2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

Convergence Analysis for an Online Recommendation System

Anh Truong, Negar Kiyavash, and Vivek Borkar

Abstract— **Online recommendation systems use votes from experts or other users to recommend objects to customers. We propose a recommendation algorithm that uses an average weight updating rule and prove its convergence to the best expert and derive an upper bound on its loss. Often times, recommendation algorithms make assumptions that do not hold in practice such as requiring a large number of the good objects, presence of experts with the exact same taste as the user receiving the recommendation, or experts who vote on all or majority of objects. Our algorithm relaxes these assumptions. Besides theoretical performance guarantees, our simulation results show that the proposed algorithm outperforms current state-of-the-art recommendation algorithm, Dsybil.**

I. INTRODUCTION

In recent years, online learning algorithm have gained popularity through online recommendation systems, such as Netflix, Digg, or Rotten Tomatoes. These systems solicit opinions from users on items such as movies or news articles. We refer to the users, who vote for items, as "experts," and the items under consideration as "objects." Usually, recommendation systems offer a satisfaction scale (e.g., five-star rating scale in Rotten Tomatoes) from which an expert can choose to make a prediction about a particular object. After receiving the ratings of experts for an object, the system calculates its own recommendation score. For example, Rotten Tomatoes releases an average rating of experts voting for an object. In our formulation of the problem, after a user, say Alice, consumes the object, her feedback is used to update weights of experts. Weight of an expert is the trust level of the rating system in the ability of the expert to rate objects. The bigger the weight of an expert, the larger is his influence on the system's recommendation. The weight evolution is thus crucial for an recommendation system. Many weight updating rules have appeared in learning literature. In most of these work the performance of the algorithm is evaluated by proving bounds on its regret [7], [8], [9], [10].

In this paper, we address the problem of designing a provably-good recommendation system. Our contributions are as follows:

- We suggest a recommendation algorithm that uses a weighted average updating of expert weights similar to that of [2]. We prove that this updating rule will converge to the best expert if only one such expert exists or alternately switch between the best experts if multiple are present. Moreover, we derive an upper bound on the loss of our proposed algorithm. Unlike Dsybil we do not require a high fraction of objects in the voting pool to be good to obtain performance guarantees.
- While in the setup of Dsybil [3], experts can only cast positive, discretely valued votes, we allow experts to rate an object with any continuous value in [0,1]. Majority of common learning algorithms assume the experts are available at all rounds. We let experts refrain from voting at some rounds as this is a more realistic assumption for online recommendation systems.
- While for the proof of convergence of our algorithm, we make some assumptions on the availability and prediction distribution of experts, these assumptions are relaxed in our simulation results. In fact, our simulations show that the proposed algorithm outperforms the current state-of-the-art algorithm, Dsybil in more general settings than were assumed in the derivation of the convergence proofs.

The paper is organized as follows. After summarizing some closely related work, we describe our problem setup in Section III. In Section IV and Section V, convergence results and upper bounds on the loss of the algorithm are derived, subsequently. Simulation results appear in VI. Finally we conclude in Section VII.

II. RELATED WORKS

Littestone and Warmuth's seminal paper [1] provided a mathematical framework for analyzing predictions with expert advice. They also established lower bounds on the performance under both binary and continuous prediction values. However, in their setting, all experts are always available, which is not a realistic assumption for online recommendation system. Moreover, Littestone and Warmuth's discount the weight of an expert any time he makes a mistake. Thus, a good expert attending most of the rounds may end up with a lower weight than a poor expert participating in few rounds.

To avoid this, Yu et al. encouraged participation of the experts in voting by increasing their weights when they provide good advice [3]. Moreover, they award experts who vote on objects with fewer votes as opposed to safe-playing experts who vote for already popular objects. Yu et al. provide an upper bound on expected loss of their algorithm. However, their analysis holds under some strong assumptions such as a high percentage of objects in the recommendations system are good (Alice likes them) and a small number of good experts have voted on a large fraction of good objects.

