight vector in the next trial becomes a mixture of all the past posteriors where the current posterior always has the largest mixture coefficient.

The key insight of our paper is to design the mixture coefficients for the past posteriors. In our main scheme the coefficient for the current posterior is $1-\alpha$ for some small $\alpha \in [0,1]$ and the coefficient for the posterior d trials in the past is proportional to α/d. Curiously enough this scheme solves Freund's open problem: When the comparators are single experts then the additional loss of our algorithms over the loss of the best comparator partition is order of the number of bits needed to encode the partition. For this scheme all past posteriors need to be stored requiring time and space $O(nt)$ at trial t. However, we show

how this mixing scheme can be approximated in time and space $O(n \ln t)$. An alternate simpler scheme has slightly weaker bounds: The coefficients of all past posteriors (there are t of them at trial t) are $\alpha\frac{1}{t}$. Now only the average of the past posteriors needs to be maintained requiring time and space $O(n)$.

We begin by reviewing some preliminaries about the expert setting and then give our main algorithm in Section 3. This algorithm contains the main schemes for choosing the mixture coefficients. In Section 4 we prove bounds for the various mixing schemes. In particular, we discuss the optimality of the bounds in relation to the number of bits needed to encode the best partition. We then discuss alternatives to the Mixing Update in Section 5 and experimentally compare the algorithms in Section 6. We conclude with a number of open problems.

2. Preliminaries

Let T denote the number of trials and n the number of experts. We will refer to the experts by their index $i \in \{1, \ldots, n\}$. At trial $t = 1, \ldots, T$, the master receives a vector x_t of n predictions, where $x_{t,i}$ is the prediction of the i-th expert. The master then must produce a prediction $\hat{y}_t$ and, following that, receives the true outcome y_t for trial t. We assume that $x_{t,i}, \hat{y}_t, y_t \in [0, 1]$.

A loss function $L : [0, 1] \times [0, 1] \to [0, \infty]$ is used to measure the discrepancy between the true outcome and the predictions. Expert i incurs loss $L_{t,i} = L(y_t, x_{t,i})$ at trial t and the master algorithm A incurs loss $L_{t,A} = L(y_t, \hat{y}_t)$. For the cumulative loss over a sequence, we will use the shorthand notation $L_{1..T,A} = \sum_{t=1}^{T} L_{t,A}$.

The weight vector maintained by the algorithm at trial t is denoted by v_t. Its elements are non-negative and sum to one, i.e. v_t is in the n-dimensional probability simplex denoted by $\mathcal{P}_n$.

Based on the current weight vector v_t, and the experts predictions x_t, the master algorithm uses a *prediction function* $\texttt{pred} : \mathcal{P}_n \times [0, 1]^n \to [0, 1]$ to compute its prediction for trial t: $\hat{y}_t = \texttt{pred}(v_t, x_t)$. In the simplest case the prediction function is an average of the experts predictions $\texttt{pred}(v, x) = v \cdot x$ (see Kivinen and Warmuth, 1999). Refined prediction functions can be defined depending on the loss function used (see Vovk 1990). The loss and prediction functions are characterized by the following definition.

Definition 1 *(Haussler et al., 1998, Vovk, 1998) Let $c, \eta > 0$. A loss function L and prediction function* pred *are (c, η)-realizable if, for any weight vector $v \in \mathcal{P}_n$, prediction vector x and outcome y,*

$$L(y, \texttt{pred}(v, x)) \leq -c \ln \sum_{i=1}^{n} v_i e^{-\eta L(y, x_i)} \quad . \tag{1}$$

For example, with the prediction function $\texttt{pred}(v, x) = v \cdot x$ the quadratic loss $L_{sq}(y, \hat{y}) = (y - \hat{y})^2$ satisfies[1] (1) with $(c, \eta) = (2, \frac{1}{2})$ and the entropic loss $L_{ent}(y, \hat{y}) = y \ln \frac{y}{\hat{y}} + (1 - y) \ln \frac{1-y}{1-\hat{y}}$ satisfies (1) with $(c, \eta) = (1, 1)$.

1. A slightly more involved prediction function shows that the quadratic loss is $(\frac{1}{2}, 2)$-realizable (see Vovk, 1990).

In the remainder we will assume the loss and prediction functions that we use are $(c, 1/c)$-realizable so that $c\eta = 1$. The two loss functions given above satisfy this criterion. This does not include the case of the absolute loss. However, the results here can essentially be extended to this loss as was done by Herbster and Warmuth (1998).

In the bounds given in this paper we will use the relative entropy as the measure of progress. For any $u, v \in \mathcal{P}_n$,

$$\Delta(u, v) = \sum_{i=1}^{n} u_i \ln \frac{u_i}{v_i} \geq 0 \ .$$

We adopt the usual convention that $0 \ln 0 = 0$. For two vectors u and v, we write $u > v$, if the "$>$" relationship holds componentwise. We use $\mathbf{0}$ to denote the all-zero vector.

The proofs in this paper rely on the following simple inequalities for the relative entropy:

Lemma 2 *For $u, v, w \in \mathcal{P}_n$ and $v, w > \mathbf{0}$,*

$$\Delta(u, v) \leq \Delta(u, w) + \ln \left(\sum_{i=1}^{n} u_i \frac{w_i}{v_i} \right) \ .$$

If $v \geq \beta w$, for some $\beta > 0$, then

$$\Delta(u, v) \leq \Delta(u, w) + \ln \frac{1}{\beta} \ .$$

Proof The first inequality follows from the concavity of the ln function and Jensen's inequality:

$$\Delta(u, v) - \Delta(u, w) = \sum_{i=1}^{n} u_i \ln \frac{w_i}{v_i} \leq \ln \left(\sum_{i=1}^{n} u_i \frac{w_i}{v_i} \right) \ .$$

For the second inequality we use the assumption that $v \geq \beta w$:

$$\sum_{i=1}^{n} u_i \frac{w_i}{v_i} \leq \sum_{i=1}^{n} u_i \frac{w_i}{\beta w_i} = \frac{1}{\beta} \ .$$

■

The first inequality is tighter than the second. However, for the sake of simplicity of the presentation we only use the second inequality throughout the paper. Chosing $u = w$ gives us

$$\Delta(w, v) \leq \ln \left(\sum_{i=1}^{n} w_i \frac{w_i}{v_i} \right) \leq \ln \frac{1}{\beta} \ .$$

Corollary 3 *Let $u \in \mathcal{P}_n$ and v be a mixture of vectors $w_q \in \mathcal{P}_n$ with $w_q > \mathbf{0}$. That is, $v = \sum_{q=1}^{t} \beta_q w_q$, where $(\beta_1, \ldots, \beta_t) \in \mathcal{P}_t$ and $> \mathbf{0}$. Then for any $1 \leq q \leq t$,*

$$\Delta(u, v) \leq \Delta(u, w_q) + \ln \frac{1}{\beta_q} \ .$$

By choosing $u = w_q$ (for $1 \leq q \leq t$), we have

$$\Delta(w_q, v) \leq \ln \frac{1}{\beta_q} \ .$$

So the mixture vector v is "close" to all w_q. Later in the paper the w_q will be posteriors from past trials. The new posterior at trial t will be a mixture of the previous posteriors and the corollary guarantees that the new posterior is close to all previous posteriors provided that the mixture coefficients β_q are not too small.

3. The Algorithms

Learning proceeds in trials. At the beginning of each trial t (see Figure 1) the master algorithm receives the prediction vector x_t from the experts and forms its own prediction $\hat{y}_t$. It then receives the correct outcome y_t and performs two weight updates. The first update is the standard *Loss Update* which is the basis for all the work in the expert framework (see Littlestone and Warmuth, 1994, and Vovk, 1990). This update, which produces an intermediate weight vector v_t^m, may be seen as a generalization of Bayes rule for updating a posterior. Indeed when $y_t \in \{0, 1\}$, the learning rate η is one and the loss is the log loss then this update becomes Bayes rule (see Appendix A for a detailed discussion).

The Loss Update allows us to prove bounds on the loss of the master algorithm in terms of the loss of the best expert (see Littlestone and Warmuth, 1994, and Vovk, 1990) or the best convex combination $\tilde{u} \in \mathcal{P}_n$ of the losses of the experts (see Kivinen and Warmuth, 1999). However in this paper the convex combination used as a comparator is allowed to change with the trial t. Let u_t be the convex combination used in trial t. The beliefs of the algorithm about the comparator sequence are modeled with a probability distribution $\beta_{t+1}(.)$. At the end of each trial t the master algorithm might need to "pick up" the computation from some previous trial. For $0 \leq q \leq t$, let $\beta_{t+1}(q)$ be the coefficient/probability given to weight vector v_q^m of trial q. Intuitively, if q is the last trial in which the comparator u_{t+1} is used then $\beta_{t+1}(q)$ should be high. In particular, $\beta_{t+1}(t)$ should be high if the comparison vector remains unchanged, i.e. $u_t = u_{t+1}$. Also if the comparison vector u_{t+1} has never appeared before then the coefficient $\beta_{t+1}(0)$ should be high because a section needs to be started with the initial weight v_0^m. Thus the second update (called the *Mixing Update*) "mixes" the previous weight vectors v_t^m. In the case of log loss the update mixes the current and the previous posteriors. However note that all posteriors are influenced by mixing that occurred in previous trials.

The probabilities $\beta_{t+1}(q)$ are specified by the specific mixing scheme to be used (see Table 1). The simplest case occurs when $\beta_{t+1}(t) = 1$ and the remaining coefficients are zero. Thus v_{t+1} simply becomes v_t^m. Following Herbster and Warmuth (1998) we call this the Static Experts case. This choice is the setup suitable when the comparator is the loss of a fixed expert (see e.g. Littlestone and Warmuth, 1994, and Vovk, 1990) or a fixed convex combination of the losses of the experts (see Kivinen and Warmuth, 1999).

Even in the case when shifting occurs the largest coefficient is naturally $\beta_{t+1}(t)$ signifying that the computation is most likely going to continue from the weight vector at the current trial t. We call any update where $\beta_{t+1}(t) = 1 - \alpha$ for some fixed small α by the name *Fixed-Share Update*. In contrast to the Static Update when v_{t+1} simply becomes the current

Parameters: $0 < \eta, c$ and $0 \leq \alpha \leq 1$
Initialization: Initialize the weight vector to $v_1 = \frac{1}{n}\mathbf{1}$ and denote $v_0^m = \frac{1}{n}\mathbf{1}$
FOR $t = 1$ **TO** T **DO**

- **Prediction:** After receiving the vector of experts' predictions x_t, predict with

$$\hat{y}_t = \texttt{pred}(v_t, x_t) \ .$$

- **Loss Update:** After receiving the outcome y_t, compute for $1 \leq i \leq n$,

$$v_{t,i}^m = \frac{v_{t,i} e^{-\eta L_{t,i}}}{\sum_{j=1}^n v_{t,j} e^{-\eta L_{t,j}}}, \qquad \text{where } L_{t,i} = L(y_t, x_{t,i}) \ .$$

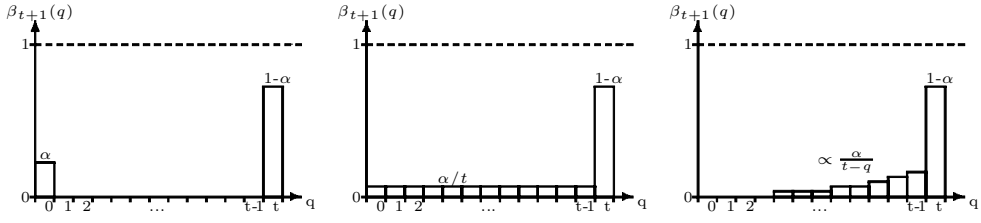
- **Mixing Update:** Choose non-negative mixture coefficients $\beta_{t+1}(q)$ $(q = 0, \ldots, t)$ such that $\sum_{q=0}^t \beta_{t+1}(q) = 1$ and compute

$$v_{t+1} = \sum_{q=0}^t \beta_{t+1}(q) v_q^m \ .$$

Figure 1: The Mixing Algorithm

Table 1: The Mixing Schemes

Name	Coefficients
Static Experts	$\beta_{t+1}(t) = 1$ and $\beta_{t+1}(q) = 0$ for $0 \leq q < t$
Fixed-Share Update	$\beta_{t+1}(t) = 1 - \alpha$ and $\sum_{q=0}^{t-1} \beta_{t+1}(q) = \alpha$
• To Start Vector	$\beta_{t+1}(0) = \alpha$
• To Past	
• Uniform Past	$\beta_{t+1}(q) = \alpha \frac{1}{t}$ for $0 \leq q < t$
• Decaying Past	$\beta_{t+1}(q) = \alpha \frac{1}{(t-q)^\gamma} \frac{1}{Z_t}$ for $0 \leq q < t$, with $Z_t = \sum_{q=0}^{t-1} \frac{1}{(t-q)^\gamma}$ and $\gamma \geq 0$



FS to Start Vector FS to Uniform Past FS to Decaying Past

posterior v_t^m, each expert first "shares" a fraction of α of its weight $v_{t,i}^m$ and the total shared

weight α is then distributed among the earlier posteriors (i.e. $\sum_{q=0}^{t-1} \beta_{t+1}(q) = 1 - \beta_{t+1}(t) = \alpha$).

