Learning Binary Relations Using Weighted Majority Voting

(Extended Abstract)

Sally A. Goldman
Department of Computer Science
Washington University
St. Louis, Mo 63130
sg@cs.wustl.edu

Manfred K. Warmuth
Dept. of Computer and Information Sciences
University of California
Santa Cruz, Ca 95064
manfred@cis.ucsc.edu

Abstract

In this paper we apply a weighted majority voting algorithm to the problem of learning a binary relation between two sets of objects. When using exponentially many weights, the mistake bound of the algorithm is essentially optimal. We present a construction where a number of copies of our algorithm divide the problem amongst themselves and learn the relation cooperatively. In this construction the total number of weights is polynomial. The mistake bounds are non-optimal (at least when compared to the best bound obtainable when computational resources are ignored) but significantly improve previous mistake bound bounds achieved by polynomial algorithms. Moreover our method can handle noise, which widens the applicability of the results.

1 INTRODUCTION

In this paper we demonstrate how weighted majority voting can be applied to obtain robust algorithms for learning binary relations. Following Goldman, Rivest and Schapire [GRS89], a binary relation is defined between two sets of objects, one of cardinality n and the other of cardinality m. For all possible pairings of objects, there is a predicate relating the two sets of variables that is either true (1) or false (0). The relation is represented as an $n \times m$ matrix M of bits, whose (i, j) entry is 1 if and only if the relation holds between the corresponding elements of the two sets. Furthermore, there are a limited number of object types. Namely, the matrix M is restricted to have at most k distinct row types amongst its n rows. (Two rows are of the same type if they agree in all columns.) This restriction is satisfied whenever there are only k types of objects in the set of n objects being considered in the relation.

We shall study the problem of learning binary relations under the standard on-line (or incremental) learning model [Lit89, Lit88]. The *learning session* consists of a set of *trials*. In each trial, the learner must predict the value of some unknown matrix entry that has been selected by the adversary[1]. After predicting the learner receives the value at the matrix entry in question as *feedback*. If the prediction disagrees with the feedback, then we say the learner has made a *mistake*. The learning session continues until the learner has predicted each matrix entry. The goal of the learner is to make as few mistakes as possible.

Since the number of binary relations is at most $2^{km}k^n$ the standard halving algorithm [BF72, Lit88, Ang88] makes at most $km + n \lg k$ mistakes[2]. Observe that the halving algorithm can be viewed as keeping $2^{km}k^n$ weights, one weight per possible binary relation. Initially, all weights start at 1, and whenever a binary relation becomes inconsistent with the current partial matrix its weight is set to 0. To make a prediction for a given matrix entry, each binary relation votes according to its bit in that entry. Finally, the halving algorithm predicts according to the majority of consistent relations (i.e. those with weight 1), and thus each mistakes halves the total weight in the system. Since the initial weight is $2^{km}k^n$ and the final weight is at least 1, at most $km + n \lg k$ mistakes can occur.

Observe that the time used to make each prediction is linear in the number of weights. Thus we are interested in algorithms that use a small number of weights in representing their hypotheses. The algorithms we

ACM COLT '93 /7/93/CA, USA

[1] The adversary, who tries to maximize the learner's mistakes, knows the learner's algorithm and has unlimited computing power.

[2] Throughout this paper we let lg denote the base 2 logarithm.

present update the weights according to a variant of the weighted majority algorithm of Littlestone and Warmuth [LW89] called WMG. We view WMG as a node that is connected to each of its inputs by a weighted edge. The inputs are in the interval $[0,1]$. An input x of weight w votes with xw for 1 and $(1-x)w$ for 0. The node "combines" the votes of the inputs by determining the total weight q_0 (respectively q_1) placed on 0 (respectively 1) and predicts with the bit corresponding to the larger of the two totals (arbitrarily in case of a tie). After receiving feedback of what the prediction should have been, then for each input the fraction of the weight placed on the wrong bit is multiplied by β, where $\beta \in [0,1)$. Thus the weight w of an input x becomes $(1-x+x\beta)w$ if the feedback is 0 and $((1-x)\beta+x)w$ if the feedback is 1. If $\beta = 0$ then the total weight halves in each trial in which the node makes a mistake and we obtain an analysis like that of the halving algorithm.