Anh Truong is with Electrical and Computer Engineering Department, University of Illinois at Urbana-Champaign, Champaign, IL 61820, USA truong3@illinois.edu

Negar Kiyavash is with Industrial Enterprise Systems Engineering Department and CSL, University of Illinois at Urbana-Champaign, Champaign, IL 61820, USA kiyavash@ad.uiuc.edu

Vivek Borkar is with Electrical Engineering Department, Indian Institute of Technology, Mumbai 400076, India borkar@ee.iitb.ac.in

The requirement of having high percentage of good objects is partially due to the fact that their suggested algorithm does not incorporate the negative feedback in updating the weights of experts.

In learning with expert literature, it is predominantly assumed that all experts are available at all rounds, some exception are the work of Freund et al. and Kleinberg et al. who consider so called specialized experts or sleeping experts, respectively in [2] and [4]. In [2], Freund et al. presented an average updating rule in which the weight of each available expert is updated according to quality of his prediction and the weighted prediction of the algorithm. In effect, the algorithm finds a pseudo expert (average expert) whose loss is equal to the average loss of available experts and follows his prediction. Kleinberg et al. in [4] assumed that in each round, all available experts have unknown but fixed payoff distribution. The algorithm then chooses the best one based on the ordered arrangement of the expected payoff. They propose algorithms that achieve nearly optimal regret bounds, i.e., up to a constant or a sublogarithmic factor.

III. Setting

We now present our framework and notations. The set of all experts is denoted by $E = \{1, 2, ..., N\}$. At each round t, the set of available experts is denoted by E_t, where $E_t \subseteq E$. We assume that E_t is stationary and that the probability distribution of available experts is symmetric, i.e., it is invariant under permutations. Indeed, in the simulation, we do not use this assumption by offering the different availabilities among experts. The weight of expert i at time t is denoted by $p_t^i \in [0,1])$, and his prediction for a given object is $x_t^i \in [0,1]$. Here, we assume that E_t and x_t are independent of each other, which is a reasonable assumption since the correctness of an expert has no relation to his availability.

At time t, knowing the predictions (i.e., opinions) of awake experts for an object, the algorithm calculates the weighted prediction for that object, $\hat{y}_t$, as follows

$$\hat{y}_t = \frac{\sum\limits_{i \in E_t} p_t^i x_t^i}{\sum\limits_{i \in E_t} p_t^i}. \tag{1}$$

Thus, the system provides a rating for every object (rated by the experts). Recommendations for the system are user specific and for different users, the system maintains different weights for the experts, based on the feedback of the corresponding user. Upon consumption of an object, the user, say Alice, provides her feedback y_t to the system, indicating her opinion about the object. Throughout this paper, y_t is assumed to be binary 0 or 1, corresponding to bad and good ratings, respectively. After obtaining Alice's feedback, the weights of experts are updated as follow:

$$p_{t+1}^i = \begin{cases} p_t^i \frac{x_t^i}{\hat{y}_t} & \text{if } i \in E_t, y_t = 1, \\ p_t^i \frac{1-x_t^i}{1-\hat{y}_t} & \text{if } i \in E_t, y_t = 0, \\ p_t^i & \text{if } i \notin E_t. \end{cases} \tag{2}$$

It is worth noting that in (2) only weights for available experts, i.e., the ones in E_t, are updated. For experts not in E_t, their weights remain unchanged. Let us consider the case when we have a good object (according to Alice, of course). At this point, a good prediction algorithm is supposed to recommend a value greater than a half. Therefore, the update rule of (2) increases the weight of an expert if he has a recommendation better than the average. A similar explanation holds for the case of bad objects.

In this paper, we define the loss of this algorithm at time t as the relative entropy log loss, i.e., the natural logarithm of the difference between the outcome and the prediction of algorithm:

$$L_t = -I\{y_t = 1\} \ln \hat{y}_t - I\{y_t = 0\} \ln(1 - \hat{y}_t). \tag{3}$$

IV. Convergence of algorithm

In this section, we analyze the weight evolution and the convergence of the algorithm. We start with experts making binary predictions and then proceed to the continuous case.