In the simplest case all of the shared weight α goes to the uniform weight vector $\frac{1}{n}\mathbf{1} = v_0^m$ leading to essentially the Fixed-Share Algorithm analyzed by Herbster and Warmuth (1998). The main contribution of this paper is to show that other choices of distributing the shared weight to the past posteriors give useful bounds. A simple choice is the average of the past weight vectors (*Uniform Past*), i.e. $\beta_{t+1}(q) = \alpha\ \frac{1}{t}$, for $0 \leq q < t$. Instead of the average a decaying sequence of coefficients leads to better bounds. The more recent weight vectors receive higher coefficients. In the case of the *Decaying Past* mixing scheme, $\beta_{t+1}(q) \approx \alpha\ \frac{1}{(t-q)^\gamma}$, for $\gamma \geq 0$.

4. Analysis of the Algorithms

In this paper we stress the bounds where the loss of the algorithm is compared to some convex combination of the losses of the experts (see e.g. Kivinen and Warmuth, 1999) rather than the loss of a single expert. Let $\beta_1(0) = 1$, $\Delta(.,.)$ be the relative entropy and L_t be the vector of the losses $L_{t,i}$ of the experts at trial t.

Lemma 4 *For any trial t, any $0 \leq q \leq t-1$ and any comparison vector $u_t \in \mathcal{P}_n$*

$$\begin{aligned} L_{t,A} &\leq L_t \cdot u_t + c\,\Delta(u_t, v_t) - c\,\Delta(u_t, v_t^m) \\ &\leq L_t \cdot u_t + c\,\Delta(u_t, v_q^m) - c\,\Delta(u_t, v_t^m) + c \ln \frac{1}{\beta_t(q)} \quad . \end{aligned}$$

Proof The first inequality follows from rewriting the r.h.s. of (1) using the assumption that $c\eta = 1$:

$$\begin{aligned} -c \ln \sum_{i=1}^n v_{t,i} e^{-\eta L_{t,i}} &= \sum_{i=1}^n u_{t,i} \left(L_{t,i} + c \ln e^{-\eta L_{t,i}} - c \ln \sum_{j=1}^n v_{t,j} e^{-\eta L_{t,j}} \right) \\ &= \sum_{i=1}^n u_{t,i} L_{t,i} + c \sum_{i=1}^n u_{t,i} \ln \frac{v_{t,i}^m}{v_{t,i}} \\ &= L_t \cdot u_t + c\,\Delta(u_t, v_t) - c\,\Delta(u_t, v_t^m) \end{aligned}$$

The second inequality follows from Corollary 3. ■

4.1 Comparison to a Fixed Convex Combination

Here we consider the case when the comparison vector $u_t = \tilde{u}$ remains unchanged. We will thus use the Static Experts mixing scheme, i.e. $\beta_{t+1}(t) = 1$ and $\beta_{t+1}(q) = 0$ for $0 \leq q < t$. This is exactly the Static Experts Algorithm since v_t^m becomes the weight vector used in the next trial, i.e. $v_{t+1} = v_t^m$. By summing Lemma 4 over all trials we get the following bound (see e.g. Kivinen and Warmuth, 1999).

Lemma 5 *For the Mixing Algorithm with the Static Expert mixing scheme (i.e. no mixing) and any $\tilde{u} \in \mathcal{P}_n$,*

$$L_{1..T,A} \leq \sum_{t=1}^{T} L_t \cdot \tilde{u} + c\,\Delta(\tilde{u}, v_1) - c\,\Delta(\tilde{u}, v_T^m) \ .$$

Proof We apply Lemma 4 (first inequality) to each trial $t = 1, \ldots, T$ and use the fact that $v_t = v_{t-1}^m$. The sum of divergences telescopes so that only the first and last terms remain. ■

Note that the r.h.s. is the same for all $\tilde{u}$. To obtain upper bounds we often choose a particular $\tilde{u}$ and drop the $-\Delta(\tilde{u}, v_T^m)$ term which is always negative. The start vector is always uniform, i.e. $v_1 = \frac{1}{n}\mathbf{1}$. If the best expert has loss L^* and $\tilde{u}$ is the unit probability vector for this expert then $\Delta(\tilde{u}, \frac{1}{n}\mathbf{1}) = \ln n$ and the above bound is at most $L^* + c \ln n$. However if k experts have loss L^* then by choosing $\tilde{u}$ uniform over these experts (and zero on the remaining experts) the bound becomes $L^* + c \ln \frac{n}{k}$. This improvement was first pointed out by Littlestone and Warmuth (1994).

4.2 Comparison to a Sequence with k Shifts

Now, we consider the situation where the comparison vector u_t of trial t is allowed to change/shift from trial to trial. A sequence of comparison vectors $u_1, \ldots, u_T \in \mathcal{P}_n$ *has k shifts* if there is a subsequence of k trials $t_1, \ldots, t_k$ where $u_{t_j} \neq u_{t_j - 1}$ and $u_{t-1} = u_t$ for all other trials $t > 1$. We define $t_0 = 1$ and $t_{k+1} = T + 1$. We now apply Lemma 4 to the case when all of the lost weight α goes to the original posterior $\frac{1}{n}\mathbf{1}$. This essentially becomes the Fixed-Share Algorithm of Herbster and Warmuth (1998).

Lemma 6 *For the Mixing Algorithm A with the Fixed-Share to the Start Vector mixing scheme and any sequence of T comparison vectors u_t with k shifts,*

$$\begin{aligned} L_{1..T,A} \quad \leq \quad & \sum_{t=1}^{T} L_t \cdot u_t + c \sum_{j=0}^{k} \left(\Delta(u_{t_j}, \frac{1}{n}\mathbf{1}) - \Delta(u_{t_j}, v_{t_{j+1}-1}^m) \right) \\ & + ck \ln \frac{1}{\alpha} + c(T - k - 1) \ln \frac{1}{1 - \alpha} \ . \end{aligned}$$

Proof We apply Lemma 4 (second inequality) for each trial. Whenever $u_t = u_{t-1}$ then we use $q = t - 1$ and $\beta_t(t-1) = 1 - \alpha$, i.e.

$$L_{A,t} \leq L_t \cdot u_t + c\,\Delta(u_t, v_{t-1}^m) - c\,\Delta(u_t, v_t^m) + c \ln \frac{1}{1 - \alpha} \ .$$

For the first trial we use $q = 0$, $\beta_1(0) = 1$, and $v_0^m = \frac{1}{n}\mathbf{1}$:

$$L_{A,1} \leq L_1 \cdot u_1 + c\,\Delta(u_1, \frac{1}{n}\mathbf{1}) - c\,\Delta(u_1, v_1^m) \ .$$

For all other beginnings of sections, i.e. the trials $t = t_j$ $(1 \leq j \leq k)$ we use $q = 0$, $\beta_{t_j}(0) = \alpha$, and $v_0^m = \frac{1}{n}\mathbf{1}$:

$$L_{A,t_j} \leq L_{t_j} \cdot u_{t_j} + c\,\Delta(u_{t_j}, \frac{1}{n}\mathbf{1}) - c\,\Delta(u_{t_j}, v_{t_j}^m) + c \ln \frac{1}{\alpha} \ .$$

Now we sum over all trials. The entropy terms within the sections telescope and only for the beginning and the end of each section a positive and a negative entropy term remains, respectively. The beginnings of the k sections each incur an additional term of $c \ln \frac{1}{\alpha}$. Also $T - k + 1$ times $u_t = u_{t-1}$ and an additional term of $c \ln \frac{1}{1-\alpha}$ is incurred. The initial trial ($t = 1$) has no additional term. ■

If we restrict ourselves to the case when the comparison vectors are focused on single experts then we essentially obtain the basic bound of Herbster and Warmuth (1998) for the Fixed Share to the Start Vector mixing scheme.

4.3 Comparison to a Sequence with k Shifts and a Pool of Size m

As in the previous section we let $t_1, \ldots, t_k$ be the subsequence of indices in the sequence of comparators $u_1, \ldots, u_T$ where shifting occurs (also $t_0 = 1$, $t_{k+1} = T + 1$ and $u_{T+1} = u_T$). Let $\tilde{u}_1, \ldots, \tilde{u}_m$ be the pool of m distinct convex combinations in $\{u_{t_1}, \ldots, u_{t_k}\}$. At trial $t_0 = 1$ the first convex combination from the pool is selected. At the remaining starting points u_{t_j} ($1 \le j \le k$) of sections either a new convex combination is selected from the pool ($m - 1$ times) or the convex combination u_{t_j} has appeared before in $\{u_{t_1}, \ldots, u_{t_{j-1}}\}$. In the latter case ($k - m + 1$ times) the convex combination *shifts back* to the end of the last section where this convex combination from the pool was the comparator. We assume that $m \ll k$ and thus most of the shifts ($k + 1 - m$ of them) are shift backs. Curiously enough the entropy terms for all trials belonging to the same convex combination telescope.

Theorem 7 *For the Mixing Algorithm A and for any sequence of T comparison vectors u_t with k shifts from a pool $\{\tilde{u}_1, \ldots, \tilde{u}_m\}$ of m convex combinations, we have (recall $\beta_1(0) = 1$)*

$$L_{1..T,A} \le \sum_{t=1}^{T} L_t \cdot u_t + c \sum_{j=1}^{m} \left(\Delta(\tilde{u}_j, \frac{1}{n}\mathbf{1}) - \Delta(\tilde{u}_j, v_{\ell_j}^m) \right) + c \sum_{t=1}^{T} \ln \frac{1}{\beta_t(q_{t-1})} \ ,$$

where ℓ_j is the last trial such that $u_t = \tilde{u}_j$ and q_t is the last of the trials $q \le t$ such that $u_{t+1} = u_q$ (we let $q_t = 0$ when no such trial exists).

Proof We apply Lemma 4 to all trials using $q = q_{t-1}$:

$$L_{t,A} \le L_t \cdot u_t + c\,\Delta(u_t, v_{q_{t-1}}^m) - c\,\Delta(u_t, v_t^m) + c \ln \frac{1}{\beta_t(q_{t-1})} \ .$$

We claim that summing the above over all trials proves the theorem. First note that the comparator u_t remains constant within each section. So as before the entropy terms within each section telescope and only for the beginning and end of each section a positive and a negative entropy term remains, respectively. Furthermore, the ending and beginning terms belonging to successive sections of the same convex combination from the pool also telescope. So for each element of the pool only two entropy terms survive: one for the beginning of the first section where it appears and one for the end of its last section. More precisely, the beginning of the first section of convex combination $\tilde{u}_j$ contributes $\Delta(\tilde{u}_j, \frac{1}{n}\mathbf{1})$ because then $q_{t-1} = 0$ by definition. Also the last trial ℓ_j of $\tilde{u}_j$ contributes $-\Delta(\tilde{u}_j, v_{\ell_j}^m)$. ■

The challenge is to design the mixing coefficients $\beta_{t+1}(q)$ so that the last sum in the above theorem is minimized. We give some reasonable choices below and discuss time and space trade offs.

Bound for the Fixed Share to Uniform Past Mixing Scheme. Consider a mixing scheme that equally penalizes all vectors in the past: $\beta_{t+1}(q) = \alpha \frac{1}{t}$ $(q = 0..t-1)$.

Corollary 8 *For the Mixing Algorithm A with the Fixed Share to Uniform Past mixing scheme and for any sequence of T comparison vectors $\boldsymbol{u}_t$ with k shifts from a pool of m convex combinations, we have*

$$L_{1..T,A} \leq \sum_{t=1}^{T} L_t \cdot \boldsymbol{u}_t + cm \ln n + ck \ln \frac{1}{\alpha} + c(T-k-1) \ln \frac{1}{1-\alpha} + ck \ln(T-1) \ .$$

Proof We use the bound of Theorem 7. Clearly, each of the m differences of divergences can be bounded as follows:

$$\Delta(\tilde{\boldsymbol{u}}_j, \frac{1}{n}\mathbf{1}) - \Delta(\tilde{\boldsymbol{u}}_j, v^m_{\ell_j}) \leq \Delta(\tilde{\boldsymbol{u}}_j, \frac{1}{n}\mathbf{1}) \leq \ln n \ .$$

So to complete the proof we still need to show that the last term in the inequality of Theorem 7 is bounded by the last three terms in the inequality of the corollary.

There are $T-k-1$ trials such that $\boldsymbol{u}_t = \boldsymbol{u}_{t-1}$. For all these trials $\beta_t(q_{t-1}) = \beta_t(t-1) = 1-\alpha$ contributing a total cost of $c(T-k-1)\ln\frac{1}{1-\alpha}$. In all the remaining trials a section is starting. For the first trial $\beta_1(0) = 1$ and no cost is incurred. If t is the index of one of the k remaining trials, which are at the start of a section, then $\beta_t(q_{t-1}) = \alpha \frac{1}{t-1}$. Thus these trials contribute at most $ck \ln \frac{1}{\alpha} + ck \ln(T-1)$. ■

Bound for the Fixed Share to Decaying Past Mixing Scheme. We now show that an improvement of the above corollary is possible by choosing $\beta_{t+1}(q) = \alpha \frac{1}{(t-q)^\gamma Z_t}$ for $0 \leq q \leq t-1$, with $Z_t = \sum_{q=0}^{t-1} \frac{1}{(t-q)^\gamma}$.

