Algorithmica (1998) 22: 112–137

Algorithmica

On Bayes Methods for On-Line Boolean Prediction[1]

N. Cesa-Bianchi,[2] D. P. Helmbold,[3] and S. Panizza[2]

Abstract. We examine a general Bayesian framework for constructing on-line prediction algorithms in the experts setting. These algorithms predict the bits of an unknown Boolean sequence using the advice of a finite set of experts. In this framework we use probabilistic assumptions on the unknown sequence to motivate prediction strategies. However, the relative bounds that we prove on the number of prediction mistakes made by these strategies hold for any sequence. The Bayesian framework provides a unified derivation and analysis of previously known prediction strategies, such as the Weighted Majority and Binomial Weighting algorithms. Furthermore, it provides a principled way of automatically adapting the parameters of Weighted Majority to the sequence, in contrast to previous ad hoc doubling techniques. Finally, we discuss the generalization of our methods to algorithms making randomized predictions.

Key Words. Boolean prediction, On-line algorithms, Bayes theory.

1. Introduction. A fundamental problem in learning theory is to predict the bits of an unknown Boolean sequence. The problem is uninteresting when the algorithm is required to minimize its worst-case number of mistakes over all sequences, as no algorithm can do better than random guessing. A richer problem results if the algorithm is given a (finite) set of models and the sequence is reasonably close to that generated by one of the models. Now interesting "relative" mistake bounds that depend on the distance between the unknown Boolean sequence and the closest model can be proven. This is sometimes referred to as the "experts" setting, since the models can be viewed as "experts" providing "advice" to the algorithm. Variants and extensions of this experts setting have been extensively studied by Littlestone and Warmuth [10], Vovk [12], Cesa-Bianchi et al. [2], [3], Haussler et al. [6], and others in the area of computational learning theory. Here we use a Bayesian approach to derive prediction algorithms with good performance in the experts settings. A crucial aspect of this work is that although the algorithms are derived by making probabilistic assumptions about the generation of the sequence to be predicted, they are analyzed in the adversarial experts setting.

In this experts setting, a "master algorithm" attempts to predict, one by one, the bits of an unknown sequence. Before predicting each bit, the master is allowed to listen to the "advice" provided by a pool of N experts. After each bit is revealed, the master

[1] A preliminary version of this paper appeared in the *Proceedings of the* 9*th Annual Conference on Computational Learning Theory*, ACM Press, New York, 1996. Part of this research was done while the first author was visiting the Institute for Theoretical Computer Science of the Technische Universitaet Graz, Austria, and the third author was visiting the University of California, Santa Cruz, USA. The first and third authors gratefully acknowledge support of ESPRIT Working Group NeuroCOLT 8556 on Neural and Computational Learning.
[2] DSI, University of Milan, via Comelico 39, 20135 Milano, Italy. {cesabian,panizza}@dsi.unimi.it.
[3] Computer Science Department, University of California, Santa Cruz, CA 95064, USA. dph@cse.ucsc.edu.

Received February 5, 1997; revised July 17, 1997. Communicated by P. Auer and W. Maass.

incurs a *loss* measuring the discrepancy between its prediction and the corresponding bit. Similarly, each of the experts incurs a loss based on its advice. For Boolean predictions we measure the performance of the experts and of the algorithm with the 0-1 loss or *discrete* loss, which simply counts the number of prediction mistakes. Instead, when the master's predictions range in the continuous interval [0,1] we use the *absolute* loss $|y - \hat{y}|$, where $\hat{y}$ is the master's prediction for bit y. Our goal is to design master algorithms whose total loss on *arbitrary* sequences is within a constant factor to that of the best expert. Thus the bounds proven are *relative* loss bounds, since they typically depend on both the number of experts and the loss of the best expert on the sequence observed. Although the bounds hold for all sequences, the value of the bound can change as the loss of the best expert increases.

Several algorithms with good relative loss bounds associate a weight with each expert and predict with a weighted combination of the experts' advice. After each bit is revealed, these algorithms slash the weights of those experts giving bad advice by a multiplicative *update factor*. We call these algorithms "multiplicative" since they use multiplicative factors to update the weights of the experts. The essential property of these multiplicative algorithms is that experts making many mistakes get their weights rapidly slashed, thus reducing their influence on the voting.

Most multiplicative algorithms use a parameter estimating the loss of the best expert on the sequence to tune their update factors. When tuned optimally, multiplicative algorithms have asymptotically optimal relative loss bounds for some 0-1 loss [3], [10] and absolute loss [2], [7] settings. Vovk [13] shows that multiplicative algorithms are optimal in a different way. He shows that if any master algorithm can achieve the relative loss bound $aL + b \log N$ (where L is the loss of the best expert, N is the number of experts, and a and b are constants), then a properly tuned multiplicative algorithm has the same relative loss bound.

In the simplest case where all predictions and outcomes are Boolean, two main multiplicative algorithms for combining the predictions of the experts have been proposed: the basic Weighted Majority (WM) algorithm [10], and the Binomial Weighting (BW) algorithm [3]. Although both algorithms use the weights in a similar way, their different update factors makes the form of their weights quite different. WM uses a fixed update factor which leads to weights that are in exponential form. In contrast, BW uses a variable update factor which depends on the current mistake count of the expert as well as on the current mistake count of the master. This update method leads to weights that are sums of binomial tails rather than exponentials.

Another difference between the WM and BW algorithms is that BW does not update the weights of the experts when it predicts correctly. Algorithms which only update the weights when they predict incorrectly are called *conservative*. The bounds for WM apply whether or not the algorithm is run conservatively.

Cesa-Bianchi et al. [3] have compared the relative bounds of the WM and BW algorithms. They showed that if both algorithms are properly tuned, then the BW algorithm has the better 0-1 relative loss bound in the experts setting. Thus, under the 0-1 loss, binomial weighting schemes seem to encode more useful information about the experts than exponential weighting schemes. In many situations (see [3]) this enables the BW algorithm to follow more quickly the predictions of the best expert and thus reduce its additional loss over the loss of the best expert.

Although both the WM and the BW algorithms are tuned through a parameter estimating the loss of the best expert, the BW algorithm relies more heavily on this information. One can easily obtain bounds on the performance of the WM algorithm, even when it is poorly tuned. In contrast, the BW algorithm is not robust: it can fail if the best expert has loss exceeding the given estimate. Thus a desirable goal is to develop on-line multiplicative algorithms that retain, at least asymptotically, the optimality properties of binomial weights while gaining the robustness of exponential weights. The purpose of this paper is to show the existence of such on-line learning algorithms and to provide more insight about the properties shared by binomial and exponential weighting schemes.

The first step in unifying the binomial and exponential weighting schemes is to consider an on-line prediction problem where explicit probabilistic assumptions are made about the sequence to be predicted. In particular, we assume that the unknown sequence is generated by first selecting an expert at random according to some prior distribution over the experts set and then by corrupting the selected expert's predictions with a noise process.[4] Different Bayes optimal prediction algorithms result from different noise assumptions, and a simple condition on the noise process guarantees that the resulting Bayes optimal algorithm is equivalent to a weighting scheme on the experts. It is then straightforward to convert these Bayesian learning algorithms into the corresponding multiplicative algorithms. The multiplicative algorithms derived in this way are then analyzed within the adversarial framework of the experts setting. This procedure enables us to obtain new algorithms as well as previously known multiplicative algorithms such as WM and BW, providing more insight into various weighting schemes.

For example, when the noise process is i.i.d. with a known rate η (i.e., each bit in the unknown sequence differs from that predicted by the selected expert with fixed probability η) the Bayes optimal algorithm for the 0-1 loss reduces to the WM [10] algorithm. When the sequence to be predicted is of known length and is equally likely to be any of the sequences within Hamming distance K of the selected expert's predictions, then we obtain the BW algorithm.

If the noise process is i.i.d. with an unknown rate selected according to a beta distribution, then we obtain another family of Bayesian prediction algorithms using mean posterior estimates and the corresponding new family of multiplicative algorithms. We call this new family of multiplicative algorithms Bayesian Binomial Weighting (BBW) algorithms. Like the BW algorithm, this new family updates its weights using an update factor that depends not only on the current mistake count of the experts, but also on the current mistake count of the master algorithm. This leads to weights that are also in binomial form. Although the original Bayesian algorithms assume a *prior* distribution on the noise rate, the derived BBW algorithms are robust, and their performance in the experts setting can be bounded even when this prior grossly misestimates the true behavior of the experts on the bit sequence.

Like the BW algorithm, we can prove better bounds for this new family of algorithms when they are run conservatively, i.e., they ignore those trials where they predict correctly and only update the experts' weights when an incorrect prediction is made.

We have also derived an adversarial technique that enables us to study in more detail

[4] Littlestone [8] has recently used a similar technique in a different setting to obtain an algorithm for predicting linearly separable boolean sequence (see the discussion in Section 4.2).

Table 1. Summary of notation used in the paper.

Notation	Description
$E_i(\boldsymbol{y}^{t-1})$	Prediction of expert E_i at time t given the sequence $\boldsymbol{y}^{t-1}$
$M_i(\boldsymbol{y}^t)$	Number of prediction mistakes made by expert E_i on the sequence $\boldsymbol{y}^t$
$M_{\text{best}}(\boldsymbol{y}^t)$	Number of prediction mistakes made by the best expert on the sequence $\boldsymbol{y}^t$
$\widehat{M}_i(\boldsymbol{y}^t)$	Number of prediction mistakes made by expert E_i on those bits of $\boldsymbol{y}^t$ that are also incorrectly predicted by the master
$\widehat{M}_{\text{best}}(\boldsymbol{y}^t)$	The minimum (over the experts) of the $\widehat{M}_i(\boldsymbol{y}^t)$ values
η	Noise rate (used in Section 3)
$\eta(\cdot,\cdot)$	Noise model (used in Section 4)

the performance of the BBW algorithms under the absolute loss. Despite the fact that we derive our technique for a special family of algorithms, we believe that this style of analysis can give insight into the performance of other algorithms as well. Although the performance bounds for this new family improve when the absolute loss measure (instead of the 0-1 loss) is used, we have been unable to show the factor of $\frac{1}{2}$ improvement that the exponential update algorithms exhibit (see [2]).

The paper is organized as follows. Section 2 describes the experts setting. The Bayesian framework and some families of Bayes optimal algorithms are discussed in Section 3. These algorithms are turned into (conservative) algorithms for the on-line prediction model in Section 4, where we also compare their mistake bounds. Finally, in Section 5 we study the performance of these on-line prediction algorithms with respect to the absolute loss.