Let us rewrite the update rule in (2) in matrix form. Let

$$p_t = \begin{bmatrix} p_t^1 \\ p_t^2 \\ \vdots \\ p_t^N \end{bmatrix},$$

then $p_{t+1} = D_t p_t$, where

$$D_t = \begin{bmatrix} d_t^1 & 0 & \dots & 0 \\ 0 & d_t^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & d_t^N \end{bmatrix}.$$

Since each expert updates his weight depending on his previous weight only, D_t is a diagonal matrix. Each diagonal element is given by (2), i.e.,

$$d_t^i = I\{i \notin E_t\} + I\{i \in E_t\} \Big[I\{y_t = 1\} \frac{x_t^i}{\hat{y}_t} + I\{y_t = 0\} \frac{1 - x_t^i}{1 - \hat{y}_t} \Big].$$

Define

$$q_t^i := I\{i \notin E_t\} + I\{i \in E_t\} \Big[I\{y_t = 1\} x_t^i + I\{y_t = 0\}(1 - x_t^i) \Big],$$

and the corresponding normalization factor

$$\bar{q}_t^i := I\{i \notin E_t\} + I\{i \in E_t\} \Big[I\{y_t = 1\} \hat{y}_t + I\{y_t = 0\}(1 - \hat{y}_t) \Big].$$

Note that $d_t^i = \frac{q_t^i}{\bar{q}_t^i}$. Hence, $D_t = Q_t \bar{Q}_t$, with

$$Q_t = \begin{bmatrix} q_t^1 & 0 & \dots & 0 \\ 0 & q_t^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & q_t^N \end{bmatrix},$$

and

$$\bar{Q}_t = \begin{bmatrix} \frac{1}{\bar{q}_t^1} & 0 & \dots & 0 \\ 0 & \frac{1}{\bar{q}_t^2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \frac{1}{\bar{q}_t^N} \end{bmatrix}.$$

Since Q_t and $\bar{Q}_t$ are diagonal matrices, by induction,

$$p_t = \Big(\prod_{m=0}^{t-1} Q_m \bar{Q}_m\Big) p_0.$$

Now, consider the ratio between weights of expert i and j at time t, $\xi_t^{ij} := \frac{p_t^i}{p_t^j} = \frac{\Big(\prod_{m=0}^{t-1} q_m^i \frac{1}{\bar{q}_m^i}\Big) p_0^i}{\Big(\prod_{m=0}^{t-1} q_m^j \frac{1}{\bar{q}_m^j}\Big) p_0^j}$.

We have,

$$\ln \xi_t^{ij} = \ln \frac{p_0^i}{p_0^j} + \sum_{m=0}^{t-1} (\ln q_m^i - \ln q_m^j) + \sum_{m=0}^{t-1} (\ln \bar{q}_m^j - \ln \bar{q}_m^i).$$

The following lemma is important for our main results.

Lemma 1. *Given the above settings and assumptions, if $E(\ln q_m^i) > E(\ln q_m^j)$ for $i \neq j$, then $p_t^j \to 0 \quad a.s.$ and $p_t^i \to 1 \quad a.s.$*

Proof. Taking the limit of $\frac{1}{t}\ln \xi_t^{ij}$ as $t \to \infty$, using the law of large numbers, with the assumptions that $\{q_m^i\}_1^\infty$ and $\{q_m^j\}_1^\infty$ are i.i.d., as are $\{\bar{q}_m^i\}_1^\infty$ and $\{\bar{q}_m^j\}_1^\infty$, we obtain

$$\frac{1}{t}\ln \xi_t^{ij} \to E(\ln q_m^i) - E(\ln q_m^j) + E(\ln \bar{q}_m^j) - E(\ln \bar{q}_m^i).$$