Corollary 9 *For the Mixing Algorithm A with the Fixed Share to Decaying Past mixing scheme with $\gamma = 1$ and for any sequence of T comparison vectors $\boldsymbol{u}_t$ with k shifts from a pool of m convex combinations, we have*

$$\begin{aligned} L_{1..T,A} \quad \leq \quad & \sum_{t=1}^{T} L_t \cdot \boldsymbol{u}_t + cm \ln n + ck \ln \frac{1}{\alpha} + c(T-k-1) \ln \frac{1}{1-\alpha} \\ & + ck \ln \frac{(T-1)(m-1)}{k} + ck \ln \ln(eT) \ . \end{aligned}$$

Proof The proof follows the proof of Corollary 8 and is given in Appendix B. ■

In Appendix D we give slightly improved bounds based on a better tuning of γ. We also allow the coefficient $\beta_{t+1}(0)$ to be equal to some constant δ (i.e. not decreasing with t). These improved tunings led to slightly better performance in practice and were used for the experiments in Section 6.

4.4 Relating the Bounds to the Number of Bits

In this section we assume the comparison vectors u_t are unit vectors. The number of bits (measured with respect to $c\ln$ instead of log) for encoding a partition with k shifts from a pool of size m is the following:

$$c\ln\binom{n}{m} + c\ln\binom{T-1}{k} + c\ln m + ck\ln(m-1) \approx cm\ln\frac{n}{m} + ck\ln\frac{T}{k} + ck\ln m. \qquad (2)$$

The first term is for selecting the m experts of the pool, the second term for encoding the boundaries of the k shifts and the last term for naming members of the pool belonging to the $k+1$ sections.

Now consider the following "direct" algorithm proposed by Freund. Run the Mixing Algorithm with the Fixed Share to Start Vector mixing scheme (i.e. the Fixed Share Algorithm of Herbster and Warmuth, 1998) on every pool/subset of m out of the n experts. Each run becomes an expert that feeds into the Mixing Algorithm with the Static Experts mixing scheme. If u_t is the comparator sequence of the best partition with k shifts from a pool of m experts then the loss of this algorithm is at most $\sum_{t=1}^{T} L_t \cdot u_t$ plus the number of bits (2). However this algorithm requires $\binom{n}{m}m$ weights which is too large for most practical applications.

In contrast, our algorithms are efficient and the bounds are still close to optimal. For example, if $\alpha = \frac{k}{T-1}$ then the bound of the Mixing Algorithm with the Fixed Share to Decaying Past ($\gamma = 1$) mixing scheme (see Corollary 9) is the loss of the best partition plus approximately

$$cm\ln n + ck\ln\frac{T}{k} + ck\ln\frac{Tm}{k} + ck\ln\ln(eT).$$

If we omit the last term, then this is at most twice the number of bits (2) and thus we solved Freund's open problem. Also note that the above bound is essentially $ck\ln\frac{T}{k} + cm\ln m$ larger than the number of bits. In the dominating first term we are paying a second time for encoding the boundaries. The same bound with the Uniform Past mixing scheme is not a constant times larger than the number of bits. Thus it seems that the mixing coefficients need to decay towards the past to obtain the best bounds.

The better Decaying Past scheme requires us to store all previous posteriors, i.e. nt weights at trial t. However in Appendix C we describe a way to approximate this scheme with a storage requirement of only $O(n\log t)$ weights, without significantly increasing the loss bounds. (Whenever $\log t \ll \binom{n}{m}$, then these methods are preferable over the direct algorithm.) The Uniform Past scheme, in contrast, only needs to store the current and the average of the past posterior (i.e. $2n$ weights).

5. Additional Observations

In this section, we will discuss some extensions and generalizations of the mixing schemes proposed above.

5.1 Mixing Several Mixing Schemes

Given two mixing schemes $\beta^1_\cdot(.)$ and $\beta^2_\cdot(.)$ we can define a new scheme

$$\beta_{t+1}(q) = \delta\ \beta^1_{t+1}(q) + (1-\delta)\ \beta^2_{t+1}(q),$$

for $0 \le q \le t$ and $\delta \in (0,1)$. In view of Lemma 2 it should be easy to see that the bound obtainable (via Theorem 7) for the new scheme $\beta_\cdot(.)$ is at most $c \ln \frac{1}{\delta}$ plus the bound obtainable via scheme $\beta^1_\cdot(.)$ and at most $c \ln \frac{1}{1-\delta}$ plus the bound obtainable via scheme $\beta^2_\cdot(.)$. So if we are not quite sure which scheme to use, we can mix them and essentially be as good as the best of both schemes.

5.2 Generalized Mixing Schemes

Notice that the bounds proven in this paper rely on the following simple property of the Mixing Update:

$$\forall q = 0, \ldots, t: \quad v_{t+1} \ge \beta_{t+1}(q) v^m_q \ . \tag{3}$$

This property is used in Lemma 2 and Corollary 3 and is the basis for our main result (Theorem 7).

The following update (called *Max Mixing Update*) also has this property and thus all bounds proven in the paper immediately hold for this update as well:

$$v_{t+1} = \frac{1}{Z_t} \max_{q=0}^{t} \left(\beta_{t+1}(q) v^m_q\right) \ ,$$

where Z_t is the normalization and the max is component-wise. Since $\max(a,b) \le a+b$ for positive a and b one can show that $Z_t \le 1$ and thus (3) is satisfied.

More generally, we can replace the maximum by other functions. For example, for any $p \ge 1$, we can use $f(a,b) = (a^p + b^p)^{1/p}$. Since we have $a^p + b^p \le (a+b)^p$ for any $a, b \ge 0$, we can see the condition (3) will still hold.

Another possible update is to minimize the relative entropy subject to the constraint defined by (3), i.e.

$$v_{t+1} = \arg \min_{\boldsymbol{v} \in C_t \cap \mathcal{P}_n} \Delta(v, v^m_t) \ ,$$

where C_t is the set of vectors satisfying (3). We call this the *Projection Mixing Update*. Such updates have been used by Herbster and Warmuth (2001) to obtain bounds for shifting in a regression setting.

Notice for all generalized mixing schemes described above we can still use the technique sketched in the Appendix C for reducing the number of weights at trial t from $O(nt)$ to $O(n \log t)$.

5.3 Variable Share and Lazy Mixing Updates

Inspired by the Variable Share Algorithm of Herbster and Warmuth (1998) we define the following *Variable Share Mixing Update*. As we shall see in the next section this algorithm is better in the experiments than the Mixing Update when the same mixing scheme is used.

$$v_{t+1,i} = \beta_{t+1}(t)^{L_{t,i}} v^m_{t,i} + F_t \sum_{q=0}^{t-1} \beta_{t+1}(q) v^m_{q,i} \ ,$$

where the losses $L_{t,i}$ must be in $[0,1]$ and F_t is a factor that assures that the $v_{t+1,i}$ sum to one. Note that the standard Mixing Update can be written as

$$v_{t+1,i} = \beta_{t+1}(t)v^m_{t,i} + \sum_{q=0}^{t-1} \beta_{t+1}(q)v^m_{q,i} \ .$$

Thus when all $L_{t,i}$ are one then $F_t = 1$ and both updates agree. Also when all $L_{t,i}$ are zero then $F_t = 0$ and $v_{t+1} = v^m_t = v_t$ which is the Static Experts Update. This shows that the new update interpolates between the Mixing Update and the Static Experts Update. In some sense this update uses a small loss of the experts as an indicator that no shift is occurring.

Another such indicator is the loss of the master algorithm itself. Indeed, when the master performs well, it is likely that no shift is occurring and there is no need to mix the posteriors. This idea leads to the *Lazy Mixing Update* which works as follows. We use a variable B_t to accumulate the loss of the master algorithm: we initialize B_1 to 0 and update with $B_{t+1} = B_t + L_{t,A}$. Only if $B_{t+1} \geq 1$ then we perform the Mixing Update and reset B_{t+1} to 0.

5.4 NP-Completeness

We now prove that the off-line algorithm for finding the best partition with experts from a pool is NP-complete. However we do not know how to use this hardness result to prove lower bounds on the loss of on-line algorithms.

Theorem 10 *The following off-line problem is NP-complete.*

Input: A number of trials T, a number of experts n, a number of shifts k, a pool size m, binary predictions $x_{t,i}$ for each expert i and trial t, and binary labels y_t for each trial t.

Question: Is there a partition of the T trials with k shifts from a pool of m convex combinations that has loss zero?

Proof The problem reduces to three-dimensional matching (see Garey and Johnson, 1979, page 221). We have $T = 3q$ trials. Trials $1, 2, \ldots, 3q$ correspond respectively to the elements $w_1, w_2, \ldots, w_q$, $r_1, r_2, \ldots, r_q$, $s_1, s_2, \ldots, s_q$. Choose the $x_{t,i}$ and y_t so that each triplet (w_j, r_k, s_ℓ) corresponds to an expert that only predicts correctly in trials j, $k+q$ and $\ell+2q$, respectively. The number of convex combinations m is q and the number of shifts k is $3q-1$.

One now can now show that a partition of loss zero corresponds to a matching and vice versa. ■

6. Experiments

In this section we discuss experiments performed on artificial data. The setup is similar to the one used by Herbster and Warmuth (1998).

6.1 Generating Datasets

The sequence of trials is partitioned beforehand into segments and a good expert is associated with each segment. We will choose the examples so that the good expert has a

smaller expected loss than the other experts. The loss of the good partition will serve as a comparator to the loss of our on-line algorithms. In the notation of the previous sections the u_t are unit-vectors specifying the good expert of trial t.

The loss is measured by the square loss. The outcome of each example is 0 or 1 with probability 1/2 and the predictions of the experts are generated randomly in $[0,1]$ as specified below. The prediction of the good expert is chosen uniformly in a small interval around the outcome. If the outcome is 0, we choose the prediction uniformly in $[0,a]$ and if the outcome is 1, we choose the predictions uniformly in $[1-a,1]$. In our experiments we chose $a = .15$ for the good expert which makes its expected square loss $a^2/3 = 0.0075$. The predictions of the other experts are chosen the same way but now $a = .5$ and the expected loss is 0.083 per trial.

6.2 Algorithms

We choose a prediction function due to Vovk (1998) which has slightly better bounds than the weighted average prediction discussed in Section 2 (see Kivinen and Warmuth, 1999). For the square loss Vovk's prediction function is motivated as follows (see Haussler et al., 1998). Define $L_0(z) = L(0,z) = z^2$ and $L_1(z) = L(1,z) = (1-z)^2$ for $z \in [0,1]$. Both these losses are monotone in $[0,1]$ so that we can define the inverse functions $L_0^{-1}(z) = \sqrt{z}$ and $L_1^{-1}(z) = 1 - \sqrt{z}$. Now we define

$$\gamma(v,y) = -c \ln \sum_{i=1}^{n} v_i e^{-\eta(y-x_i)^2} .$$

Then the prediction is computed as

$$\mathtt{pred}(v,x) = \frac{L_0^{-1}(\gamma(v,0)) + L_1^{-1}(\gamma(v,1))}{2} .$$

This prediction function together with the square loss is $(1/2,2)$-realizable, i.e the learning rate η is 2 and $c = 1/2$. In contrast, the weighted average prediction is only $(2,1/2)$-realizable. In this paper we assumed that $c\eta = 1$. In this case c is the factor in front of the additional loss incurred by the algorithm (see for example Theorem 7). Thus the smaller c, the better the bounds.

We consider the various mixing schemes introduced in this paper. The parameters such as α, δ or γ are all set to their optimal value, namely $\alpha^* = \frac{k}{T-1}$, $\delta^* = \frac{m-1}{k}$ and $\gamma^* = 1 - \ln^{-1} \frac{k-m+1}{m-1}$ (see Appendix D for details).

6.3 Format of the Results

We present two types of plots. The first type is a total loss plot which depicts the cumulative loss of various algorithms as a function of the trial number. The best partition is indicated along the x-axes as follows. We plot vertical lines at each shift in the best partition and write the index of the best expert at the beginning of each section.