We apply WMG to our problem in two different constructions. The first one uses one node and k^n weights. Thus the number of weights is still exponential but significantly lower than the number of weights of the halving algorithm. For this case, the analysis is straightforward and the bounds achieved are essentially optimal. The purpose of this construction is to show what is possible when computational resources are cheap. In the second construction we use one node for each of the n rows and one weight for each pair of rows (i.e. $\binom{n}{2}$ weights). Proving bounds for the second construction is much more involved and is the focus of the paper. The bounds obtained are non-optimal when compared to those obtained when computation time is not a concern. However, our bounds are significantly better than previous bounds obtained by a polynomial algorithm. We first discuss both constructions for the case when $\beta = 0$ and then discuss a noisy variant of our problem when $\beta > 0$ becomes useful.

In the first construction we use one weight per partition of the n rows into at most k row types (i.e. k^n weights). Initially all weights are set to 1. To make a prediction for a new matrix entry, each partition (with non-zero weight) votes as follows: If a column of the row type to which the new entry belongs has already been set then vote with the value of this bit, otherwise, vote with 1/2 causing the weight to be split between the votes of 0 and 1. Our algorithm predicts according to the weighted majority of these votes (see Figure 1). By selecting $\beta = 0$, the weight of partitions that predict incorrectly (and are thus inconsistent with the partial matrix) are set to zero and the weights of all partitions that split their vote are halved. After all entries of the target matrix are known, the correct partition has weight at least 2^{-km} since it never predicted incorrectly, and split its vote at most km times. Since the initial weight is k^n, we obtain the mistake bound of $km + n \lg k$ just as for the halving algorithm. (Actually, one can show that when $\beta = 0$ then the first construction simulates the halving algorithm with k^n weights instead of $2^{km}k^n$ weights). Note that the mistake bound of the halving algorithm is essentially optimal since Goldman et al. [GRS89] prove an information-theoretic lower bound of $\Omega(km + (n-k)\lg k)$ mistakes. While this construction has assumed that an upperbound on k is known, if no such bound is provided the standard doubling trick can be applied.

In the second construction one weighted majority node is used per row of the matrix, and one edge between each pair of nodes. Thus, unlike the first construction, no knowledge of k is needed. Let $e_{ii'}$ denote the edge between the node for row i and the node for row i', and let $w(e)$ to denote the weight of edge e. The node for row i dictates the predictions for all entries in row i. So the nodes, in some sense, partition the learning problem amongst themselves. Assume M_{ij} is the next value to predict. To make its prediction, the node for row i combines the votes of the $n-1$ inputs from the other nodes. Each node $i' \neq i$ votes with the weight $w(e_{i'i})$ for the bit at $M_{i'j}$. If the bit $M_{i'j}$ is unknown (corresponding to this input being 1/2) then the weight is split between the two votes (see Figure 2). If a mistake occurs then only the $n-1$ weights connected to node i are adjusted using a multiplicative weight update scheme. Thus we are running n copies of the WMG in parallel, where in each trial only one of the copies makes a prediction. We show that this parallel composition of weighted majority style algorithm "learns" in the sense that if there are few row types in the matrix then the algorithms eventually cooperate to make few mistake altogether. This holds even if the adversary gets to choose in what order the entries of the matrix M are uncovered.[3]

We demonstrate a technique to adapt the worst-case mistake bounds proven for the weighted majority algorithm WMG to this parallel application. We show when $\beta = 0$ then second construction obtains a mistake bound of $km + \min\{\frac{n^2}{2}\lg k, n\sqrt{3m\lg k}\}$ using only $\binom{n}{2}$ weights. Here k is the size of the smallest partition consistent with the whole matrix. The best previous bound for a polynomial algorithm was $km + n\sqrt{(k-1)m}$ [GRS89]. An interesting aspect of our problem besides its parallel nature is that when node i is to predict for entry M_{ij} then not all other $n-1$ entries in the j-th column may have been uncovered. Such unknown ("sleeping") entries are naturally set to 1/2, leading to split votes.

[3] A version of the above algorithm following the second construction is described in detail in Figure 3. For the sake of simplicity it is parameterized by an update the factor $\gamma = 2\beta/(1+\beta)$ instead of β. See Section 2 for a discussion of this algorithm called *Learn-Relation*(γ).