However, since we have assumed that the distribution of available experts is symmetric, $\bar{q}_m^i$ and $\bar{q}_m^j$ have the same distribution. Hence, $E(\ln \bar{q}_m^i) = E(\ln \bar{q}_m^j)$. Therefore,

$$\frac{1}{t}\ln \xi_t^{ij} \to E(\ln q_m^i) - E(\ln q_m^j).$$

It follows that if $E(\ln q_m^i) > E(\ln q_m^j)$, then $\ln \xi_t^{ij} \to \infty$, i.e., $\frac{p_t^i}{p_t^j} \to \infty$. Note that from the update rule (2), $\sum_{i=1}^n p_t^i = 1$ for every t. Therefore,

$$p_t^j \to 0 \quad a.s. \text{ and } p_t^i \to 1 \quad a.s. \tag{4}$$

□

Now, we consider whether the above inequality, $E(\ln q_m^i) > E(\ln q_m^j)$, holds. Define the time proportion of occurrence of a set of experts A as follows:

$$\Gamma(A) := \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} I\{E_t = A\} \quad \forall A \subseteq \{1, 2, ..., N\}.$$

Note that Γ is a probability measure and that the above definition is well defined since E_t is stationary. Define the accuracy of expert i, μ^i, to be the probability of correct predictions, with respect to user Alice.

A. Binary predictions

We first analyze the algorithm where x_t^i is allowed to take only two values, ε and $1-\varepsilon$, depending on the values of outcome y_t. Here, ε is a small positive value to guarantee that $\ln \varepsilon$ does not diverge to infinity. Consider the case when expert i is correct, which happens with probability μ^i. If $y_t = 1$, then $x_t^i = 1-\varepsilon$, and $q_t^i = 1-\varepsilon$. Similarly, if $y_t = 0$, then $x_t^i = \varepsilon$, and $q_t^i = 1-\varepsilon$.

A similar derivation holds for the case when the expert is not correct. It follows that, if $i \in E_t$

$$q_m^i = \begin{cases} 1-\varepsilon & \text{w.p } \mu^i, \\ \varepsilon & \text{w.p } 1-\mu^i. \end{cases} \tag{5}$$

So, the expected value of $\ln q_m^i$ is given by

$$\begin{aligned} E(\ln q_m^i) &= \sum_{A: i\in A} \Gamma(A)\Big(\mu^i \ln(1-\varepsilon) + (1-\mu^i)\ln\varepsilon\Big) \\ &= \Big(\sum_{A: i\in A} \Gamma(A)\Big)\Big(\mu^i \ln(1-\varepsilon) + (1-\mu^i)\ln\varepsilon\Big). \end{aligned} \tag{6}$$

Let us define the availability of expert i as the proportion of time that expert i is available, which is the first factor of (6). Also define the "index" of an expert i as the second factor in the right-hand side in (6), which is a non-decreasing function of his accuracy. The best expert is then defined as the expert whose index is highest, i.e., $E(\ln q_m^i) > E(\ln q_m^j)$ for all $j \neq i$. Our main results for the binary case is as follows.

Theorem 1. *In the binary setting, if there is only one best expert, then the algorithm converges to that expert. If there exists more than one best expert, then the algorithm alternates between those experts.*

Proof. We begin with the first case when there is only one expert attaining the highest index. From Lemma 1, the weight of expert j converges to zero for all $j \neq i$, while the weight of expert i converges to 1. The algorithm then follows that best expert. Note that from (6), the performance of an expert depends on both his accuracy and availability. In our convergence analysis, it has been assumed that the distribution of available experts is symmetric and hence all experts are equally available. Therefore, the algorithm will follow the most accurate expert.[1]

For the second case, without loss of generality, assume that there exists more than one expert who achieves the maximum

[1]In the simulation results, we present a more interesting comparison of the performance of our algorithm where both availability and accuracy are taken into account. There, the index is redefined as the product of accuracy and availability of an expert. The "best" expert must achieve the highest product. Thus, this index encourages experts to vote for objects, since it prefers experts who both participate in voting as well as provide correct recommendations.

value in (6), i.e., there exist experts i, j such that $E(\ln q_m^i) = E(\ln q_m^j) > E(\ln q_m^k)$ for all $k \neq i, j$. The proof is based on the law of the iterated logarithms. We recall the law of iterated logarithm (LIL): Given X_n i.i.d., variance σ^2, let

$$S_n := \sum_{m=1}^{n} X_m.$$

Then,

$$\limsup_{n\to\infty} \frac{S_n}{\sqrt{n \log\log n}} = \sqrt{2}\sigma \quad a.s.$$