The second type of plot displays the weights of the experts maintained by the master algorithm where the x-axis is again the trial number. Only the larger weights appear in the plots. Those weights usually correspond to the experts in the small pool of experts used in

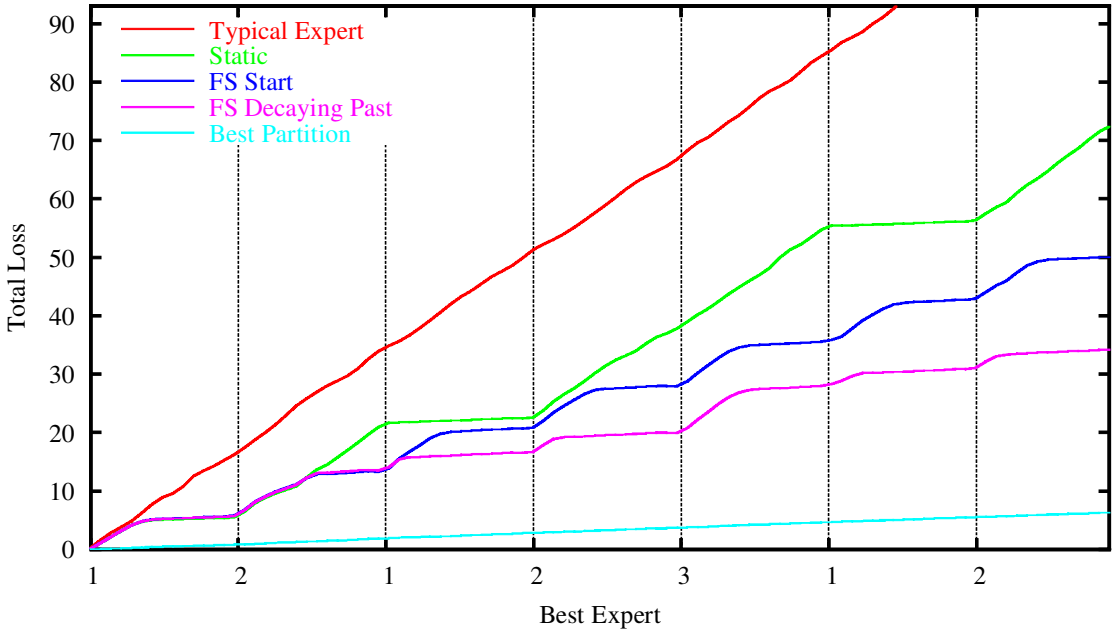


Figure 2: Total losses obtained by the different mixing schemes. The parameters are $T = 1400$, $n = 20000$, $k = 6$, $m = 3$. The numbers below the x-axis indicate the index of the best expert in the current segment.

the best partition. We always use a different color for each expert from the pool. We also depicted in these plots the maximum weight of any experts outside the pool and the weight of a typical expert outside the pool. Often we use a logarithmic scale of the weights (on the y-axis) to emphasize small weights.

6.4 Basic Experiment

We use $T = 1400$ trials and $n = 20000$ experts, $m = 3$ of which constitute the experts in the pool. W.l.o.g. these are the first three experts. Thus the best partition only uses the first three unit vectors. The best partition has 7 segments of 200 trials each. We thus have $k = 6$ shifts. The sequence of best experts is $1, 2, 1, 2, 3, 1, 2$. This means that at trials 1, $t_1 = 201$ and $t_4 = 1001$ the three experts of the pool have small loss for the first time, while at trials 401, 601, 801 and 1201 we are shifting back to a previously used expert from the pool. In Figure 2 we plot the total loss of the different algorithms as a function of the trial number. The top curve is the total loss of a typical expert and the bottom curve is the total loss of the best partition. The slope always corresponds to the loss per trial.

As expected, the Static Experts Algorithm simply learns the weight of the expert belonging to the first section and then "gets stuck" with that expert. It has the optimal rate of loss (slope of bottom curve) in all later segments in which the first expert is active and the slope of the top curve in the remaining sections. The total loss curve of the Fixed Share to Start Vector mixing scheme has a bump at the beginning of each section but is able to

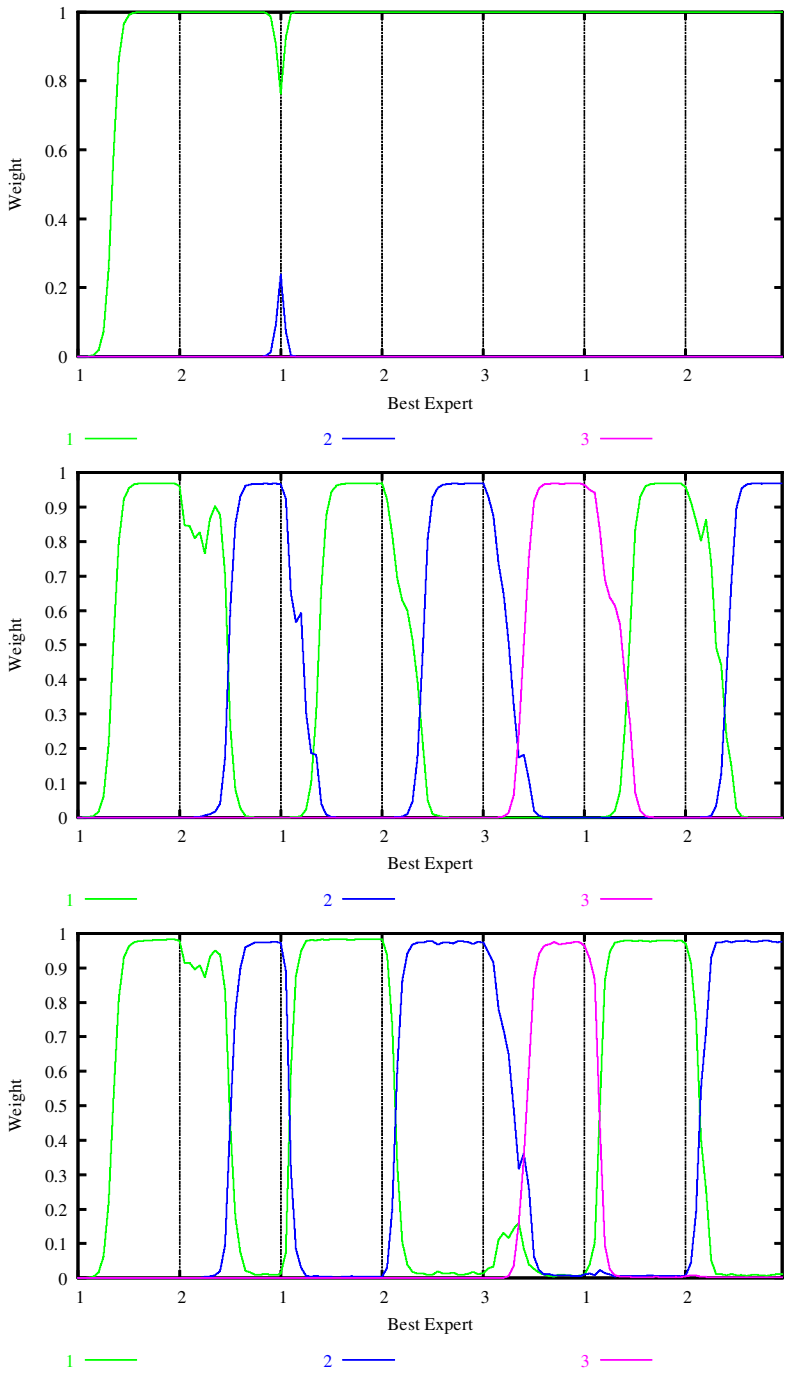


Figure 3: Weights for the different mixing schemes (top: Static Experts Algorithm, middle: Fixed Share to Start Vector, bottom: Fixed Share to Decaying Past). The line marked "1" depicts the log weight of expert 1.

recover to the optimum slope after each shift. The bumps in its total loss curve are roughly of equal size. However note that the Fixed Share to Decaying Past mixing scheme is able to recover faster when the sequence shifts back to a previously active expert from the pool. Thus in trials 401, 601, 801 and 1201 the bumps in its total loss curve are smaller than the bumps in the curve for the Fixed Share to Start Vector mixing scheme.

In support to the above explanation, we plotted the weights maintained by the algorithms in Fig. 3. Clearly the Static Experts Algorithm (top plot) is unable to track the shifts. In the Fixed Share to Start Vector mixing scheme (middle plot) the weight of each expert is picked up exponentially fast in each segment. Note that the pickup is similar in each segment. However in the Fixed Share to Decaying Past mixing scheme (bottom plot) the weights of the experts are picked up faster in sections where we return to an expert that has already been used in an earlier section.

To focus on the small weights we plot the weights of the Fixed Share to Start Vector and the Decaying Past mixing schemes on a logarithmic scale (Figure 4). Note that if an expert of the pool was active then its weight remains at an elevated level above the maximum weight of all $n - 3$ experts outside the pool. From the elevated level the weight can be picked up more quickly when the same expert becomes active again. This corresponds to the smaller bumps in the total loss curve of the Decaying Past mixing scheme for the sections when an expert becomes active again (see Figure 2).

For all the experiments we conducted, the Uniform Past mixing scheme essentially has the same performance as the Decaying Past mixing scheme (plots not shown), although the bounds we proved for the former scheme are slightly worse than those proven for the latter scheme. Similar performance was also obtained with the Max Mixing Updates.

6.5 Many Experts Remembered

We next consider an experiment where the master algorithm has to "remember" a larger number of experts for a longer period of time. We use a pool of size $m = 10$ and cycle through these experts three times where each segment has length 200. Each of the 10 experts thus has to be remembered for 2000 trials.

Figure 5 shows the total loss plots. We notice that in the 10 section (the first cycle), the curve of the Fixed Share to Start Vector has slightly lower loss than the fancier Fixed Share to Decaying Past mixing scheme. This is because the latter more sophisticated mixing scheme "expects" a return to previously used experts. However, the first 10 experts are all new. When the 10 experts are repeated again (for the second cycle) then the fancier scheme clearly has lower loss. This shows that the memory effect is still noticeable over a long period of time. Figure 6 shows the weights on a logarithmic scale. On the top we plot the weights of the Fixed Share to Start Vector. The behavior of the algorithm is roughly the same in all sections indicating that the experts in the pool are not "remembered". In contrast, the Fixed Share to Decaying Past mixing scheme behaves differently in the first versus the repeating cycles. The weights are successively raised in the first cycle but then remain at an intermediate level so that they can be picked up faster in the repeating cycles. In some sense the Fixed Share to Decaying Past mixing scheme "caches" the weights of experts that have done well sometime in the past.

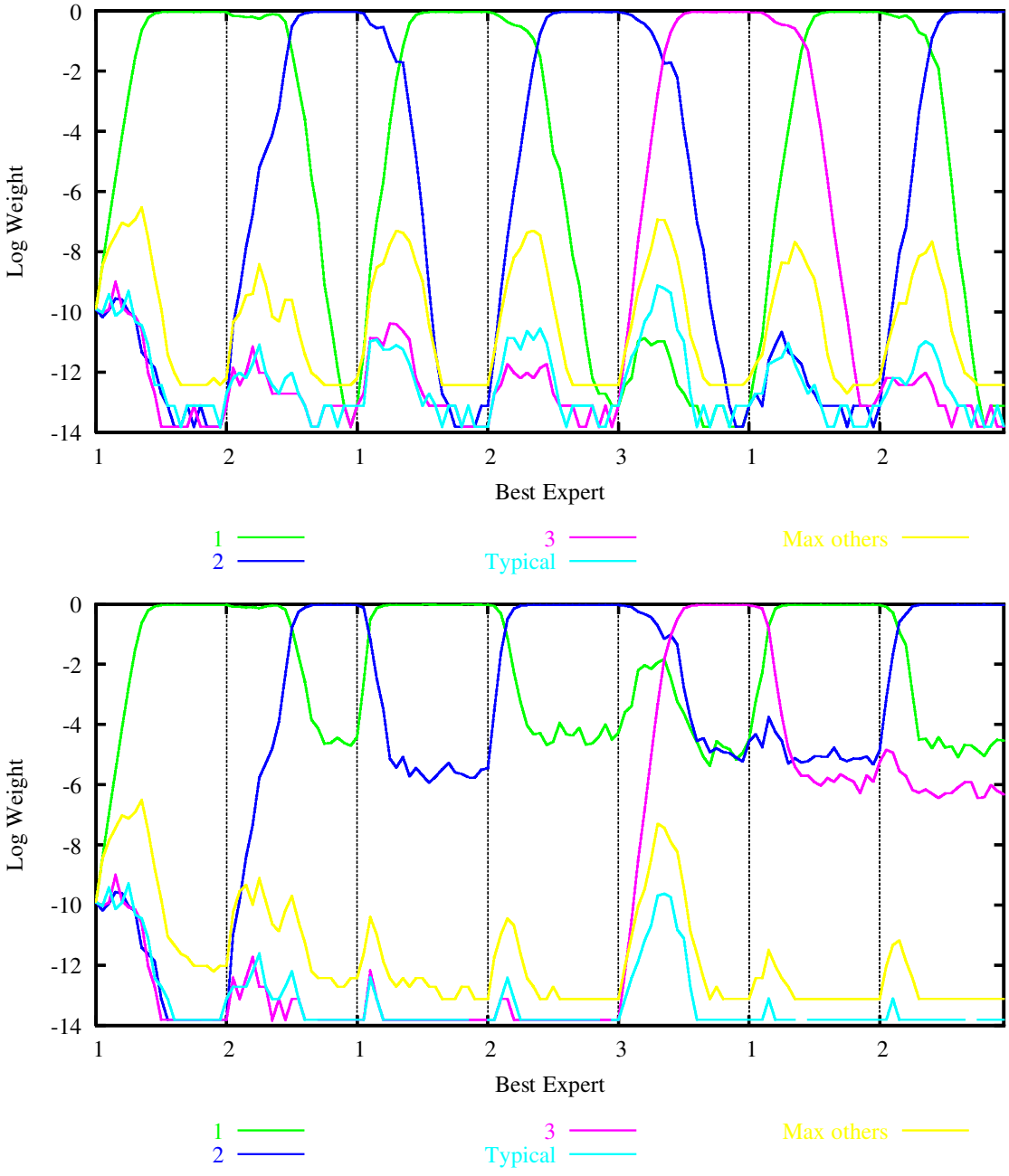


Figure 4: Log weights for the different updates (top: Fixed Share to Start Vector, bottom: Fixed Share to Decaying Past).