Notation. Throughout, log and ln denote the binary and natural logarithms, respectively, and $\mathbb{N}$ denotes the nonnegative integers. Let $\{0,1\}^*$ be the set of all Boolean sequences of finite (including zero) length and let $\{0,1\}^\infty$ be the set of all Boolean sequences of infinite length. We use $\boldsymbol{y}$ to denote a Boolean sequence of finite unspecified length and $\boldsymbol{y}^\ell$ to denote a Boolean sequence of length ℓ. When $\boldsymbol{y}$ is set by the surrounding context, $\boldsymbol{y}^t$ denotes the length t prefix of $\boldsymbol{y}$ and y_t denotes the tth bit of $\boldsymbol{y}$. The empty sequence of length 0 is denoted by $\boldsymbol{y}^0$. An *on-line* (*Boolean*) *predictor* is any function from $\{0,1\}^*$ to $\{0,1\}$. For an on-line predictor A we define

$$M_A(\boldsymbol{y}^\ell) = |\{t : A(\boldsymbol{y}^{t-1}) \neq y_t, 1 \leq t \leq \ell\}|$$

as the number of prediction mistakes made by A on the sequence $\boldsymbol{y}^\ell$. Table 1 contains a summary of some other notation introduced later in the paper.

2. An Overview of the Experts Setting. In the on-line prediction framework of Littlestone and Warmuth [10] there are N *experts* $\{E_1, \ldots, E_N\}$, each of which is an on-line predictor E_i: $\{0,1\}^* \to \{0,1\}$. At each time step t, the "master predictor" combines the experts' *advice*, $E_1(\boldsymbol{y}^{t-1}), \ldots, E_N(\boldsymbol{y}^{t-1})$, to produce its own prediction $\hat{y}_t$ for bit y_t. Note that the advice of each expert can depend on the previously seen bits of $\boldsymbol{y}$ (but

not on the "future" ones). Two examples of experts allowed by the above definition are the one that always predicts the previous bit of $\boldsymbol{y}$, and the expert that predicts with the exclusive-or of the previously seen bits. Furthermore, the master algorithm does not get the structure or definition of the various experts, but only their advice on the current (and previous) bit(s) of $\boldsymbol{y}$.

In the simple case where the master's predictions are boolean, i.e., $\hat{y}_t \in \{0, 1\}$, the performance of the master algorithm is measured by the number of mistakes it makes when both the sequence of bits and the predictions of the experts are chosen by an adversary. Thus the goal is to prove mistake bounds on the master algorithms which hold for all sequences. Clearly, some sequences (such as when all the experts predict perfectly) are much easier on the algorithm than others (such as when the experts' advice is not correlated with the bits to be predicted). One obvious way to measure the difficulty of a sequence is with the number of mistakes made by the best expert on the sequence. Thus meaningful mistake bounds in the experts setting depend on both N, the number of experts, and the number of mistakes made by the best expert on the actual sequence of bits observed. Although the setting is adversarial, it is inappropriate to call the analysis *worst case*, since the resulting bounds depend on the particular sequence observed. We use the term *relative bounds* (as advocated by Yoav Freund) for this style of analysis since the value of the algorithm's mistake bound is relative to the "difficulty" of the sequence being predicted.

Different algorithms for the on-line prediction model with experts have recently been proposed [2], [7], [10], [12], both for the simple setting in which all the predictions are Boolean and for the more general setting in which the experts' advice and/or the master's predictions are chosen in the interval [0,1]. All these algorithms share the same general multiplicative weighting scheme which we mentioned in the Introduction. For each expert E_i and for each time step t a weight $w_i(t)$ is maintained. These weights are used to combine the advice of the N experts on bit y_t in order to produce the master algorithm's own prediction. After receiving the actual value of y_t, the master algorithm may adjust the voting weight of each expert E_i, multiplying it by a suitable update factor.

As outlined above, the two main ingredients in designing master algorithms are the *weighting scheme*, used to weight the advice of the experts, and the *prediction function*, used by the master algorithm to convert the weighted average of experts' advice into a prediction. Bayes theory provides a clear and theoretically sound basis to derive master algorithms whose weighting schemes and prediction functions meet certain requirements.

3. The Bayesian Framework. Following the work of Haussler and Barron [5], in this section we present a general Bayesian framework for sequentially predicting Boolean sequences by combining the advice of a finite set of experts. Throughout the section we assume that a "noise model" is associated with each expert and that the sequence $\boldsymbol{y}^\ell$ is generated by first selecting an expert at random (according to some prior distribution Q over the set $\{E_1, \ldots, E_N\}$) and then by corrupting that expert's predictions using its associated noise model. In other words, each bit of the unknown sequence $\boldsymbol{y}^\ell = y_1, \ldots, y_\ell$ differs from that predicted by the selected expert with a probability that depends on the assumed noise model for that expert.

The Bayes decision rule is an optimal prediction strategy (i.e., it minimizes the probability of an incorrect prediction or the expected total number of mistakes) in this probabilistic setting. On each round t, this rule simply outputs the label with the highest posterior probability (having seen $\boldsymbol{y}^{t-1}$), i.e.,

$$\hat{y}_t = \arg\max_{y \in \{0,1\}} P(y \mid \boldsymbol{y}^{t-1}), \tag{1}$$

where P is a probability distribution over $\{0,1\}^\infty$ (equipped with the natural σ-algebra) that is determined by the experts' predictions and by the noise model. When P is a distribution over $\{0,1\}^\infty$, we use $P(\boldsymbol{y}^t)$ to denote the measure of the set of all infinite sequences with prefix $\boldsymbol{y}^t$.

Rule (1) is especially easy to compute when we express the probability $P(y \mid \boldsymbol{y}^{t-1})$ as a sum over the expert set $\{E_1, \ldots, E_N\}$,

$$P(y \mid \boldsymbol{y}^{t-1}) = \sum_{i=1}^{N} P(y \mid \boldsymbol{y}^{t-1}, E_i) P(E_i \mid \boldsymbol{y}^{t-1}).$$

To simplify our notation we abbreviate $P(\boldsymbol{y}^t \mid E_i)$ by $P_i(\boldsymbol{y}^t)$, so

$$P(y \mid \boldsymbol{y}^{t-1}) = \sum_{i=1}^{N} P_i(y \mid \boldsymbol{y}^{t-1}) P(E_i \mid \boldsymbol{y}^{t-1}).$$

The first factor in each summand, $P_i(y \mid \boldsymbol{y}^{t-1})$, is simply the probability, under expert E_i, that the tth bit of the sequence will be y given that the previous $t-1$ bits were $y_1, \ldots, y_{t-1}$. In particular, if $y = E_i(\boldsymbol{y}^{t-1})$, where $E_i(\boldsymbol{y}^{t-1})$ is the prediction of expert E_i at time t, then we can view the probability $P_i(y \mid \boldsymbol{y}^{t-1})$ as measuring the chance (under the assumed noise model for expert E_i) that the bit y_t is *not* corrupted. Similarly, $P_i(1-y \mid \boldsymbol{y}^{t-1})$ is the chance that the bit y_t is corrupted by the noise model associated with expert E_i. The second factor, $P(E_i \mid \boldsymbol{y}^{t-1})$, represents the posterior probability that expert E_i was selected given that we have observed the prefix $\boldsymbol{y}^{t-1}$. Thus each term in the sum $\sum_{i=1}^{N} P_i(y \mid \boldsymbol{y}^{t-1}) P(E_i \mid \boldsymbol{y}^{t-1})$ represents the contribution of one expert, weighted according to its posterior probability, to the chance that the next outcome will be $y \in \{0,1\}$. By applying the Bayes rule, we can compute the posterior probability $P(E_i \mid \boldsymbol{y}^{t-1})$ in terms of the prior over the experts, $Q(E_i)$, and the likelihood $P_i(\boldsymbol{y}^{t-1})$ of $\boldsymbol{y}^{t-1}$ under E_i,

$$P(E_i \mid \boldsymbol{y}^{t-1}) = \frac{P_i(\boldsymbol{y}^{t-1}) Q(E_i)}{\sum_{j \in \{1,\ldots,N\}} P_j(\boldsymbol{y}^{t-1}) Q(E_j)}, \tag{2}$$

where $P_i(\boldsymbol{y}^0)$ is defined to be 1. Although the following trivially generalizes to nonuniform priors, the presence of the prior complicates the notation and obscures the points we wish to make. Therefore we now assume that the prior $Q(\cdot)$ is uniform, so $Q(E_i) = 1/N$ for $i = 1, \ldots, N$. Since the uniform prior makes the factor multiplying $P_i(\boldsymbol{y}^{t-1})$ in (2) the same for all i, the Bayes optimal prediction $\hat{y}_t$ can be written in the following equivalent form:

$$\hat{y}_t = \arg\max_{y \in \{0,1\}} \sum_{i=1}^{N} P_i(y \mid \boldsymbol{y}^{t-1}) P_i(\boldsymbol{y}^{t-1}). \tag{3}$$

On-Line Predictor GBP

Input: A set $E_1, \ldots, E_N$ of experts and a noise model defining distributions $P_1, \ldots, P_N$ over $\{0,1\}^\infty$.

For $t = 1, 2 \ldots$

1. If $\sum_{i=1}^N P_i(1 \mid \boldsymbol{y}^{t-1}) P_i(\boldsymbol{y}^{t-1}) \geq \sum_{i=1}^N P_i(0 \mid \boldsymbol{y}^{t-1}) P_i(\boldsymbol{y}^{t-1})$, then predict 1 for bit y_t; otherwise predict 0.
2. Observe bit y_t.

Fig. 1. The general Bayesian predictor GBP.

As shown by (3), the Bayesian predictor uses the distributions $\{P_i\}_{i=1}^N$ to determine the conditional probabilities of each label, and predicts with the most likely one. The "general Bayesian predictor" or GBP which is derived from schema (3) is summarized in Figure 1. Observe that the prediction $\hat{y}_t$ chosen by GBP minimizes the "mistake probability" according to the current mixture $\{P(E_i \mid \boldsymbol{y}^{t-1})\}_{i=1}^N$. That is,

$$P(y_t \neq \hat{y}_t \mid \boldsymbol{y}^{t-1}) = \min\{P(y_t \neq 0 \mid \boldsymbol{y}^{t-1}),\ P(y_t \neq 1 \mid \boldsymbol{y}^{t-1})\}. \tag{4}$$

Although the predictor's goal in a Bayesian framework is to minimize the mistake probability or the expected total number of mistakes, Bayesian algorithms can also be analyzed in an adversarial setting. The following well-known result (see, e.g., [5]) bounds the performance of the Bayesian predictor GBP within the adversarial experts setting. This result shows that the number of prediction mistakes made by GBP on any sequence $\boldsymbol{y}$ is bounded by the log-likelihood of $\boldsymbol{y}$ under the best model. Precisely, we have the following:

LEMMA 3.1. *For any positive integer N, let $\{P_1, \ldots, P_N\}$ be a set of N probability distributions over $\{0,1\}^\infty$. Then, for any sequence $\boldsymbol{y}^\ell \in \{0,1\}^\ell$, $\ell \in \mathbb{N}$, the total number of prediction mistakes made by* GBP *on $\boldsymbol{y}^\ell$ is at most*

$$M_{\mathrm{GBP}}(\boldsymbol{y}^\ell) \leq \log N + \min_{1 \leq i \leq N} \log \frac{1}{P_i(\boldsymbol{y}^\ell)}. \tag{5}$$

PROOF. Note that $P(\boldsymbol{y}^t) \leq P(\boldsymbol{y}^{t-1})$ holds for all $1 \leq t \leq \ell$ and that (4) implies $P(\boldsymbol{y}^t) \leq P(\boldsymbol{y}^{t-1})/2$ for all t such that $y_t \neq \hat{y}_t$. Therefore,

$$\max_{1 \leq i \leq N} \left[\frac{P_i(\boldsymbol{y}^\ell)}{N} \right] \leq P(\boldsymbol{y}^\ell) \leq 2^{-m},$$

where $m = M_{\mathrm{GBP}}(\boldsymbol{y}^\ell)$ and $P(\boldsymbol{y}^\ell) = (1/N) \sum_{i=1}^N P_i(\boldsymbol{y}^\ell)$. Solving the above with respect to $M_{\mathrm{GBP}}(\boldsymbol{y}^\ell)$ gives the bound. □

Different noise models produce different versions of the Bayes optimal predictor (GBP). We now present two versions of the Bayesian predictor GBP which are used in section 4 to derive master algorithms.