Also,

$$\liminf_{n\to\infty} \frac{S_n}{\sqrt{n \log\log n}} = -\sqrt{2}\sigma \quad a.s.$$

Thus, the sum S_n will change signs infinitely often. Applying LIL for $X_m = \ln q_m^i$, along with the fact that $Prob\{p_t^i = p_t^j\} = 0$, one can see that $(\ln q_m^i - \ln q_m^j) > 0$ or $(\ln q_m^i - \ln q_m^j) < 0$ infinitely often. Therefore the scheme will switch between the two experts infinitely often. However, S_n can change only by a bounded amount in each step, from which it follows that the switchings become rarer and rarer, i.e., the algorithm stays with the same expert for longer and longer times. □

B. Continuous predictions

When x_t^i can take continuous values in $[0, 1]$, μ_i needs to be analyzed to a finer degree. For a given tolerance a, define the corresponding accuracy $\mu_i(a)$ of an expert i as the percentage of time that his recommendations lie within a distance a from Alice's feedback, i.e., $|x_t^i - y_t| < a$.

Suppose first that expert i is correct. If $y_t = 1$, then x_t^i is not far from 1 by a distance greater than a, i.e., $x_t^i > 1 - a$, and x_t^i is assumed to be uniformly distributed in $[1-a, 1]$. It follows that $q_t^i = x_t^i$ and q_t^i has the same distribution in this interval. Similarly, if $y_t = 0$, then $x_t^i < a$, and x_t^i is assumed to be uniformly distributed in $[0, a]$. Therefore, $q_t^i = 1 - x_t^i$ and q_t^i has uniform distribution in $[1 - a, 1]$.

A similar derivation holds for the case when expert i is not correct. Specifically, if $y_t = 1$, then $x_t^i < 1 - a$, $q_t^i = x_t^i$ and x_t^i is assumed to be uniformly distributed in $[1 - a, 1]$. If $y_t = 0$, then $x_t^i > a$, $q_t^i = 1 - x_t^i$ and x_t^i is assumed to be uniformly distributed in $[0, a]$.

The probability density function of q_t^i is derived as

$$f_{q^i}(x) = \begin{cases} \frac{1-\mu^i(a)}{1-a} & \text{if } x \in [0, 1-a], \\ \frac{\mu^i(a)}{a} & \text{if } x \in [1-a, 1], \\ 0 & \text{otherwise .} \end{cases}$$

The index of the expert i is given by

$$E(\ln q_t^i) = \Big(\sum_{A: i\in A} \Gamma(A)\Big)\Big(\frac{\mu^i(a)}{a}\int_{1-a}^{1} \ln x dx + \frac{1-\mu^i(a)}{1-a}\int_{0}^{1-a} \ln x dx\Big). \tag{7}$$

The same results as in Theorem 1 can be applied for this case with continuous values.

V. Upper bound on loss

We analyze the loss of the algorithm defined by (3). Let us rewrite the update rule in (2),

$$\begin{aligned} p_{t+1}^i &= p_t^i I\{i \notin E_t\} \\ &+ p_t^i I\{i \in E_t\}\left(I\{y_t = 1\}\frac{x_t^i}{\hat{y}_t} + I\{y_t = 0\}\frac{1 - x_t^i}{1 - \hat{y}_t}\right) \\ &= p_t^i f(x_t^i, y_t). \end{aligned}$$

By induction, the weight of expert i at time n is

$$p_n^i = p_0^i \prod_{t=1}^{n} f(x_t^i, y_t) = \frac{1}{N}\prod_{t=1}^{n} f(x_t^i, y_t). \tag{8}$$