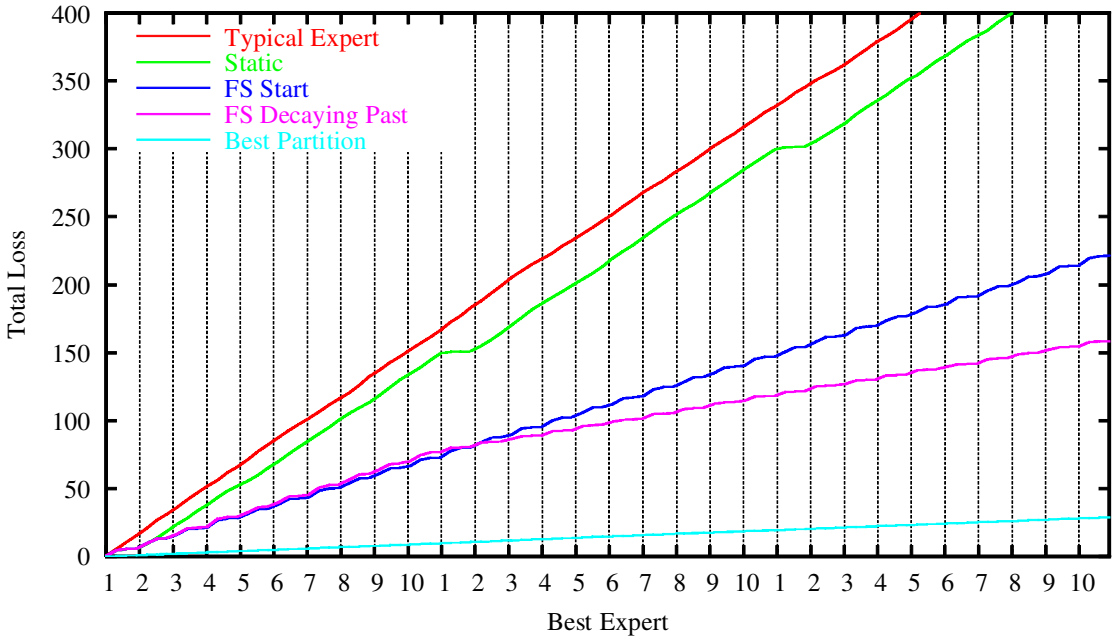


Figure 5: The memory effect appears even for larger pools and longer periods of time. The parameters are $T = 6000$, $n = 20000$, $k = 29$ (every 200 trials), $m = 10$.

6.6 Long Term versus Short Term Memory

In order to better understand the effect of the Mixing Update, we now consider the case of a single shift. The pool has size 2 and we start with expert 1. At a certain point, we shift to expert 2 and keep it for the rest of the trials. Figure 7 gives the total loss plots for this experiment. The interesting part of the experiment is when we monitor the weights maintained by the different mixing schemes (see Fig. 8). Indeed, the Fixed Share to Decaying Past displays two different forms of memory. The first one is a "short-term memory" due to the loss update. At the beginning of each section the weight of the good expert is picked up exponentially fast (linear in the log plot). Also the weight of the previously good expert decays exponentially fast at the beginning of the next section. The second is a "long-term memory" due to the Fixed Share to Decaying Past mixing scheme. It keeps the weight of the first expert at a higher level so that the recovery is faster if this expert is reused in the future. The decrease of the long term memory is much slower than that of the short term memory.

6.7 Cumulative Long-term Memory

The short and long-term memory can work together in intricate ways. We consider again a pool of size 2. Now the first expert is used in large sections of 200 trials while the second expert is used in smaller sections of size 50. The two experts are alternately active. Once again the total loss plots show the superiority of the Fixed Share to Decaying Past mixing scheme. Also, this setting seems to be poorly handled by the Fixed Share to Start

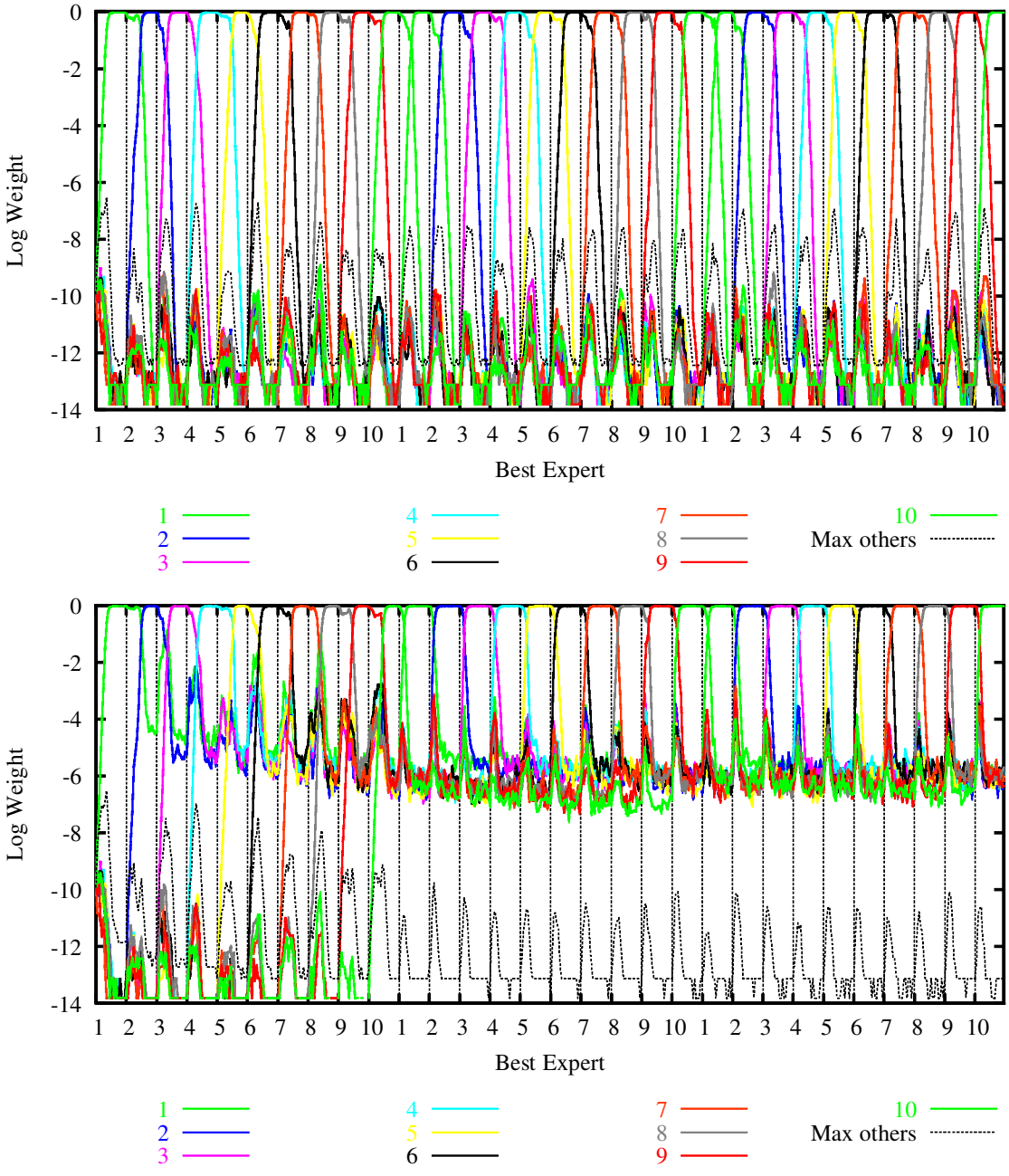


Figure 6: Log weights for the different updates (top: Fixed Share to Start Vector, bottom: Fixed Share to Decaying Past). "max others" is the maximum weight of any expert outside the pool of 10 experts.

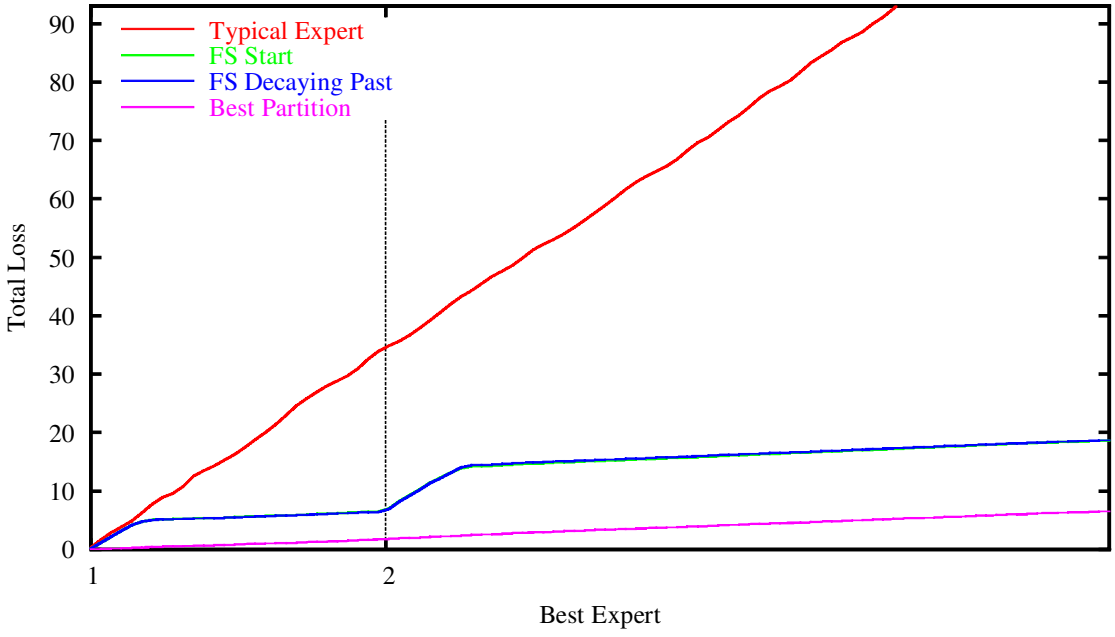


Figure 7: With only one shift the two Fixed Share schemes perform identically. The parameters are $T = 1400$, $n = 20000$, $k = 1$, $m = 2$.

Vector mixing scheme since it is outperformed by the simple Static Experts Algorithm which essentially treats the short experts as noise.

In order to understand this phenomenon, we can look at the plots of the weights on a logarithmic scale (Fig. 10). It turns out that the small sections are too small for the weight of the second expert to get high enough. Thus the Fixed Share to Start Vector is unable to switch between the two experts. The Fixed Share to Decaying Past mixing scheme faces the same problem in the first few sections. However, the long-term memory effect is somewhat cumulative which allows the weight of the second expert to be ratcheted up to a level which increases over time so that after 5 short sections the intermittent expert can recover in the short sections.

6.8 Parallel Cumulative Memory Effect

Interestingly enough, the cumulative memory effect can happen simultaneously for a number of experts. We performed an experiment where the pool contains 9 experts. During all odd numbered sections expert 1 is active and the length of these sections is all 140 (which is long). The even numbered sections are only of length 60 each and we cycle through the remaining 8 experts in these sections. The pattern is thus: expert 1 for $T = 1, \ldots, 200$, expert 2 for $t = 201, \ldots, 260$, expert 1 again for $t = 261, \ldots, 400$, expert 3 for $t = 400, \ldots, 460$, and so on. The parameters of the experiment are $T = 21000$ trials, $n = 20000$ experts, $k = 208$ shifts, and $m = 9$ experts in the pool. The log-weight plots (Fig. 11) show that the weights of the interspersed experts 2 through 9 are ratcheted up in parallel.