Algorithm η-GBP. Perhaps the simplest noise model is when the expert's predictions are corrupted by an i.i.d. noise process with fixed rate $0 \leq \eta < \frac{1}{2}$. Using this noise model, the probabilities in (3) are easily seen to be

$$P_i(y \mid \boldsymbol{y}^{t-1}) = \begin{cases} \eta & \text{if} \quad y \neq E_i(\boldsymbol{y}^{t-1}), \\ 1-\eta & \text{otherwise;} \end{cases}$$

and $P_i(\boldsymbol{y}^{t-1}) = \prod_{t'=1}^{t-1} P_i(y_{t'} \mid \boldsymbol{y}^{t'-1}) = \eta^{k_i}(1-\eta)^{\ell-k_i}$, where k_i is the number of mistakes made by E_i on the sequence $\boldsymbol{y}^{t-1}$. We call the Bayesian predictor that uses the above probabilities η-GBP.

Algorithm $\beta_{a,b}$-GBP. The Bayesian predictor η-GBP is based on the simplistic assumption that the noise model is characterized by a fixed rate $0 \leq \eta < \frac{1}{2}$. However, in many applications such strong knowledge is not available and thus a different solution for the prediction problem must be sought. In these cases the Bayesian paradigm suggests that one should assume a prior distribution on the parameter space and use this prior together with evidence from the sequence to estimate the noise rate[5] η dynamically. Thus we now assume that the sequence $\boldsymbol{y}^{\ell}$ is generated by first selecting the relevant expert and the noise rate from the appropriate prior distributions, and then using the selected noise rate to corrupt the predictions of the relevant expert. The Bayes optimal prediction (3) is then computed using the posterior mean as an estimate of η. The only problem we are left with is the choice of the prior μ for η. It is convenient to choose μ so that the posterior distribution $P_i(\cdot \mid \boldsymbol{y}^t)$ with respect to μ is easily computable for each expert E_i. This of course depends on the interaction between μ and the distribution P_i. A posterior $P_i(\cdot \mid \boldsymbol{y}^t)$ whose distribution is in the *same family* as the prior μ is found whenever μ is chosen in the conjugate family of distributions for P_i (see, e.g., p. 130 of [1]). We now describe the Bayes optimal predictor that results when the beta distribution is chosen as the prior.

Let the prior distribution on η be a beta distribution with parameters $a, b > 0$ where now η ranges in the interval $[0, 1]$. The corresponding density function with respect to the dominating Lebesgue measure on $[0, 1]$ is

$$\beta_{a,b}(\eta) = \frac{\eta^{a-1}(1-\eta)^{b-1}}{\int_0^1 \xi^{a-1}(1-\xi)^{b-1}\, d\xi},$$

and its expected value is $a/(a+b)$. Observe that by changing the setting of the parameters a and b we obtain different distributions. For instance, choices of a and b for which $a > b$ correspond to a distribution skewed to the right, and vice versa. The noise model now picks a noise rate using the $\beta_{a,b}(\eta)$ distribution, and the outcomes are then produced by corrupting the chosen expert's predictions with i.i.d. noise at the chosen rate.

We can now compute the probabilities in (3) using the new noise model by considering the two cases $E_i(\boldsymbol{y}^{t-1}) = y$ and $E_i(\boldsymbol{y}^{t-1}) \neq y$. Recall that each distribution P_i is

[5] Although we use the same notation for η the reader should observe that in this Bayesian context η is actually a random variable, rather than a fixed value. For a detailed description of the Bayesian paradigm see [11].

completely determined by the predictions of expert E_i and the assumed noise model. Let $\hat{P}_i$ be the product measure $P_i \times \beta_{a,b}$ and let k_i be the total number of prediction mistakes made by E_i on $\boldsymbol{y}^{t-1}$. Then

$$
\begin{aligned}
(6) \qquad P_i(y \mid \boldsymbol{y}^{t-1}) &= \int_0^1 \hat{P}_i(y \mid \boldsymbol{y}^{t-1}, \eta)\, \hat{P}_i(\eta \mid \boldsymbol{y}^{t-1})\, d\eta \\
&= |E_i(\boldsymbol{y}^{t-1}) - y| \int_0^1 \eta\, \hat{P}_i(\eta \mid \boldsymbol{y}^{t-1})\, d\eta \\
&\quad + (1 - |E_i(\boldsymbol{y}^{t-1}) - y|) \int_0^1 (1-\eta)\, \hat{P}_i(\eta \mid \boldsymbol{y}^{t-1})\, d\eta \\
&= |E_i(\boldsymbol{y}^{t-1}) - y| \left(\frac{k_i + a}{t - 1 + a + b} \right) \\
&\quad + (1 - |E_i(\boldsymbol{y}^{t-1}) - y|) \left(1 - \frac{k_i + a}{t - 1 + a + b} \right).
\end{aligned}
$$

The last equality follows by noting that when the noise process is i.i.d. the posterior density of η is in the same form as the $\beta_{a,b}$ prior density with the parameters a and b replaced by $a' = k_i + a$ and $b' = t - 1 - k_i + b$. The estimate given by (6) is usually referred to as the *mean posterior estimate*. Note that k_i is zero when $t = 1$, so the estimate produced by (6) is $a/(a+b)$, the mean of the beta prior density. The simplicity of this method of estimation is one of the reasons beta densities (which are a special case of the Dirichlet densities) are so attractive. From (6) it is easy to see that the probability $P_i(\boldsymbol{y}^{t-1})$ reduces to

$$
(7) \qquad P_i(\boldsymbol{y}^{t-1}) = \frac{\left[\prod_{k=0}^{k_i-1}(k+a)\right]\left[\prod_{m=0}^{t-2-k_i}(m+b)\right]}{\prod_{n=0}^{t-2}(n+a+b)}.
$$

When the distributions in (3) are computed as described by (6) and (7), we obtain a new family of Bayes optimal predictors which we call $\beta_{a,b}$-GBP. As we will see in the next section, the $\beta_{a,b}$-GBP predictor can be rephrased as a multiplicative weighting scheme, leading to a new family of multiplicative algorithms for the adversarial experts setting.

4. Multiplicative Weighting Algorithms. We are now ready to formulate and analyze the class of multiplicative algorithms that arise from the Bayesian prediction framework discussed in Section 3. Since we present algorithms derived from Bayesian learning algorithms, probabilities will still appear in the formal descriptions of the algorithms. However, the relative loss bounds we prove do not make probabilistic assumptions about how the bit sequence to be predicted is generated. That is, we analyze the performance of the derived algorithms in the adversarial framework of the experts setting.

In the first part of the section we investigate the properties shared by all weighting schemes derived from the Bayesian paradigm (3). Section 4.2 contains a simple modification to these Bayesian weighting schemes that leads to improved bounds. This

modification allows us to rederive existing master algorithms as well as a new family of binomial prediction algorithms.

4.1. *From Bayesian Prediction to Weighting Schemes.* This section shows how the Bayesian predictor GBP can be used in the experts setting. Recall that GBP computes the sums $\sum_{i=1}^{N} P_i(y \mid \boldsymbol{y}^{t-1}) \cdot P_i(\boldsymbol{y}^{t-1})$ for bit $y = 0$ and for bit $y = 1$, and then predicts the value having the largest sum. This suggest that when predicting the bit y_t in the experts setting, each expert should have the weight $P_i(\boldsymbol{y}^{t-1})$. Moreover, the "vote" of expert E_i should be split between the predictions $E_i(\boldsymbol{y}^{t-1})$ and $1-E_i(\boldsymbol{y}^{t-1})$ in the same proportions that $P_i(E_i(\boldsymbol{y}^{t-1}) \mid \boldsymbol{y}^{t-1})$ and $1 - P_i(E_i(\boldsymbol{y}^{t-1}) \mid \boldsymbol{y}^{t-1})$ split unit probability. Expert E_i's weight should then be multiplied by the update factor $P_i(y_t \mid \boldsymbol{y}^{t-1})$ after the value y_t is revealed. Note that the update factor $P_i(y_t \mid \boldsymbol{y}^{t-1})$ is equal to $P_i(E_i(\boldsymbol{y}^{t-1}) \mid \boldsymbol{y}^{t-1})$ when E_i's advice on bit y_t is not corrupted and to $1 - P_i(E_i(\boldsymbol{y}^{t-1}) \mid \boldsymbol{y}^{t-1})$ when E_i's advice is corrupted by the noise model associated to expert E_i. It is now quite easy to rephrase our Bayesian predictor GBP in term of weighted voting schemes. Once one has specified a "noise model," the prediction method and update factors follow naturally.

To make this more precise, we start by reinterpreting the noise model as an update factor. Our goal is to use different update factors for the various experts' weights depending on the accuracy of each expert's previous predictions on the observed sequence. Therefore, a *noise model* in the experts framework is a parametrized update factor $\eta(t, k) \in (0, 1)$, where the first argument t measures the length of the (previously) observed sequence and the second argument k counts the number of prediction mistakes made by the expert on the (previously) observed sequence. Hence, at the end of the first trial each expert's weight will be multiplied by the update factor $\eta(0, 0)$ if its advice was wrong, or by $1 - \eta(0, 0)$ if it predicted correctly. Similarly, after observing y_t on trial t, the probability (or weight) $P_i(\boldsymbol{y}^{t-1})$ of expert E_i should be multiplied by the update factor $\eta(t-1, M_i(\boldsymbol{y}^{t-1}))$ if $E_i(\boldsymbol{y}^{t-1}) \neq y_t$ and by $1 - \eta(t-1, M_i(\boldsymbol{y}^{t-1}))$ otherwise, where $M_i(\boldsymbol{y}^{t-1})$ is the number of prediction mistakes made by E_i on $\boldsymbol{y}^{t-1}$, i.e.,

$$M_i(\boldsymbol{y}^{t-1}) = |\{s : E_i(\boldsymbol{y}^{s-1}) \neq y_s,\ 1 \leq s \leq t-1\}|.$$

Note that for each expert E_i we have, at each trial t, two possible update factors which sum to 1. Several master algorithms where the update factors are *unnormalized*, i.e., they do not sum to 1, have been proposed and analyzed. However, normalized update factors arise naturally in weighting schemes derived from the Bayesian paradigm.