Now, define the 'current best' expert by $i_t^* = \arg\max_{i\in E_t} \alpha_t^i \beta_t^i$, where

$$\alpha_t^i = \frac{\sum_{t'=1}^{t} I\{i\in E_{t'}\}}{t},$$

$$\beta_t^i = \frac{\sum_{t'=1}^{t} \left[I\{y_{t'}=1\}\ln x_{t'}^i + I\{y_{t'}=0\}\ln(1-x_{t'}^i)\right]}{\sum_{t'=1}^{t} I\{i\in E_{t'}\}}.$$

Intuitively, α_t^i is the availability accumulated up to time t, or the number of occurrences of expert i over t rounds. β_t^i is the accuracy accumulated up to time t of expert i, calculated by his performance over the number of predictions up to t. By this definition, the 'current best' expert is the one who is currently available, and whose product of accumulated availability and accuracy up to time t is largest. We will compare the loss of the algorithm to that of the 'current best' expert. We have:

$$\begin{aligned} &\sum_{t=1}^{n} l(\hat{y}_t) - \sum_{t=1}^{n} l(x_t^{i_t^*}) = \\ &= \sum_{t=1}^{n} [-I\{y_t = 1\}\ln \hat{y}_t - I\{y_t = 0\}\ln(1 - \hat{y}_t)] \\ &- \sum_{t=1}^{n} I\{i_t^* \in E_t\}\left[-I\{y_t = 1\}\ln x_t^{i_t^*} - I\{y_t = 0\}\ln(1 - x_t^{i_t^*})\right] \\ &= \sum_{t=1}^{n} I\{i_t^* \in E_t\}\left[I\{y_t = 1\}\ln\frac{x_t^{i_t^*}}{\hat{y}_t} + I\{y_t = 0\}\ln\frac{1 - x_t^{i_t^*}}{1 - \hat{y}_t}\right] \\ &= \sum_{t=1}^{n} \ln f(x_t^{i_t^*}, y_t) = \ln\frac{1}{N}\prod_{t=1}^{n} f(x_t^{i_t^*}, y_t) + \ln N, \end{aligned} \tag{9}$$

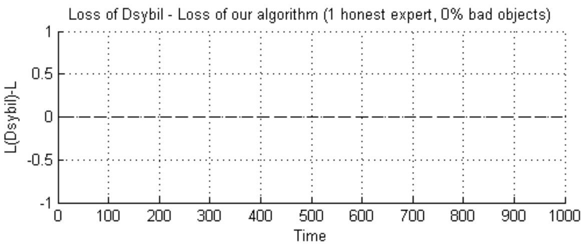

Fig. 1. Comparison of loss with one honest expert and 0% of bad objects.

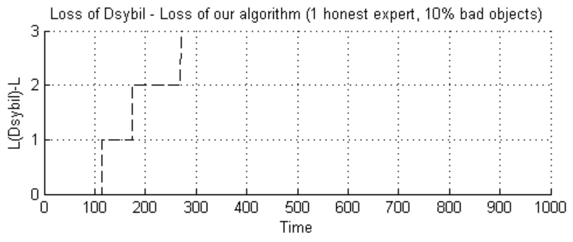

Fig. 2. Comparison of loss with one honest expert and 10% of bad objects.

where p_0 is the initial weight of the 'current best' expert, which is $1/N$. Without loss of generality, assume that the set of experts ranked by the descending order of accuracy is $\{1, 2, ..., N\}$. Obviously, if expert 1 is always present, then the current best expert is expert 1. Otherwise, the current expert could be any expert whose accumulated product is currently highest. Thus, this current best expert need not be expert 1.

Now assume that expert 1 is always available. Recall that the weight of each expert is always less than one, so $\ln p_n^1 \leq 0$, or equivalently, from (8) $\ln \frac{1}{N} \prod_{t=1}^{n} f(x_t^1, y_t) \leq 0$. Thus due to the above fact, $\ln \frac{1}{N} \prod_{t=1}^{n} f(x_t^{i_t^*}, y_t) \leq \ln \frac{1}{N} \prod_{t=1}^{n} f(x_t^1, y_t) \leq 0$.

From (9), $\sum_{t=1}^{n} l(\hat{y}_t) - \sum_{t=1}^{n} l(x_t^{i_t^*}) \leq \ln N$.