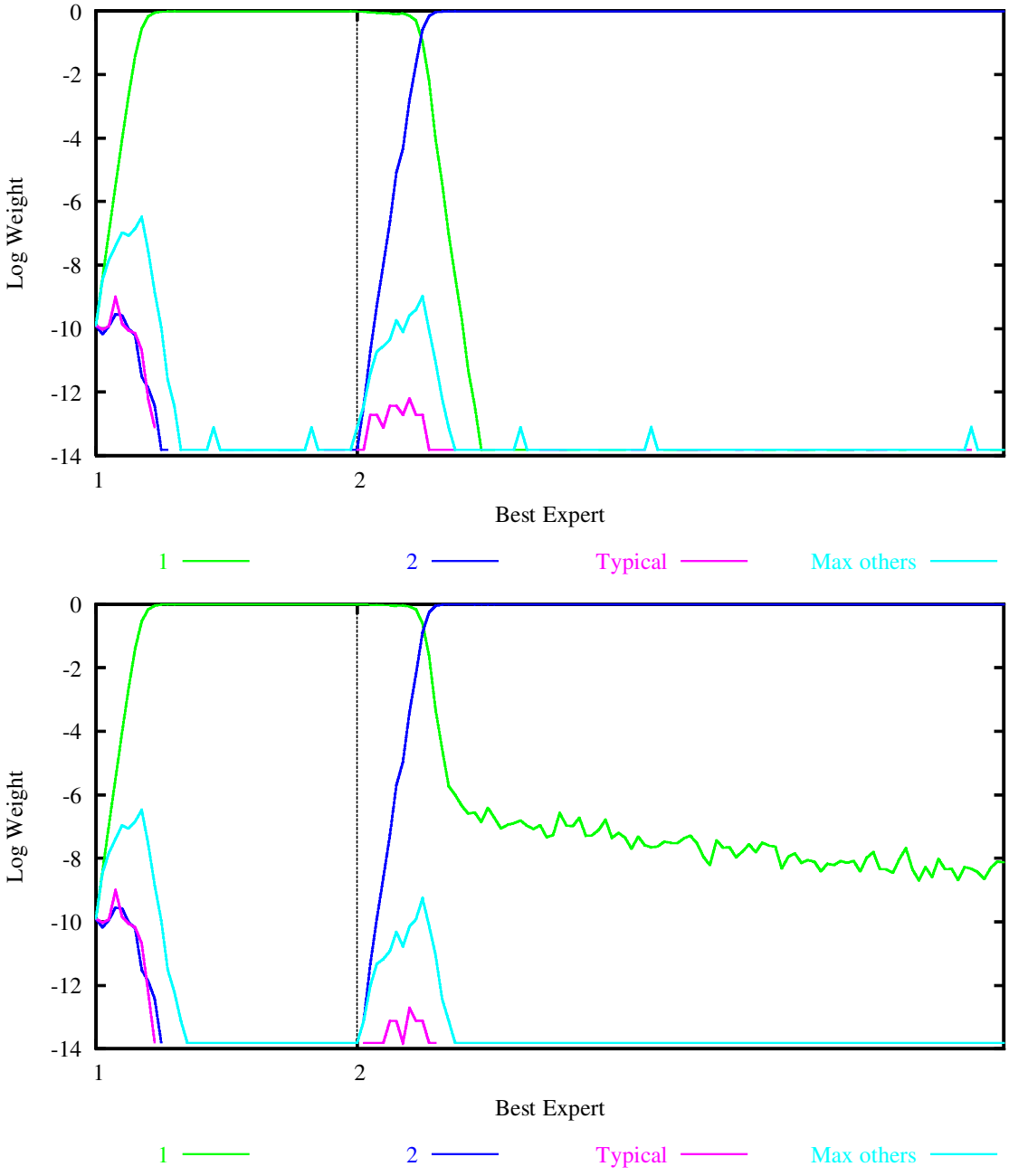


Figure 8: Short term versus long term memory effect (top: Fixed Share to Start Vector, bottom: Fixed Share to Decaying Past).

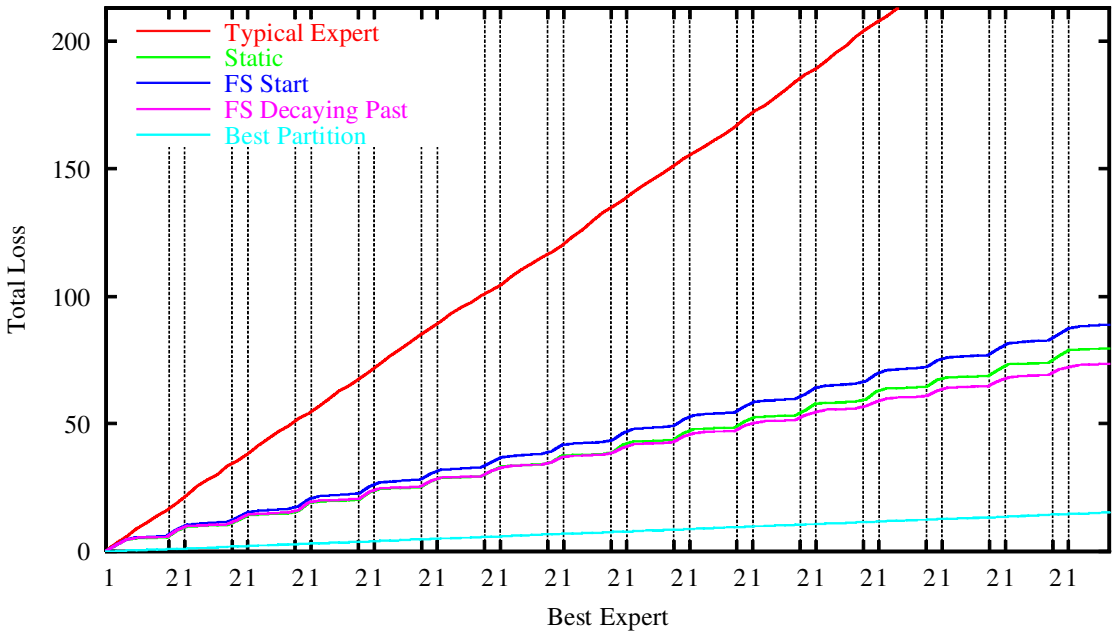


Figure 9: Sections with different sizes: long section of 200 trials with shorter ones of 50 trials interleaved. A cumulative memory effect appears across time. The parameters are $T = 3200$, $n = 20000$, $k = 30$, $m = 2$.

6.9 Variable Share

Finally, we implement the Variable Share version of the mixing schemes (Section 5.3) and, as expected, we get some improvement. For this we return to the setting of the basic experiment of section 6.4. Figure 12 compares the loss plots for the Fixed Share and Variable Share versions of the Start Vector and Decaying Past schemes.

The improvement is even clearer when one looks at the weight plots of Fig. 13 in comparison to those of Fig. 3.

7. Conclusion

Building on the work of Herbster and Warmuth, we have shown that by mixing the past posteriors, we can significantly reduce the cost of comparator shifts when the comparators are from a small pool of convex combinations of the experts. We showed that the total loss for the Fixed Share to Decaying Past mixing scheme is at most the loss of the best partition plus the number of bits needed to encode the best partition (including the boundaries of the sections) plus (a second time) the number of bits needed to encode the boundaries. A good approximation of this mixing scheme requires time and space $O(n \ln t)$ at trial t.

We are investigating whether the cost of paying for the boundaries a second time can be reduced. The off-line problem of finding a partition with k shifts from a pool of m convex combinations and small loss is NP-hard. However we do not know how to obtain lower bounds from this hardness result.

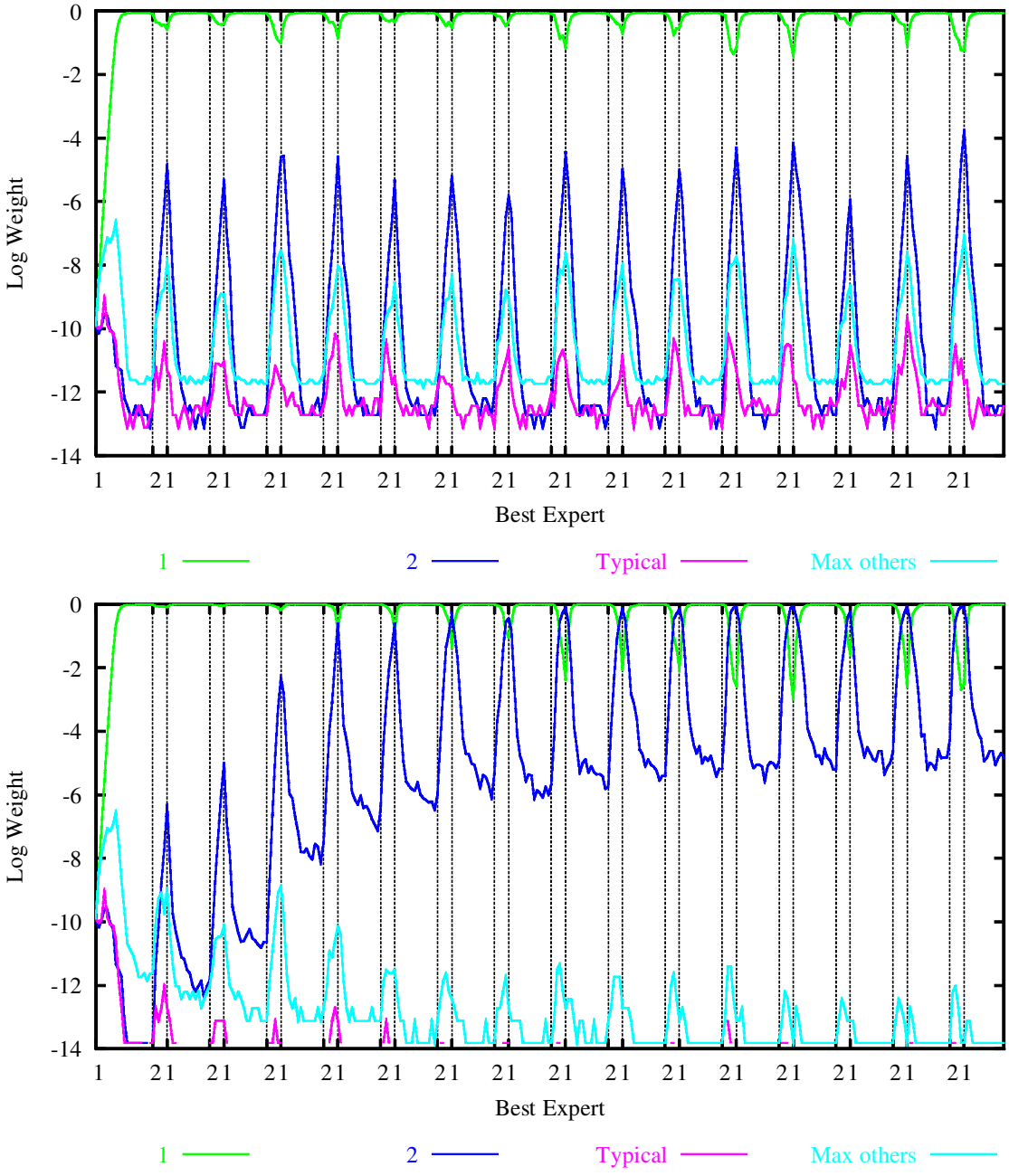


Figure 10: An illustration of the cumulative memory effect (top: Fixed Share to Start Vector, bottom: Fixed Share to Decaying Past).

Another theoretical question is whether the new mixing schemes can be explained in terms of priors over partition experts. Indeed, for the Fixed Share Algorithm of Herbster and Warmuth (1998) (called Fixed Share to Start Vector mixing scheme in this paper) it has been proven by e.g. Vovk (1999), Herbster (1998) that using one weight per partition gives an update that collapses to the latter efficient algorithm. However for the general mixing update we only gave a partial Bayesian interpretation in this paper (see Appendix A).

As discussed in Section 5, the Mixing Update is rather robust. We showed that mixing schemes can be mixed to form new schemes that are not much worse than the best of the original schemes.

The mixing schemes we propose can be extended in various ways. For example one could cluster the past weight vectors in order to limit the number of weights to be stored and to improve the identification of the convex combinations in the best comparator sequence. Also, one could incorporate prior and on-line knowledge about how the best convex combination is changing into the mixing schemes. One way of doing so is to modify the decay coefficients.

Another open question is whether the parameters α, γ, and (in the case of absolute loss) η can be tuned on-line using techniques from on-line learning (see e.g. Cesa-Bianchi et al., 1998, Auer et al., 2000) and universal coding (see e.g. Willems, 1996, Shamir and Merhav, 1999). Following Herbster and Warmuth (1998), slightly improved upper bounds should be obtainable for the Variable Share modification of the updates when the losses of the experts lie in $[0, 1]$. So far we could only show that the Variable Share modifications perform better experimentally. Also it should be possible to formally prove lower bounds on the loss of any on-line algorithm in terms of the number of bits needed to encode the partition.

Finally, the loss update analyzed in this paper belongs to the *Exponentiated Gradient* (EG) family of updates (see e.g. Kivinen and Warmuth, 1999).[2] Any update in the EG family is derived and analyzed with the relative entropy as a measure of progress (see e.g. Kivinen and Warmuth, 1997). Thus by Lemma 2 the mixing update can be used with any member of this family such as EG with square loss for linear regression (see e.g. Kivinen and Warmuth, 1997) or normalized Winnow (see e.g. Helmbold et al., 1999).

A next goal would be to adapt the mixing updates to other families of updates such as EGU family (analyzed with the unnormalized entropy) and the BEG family (analyzed with the componentwise sum of binary entropies) (see e.g. Kivinen and Warmuth, 1997, Bylander, 1997).

Acknowledgments

This research was done while the first author was visiting UC Santa Cruz. The authors are grateful to Yoav Freund for posting an open problem which inspired this research. We also thank Mark Herbster for many inspiring discussions and John Langford for helping us strengthen Lemma 2. This research was supported by NSF grant CCR 9821087.