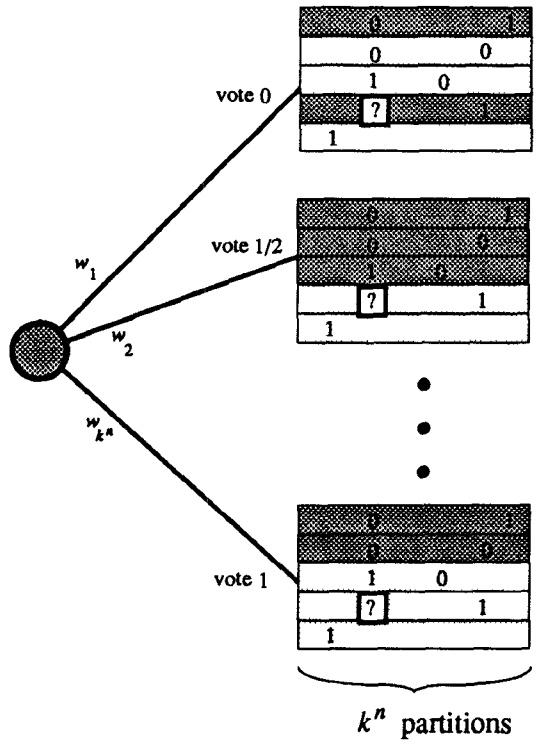

Figure 1: This figure illustrates how the voting works in our first construction. In this example $k = 2$. We use the two degrees of shading to indicate how the rows of the matrices are partitioned. Thus on the right we have shown the partially known matrix under three different partitions. Just to the left of each partition we show its vote for the unknown matrix entry. Recall that a partition voting with 1/2 can be viewed as a vote of 1 with half of its weight and a vote of 0 with half of its weight. So the overall prediction made in this example is 1 if and only if $w_2/2 + \cdots + w_{k^n} \geq w_1 + w_2/2 + \cdots$.

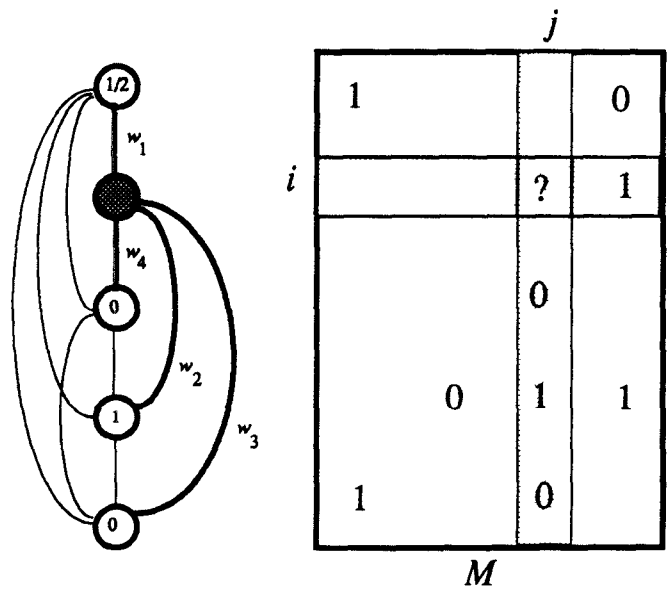

Figure 2: This figure illustrates how predictions are made in our second construction. The weighted majority node corresponding to row i (heavily shaded) is used to predict M_{ij}. The predictions of the inputs coming from the nodes are shown. So for this example the shaded node predicts 1 if $w_1/2 + w_2 \geq w_1/2 + w_3 + w_4$ and predicts 0 otherwise.

Learn-Relation(γ)

For all i, j such that $(i \neq j)$ initialize $w(e_{ij}) = 1$
To make a prediction for M_{ij}
    For each row $r \neq i$ such that $w(e_{ir}) > 0$
        If M_{rj} is not known then r predicts that $M_{ij} = 1/2$
        If $M_{rj} = 1$ then r predicts that $M_{ij} = 1$
        If $M_{rj} = 0$ then r predicts that $M_{ij} = 0$
    Let R_0 contain rows that predict $M_{ij} = 0$
    Let R_1 contain the algorithms that predict $M_{ij} = 1$
    If $\sum_{r \in R_1} w(e_{ir}) \geq \sum_{r \in R_0} w(e_{ir})$ predict 1
    Else predict 0
Receive correct value for M_{ij}
Update weights as follows
    For each $r \neq i$ such that $w(e_{ir}) > 0$
        If r made a correct prediction then let $w(e_{ir}) \leftarrow (2 - \gamma) w(e_{ir})$
        Else if r predicted incorrectly then let $w(e_{ir}) \leftarrow \gamma w(e_{ir})$

Figure 3: Our polynomial prediction algorithm for learning binary relations.

There are many relatives of the basic weighted majority algorithm that we could use within our two constructions such as the probabilistic variants due to Vovk [Vov90] (see also Cesa-Bianchi et al. [CBFH+93]) and WMR of Littlestone and Warmuth [LW89]. Also for example, WMC of Littlestone and Warmuth [LW89] handles inputs and predictions in [0,1] and the algorithm of Littlestone, Long and Warmuth [LLW91] selects linear combinations of the inputs. (Incidentally all these on-line prediction algorithms have multiplicative weight updates in common.) We chose the simplest setting for showing the usefulness of our method: the entries of the matrix are binary and the predictions must be deterministic.