The above interpretation of the noise model as an update factor leads to the master predictor BAY sketched in Figure 2, which is derived from the Bayesian predictor GBP defined in Section 3. Note that in Figure 2 we use "$\boldsymbol{y}, b$" to denote the sequence resulting when bit b is appended to the end of the sequence $\boldsymbol{y}$. Thus $P_i(\boldsymbol{y}^{t-1}, b)$ is the probability, according to P_i, of the sequence $(y_1, \ldots, y_{t-1}, b)$. Furthermore, $P_i(\boldsymbol{y}^{t-1}, b) = P_i(b \mid \boldsymbol{y}^{t-1}) P_i(\boldsymbol{y}^{t-1})$ and $P_i(b \mid \boldsymbol{y}^{t-1})$ is either $\eta(t-1, M_i(\boldsymbol{y}^{t-1}))$ or $1 - \eta(t-1, M_i(\boldsymbol{y}^{t-1}))$ depending on whether or not $b = E_i(\boldsymbol{y}^{t-1})$. The relative loss bound (5) of GBP carries over to BAY leading to the bound

$$M_{\mathrm{BAY}}(\boldsymbol{y}^{\ell}) \leq \log N + \min_{1 \leq i \leq N} \log \frac{1}{P_i(\boldsymbol{y}^{\ell})}. \tag{8}$$

On-Line Predictor BAY

Input: A set $E_1, \ldots, E_N$ of experts and a noise model η.
Initialization: Let $P_i(\boldsymbol{y}^0) = 1$ and $\eta_i = \eta(0, 0)$ for each $i = 1, \ldots, N$.

For $t = 1, 2 \ldots$

1. For each $i = 1, \ldots, N$
   let $P_i(\boldsymbol{y}^{t-1},\ E_i(\boldsymbol{y}^{t-1})) = (1-\eta_i)P_i(\boldsymbol{y}^{t-1})$ and $P_i(\boldsymbol{y}^{t-1}, 1-E_i(\boldsymbol{y}^{t-1})) = \eta_i P_i(\boldsymbol{y}^{t-1})$.
2. If $\sum_{i=1}^N P_i(\boldsymbol{y}^{t-1},\ 1) \geq \sum_{i=1}^N P_i(\boldsymbol{y}^{t-1},\ 0)$, then predict 1 for bit y_t; otherwise predict 0.
3. Observe bit y_t.
4. For each $i = 1, \ldots, N$
   compute the new update factors $\eta_i = \eta(t, M_i(\boldsymbol{y}^t))$.

Fig. 2. The master predictor BAY.

4.2. *Conservative Algorithms*. This subsection presents a modification to the master algorithm BAY leading to improved performance bounds. Bound (8) relies on the facts that:

1. Every time BAY makes a mistake, the sum of the probabilities assigned to the observed sequence by each expert is (at least) halved.
2. After ℓ bits have been revealed, this sum is at least $P_i(\boldsymbol{y}^\ell)$ for any $1 \leq i \leq N$.

However, the probability (or weight) P_i, of any expert E_i, often drops even when BAY predicts correctly. This can be partially remedied by using a conservative variant of BAY which we call CBAY. This conservative variant skips the weight update step whenever it predicts correctly, essentially ignoring those trials where it makes a correct prediction. Since a Bayesian algorithm makes its prediction based on all of the information acquired during the previous trials, this variant can no longer be considered a Bayesian algorithm (with respect to the original assumptions). This encourages us to use a different (nonprobabilistic) notation for its weights.

Littlestone and Mesterharm [8], [9] independently explored a slightly different path to obtain a closely related family of algorithms. They call these algorithms "Apobayesian" in order to emphasize their Bayesian roots, while making it clear that they are are not themselves Bayesian (in the usual sense). Their algorithms are designed for the more general problem of learning disjunctions or other linearly separable functions of the expert's predictions, and their analysis revolves around measuring the progress toward a target distribution. Although the problems they consider are more general, we are able to obtain better bounds for the simpler experts setting considered here.

Before analyzing the conservative variants of our algorithms, we need the following definition. Let

$$\widehat{M}_i(\boldsymbol{y}^\ell) = |\{t : 1 \leq t \leq \ell,\ E_i(\boldsymbol{y}^{t-1}) \neq y_t \wedge \mathrm{CBAY}(\boldsymbol{y}^{t-1}) \neq y_t\}|$$

be the number of prediction mistakes made by E_i in previous trials where the master made a mistake as well.

On-Line Predictor CBAY

Input: A set $E_1, \ldots, E_N$ of experts and a noise model η.
Initialization: Let $w_i(0) = 1$ and $\eta_i = \eta(0, 0)$ for each $1 \le i \le N$. Set the mistake counter m to 0.

For $t = 1, 2, \ldots$

1. For each $i = 1, \ldots, N$
   let $w_i^{x_i} = (1 - \eta_i) w_i(t-1)$ and let $w_i^{1-x_i} = \eta_i w_i(t-1)$, where $x_i = E_i(\boldsymbol{y}^{t-1})$.
2. If $\sum_{i=1}^N w_i^1 \ge \sum_{i=1}^N w_i^0$, then predict 1 for bit y_t; otherwise predict 0.
3. Observe bit y_t.
4. If a *mistake occurred*, then set $w_i(t) = w_i^{y_t}$, compute the new update factors $\eta_i = \eta(m+1, \widehat{M}_i(\boldsymbol{y}^t))$, and increment the mistake counter m by 1.
5. If no *mistake occurred*, then for each $i = 1, \ldots, N$
   set $w_i(t) = w_i(t-1)$. (The update factors also remain unchanged.)

Fig. 3. The conservative master predictor CBAY.

Despite the change in notation, algorithm CBAY in Figure 3 is similar to the master predictor BAY of Figure 2. The essential difference is that CBAY skips the update step whenever its prediction is correct. The algorithm CBAY uses a *weight* w_i (initially set to 1) for each expert E_i. After observing any sequence $\boldsymbol{y}^t$, the current weight $w_i(t)$ of expert E_i corresponds to the probability P_i (according to the noise model) assigned to the subsequence of $\boldsymbol{y}^t$ consisting of only those trials where CBAY made a mistake. Unlike the P_i's computed by BAY, the weights used by CBAY cannot be defined independently for each expert: their values depend on when CBAY makes mistakes, which in turn depends on the behavior of the other experts. Therefore, the weights $w_i(t)$ have an implicit dependence on the particular run of CBAY.

The analysis of CBAY is easy, as shown by the following two facts.

FACT 4.1. *For all noise models $\eta(\cdot, \cdot)$, for all sequences $\boldsymbol{y} \in \{0,1\}^\infty$, for all sets of N experts, and for all integers $t \ge 0$,*

$$M_{\mathrm{CBAY}}(\boldsymbol{y}^t) \le \log N + \min_{1 \le i \le N} \log \frac{1}{w_i(t)}. \tag{9}$$

PROOF. Fix the noise model $\eta(\cdot, \cdot)$, the sequence $\boldsymbol{y}$ to be predicted, and the set of N experts. For all $t \ge 1$, let $W_t = \sum_{i=1}^N w_i(t-1)$ be the total weight of the experts at the beginning of trial t. Note that if CBAY makes a mistake predicting y_t, then $W_{t+1} \le W_t/2$. Since $W_1 = N$ and $w_i(t-1) \le W_t$ for any $1 \le i \le N$, we have $\max_{1 \le i \le N} w_i(t) \le N/2^{m_t}$, where m_t is $M_{\mathrm{CBAY}}(\boldsymbol{y}^t)$. Now, solving for m_t yields the required bound. □

Bound (9) contains $w_i(t)$, a quantity that depends on the particular run of the algorithm. We now convert this bound into a form that is easier to apply.

FACT 4.2. *For all noise models $\eta(\cdot,\cdot)$, if $f(\cdot,\cdot)$ is a nonnegative function such that, for all $\boldsymbol{y} \in \{0,1\}^\infty$, for all sets of N experts, and for all integers $t \geq 0$,*

$$w_i(t) \geq f(M_{\mathrm{CBAY}}(\boldsymbol{y}^t), \widehat{M}_i(\boldsymbol{y}^t)),$$

then

$$M_{\mathrm{CBAY}}(\boldsymbol{y}^t) \leq \max\left\{q \in \mathbb{N} : q \leq \log N + \min_{1\leq i\leq N} \log \frac{1}{f(q, \widehat{M}_i(\boldsymbol{y}^t))}\right\}. \tag{10}$$

PROOF. From Fact 4.1 and the definition of f it is easy to see that $M_{\mathrm{CBAY}}(\boldsymbol{y}^t)$ belongs to the set of integers

$$\left\{q \in \mathbb{N} : q \leq \log N + \min_{1\leq i\leq N} \log \frac{1}{f(q, \widehat{M}_i(\boldsymbol{y}^t))}\right\}.$$

Hence, $M_{\mathrm{CBAY}}(\boldsymbol{y}^t)$ is at most the maximum integer in this set. □

Fact 4.2 gives a general bound on the number of prediction mistakes made by CBAY. We now apply it to different noise models to obtain previously known bounds for WM and BW.

4.3. *A Revisitation of Some Known Algorithms*

Weighted Majority (*WM*). We now analyze CBAY with the simple i.i.d. noise model used by η-GBP of Section 3. For this algorithm, bound (10) becomes linear in the loss of the best expert. Let $\eta(t,k) = \eta$ for all $t, k \in \mathbb{N}$, where $0 < \eta < \frac{1}{2}$ is a constant. In this noise model we multiply the weight of any expert that made a mistake in the last trial by η and the weight of any expert that was correct in the last trial by $1-\eta$. The weights for the conservative case are as follows. After any number t of trials we have that $w_i(t) = \eta^{\hat{M}_i(\boldsymbol{y}^t)}(1-\eta)^{m_t - \hat{M}_i(\boldsymbol{y}^t)}$ where $m_t = M_{\mathrm{CBAY}}(\boldsymbol{y}^t)$. Let $M_{\mathrm{best}}(\boldsymbol{y}^t) = \min_{1\leq i\leq N} M_i(\boldsymbol{y}^t)$ and $\widehat{M}_{\mathrm{best}}(\boldsymbol{y}^t) = \min_{1\leq i\leq N} \widehat{M}_i(\boldsymbol{y}^t)$. By applying Fact 4.2 with $f(m,k) = \eta^k(1-\eta)^{m-k}$ we obtain the following bound:

$$M_{\mathrm{CBAY}}(\boldsymbol{y}^t) \leq \max\{q \in \mathbb{N} : q \leq \log N - (q - \widehat{M}_{\mathrm{best}}(\boldsymbol{y}^t))\log(1-\eta) - \widehat{M}_{\mathrm{best}}(\boldsymbol{y}^t)\log\eta\}. \tag{11}$$

After a suitable reordering we find that

$$M_{\mathrm{CBAY}}(\boldsymbol{y}^t) \leq \left\lfloor \frac{\log N + \widehat{M}_{\mathrm{best}}(\boldsymbol{y}^t)\cdot\log((1-\eta)/\eta)}{1+\log(1-\eta)} \right\rfloor. \tag{12}$$

As $\widehat{M}_{\mathrm{best}}(\boldsymbol{y}) \leq M_{\mathrm{best}}(\boldsymbol{y})$ for any $\boldsymbol{y}$, this shows an upper bound on the number of mistakes of CBAY of the form $a\log N + bM_{\mathrm{best}}(\boldsymbol{y})$, where a and b depend on η only. Vovk [13] studies those pairs (a,b) where mistake bounds of the form $a\log N + bM_{\mathrm{best}}(\boldsymbol{y})$ are achievable. Littlestone and Warmuth [10] previously obtained a bound identical to (12) for their WM algorithm which is equivalent to this version of our CBAY algorithm (even though the Bayesian origin is not so explicit in the definition of WM as it is in the definition of CBAY.)