2. For any convex combination v of experts use the loss $v \cdot L$ where L is the vector of losses of the experts.

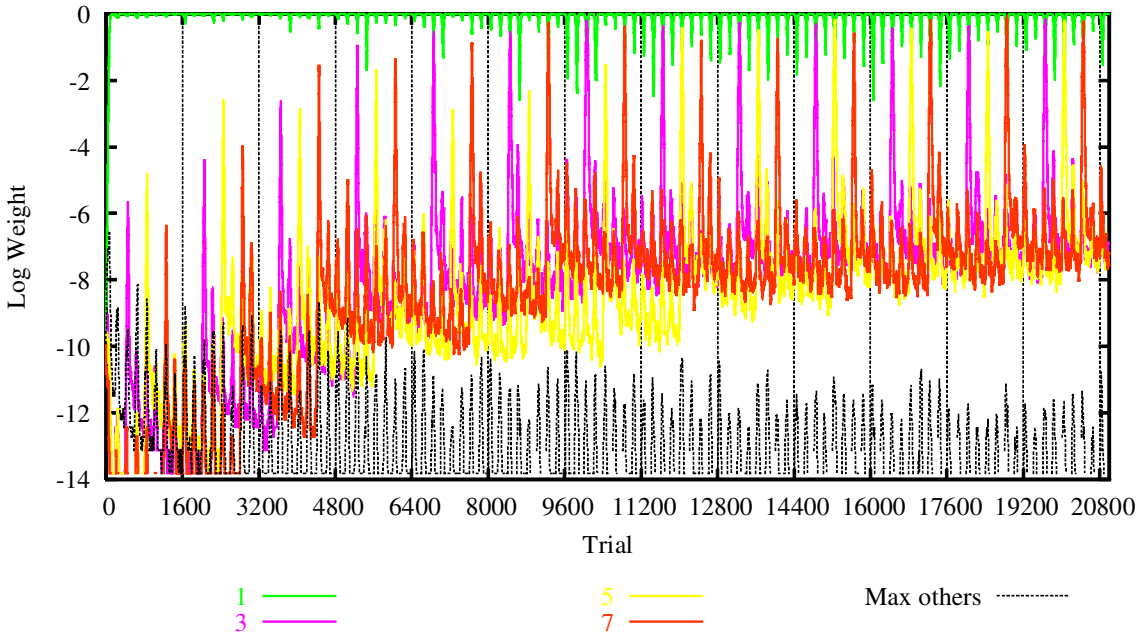


Figure 11: An illustration of the parallel cumulative memory effect with the Fixed Share to Decaying Past mixing scheme. Parameters: $T = 21000$, $n = 20000$, $k = 208$, $m = 9$. Only the weights of 4 of the 9 experts in the pool are displayed. The other ones have a similar behaviour.

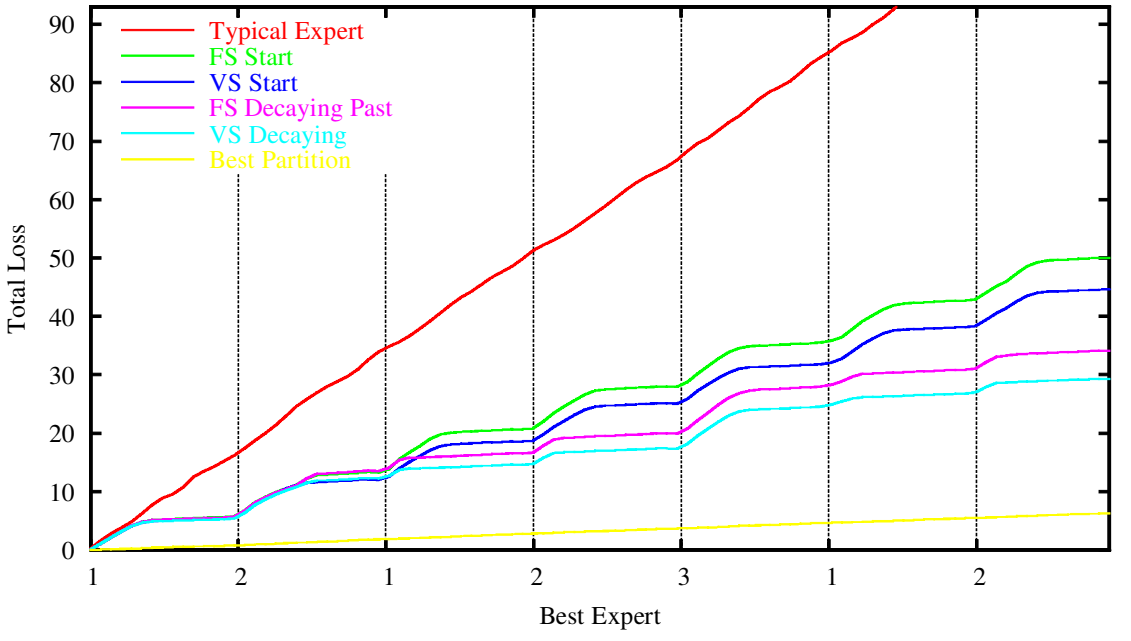


Figure 12: Comparison between Fixed Share and Variable Share mixing schemes.

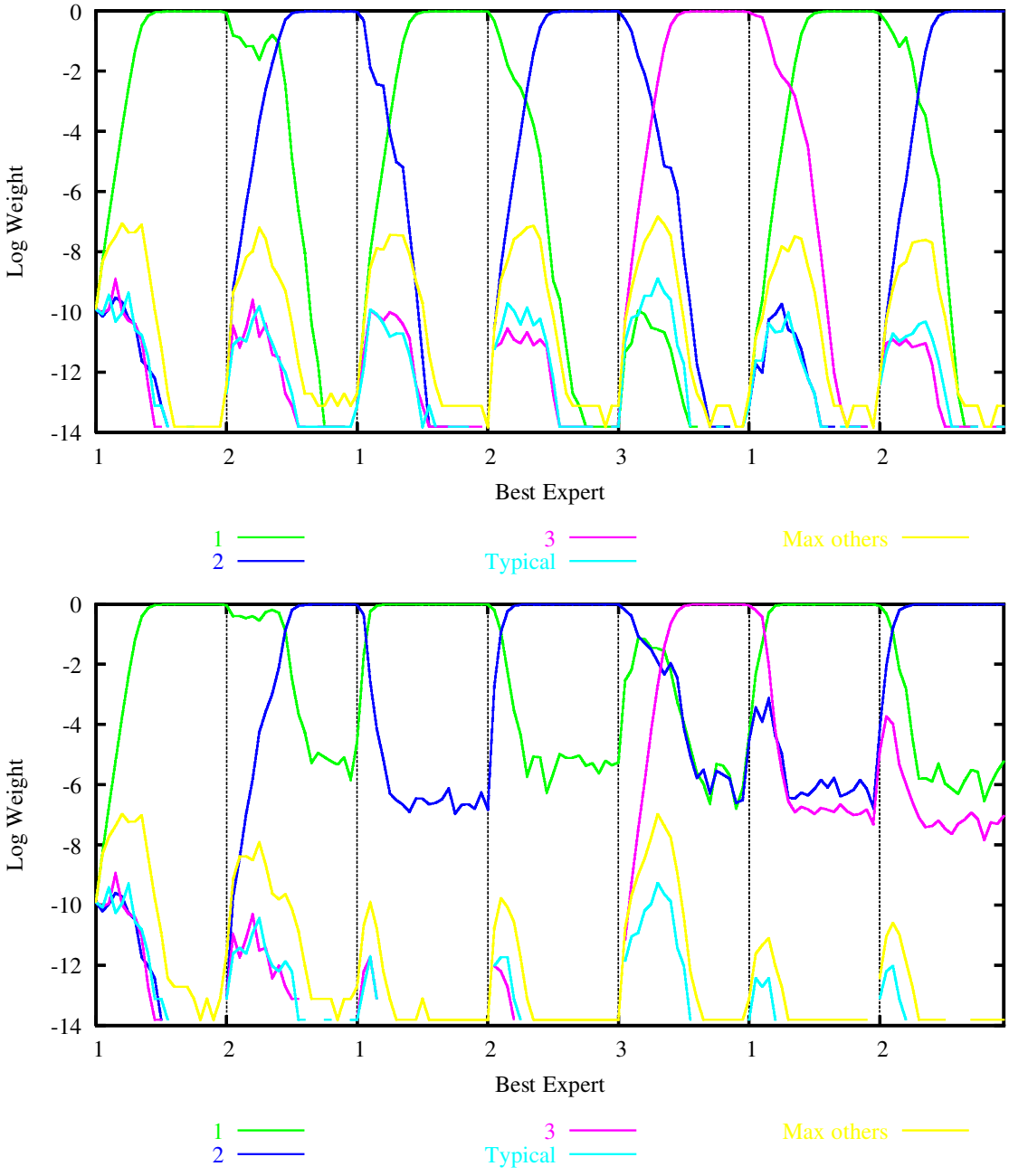


Figure 13: Log weights for the Variable Share version of the Mixing Update (top: Variable Share to Start Vector, bottom: Variable Share to Decaying Past).

Appendix A: A Bayesian Interpretation

In this section, we give a partial Bayesian interpretation of the updates given in paper. We interpret the i-th component of the weight vector as the probability that the current expert generating the outcomes is expert i, conditioned on the past outcomes that were already observed. It is known that in the case of the log-loss the standard loss update can be seen as a direct application of Bayes rule.

We prove that this still holds for the Fixed Share to Start Vector mixing scheme. However for the general Mixing Update we only obtain a Bayesian interpretations after some approximations.

Assume the following probabilistic model. At trial t an expert from $\{1, \ldots, n\}$ is chosen (denoted as the random variable E_t) and this expert generates a binary label (the random variable Y_t). We want to investigate under what assumptions does the following hold

$$v_{t,i} = P(E_t = i | y_{t-1}) \text{ and } v^m_{t,i} = P(E_t = i | y_t) \ . \tag{4}$$

Assume the loss function is the log-loss, i.e. $L_{t,i} = -\ln P(y_t | E_t = i, y_{t-1})$ and $\eta = 1$. By Bayes rule,

$$\begin{aligned} P(E_t = i | y_t) &= \frac{P(y_t | E_t = i, y_{t-1}) P(E_t = i | y_{t-1})}{\sum_{j=1}^n P(y_t | E_t = j, y_{t-1}) P(E_t = j | y_{t-1})} \\ &= \frac{e^{-\eta L_{t,i}} P(E_t = i | y_{t-1})}{\sum_{j=1}^n e^{-\eta L_{t,j}} P(E_t = j | y_{t-1})} \ . \end{aligned}$$

This is the loss update and is consistent with equations (4).

We now model how the expert E_{t+1} for the next trial is chosen. We pick a number $q \in \{0, \ldots, t\}$, where outcome q has probability $\beta_{t+1}(q)$. If q is non-zero, then we switch to the expert of trial q, i.e. $E_{t+1} = e_q$. If $q = 0$, then we switch to a random expert. This gives the following expression:

$$P(E_{t+1} = i | y_t) = \frac{\beta_{t+1}(0)}{n} + \sum_{q=1}^{t} \beta_{t+1}(q) P(E_q = i | y_t) \ . \tag{5}$$

Note that summing the r.h.s. of the above expression over $i = 1..n$ gives probability one. However for the above update we need to maintain the probabilities $P(E_q = i | y_t)$ for all $q = 1, \ldots, t$ and all $i = 1, \ldots, n$. We do not know how to do this efficiently.

In the case of the Fixed Share to the Start Vector mixing scheme, only $\beta_{t+1}(0) = \alpha$ and $\beta_{t+1}(t) = 1 - \alpha$ are non-zero and thus

$$P(E_{t+1} = i | y_t) = \frac{\alpha}{n} + (1 - \alpha) P(E_t = i | y_t) \ .$$

So via the weight interpretation (4) the above is the Share Update of the Fixed Share to the Start Vector mixing scheme.

In the general case, if we use the following approximation (for $0 < q < t$)

$$P(E_q = i | y_t) \approx P(E_q = i | y_q) \ ,$$

then

$$P(E_{t+1} = i|y_t) \approx \frac{\beta_{t+1}(0)}{n} + \sum_{q=1}^{t} \beta_{t+1}(q) P(E_q = i|y_q) \ .$$

Notice that the r.h.s. still sums to one when summed over i and the above is the Mixing Update with the weight interpretation of (4).

We do not know how to obtain a Bayesian interpretation for the Mixing Update without using the above approximation.

Appendix B: Proof of Corollary 9

We proceed as in the proof of Corollary 8. There are $T-k-1$ trials such that $u_t = u_{t-1}$. For all these trials $\beta_t(q_{t-1}) = \beta_t(t-1) = 1-\alpha$, contributing a total cost of $c(T-k-1)\ln\frac{1}{1-\alpha}$. In all the remaining trials a section is starting. For the first trial $\beta_1(0) = 1$ and no cost is incurred. All k other sections start at a trial $1 < t \leq T-1$ and $\beta_t(q_{t-1}) = \alpha\ \frac{1}{(t-1-q_{t-1})Z_{t-1}}$. The normalization factor Z_t can be bounded as follows:

$$Z_{t-1} \ = \ \sum_{q=0}^{t-2} \frac{1}{t-1-q} \ = \ \sum_{q=1}^{t-1} \frac{1}{q} \ \leq \ 1 + \int_1^{t-1} \frac{dx}{x} \ = \ 1 + \ln(t-1) \leq 1 + \ln(T-1).$$

So the last term of the bound of Theorem 7 is upper bounded by

$$c(T-1-k)\ln\frac{1}{1-\alpha} + ck\ln(1+\ln(T-1)) + ck\ln\frac{1}{\alpha} + c\sum_{i=1}^{k}\ln(t_i - 1 - q_{t_i-1}). \qquad (6)$$

To complete the proof it suffices to show the claim that the last sum above is at most $ck\ln\frac{(T-1)(m-1)}{k}$.