One can see that by applying the average update rule of (1), the algorithm suffers an amount of loss no more than the loss of the 'current best' expert plus the term $\ln N$.

VI. Simulation results

We consider a data set consisting of 20 objects, 10 experts and 1000 rounds. In other words, each object has a 1000×10 rate table containing the votes of 10 experts over 1000 rounds.

Table I gives an example of the recommendations for an object. The predictions of experts have values in $[0, 1]$, while outcomes (user's feedback after consumption) take only two values, 0 or 1. In this simulation, we attempt to create

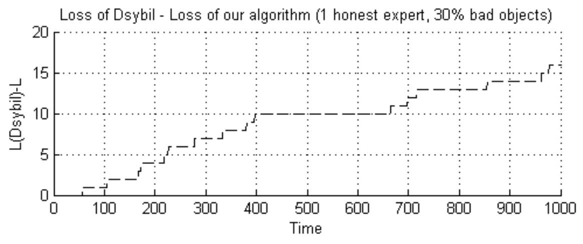

Fig. 3. Comparison of loss with one honest expert and 30% of bad objects.

different scenarios for the distribution of experts' predictions. In addition, we change the accuracy and availability of experts, as well as the fraction p of bad objects to compare with Dsybil [3]. One of the problems in comparing these algorithms is that the definitions of loss for them is different. Therefore, we consider the following common definition, the total number of mistakes the algorithm makes is $L_t = \sum_{v=1}^{t} |y_v - round(\hat{y}_v)|$, where "round" is a function that approximates a real number by its nearest integer. In Dsybil, $round(\hat{y}_v) = \hat{y}_v$ since $\hat{y}_v$ is 0 or 1.

In this simulation, we consider the following two cases. In the first case, there is one expert who is always correct. That is, the expert is not necessarily awake at all times, but whenever he is, he provides the right recommendation. In the second case, there are two best experts who are correct almost always. Figure 1 shows the comparison of the losses when there is no bad object for the case of a single best expert. In this case, Dsybil and our algorithm only need to follow the best expert and have the same performances. However, if the fraction of bad objects is increased (to 10% in Figure 2 and 30% in Figure 3) while still assuming a single correct expert, our algorithm outperforms Dsybil due to the following explanation.

In our algorithm, all experts express their opinions regardless of quality of the object, while in Dsybil, the experts only vote on good objects, and do not provide any recommendations for the bad objects. Thus, our algorithm still follows the "best" expert (one with recommendation close to 0) and does not suffer any additional loss. In contrast, knowing that the object is bad, the "best" experts in Dsybil do not provide their opinion concerning the object. Dsybil therefore must depend on other experts (who are usually not honest experts) rather than the "best" experts, and the recommendations will consequently follow the opinions of others. The additional loss is partially induced by this shortcoming of Dsybil. Therefore for Dsybil to perform acceptably a high fraction of the objects must be good. Clearly, such a requirement is quite unrealstic for an online recommendation system.

In figure 4, we analyze the case where 2 best experts have

TABLE I

RECOMMENDATIONS OF EXPERTS FOR AN OBJECT.

Outcome	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}
1	0.15	0.2	0.85		0.23	0.6	0.9	0.95	0.7	
1	0.5		0.8	0.4	0.83		0.2	0.5	0.6	0.9
0	0.4	0.1	0.15	0.06	0.5	0.8			0.5	0.4
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

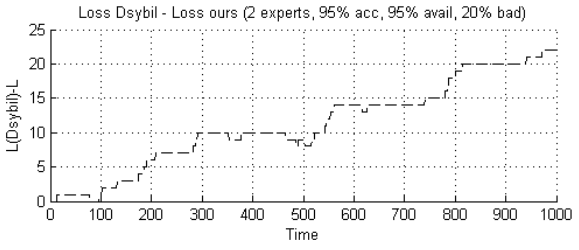

Fig. 4. Comparison of loss with two best experts and 20% of bad objects.