Binomial Weighting (*BW*). We now define a noise model such that our CBAY becomes the same as the BW algorithm of [3]. Let $\binom{r}{\leq s}$ be a shorthand for $\sum_{i=0}^{s}\binom{r}{i}$. For each $q, K \in \mathbb{N}$, define the noise model

$$\eta_{q,K}(r,s) = \binom{q-r-1}{\leq K-s-1} \Big/ \binom{q-r}{\leq K-s}.$$

Note that $\eta_{q,K}(r,s) = 0$ for $r \geq q$ or $s \geq K$. Thus for certain bit sequences $\boldsymbol{y}$, it may happen that $w_i(t) = 0$. If this happens for all experts E_i, then according to (3) BAY (or CBAY) predicts arbitrarily on all future bits. Since the algorithm is predicting arbitrarily it is not surprising that the bounds (8) and (9) become vacuous. Hence, instances of BAY or CBAY using noise model $\eta_{q,K}$ should only be used to predict sequences $\boldsymbol{y}$ for which there is at least one expert E_i whose weight remains positive throughout the prediction process. In other words, the algorithm requires advance knowledge of an upper bound K on the number of mistakes made by the best expert. The next result is equivalent to Theorem 1 of [3].

THEOREM 4.1. *For all $K \in \mathbb{N}$ and for all $\boldsymbol{y}^\ell \in \{0,1\}^\ell$, $\ell \in \mathbb{N}$, if* CBAY *is run on $\boldsymbol{y}^\ell$ using the noise model $\eta_{m+1,K}$, where*

$$m = \max\left\{q \in \mathbb{N} : q \leq \log N + \log\binom{q}{\leq K}\right\} \tag{13}$$

and $\widehat{M}_{\text{best}}(\boldsymbol{y}^\ell) \leq K$, then the number of mistakes made by CBAY *is upper bounded by m, i.e., $M_{\text{CBAY}}(\boldsymbol{y}^\ell) \leq m$.*

PROOF. Choose $\boldsymbol{y} \in \{0,1\}^\ell$, set m as in (13), and for any nonnegative integers r and s define

$$f(r,s) = \binom{m+1-r}{\leq K-s} \Big/ \binom{m+1}{\leq K}.$$

As $f(r,s) = 0$ for $r > m+1$, Fact 4.2 does not give a useful bound here. To prove the theorem, we assume to the contrary that CBAY makes $m+1$ mistakes on the sequence $\boldsymbol{y}^\ell$. Using Fact 4.1, we find that

$$M_{\text{CBAY}}(\boldsymbol{y}^\ell) \leq \log N + \min_{1\leq i\leq N} \log\frac{1}{w_i(\ell)}. \tag{14}$$

Now, it is not hard to see that, for each $t = 1, \ldots, \ell$ and each $1 \leq i \leq N$,

$$w_i(t) = f(M_{\text{CBAY}}(\boldsymbol{y}^t), \widehat{M}_i(\boldsymbol{y}^t)).$$

Since $\widehat{M}_{\text{best}}(\boldsymbol{y}^\ell) \leq K$ and $M_{\text{CBAY}}(\boldsymbol{y}^\ell) = m+1$ (by the assumption) there is at least one index $1 \leq i^* \leq N$ such that

$$w_{i^*}(\ell) = f(m+1, \widehat{M}_i(\boldsymbol{y}^\ell)) = 1 \Big/ \binom{m+1}{\leq K}. \tag{15}$$

Combining (14) and (15) we see that

$$2^{m+1} \leq N\binom{m+1}{\leq K},$$

contradicting the definition of m given in (13) and completing the proof. □

4.4. *Bayesian Binomial Weighting* (*BBW*). In the previous section we unified two known algorithms, WM and BW, by reformulating them as different versions of CBAY. In this section we exploit Bayes theory to derive an improved version of CBAY by using the same noise model as the $\beta_{a,b}$-GBP algorithm described in Section 3.

Clearly, bound (12) depends on how the parameter η is chosen as a function of N and $\widehat{M}_{\text{best}}(\boldsymbol{y}^\ell)$. When the algorithm has no knowledge about the magnitude of $\widehat{M}_{\text{best}}(\boldsymbol{y}^\ell)$, a reasonable choice is $\eta = \frac{1}{3}$. If the algorithm knows in advance a bound K on $\widehat{M}_{\text{best}}(\boldsymbol{y}^\ell)$, tighter bounds can be obtained by choosing the parameter η as a function of N and of the bound K. Vovk [12] has given an implicit formula for the value η^* that minimizes (12). More precisely, he has shown that when $\eta = K/M$, where $x = M$ is the unique solution of the equation

$$x = \log N + x \cdot H\left(\frac{K}{x}\right) \tag{16}$$

and H is the binary entropy function, $H(\alpha) = -\alpha \log(\alpha) - (1-\alpha)\log(1-\alpha)$, then CBAY run with this constant value of η has a mistake bound of M.

We also remark that an explicit approximation to η^*, yielding the more tractable bound

$$M_{\text{CBAY}}(\boldsymbol{y}^\ell) \le 2K + 2\sqrt{K \ln N} + \log N, \tag{17}$$

has been given by Cesa-Bianchi et al. in Lemma 4 of [2].

Bound (17) has been derived for an algorithm that requires the knowledge of an upper bound K on the number of mistakes made by the best expert. The algorithm uses this additional information to tune the noise rate optimally in its noise model. We now consider a different noise model where the update factor varies over trials and is potentially different for each expert E_i. Using this noise model, CBAY achieves a mistake bound close to (17) *without* using any previous knowledge about the sequence $\boldsymbol{y}^\ell$ to predict. Thus we obtain a more principled method than ad hoc doubling techniques for automatically determining the proper update factors.

THEOREM 4.2. *For all positive integers a and b and for all $\boldsymbol{y}^\ell \in \{0,1\}^\ell$, $\ell \in \mathbb{N}$, if* CBAY *is run on $\boldsymbol{y}^\ell$ using the noise model*

$$\eta_{a,b}(r,s) = \frac{s+a}{r+a+b}, \tag{18}$$

then $M_{\text{CBAY}}(\boldsymbol{y}^\ell)$ is at most

$$\max\left\{q \in \mathbb{N} : q \le \log N + \log\binom{q+a+b-2}{k^*+a-1} + \log\left(1+\frac{q}{a+b-1}\right) - \log\binom{a+b-2}{a-1}\right\},$$

where $k^* = \widehat{M}_{\text{best}}(\boldsymbol{y}^\ell)$.

PROOF. Fix a and b and let $r = M_{\text{CBAY}}(\boldsymbol{y}^\ell)$ when CBAY uses the noise model (18). Also let E_i be an expert such that $\widehat{M}_i(\boldsymbol{y}^\ell) = k^*$. Then one can see that the final weight

of expert E_i satisfies

$$w_i(\ell) = \frac{\left[\prod_{k=0}^{k^*-1}(k+a)\right]\left[\prod_{m=0}^{r-k^*-1}(m+b)\right]}{\prod_{n=0}^{r-1}(n+a+b)}. \tag{19}$$

Note that the ratio given in (19) is the conservative analog of the ratio given in (7) of Section 3. The bound of the theorem immediately follows by applying Fact 4.2 with $f(r,k^*)$ equal to the right-hand side of (19) and by applying the following chain of equalities:

$$\begin{aligned}
\log\frac{1}{w_i(\ell)} &= \log\prod_{n=0}^{r-1}(n+a+b) - \log\prod_{k=0}^{k^*-1}(k+a) - \log\prod_{m=0}^{r-1-k^*}(m+b) \\
&= \log\frac{(r+a+b-1)!}{(a+b-1)!} - \log\frac{(k^*+a-1)!}{(a-1)!} - \log\frac{(r-k^*+b-1)!}{(b-1)!} \\
&= \log\frac{(r+a+b-2)!}{(k^*+a-1)!\,(r-k^*+b-1)!} + \log\frac{r+a+b-1}{a+b-1} \\
&\quad - \log\frac{(a+b-2)!}{(a-1)!\,(b-1)!} \\
&= \log\binom{r+a+b-2}{k^*+a-1} + \log\left(1+\frac{r}{a+b-1}\right) - \log\binom{a+b-2}{a-1}. \quad \square
\end{aligned}$$

Since each expert's final weight (as given by (19)) is the probability that the expert generated the subsequence of $\boldsymbol{y}^\ell$ where the master predicted incorrectly, Theorem 4.2 can also be proven by expressing these probabilities as ratios of beta functions.

Specific choices for the parameters a and b yield the following bounds.

COROLLARY 4.1. *For all $\boldsymbol{y}^\ell \in \{0,1\}^\ell$, $\ell \in \mathbb{N}$, if* CBAY *is run on $\boldsymbol{y}^\ell$ using the noise model* (18) *with $a = b = 1$, then the number of mistakes is at most*

$$M_{\mathrm{CBAY}}(\boldsymbol{y}^\ell) \le \max\left\{q \in \mathbb{N} : q \le \log N + \log\binom{q}{\widehat{M}_{\mathrm{best}}(\boldsymbol{y}^\ell)} + \log(q+1)\right\}. \tag{20}$$

Furthermore, if CBAY *is run on the same sequence with the setting $a = 1$ and $b = 2$, then the number of mistakes is at most*

$$\max\left\{q \in \mathbb{N} : q \le \log N + \log\binom{q}{\widehat{M}_{\mathrm{best}}(\boldsymbol{y}^\ell)} + \log(q+1)\right\} - 1.$$

One might be tempted to optimize the choice of parameters a and b in Theorem 4.2 by guessing the number of mistakes made by the best expert on the sequence to predict. In Section 4.5 we show that this tuning cannot provide a significant advantage on all bit sequences. In particular, we show that the bound on CBAY using one of the beta priors from Corollary 4.1 has the same leading term as the bound for the optimally tuned WM algorithm.

4.5. *Comparison*. The results of Section 4.4 raise two natural questions. First, how can one optimize the choice of a and b in the bound of Theorem 4.2? Second, how does this bound (or its special cases in Corollary 4.1) compare with the bounds (12) and (13)? Note that bound (13) assumes the knowledge of an upper bound K on the number of mistakes made by the best expert. As already mentioned in Section 4.4, the knowledge of such a bound K can also be used to tune the parameter η in (12).

Cesa-Bianchi et al. [3] showed that both (12) with tuned parameters and (13) are asymptotically optimal in a sense that will be made precise in a moment. In this section we state that the bounds of Corollary 4.1 are asymptotically optimal as well, even though the algorithm does not require an upper bound K on the number of mistakes made by the best expert. As a side-effect of this result we have that no tuning of the parameters a and b in the bound of Theorem 4.2 can get, at least asymptotically, an advantage over the bounds proven in Corollary 4.1.