For each of the m convex combinations $\tilde{u}_j$, we compute the total length of all shift backs:

$$\sum_{\boldsymbol{u}_{t_i} = \tilde{\boldsymbol{u}}_j} (t_i - 1 - q_{t_i-1}) \leq T - 1 - s_j,$$

where s_j is the total length of all sections corresponding to $\tilde{u}_j$. By summing over all the elements of the pool we have

$$\sum_{i=1}^{k}(t_i - 1 - q_{t_i-1}) \leq m(T-1) - \sum_{j=1}^{k} s_j \ = \ (m-1)(T-1).$$

Since the last sum of (6) has k summands and since ln is concave, the sum is maximized when all shift backs have length $\frac{(T-1)(m-1)}{k}$. This proves the claim.

Appendix C: Keeping the Number of Weights Small

In this section we examine the storage requirement for the different updates and propose a method for reducing it.

Notice that the Static Experts Algorithm requires to maintain a vector of n weights. Ideally, we would like the other scheme to have a storage complexity of order $O(n)$.

For the Fixed Share to Uniform Past mixing scheme it suffices to store the average of the past vectors v_q^m (for $0 \leq q < t$). Indeed, it is possible to update this average simply based on the current weight vector. Thus only $2n$ weights have to be stored. More precisely, we have maintain the weight vector v_t and the vector of the average of all the past vectors, denoted r_t. The Mixing Update is then computed as

$$v_{t+1} = (1-\alpha)v_t^m + \alpha r_t \ ,$$

where the average r_t is computed via

$$r_{t+1} = \frac{(t-1)r_t + v_t^m}{t} \ .$$

For the Fixed Share Update to Decaying Past mixing scheme it is not possible to use such a strategy and one needs to store all the past weight vectors in order to perform the Mixing Update. The storage requirement at trial t is thus $O(nt)$.

However, we will sketch an approximate version of this update which has essentially the same loss bound while requiring only $O(n \ln t)$ weights to be stored.

In the case of the Uniform Past mixing scheme all past weight vectors have the same coefficient and thus can be collapsed into an average weight vector. For the best bounds we need some decaying towards the past. However the value of the coefficients for the past weight vectors only enters logarithmically into the bounds. So we can group past posteriors into large blocks. For each block we only keep the average weight vector and the mixture coefficient for the whole block is the smallest mixture coefficient of all vectors in the block.

We maintain a linked list of blocks whose lengths are powers of 2. This list contains at most two blocks for each power of 2 and at least one block for each power (up to the maximal power). For each block only the average weight vector is stored. Each time a new weight vector is added, a new block with power zero is created and added to the end of the list. If there are already three blocks of power 0, then the previous two blocks of power 0 are collapsed into a block with power 1. The algorithm proceeds down the list. If a third block of power q is created, then the previous two are collapsed into a block of power $q+1$. Whenever two blocks are collapsed, their weight vectors are averaged.

It can be proven that the maximum number of nodes in the list is $1 + 2\lfloor \log t \rfloor$. Also the cost per boundary in Theorem 7 is $\ln \frac{1}{\beta_{t+1}(q_t)}$ and when applying the above method to the Decaying Past mixing scheme this cost is increased by at most $\gamma c \ln 2$.

Appendix D: Tuning Additional Parameters for the Decaying Past Mixing Scheme

In this section we introduce a sophisticated version of the Decaying Past mixing scheme which gives better performance in practice. This scheme is controlled by three parameters α, δ and γ for which we will give optimal values.

The scheme of interest is defined as follows. For $t = 1, \ldots, T$, let α and δ be real numbers in $(0,1)$ and let γ be a positive real number such that $\gamma \neq 1$. We define

$$\beta_{t+1}(t) = 1 - \alpha \ ,$$

and for $0 < q < t$,

$$\beta_{t+1}(q) = (1-\delta)\alpha \, \frac{1}{(t-q)^\gamma} \frac{1}{Z_t} \ ,$$

and

$$\beta_{t+1}(0) = \alpha\delta \, .$$

Like in the original scheme, Z_t is a normalizing factor defined as

$$Z_t = \sum_{q=1}^{t-1} \frac{1}{(t-q)^\gamma} \ .$$

Now let's state the bound corresponding to this scheme, which is also a corollary of Theorem 7.

Corollary 11 *For the Mixing Algorithm using the above mixing scheme , for $\alpha, \delta \in (0,1)$ and $\gamma > 0, \gamma \neq 1$, we have for any sequence of T comparison vectors $\boldsymbol{u}_t$ with k shifts from a pool $\{\tilde{\boldsymbol{u}}_1, \ldots, \tilde{\boldsymbol{u}}_m\}$ of m convex combinations, we have*

$$\begin{aligned} L_{1..T,A} \ \leq \ & \sum_{t=1}^{T} L_t \cdot \boldsymbol{u}_t + c \sum_{j=1}^{m} \left(\Delta(\tilde{\boldsymbol{u}}_j, \frac{1}{n}\mathbf{1}) - \Delta(\tilde{\boldsymbol{u}}_j, \boldsymbol{v}_{\ell_j}^m) \right) \\ & + ck \ln \frac{1}{\alpha} + c(T-k-1) \ln \frac{1}{1-\alpha} \\ & + c(m-1) \ln \frac{1}{\delta} + c(k-m+1) \ln \frac{1}{1-\delta} \\ & + c(k-m+1) \left(\gamma \ln \frac{(T-1)(m-1)}{k-m+1} + (1-\gamma) \ln(T-1) + \ln \frac{1}{1-\gamma} \right) \ . \end{aligned}$$

Moreover, the optimal choices of the parameters are

$$\alpha^* = \frac{k}{T-1} \ ,$$

$$\delta^* = \frac{m-1}{k} \ ,$$

and

$$\gamma^* = \frac{\ln \frac{k-m+1}{m-1} - 1}{\ln \frac{k-m+1}{m-1}} < 1 \ .$$

Choosing these values for the parameters, the bound becomes

$$\begin{aligned} L_{1..T,A} \ \leq \ & \sum_{t=1}^{T} L_t \cdot \boldsymbol{u}_t + c \sum_{j=1}^{m} \left(\Delta(\tilde{\boldsymbol{u}}_j, \frac{1}{n}\mathbf{1}) - \Delta(\tilde{\boldsymbol{u}}_j, \boldsymbol{v}_{\ell_j}^m) \right) \\ & + ck \ln \frac{T-1}{k} + c(T-k-1) \ln \frac{T-1}{T-k-1} \\ & + c(m-1) \ln \frac{k}{m-1} + c(k-m+1) \ln \frac{k}{k-m+1} \\ & + c(k-m+1) \left(\ln \frac{e(T-1)(m-1)}{k-m+1} + \ln \ln \frac{k-m+1}{m-1} \right) \ . \end{aligned}$$

Proof We proceed as in the proof of Corollary 9. There are $T-k-1$ trials such that $u_t = u_{t-1}$. For all these trials $\beta_t(q_{t-1}) = \beta_t(t-1) = 1-\alpha$, contributing a total cost of $c(T-k-1)\ln\frac{1}{1-\alpha}$. In all the remaining trials a section is starting. For the first trial $\beta_1(0) = 1$ and no cost is incurred. Among the remaining k trials where a section is started, $m-1$ correspond to the introduction of a new convex combination and the corresponding cost is $c\ln\frac{1}{\alpha\delta}$ while for the other $k-m+1$ trials, we have $\beta_t(q_{t-1}) = (1-\delta)\alpha\,\frac{1}{(t-1-q_{t-1})^\gamma Z_{t-1}}$. The trials t where the sections are started verify $1 < t \le T-1$ and we can bound the normalization factor as

$$Z_{t-1} = \sum_{q=1}^{t-2}\frac{1}{(t-1-q)^\gamma} = \sum_{q=1}^{t-2}\frac{1}{q^\gamma} \le 1 + \int_1^{t-2}\frac{dx}{x^\gamma}\ ,$$

so that

$$Z_{t-1} \le 1 + \frac{(T-1)^{1-\gamma}-1}{1-\gamma} \le \frac{(T-1)^{1-\gamma}}{1-\gamma}\ .$$

Then the last term of the bound of Theorem 7 is upper bounded by

$$\begin{aligned}
&c(T-k-1)\ln\frac{1}{1-\alpha} + c(k-m+1)(1-\gamma)\ln(T-1)\\
&+c(k-m+1)\ln\frac{1}{(1-\delta)\alpha(1-\gamma)} + c(m-1)\ln\frac{1}{\alpha\delta} + c\sum_{i\in I}\gamma\ln(t_i-1-q_{t_i-1})\ ,
\end{aligned}$$

where I denotes the set of indices i such that $q_{t_i} > 0$ (i.e. a convex combination from the pool is reused).

The same argument as in the proof of Corollary 9 shows that the last sum above which contains $k-m+1$ elements is upper bounded by $c\gamma(k-m+1)\ln\frac{(T-1)(m-1)}{k-m+1}$ which completes the proof of the first part of the corollary.

Now differentiating the bound with respect to the various parameter yields by simple algebra the optimal values

$$\alpha^* = \frac{k}{T-1}\ , \qquad \delta^* = \frac{m-1}{k}\ ,$$

and

$$\gamma^* = \frac{\ln\frac{k-m+1}{m-1}-1}{\ln\frac{k-m+1}{m-1}} < 1\ .$$

Replacing these values in the bound yields

$$\begin{aligned}
L_{1..T,A} \ \le\ & \sum_{t=1}^{T} L_t\cdot u_t + c\sum_{j=1}^{m}\left(\Delta(\tilde{u}_j,\frac{1}{n}\mathbf{1}) - \Delta(\tilde{u}_j, v^m_{\ell_j})\right)\\
&+ck\ln\frac{T-1}{k} + c(T-k-1)\ln\frac{T-1}{T-k-1}\\
&+c(m-1)\ln\frac{k}{m-1} + c(k-m+1)\ln\frac{k}{k-m+1}\\
&+c(k-m+1)\left(1+\ln\frac{(T-1)(m-1)}{k-m+1} + \ln\ln\frac{k-m+1}{m-1}\right)\ ,
\end{aligned}$$

which concludes the proof. ■

References

P. Auer, N. Cesa-Bianchi, and C. Gentile. Adaptive and self-confident on-line learning algorithms. NeuroCOLT Technical Report NC-TR-00-083, 2000. An extended abstract appeared in *Proc. 13th Ann. Conf. Computational Learning Theory.*

P. Auer and M. K. Warmuth. Tracking the best disjunction. *Journal of Machine Learning*, 32(2):127–150, August 1998. Special issue on concept drift.

T. Bylander. The binary exponentiated gradient algorithm for learning linear functions. In *Proc. of the 10th Annu. Conf. on Comput. Learning Theory*, 184–192, ACM, 1997.

N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth. How to use expert advice. *Journal of the ACM*, 44(3):427–485, 1997.

N. Cesa-Bianchi, D. P. Helmbold, and S. Panizza. On Bayes methods for on-line boolean prediction *Algorithmica*, 22(1/2):112–137, 1998.

Y. Freund. Private Communication, 2000.
Also posted on `http://www.learning-theory.org/` .

M. R. Garey and D. J. Johnson. Computers and intractability: a guide to the theory of NP-completeness. W. H. Freeman, 1979.

D. Haussler, J. Kivinen, and M. K. Warmuth. Sequential prediction of individual sequences under general loss functions. IEEE Transactions on Information Theory, 44(2):1906–1925, September 1998.

D. Helmbold, S. Panizza, M. K. Warmuth. Direct and indirect algorithms for on-line learning of disjunctions. In Paul Fischer and Hans Ulrich Simon, editors, *Computational Learning Theory: 4th European Conference (EuroCOLT '99)*, Berlin, March 1999. Springer. To appear in *Theoretical Computer Science.*

M. Herbster. Private Communication, 1998.

M. Herbster and M. K. Warmuth. Tracking the best expert. *Journal of Machine Learning*, 32(2):151–178, August 1998. Special issue on concept drift.

M. Herbster and M. K. Warmuth. Tracking the best linear predictor. In *Journal of Machine Learning Research*, 1:281-309, 2001.

J. Kivinen and M. K. Warmuth. Additive versus exponentiated gradient updates for linear prediction. In *Journal of Information and Computation*, 132(1):1–64, January 1997.

J. Kivinen and M. K. Warmuth. Averaging expert predictions. In Paul Fischer and Hans Ulrich Simon, editors, *Computational Learning Theory: 4th European Conference (EuroCOLT '99)*, pages 153–167, Berlin, March 1999. Springer.

N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

G. Shamir and N. Merhav. Low complexity sequential lossless coding for piecewise stationary memoryless sources. *IEEE Trans. Info. Theory*, 45:1498–1519, 1999.

Y. Singer. Switching portfolios. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI'98)*, pages 488–495, Morgan Kaufmann, San Francisco, CA, 1998.

V. Vovk. Aggregating strategies. In *Proc. 3rd Annu. Workshop on Comput. Learning Theory*, pages 371–383. Morgan Kaufmann, 1990.

V. Vovk. A game of prediction with expert advice. *Journal of Computer and System Sciences*, 56:153–173, 1998.

V. Vovk. Derandomizing stochastic prediction strategies. *Machine Learning*, 35(3):247–282, 1999.

F. M. J. Willems. Coding for a binary independent piecewise-identically-distributed source. *IEEE Trans. Info. Theory*, 42(6):2210–2217, 1996.