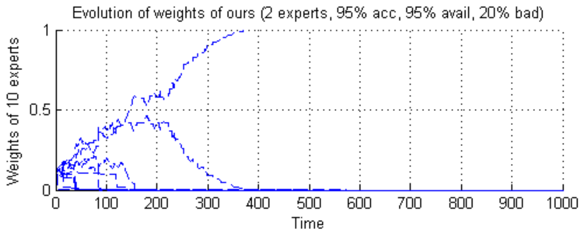

Fig. 6. Evolution of weights in Our algorithm with two best experts.

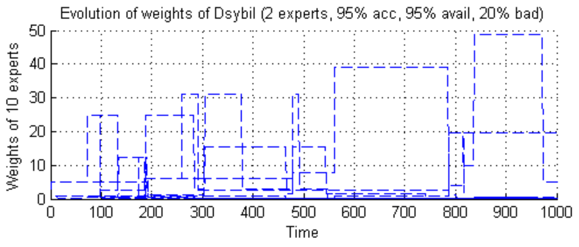

Fig. 5. Evolution of weights in Dsybil with two best experts.

95% accuracy and availability and show that our algorithm still outperforms Dsybil because it converges faster to the one of the best experts. Figure 5 and figure 6 shows that Dsybil keeps jumping between two experts and thus suffers a higher loss while our algorithm tracks one of the best experts after some number of rounds.

VII. CONCLUSION

We proposed a recommendation algorithm that uses an average-weight update rule and proved its convergence to the best expert and derived an upper bound on its loss. Besides theoretical performance guarantees, we relaxed some commonly made assumptions in the literature that do not hold in practice. For instance, existance of a large fraction of good objects in the voting pool, presence of experts with the exact same taste as the user, or experts who vote on all or majority of objects. Our simulation results show that the proposed algorithm outperforms current state-of-the-art recommendation algorithm, Dsybil even in more general setup than what was imposed for deriving our theoretical results.

VIII. ACKNOWLEDGMENTS

The authors would like to express their thanks to Prof. P. R. Kumar for helpful discussions. This work was supported in part by AFOSR FA9550-10-1-0345 and General Motors India Lab., Bangalore.

REFERENCES

[1] N. Littlestone, M. K. Warmuth, *The weighted majority algorithm*, Proceedings of the 30th Annual Symposium on Foundations of Computer Science, NC, 30 Oct-1 Nov, 1989.
[2] Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth, *Using and combining predictors that specialize*, Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997.
[3] H. Yu, C. Shi, M. Kaminsky, P. B. Gibbons, and F. Xiao, *DSybil: Optimal Sybil-Resistance for Recommendation Systems*, Proceedings of the IEEE Symposium on Security and Privacy, Oakland 2009.
[4] R. D. Kleinberg, A. Niculescu-Mizil, Y. Sharma, *Regret Bounds for Sleeping Experts and Bandits*, 21st Annual Conference on Learning Theory - COLT 2008, Helsinki, Finland, July 9-12, 2008.
[5] N. C. Bianchi, G. Lugosi, *Prediction, Learning, and Games*, Cambridge University Press, Oakland 2006.
[6] N. C. Bianchi, G. Lugosi, *Universal Prediction*, Cambridge University Press, Oakland 2006.
[7] A. Blum, Y. Mansour, *From External to Internal Regret*, The Journal of Machine Learning Research, Vol 8, 12/1/2007.
[8] D. Haussler, J. Kivinen, and M. Warmuth, *Tight Worst-Case Loss Bounds For Predicting With Expert Advice*, Technical Report, University of California at Santa Cruz, Santa Cruz, CA, USA, 1994.
[9] V. G. Vovk, *Aggregating strategies*, Proceedings of the third annual workshop on Computational learning theory (COLT '90), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 371-386, 1990.
[10] C. Bianchi, Y. Freund, D. Helmbold, D. Haussler, R. E. Schapire, and M. K. Warmuth, *How to use expert advice*, Proceedings of the twenty-fifth annual ACM symposium on Theory of computing (STOC '93), ACM, New York, NY, USA, 382-391, 1993.