To define asymptotical optimality we use the equivalence relation "$\sim$", between infinite sequences of positive integers, defined by $\langle a_i \rangle \sim \langle b_i \rangle$ if and only if

$$\lim_{i \to \infty} \frac{a_i}{b_i} = 1.$$

Let $F = F(N_i, k_i)$ be the function whose value on (N_i, k_i) is the right-hand side of (20) when $N = N_i$ and $\widehat{M}_{\text{best}}(\boldsymbol{y}^\ell) = k_i$. Similarly, let $G = G(N_i, k_i)$ be the function whose value on (N_i, k_i) is the right-hand side of (13) when $N = N_i$ and $K = k_i$. Using Theorem 3 of [3] and an adaptation of the proof of Theorem 4 of [3] we can show the following.

THEOREM 4.3. *For any sequence $\langle (N_i, k_i) \rangle_{i>0}$ of positive integers $N_i \geq 2$ and k_i such that either $\lim_{i\to\infty} k_i = \infty$ or $\lim_{i\to\infty} N_i = \infty$ there exist a sequence $\langle m_i \rangle$ of positive integers and a sequence $\langle \mathcal{E}_i \rangle$ of sets of experts, with $|\mathcal{E}_i| = N_i$, such that the following hold*:

1. *For each $i > 0$ and for any deterministic on-line prediction algorithm A, there is a sequence $\boldsymbol{y} \in \{0,1\}^*$ for which $\widehat{M}_{\text{best}}(\boldsymbol{y}) \leq k_i$ and $M_A(\boldsymbol{y}) \geq m_i$.*
2. *$\langle F(N_i, k_i) \rangle \sim \langle G(N_i, k_i) \rangle \sim \langle m_i \rangle$.*

PROOF. Omitted.

This theorem is especially interesting in light of recent work by Vovk [13]. He considers a family of expert learning games parametrized by the real values c_1 and c_2. The learner wins a particular (c_1, c_2) game if, for all N, for all sets of N experts, and for all sequences $\boldsymbol{y}$, it makes at most $c_1 \log N + c_2 M_{\text{best}}(\boldsymbol{y})$ mistakes on the sequence $\boldsymbol{y}$ (recall that $M_{\text{best}}(\boldsymbol{y})$ is the number of mistakes made by the best of the N experts on $\boldsymbol{y}$, thus $M_{\text{best}}(\boldsymbol{y})$ is an upper bound on $\widehat{M}_{\text{best}}(\boldsymbol{y})$). Similarly, the adversary wins a particular (c_1, c_2) game if for each master algorithm there is a set of N experts and a sequence $\boldsymbol{y}$ such that the master algorithm makes more than $c_1 \log N + c_2 M_{\text{best}}(\boldsymbol{y})$ mistakes on the sequence $\boldsymbol{y}$.

Bound (12) shows that CBAY, assuming an i.i.d. noise model with known rate η, is a

winning strategy for the learner in the game with parameters

$$\left(\frac{1}{1+\log(1-\eta)}, \frac{\log((1-\eta)/\eta)}{1+\log(1-\eta)}\right).$$

In addition to showing this result for WM, Vovk also shows that, when the algorithm's predictions are in $\{0, 1\}$, these pairs define the border between games where the learner has a winning strategy and games where the adversary wins. Vovk has extended these results to other loss functions.

It is important to realize that different points on this border between where the learner wins and where the adversary wins lead to different performances on different sequences. For example, taking $\eta = \frac{1}{3}$ corresponds to the values $c_1 \approx 2.4$ and $c_2 = 1$ while $\eta = \frac{1}{4}$ leads to $c_1 \approx 1.7$ and $c_2 \approx 1.6$. Thus when $M_{\text{best}}(\boldsymbol{y})$ is large compared with $\log N$, the bound for $\eta = \frac{1}{4}$ can be more than half again as large as the bound for $\eta = \frac{1}{3}$. Similarly, the bound for $\eta = \frac{1}{3}$ can be nearly half again as large as the bound for $\eta = \frac{1}{4}$ when $M_{\text{best}}(\boldsymbol{y})$ is small compared with $\log N$.

Theorem 4.3 illustrates the power of algorithm CBAY with the $\eta_{a,b}(r, s)$ noise model of (18) when $a = b = 1$. The bound for this version of CBAY is asymptotically the same as the bound (13) for BW which, as we mentioned above, is asymptotically equivalent to the bound (12) for WM using the best value of η for that particular sequence.

In the continuous prediction case, Cesa-Bianchi et al. [3] showed how a complicated doubling trick can be used to reestimate repeatedly the best choice for η. In our discrete loss setting, their bound on the loss of the master becomes

$$2M_{\text{best}}(\boldsymbol{y}) + 8\sqrt{M_{\text{best}}(\boldsymbol{y}) \ln N} + 5.6 \ln N. \tag{21}$$

Actually, their doubling scheme allows a limited tradeoff between the constant in front of the $\sqrt{M_{\text{best}}(\boldsymbol{y}) \ln N}$ term and the constant in front of the $\ln N$ term. The former constant can be reduced to about $\frac{20}{3}$ at the cost of letting the latter constant go to infinity.

Our bounds for algorithm CBAY are in an inconvenient implicit form. By making several overapproximations, we can convert them into a more comparable form. This bound is an improvement over the doubling scheme unless $M_{\text{best}}(\boldsymbol{y})$ is very large with respect to $\ln N$.

THEOREM 4.4. *For all $\boldsymbol{y} \in \{0, 1\}^*$, if* CBAY *is run on $\boldsymbol{y}$ using the noise model* (18) *with $a = b = 1$ and $M_{\text{best}}(\boldsymbol{y}) \geq 1$, then the number of mistakes is at most*

$$M_{\text{CBAY}}(\boldsymbol{y}) \leq 2M_{\text{best}}(\boldsymbol{y}) + 3\sqrt{M_{\text{best}}(\boldsymbol{y}) \ln N} + 3M_{\text{best}}(\boldsymbol{y})^{2/3} + \log N + 3. \tag{22}$$

PROOF SKETCH. Let $k = M_{\text{best}}(\boldsymbol{y})$. Using bound (20), it suffices to show that when $q = 2k + \log(N) + 3\sqrt{k \ln N} + 3k^{2/3} + 3$ we have

$$g(N, k) = q - \log N - \log \binom{q}{k} - \log(q + 1) > 0.$$

We first approximate the binomial coefficient $\binom{q}{k}$ with $(q/(q-k))^{q-k}(q/k)^k$ and then show that the derivative of $g(N, k)$ with respect to $N \geq 2$ is positive. This requires two

approximations to the logarithms depending on whether or not $\log N > k$. Once it is known that the derivative with respect to N is positive, it suffices to show that $g(2, k)$ is positive. This can be verified numerically for $1 \le k \le 1000$. For larger k, we show that $g(2, k) > 0$ analytically using fifth-order approximations to the logarithms.

Although straightforward once one has the appropriate approximations, the algebra involved in the proof is exceedingly tedious. We therefore suggest that the interested reader verify the theorem by "plot" using a suitable tool such as Maple or Mathematica. □

When $M_{\text{best}}(\boldsymbol{y}) = 0$, bound (22) does not apply, and the implicit bound of Corollary 4.1 is at least $\log N + \log\log N$. However, one can show that, for the special case $M_{\text{best}}(\boldsymbol{y}) = 0$,

$$M_{\text{CBAY}}(\boldsymbol{y}) \le \log(N) + 3\sqrt{\log N},$$

as $q > \log(N) + \log(q + 1)$ when $q = \log(N) + 3\sqrt{\log N}$.

We used the $\beta_{1,1}$ prior to obtain the bound (22). Better exponents on the $3M_{\text{best}}(\boldsymbol{y})^{2/3}$ can be achieved using different priors (at the cost of increasing the term's constant). However, we were unable to find a prior for which the exponent drops to $\frac{1}{2}$. It is unclear if this is a natural property of the algorithm or an artifact of the approximations used in the proof of Theorem 4.4.

5. Performance under Absolute Loss. In this section we consider variants of BAY that output predictions in the range $[0, 1]$. We measure the loss incurred by these master algorithms when their prediction is $\hat{y}_t \in [0, 1]$ and the correct bit is $y_t \in \{0, 1\}$ with the absolute loss $|\hat{y}_t - y_t|$. Note that if the master algorithm's predictions are always in $\{0, 1\}$ (like those of our experts), then the absolute loss and the 0-1 loss measures are identical. Furthermore, if the Boolean sequence is generated probabilistically according to the model described in Section 3, the Bayes optimal prediction, minimizing the expected absolute loss, is always found in $\{0, 1\}$. Therefore, algorithms which "hedge their bets" by choosing their predictions in $[0, 1]$ are not strictly Bayesian.

We use $L_A(\boldsymbol{y}^\ell)$ to denote the total absolute loss of the master algorithm A on a sequence $\boldsymbol{y}^\ell$, so if $\hat{y}_t$ is the prediction of A for bit y_t, then $L_A(\boldsymbol{y}^\ell) = \sum_{t=1}^{\ell} |\hat{y}_t - y_t|$. Observe that if a master algorithm uses a biased coin to predict 1 with probability $\hat{y}_t$ and 0 with probability $1 - \hat{y}_t$ (rather than outputting the value $\hat{y}_t \in [0, 1]$), then $|\hat{y}_t - y_t|$ is the probability (with respect to the coin flip) that the randomized master will make an incorrect prediction when y_t is the correct bit. Furthermore, the total absolute loss $\sum_{t=1}^{\ell} |\hat{y}_t - y_t|$ of the deterministic master predicting $\hat{y}_t \in [0, 1]$ equals the expected total number of mistakes made by the randomized master.

We consider variants of the master algorithm BAY (described in Figure 2) using arbitrary noise models $\eta(\cdot, \cdot)$ to output predictions that range in the continuous interval $[0, 1]$. Each such master predictor computes the probabilities $P(0 \mid \boldsymbol{y}^{t-1})$ and $P(1 \mid \boldsymbol{y}^{t-1})$. However, instead of outputting a bit, the master algorithms of this section output a prediction

$$\hat{y}_t = \frac{F(P(1 \mid \boldsymbol{y}^{t-1}))}{F(P(0 \mid \boldsymbol{y}^{t-1})) + F(P(1 \mid \boldsymbol{y}^{t-1}))}, \tag{23}$$

where F: $[0, 1] \to \mathbb{R}^+$ is some monotonically increasing function. One natural choice for F is the identity function, and different choices of F lead to different master algorithms.

A good choice for F is the sigmoidal function, $-\log(1-x)$, previously used in [2] and [12]. This function has a nice theoretical motivation, as it is the amount of information (measured in bits) gained when an event with probability $1 - x$ occurs. For this reason, we call the master predictor whose predictions are produced using (23) with $F(x) = -\log(1 - x)$ the IGAIN master algorithm.

Using standard techniques from [2], it is easy to prove the following result.

THEOREM 5.1. *For all $\ell \in \mathbb{N}$ and $\boldsymbol{y}^\ell \in \{0, 1\}^\ell$, if* IGAIN *is run on $\boldsymbol{y}^\ell$ using any noise model, then*

$$L_{\text{IGAIN}}(\boldsymbol{y}^\ell) \le \tfrac{1}{2}\left[\log N + \min_{1 \le i \le N} \log \frac{1}{P_i(\boldsymbol{y}^\ell)}\right]. \tag{24}$$

PROOF. For $1 \le t \le \ell$, we set $r_t = P(y_t \mid \boldsymbol{y}^{t-1})$, so $P(1 - y_t \mid \boldsymbol{y}^{t-1}) = 1 - r_t$. We further define $W_t = \sum_{i=1}^N P_i(\boldsymbol{y}^{t-1})$ to be the total weight of the experts at the beginning of trial t. Since at each trial t IGAIN's prediction for bit y_t is

$$\hat{y}_t = \frac{-\log P(0 \mid \boldsymbol{y}^{t-1})}{-\log(P(0 \mid \boldsymbol{y}^{t-1})) - \log(P(1 \mid \boldsymbol{y}^{t-1}))},$$

the total absolute loss incurred by IGAIN on the prediction sequence $\boldsymbol{y}^\ell$ is

$$L_{\text{IGAIN}}(\boldsymbol{y}^\ell) = \sum_{t=1}^{\ell} |y_t - \hat{y}_t| = \sum_{t=1}^{\ell} \frac{-\log r_t}{-\log(r_t) - \log(1 - r_t)}.$$

Using the fact that the function $-\log(r_t(1 - r_t))$ is minimized for $r_t = \frac{1}{2}$, we can upper bound the total loss of IGAIN by

$$L_{\text{IGAIN}}(\boldsymbol{y}^\ell) \le \tfrac{1}{2} \sum_{t=1}^{\ell} \log \frac{1}{r_t}. \tag{25}$$

On the other hand, the construction of IGAIN ensures that $W_{t+1} = W_t\, r_t$ on each trial t. Hence the total final weight of the experts is $W_{\ell+1} = W_1(\prod_{t=1}^{\ell} r_t)$. Since $W_1 = N$ and $W_{\ell+1} \ge P_i(\boldsymbol{y}^\ell)$ it follows that

$$P_i(\boldsymbol{y}^\ell) \le W_{\ell+1} \le N \prod_{t=1}^{\ell} r_t. \tag{26}$$

Solving (26) with respect to $\prod_{t=1}^{\ell} r_t$ and taking the log of both sides yields

$$\sum_{t=1}^{\ell} \log \frac{1}{r_t} \le \log N + \log \frac{1}{P_i(\boldsymbol{y}^\ell)}. \tag{27}$$

Substituting the upper bound in (27) into the right-hand side of (25) we obtain

$$L_{\text{IGAIN}}(\boldsymbol{y}^\ell) \le \tfrac{1}{2}\left(\log N + \log \frac{1}{P_i(\boldsymbol{y}^\ell)}\right). \tag{28}$$

The bound of the theorem then follows by observing that inequality (28) holds for all $i = 1, \ldots, N$. □

Note that (24) is exactly half of the 0-1 loss bound (8) for algorithm BAY. It would be nice to devise an algorithm whose expected number of mistakes is at most half the right-hand side of (10), i.e., the bound for CBAY that does not depend directly on the sequence length ℓ. Unfortunately, so far this has been shown (see [2] and [12]) only for the simple noise model $\eta(t,k) = \eta$ for all $t, k \in \mathbb{N}$ ($0 \le \eta < \frac{1}{2}$). Proving expected mistake bounds equal to half the right-hand side of (20) when the algorithm is allowed to make predictions in the $[0,1]$ interval remains an open problem.

The reasons for this difficulty are subtle and relate to the use of the conservativeness to remove the dependence on ℓ in the bounds. When the algorithm's predictions are in $\{0,1\}$, the algorithm is either right or wrong. If the prediction is between 0 and 1, then the algorithm is always partly wrong and must adjust the weights of the experts to help identify the best expert. On the other hand, if the weights of the experts are in exponential form $\eta^t(1-\eta)^{t-k}$, as when the noise model is $\eta(t,k) = \eta$ for all $t, k \in \mathbb{N}$, then bounds can be proven without resorting to conservativeness. This is because, for $\eta \in (0, \frac{1}{2})$, the weights of all experts can be scaled up at each trial by the factor $(1-\eta)$ without changing the algorithm's predictions. This scaling prevents the drop in weight of those experts predicting correctly, yet the fact that no weight increases preserves the proofs of the bounds.

Note that bound (24) is trivial when $\ell < 1/2(\log N + \min_{1 \le i \le N} \log(1/P_i(\boldsymbol{y}^\ell)))$. We can use a game-theoretic analysis to obtain meaningful bounds for all ℓ. Here we present such an analysis for a slightly different prediction rule which is easier to analyze. Instead of the sigmoidal $-\log(1-x)$ we use the piecewise linear function (similar to the one used in [4])

$$F(x) = \begin{cases} 0 & \text{if} \quad x < (1 - \ln 2)/2, \\ 1 & \text{if} \quad x > (1 + \ln 2)/2, \\ x/\ln 2 + \frac{1}{2}(1 - 1/\ln 2) & \text{otherwise.} \end{cases} \tag{29}$$

This analysis shows that the best strategy for an adversary trying to maximize the total absolute loss of the algorithm is either to spread the information gain evenly over the sequence $\boldsymbol{y}^\ell$, or concentrate it on a subset of the sequence.

Let LIN be the master predictor based on (23) and (29).

THEOREM 5.2. *For all $\boldsymbol{y}^\ell \in \{0,1\}^\ell$, $\ell \in \mathbb{N}$, if* LIN *is run on $\boldsymbol{y}^\ell$ using an arbitrary noise model, then we have*

$$L_{\text{LIN}}(\boldsymbol{y}^\ell) \le \begin{cases} \ell \cdot C & \textit{if} \quad \ell < \log N + \log 1/P_{\text{best}}(\boldsymbol{y}^\ell), \\ \frac{1}{2}(\log N + \log 1/P_{\text{best}}(\boldsymbol{y}^\ell)) & \textit{otherwise;} \end{cases}$$

where

$$C = \min\left\{1,\ \max\left\{\frac{\ln 2 + 1 - 2\sqrt[\ell]{P_{\text{best}}(\boldsymbol{y}^\ell)/N}}{2\ln 2},\ 0\right\}\right\}$$

and $P_{\text{best}}(\boldsymbol{y}^\ell) = \max_{1\le i\le N} P_i(\boldsymbol{y}^\ell)$.

When $\ell \ge \log N + \log 1/P_{\text{best}}(\boldsymbol{y}^\ell)$, Theorem 5.2 gives an upper bound on the total loss of the LIN algorithm that matches the bound given in Theorem 5.1 for IGAIN. However, if the length of the prediction sequence satisfies $\ell < \log N + \log 1/P_{\text{best}}(\boldsymbol{y}^\ell)$, then Theorem 5.2 gives a tighter bound.

Before proving Theorem 5.2 we need some definitions. Let $W_t = \sum_{i=1}^N P_i(\boldsymbol{y}^{t-1})$ be the total weight of the experts at the beginning of trial t. Recall that, by definition of the noise model, $W_1 = N$ and $W_{t+1} \le W_t$ holds for all t. We now turn the on-line prediction model into a game where a predictor plays against an adversary choosing both the experts' advice and the bits to be predicted. As Theorem 5.2 must hold for arbitrary noise models, we also assume that the adversary can choose the experts' update factors. However, as the loss bound in Theorem 5.2 is parametrized with respect to the final weight of the best expert, we require that $W_{\ell+1}/W_1 = c$ hold at the end of the game, where the constant $c \in (0, 1)$ is a parameter of the game. A second parameter of the game is the number ℓ of *trials*. On each trial t of the game:

1. For each $i = 1, 2, \ldots, N$, the adversary chooses the advice $E_i(\boldsymbol{y}^{t-1})$ and the update factor η_i for expert E_i.
2. The predictor computes a value $\hat{y}_t \in [0, 1]$.
3. The adversary chooses a bit $y_t \in \{0, 1\}$ and the predictor is charged a loss of $|\hat{y}_t - y_t|$.

It should be clear that any upper bound on the total loss incurred by algorithm LIN playing the predictor for ℓ trials of this game is an upper bound on the total loss incurred by LIN on any sequence $\boldsymbol{y}^\ell \in \{0, 1\}^\ell$ when N experts and an arbitrary noise model are used.

The predictor's goal in a game with parameters c and ℓ is to minimize its total loss over the ℓ trials while the adversary's goal is to maximize it. Let MAXLOSS(ℓ, c) be the maximum loss the adversary can force on the predictor when the game is played with parameters c and ℓ. Furthermore, for each trial t, let r_t be the fraction W_{t+1}/W_t of the total weight voting for the correct value y_t. Note that $\prod_{t=1}^\ell r_t = c$. For our purposes, it is convenient to denote the loss $|\hat{y}_t - y_t|$ of the predictor at trial t by LOSS(r_t). It is not difficult to see that, for algorithm LIN,

$$\text{LOSS}(r_t) = \min\left\{1,\ \max\left\{\frac{\ln(2) + 1 - 2r_t}{2\ln(2)},\ 0\right\}\right\}.$$

We begin with the following simple claim which establishes the best strategy for the adversary when the game consists of two trials. In this case, depending upon the value of the parameter c, the best strategy is either to split c evenly among the two trials or to concentrate c on only one trial and to give up in the other trial.

CLAIM 5.1. *For any fixed* $c \in (0, 1)$, $\text{MAXLOSS}(2, c) = \max\{2\text{LOSS}(\sqrt{c}), \text{LOSS}(c)\}$.

PROOF. Let r, s be the fraction of the weight voting for the correct outcome on the first and second trial, respectively. Since c is the fraction of the weight left at the end of the game, it follows that $rs = c$. Therefore the total loss of the algorithm is

$$\min\left\{1,\ \max\left\{\frac{\ln(2)+1-2r}{2\ln(2)},\ 0\right\}\right\}+\min\left\{1,\ \max\left\{\frac{\ln(2)+1-2(c/r)}{2\ln(2)},\ 0\right\}\right\}.$$

The claim then follows from the fact that the function $(\ln(2)+1-2r)/(2\ln(2))+(\ln(2)+1-2(c/r))/(2\ln(2))$ is maximized when $r=\sqrt{c}$. □

From Claim 5.1 it immediately follows that, if $c \le c^* = (\ln^2 2/4)((2/\ln 2) - \sqrt{2}\sqrt{1-\ln 2}/\ln 2)^2$, then $\text{MAXLOSS}(2, c) = 2\,\text{LOSS}(\sqrt{c})$. Namely, the best strategy for the adversary is to split c evenly among the two trials. Similarly if $c \ge c^*$, then $\text{MAXLOSS}(2, c) = \text{LOSS}(c)$, as the adversary's best strategy is to concentrate c on only one trial and to give up in the other trial.

Claim 5.1 establishes the best strategy for the adversary when $\ell = 2$. The next result generalizes this strategy to the case $\ell > 2$. A reasonable generalization consists in setting $r_1 = \cdots = r_\ell = \sqrt[\ell]{c}$. However, as in the $\ell = 2$ case, for some choices of c the adversary maximizes the predictor's loss by "giving up" some trials and concentrating the fraction c on the remaining trials. For each fixed $c \in (0, 1)$, let $\ell^* = \ell^*(c)$ be the smallest integer ℓ achieving $\text{MAXLOSS}(\ell, c) = \max_{t\ge 1} \text{MAXLOSS}(t, c)$. Observe that, for any parameter $0 < c < 1$ of the game, the value $\ell^*(c)$ achieving the maximum loss is finite. This follows from the fact that the total weight of the experts drops by at least a factor of $(1 - \ln 2)/2$ on each of the trials where the algorithm incurs a positive loss.

CLAIM 5.2. *For any fixed* $c \in (0, 1)$,

$$\text{MAXLOSS}(\infty, c) = \max_{\ell\ge 1} \text{MAXLOSS}(\ell, c) = \ell^*\,\text{LOSS}(\sqrt[\ell^*]{c}).$$

PROOF. Let $S = (r_1, \ldots, r_{\ell^*})$, where $r_t = W_{t+1}/W_t$ for each $t = 1, \ldots, \ell^*$, be the sequence of splits maximizing the predictor's total loss. Without loss of generality, we can assume $r_t < 1$ for all $1 \le t \le \ell^*$. For the purpose of contradiction, assume that there is a t such that $r_t \ne \sqrt[\ell^*]{c}$. Then, as $\prod_t r_t = c$ must hold, there must be $t' \ne t$ such that $r_t \ne r_{t'}$. Without loss of generality let $t = 1$ and $t' = 2$. We now proceed by case analysis.

Case 1: $(1-\ln 2)/2 < r_1, r_2 < (\ln 2+1)/2$. Using Claim 5.1 we can consider the two following subcases. If $r_1 \cdot r_2 \le c^*$, then $\text{MAXLOSS}(2, r_1 \cdot r_2) = 2\,\text{LOSS}(\sqrt{r_1 r_2})$. Therefore the sequence $S' = (r'_1, r'_2, r_3, \ldots, r_{\ell^*})$, where $r'_1 = r'_2 = \sqrt{r_1 r_2}$ gives a higher total loss, contradicting the fact that S is the sequence maximizing the total loss. On the other hand, if $r_1 \cdot r_2 \ge c^*$, then $\text{MAXLOSS}(2, r_1 \cdot r_2) = \text{LOSS}(r_1 \cdot r_2)$, therefore the sequence $S' = (r'_1, r'_2, r_3, \ldots, r_{\ell^*})$, where $r'_1 = 1$ and $r'_2 = r_1 \cdot r_2$, can give a higher total loss. Now, if S is different from S', then the assumption that S' is the sequence maximizing the predictor's loss is contradicted. If $S = S'$, it must be the case that $r_1 = r'_1$ and $r_2 = r'_2$ or vice versa. However, both these cases contradict the assumption $r_1, r_2 < 1$.

Case 2. When either r_1 or r_2 lies outside the interval $[(1-\ln 2)/2,\ (\ln 2)+1)/2]$, the proof proceeds similarly to that of case 1. □

An explicit formula for the maximum loss incurred by the LIN predictor is given by the following result, which easily follows from Claim 5.2.

COROLLARY 5.1. *For any fixed* $c \in (0,1)$,

$$\text{MAXLOSS}(\infty, c) = \max\left\{\left\lceil \log\left(\frac{1}{c}\right)\right\rceil \text{LOSS}(\sqrt[\lceil \log(1/c)\rceil]{c}),\ \left\lfloor \log\left(\frac{1}{c}\right)\right\rfloor \text{LOSS}(\sqrt[\lfloor \log(1/c)\rfloor]{c})\right\}.$$

PROOF. From Claim 5.2 and the fact that the function $\ell\,\text{LOSS}(\sqrt[\ell]{c})$ is maximized when $\ell = \log(1/c)$ we obtain

$$\text{MAXLOSS}(\infty, c) \le \text{LOSS}(\sqrt[\log(1/c)]{c}) \log\frac{1}{c}.$$

The thesis then follows from the fact that the function $\ell\,\text{LOSS}\left(\sqrt[\ell]{c}\right)$ is concave. □

In order to prove Theorem 5.2 we need one further claim which establishes the strategy played by the adversary as a function of the number ℓ of trials to be played. Note that, for any $0 < c < 1$, $\ell^*(c)$ is the minimum number of trials for which the adversary is able to apply its best strategy. Thus, when $\ell < \ell^*(c)$ the adversary does not have enough trials to apply such an optimal strategy and the loss incurred by the algorithm is smaller.

CLAIM 5.3. *For any fixed* $c \in (0,1)$, *the following hold*:

$$\forall \ell \le \ell^*, \qquad \text{MAXLOSS}(\ell, c) = \ell\,\text{LOSS}(\sqrt[\ell]{c}),$$

$$\forall \ell \ge \ell^*, \qquad \text{MAXLOSS}(\ell, c) = \ell^*\,\text{LOSS}(\sqrt[\ell^*]{c}).$$

We are now ready to prove the main theorem of this section.

PROOF OF THEOREM 5.2. Let $\boldsymbol{y}^\ell$ be the sequence of length ℓ to be predicted and let $c = W_{\ell+1}/W_1$ be the fraction of the initial weight left after all bits have been predicted. Recall that $W_t = \sum_{i=1}^N P_i(\boldsymbol{y}^{t-1})$ and $P_{\text{best}} = P_{\text{best}}(\boldsymbol{y}^\ell) = \max_{1\le i\le N} P_i(\boldsymbol{y}^\ell)$. Clearly, $W_{\ell+1} \ge P_{\text{best}}$ implying $c \ge P_{\text{best}}/N$. This in turn implies

$$\text{MAXLOSS}(\ell, c) \le \text{MAXLOSS}\left(\ell, \frac{P_{\text{best}}}{N}\right). \tag{30}$$

Now, when $\log N/P_{\text{best}} \le \ell$, it follows from Claim 5.3 that

$$\text{MAXLOSS}\left(\ell, \frac{P_{\text{best}}}{N}\right) \le \text{LOSS}\left(\left(\frac{P_{\text{best}}}{N}\right)^{1/\log(N/P_{\text{best}})}\right) \cdot \log\frac{N}{P_{\text{best}}} = \tfrac{1}{2}\log\frac{N}{P_{\text{best}}},$$

which substituted in (30) gives the first of the two bounds of the theorem. When $\log N/P_{\text{best}} > \ell$, we can obtain a better bound since the adversary does not have enough trials to apply its best strategy. In this case it follows that

$$\begin{aligned}\text{MAXLOSS}\left(\ell, \frac{P_{\text{best}}}{N}\right) &= \ell \,\text{LOSS}\left(\sqrt[\ell]{\frac{P_{\text{best}}}{N}}\right) \\ &= \ell \cdot \min\left\{1,\ \max\left\{\frac{\ln 2 + 1 - 2\sqrt[\ell]{P_{\text{best}}/N}}{2\ln 2},\ 0\right\}\right\},\end{aligned}$$

which substituted in (30) proves the second bound of the theorem. □

6. Conclusions. We have examined the problem of predicting the bits of an unknown sequence using the advice of experts. If the unknown sequence is viewed as a corrupted version of an expert's advice, then Bayesian predictors can be defined. Bounds on the performance of these predictors can be improved by making the predictors conservative. Several existing algorithms, such as the WM and BW algorithms, can be derived in this way.

If we assume that the corruption is due to i.i.d. noise with a rate which is unknown, but drawn from a beta distribution, then we can define algorithms based on mean posterior estimates. We show that algorithms of this type are asymptotically optimal under the 0-1 loss, and they have the same asymptotic performance as existing algorithms which require more information about the true noise rate.

When the algorithms are allowed to "hedge their bets" by predicting a value in $[0, 1]$ we use the absolute loss to measure their performance. The analysis of one such algorithm (algorithm LIN of Section 5) involves interesting techniques. We show that the best adversaries split the loss evenly over some of the trials and inflict no loss on the others. When we started this research we hoped to find an algorithm for the experts setting with absolute loss bounds exactly half of the 0-1 loss bounds for the (conservative) BW algorithm. This seemed an attractive problem since the BW algorithm has the best known 0-1 loss bounds in the experts setting (although it does require significant side information), and the other important algorithms all had absolute loss variants with bounds equal to exactly one-half of their 0-1 loss bounds. It remains open whether or not there is an algorithm whose absolute loss is at most half of the BW algorithm's 0-1 loss bound.

Acknowledgments. We would like to thank Nick Littlestone, Manfred Warmuth, Mark Herbster, and Vladimir Vovk for many helpful conversations. In addition we are indebted to Robert Schapire for pointing out the connection between our work and Nick Littlestone's Apobayesian algorithm, as well as to Claudio Gentile and the anonymous referees for their many helpful suggestions.

References

[1] J.O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer Verlag, New York, 1985.

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

[3] N. Cesa-Bianchi, Y. Freund, D.P. Helmbold, and M.K. Warmuth. On-line prediction and conversion strategies. *Machine Learning*, 25:71–110, 1996.

[4] M. Feder, N. Merhav, and M. Gutman. Universal prediction of individual sequences. *IEEE Transactions on Information Theory*, 38:1258–1270, 1992.

[5] D. Haussler and A. Barron. How well does the Bayes method work in on-line predictions of $\{+1, -1\}$ values? In *Proceedings of 3rd NEC Symposium*, pages 74–100. SIAM, Philadelphia, PA, 1993.

[6] D. Haussler, J. Kivinen, and M.K. Warmuth. Tight worst-case loss bounds for predicting with expert advice. In *Proceedings of the 2nd European Conference on Computational Learning Theory*, pages 69–83. Lecture Notes on Artificial Intelligence, Vol. 904. Springer-Verlag, New York, 1995.

[7] J. Kivinen and M.K. Warmuth. Using experts for predicting continuous outcomes. In *Proceedings of the First Euro-COLT Workshop*. The Institute of Mathematics and Its Applications, Oxford, 1994.

[8] N. Littlestone. Mistake-Driven Bayes Sports: Bounds for Symmetric Apobayesian Learning Algorithms. Technical report, NEC Research Institute, Princeton, NJ, 1996.

[9] N. Littlestone and C. Mesterharm. An apobayesian relative of Winnow. In *Advances in Neural Information Processing Systems* 9. MIT Press, Cambridge, MA, 1997.

[10] N. Littlestone and M.K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108:212–261, 1994.

[11] M.J. Schervish. *Theory of Statistics*. Springer Verlag, New York, 1995.

[12] V.G. Vovk. Aggregating strategies. In *Proceedings of the 3rd Annual Workshop on Computational Learning Theory*, pages 372–383, 1990.

[13] V.G. Vovk. A game of prediction with expert advice. In *Proceedings of the 8th Annual Conference on Computational Learning Theory*, pages 51–60, 1995.