A Generalization: Non-Pure Relations. In the second half of our paper we show how the robust nature of WMG enables us to solve an important generalization of the problem of learning binary relations with noisy data. To motivate this problem we briefly review the allergist example given by Goldman et al. [GRS89]. Consider an allergist with a set of patients to be tested for a given set of allergens. Each patient is either *highly allergic*, *mildly allergic*, or *not allergic* to any given allergen. The allergist may use either an *epicutaneous* (scratch) test in which the patient is given a fairly low dose of the allergen, or an *intradermal* (under the skin) test in which the patient is given a larger dose of the allergen. What options does the allergist have in testing a patient for a given allergen? He could just perform the intradermal test (option 0). Another option (option 1) is to perform an epicutaneous test, and if it is not conclusive, then perform an intradermal test. Which option is best? If the patient has no allergy or a mild allergy to the given allergen, then option 0 is best, since the patient need not return for the second test. However, if the patient is highly allergic to the given allergen, then option 1 is best, since the patient does not experience a bad reaction. The allergist's goal here is to minimize the number of prediction mistakes in choosing the option to test each patient for each allergen. Although Goldman et al. explore several possible methods for the selection of the presentation order, here we only consider the standard worst-case model in which an adversary determines the order in which the patient/allergen pairs are presented.

While this example is generally convincing, there is an assumption that is a clear oversimplification. Namely, they assume that there are a common set of "allergy types" that occur often and that most people fit into one of these allergy types. Thus the allergy types become the row types of the matrix. However, while it is

true that often people have very similar allergies, there are not really pure allergy types. In other words, it is unreasonable to assume that all rows of the same "type" are identical but rather they are just close to each other. Without this flexibility one may be required to have most patient's allergies correspond to a distinct allergy type. Henceforth, we shall refer to original formulation of the problem of learning binary relations in which all row types are "pure" as *learning pure relations*.

We propose the following generalization of this problem. Suppose that the rows of the matrix are partitioned into a set of k clusters $S_1, \ldots, S_k$. For each cluster we define a distance measure

$$d(S_i) = \min_{c_i \in \{0,1\}^n} \sum_{s \in S_i} H(s, c_i)$$

where $H(s, c_i)$ is the Hamming distance between s and c_i. Thus one can view the c_i at which the sum is minimized as the center point for the cluster such that the total number of disagreements $d(S_i)$ between each row and the center is minimized. Finally for a given partition p we define the *noise* α_p as $\sum_{S_i \in p} d(S_i)$, and the *size* k_p as the number of clusters in partition p. We refer to this problem as *learning non-pure relations*. Due to the robust nature of WMG, we can use both constructions to learn non-pure relations by simply using a non-zero update factor.

We now discuss both constructions when applied to the problem of learning non-pure relations and give bounds for each. The key to our approach is to view the discrepancies between the row templates and the actual rows as noise and thus the robust nature of the weighted majority algorithm enables us to handle such noise. The additional flexibility provided by this formulation will greatly reduce the mistake bounds that one can obtain when using the original formulation of Goldman et al. [GRS89] by reducing the number of row types by viewing minor differences as noise.

To demonstrate our basic approach, we now show that our first construction (i.e. the one using k^n weights) can learn a non-pure relation by making at most

$$\min\left\{k_p m + \frac{n \lg k}{\lg \frac{2}{1+\beta}} + \frac{\alpha_p \lg \frac{1}{\beta}}{\lg \frac{2}{1+\beta}}\right\}$$

mistakes in the worst case, where the minimum is taken over all partitions p of size at most k and k_p denotes the size and α_p the noise of partition p. The first construction for the noisy case still uses a single copy of the weighted majority algorithm where it has one weight for each partition. When a partition predicts incorrectly, its weight is multiplied by β. A partition that splits its vote has its weight multiplied by $(1+\beta)/2$. (Half of its weight remains unchanged and the other half is multiplied by β.) Observe that the true partition p predicts incorrectly at most α_p times and splits it vote at most km times. Thus the final weight in the system is at least $\beta^{\alpha_p}\left(\frac{1+\beta}{2}\right)^{km}$. Since the initial weight in the system is k^n and for each trial in which a mistake occurs the total weight after the trial is at most $(1+\beta)/2$ times the total weight before the trial, the stated bound holds. Finally, by applying the results of Cesa-Bianchi et al. [CBFH+93]) we can tune β as a function of an upper bound α on the noise leading to a worst-case mistake bound[4] of

$$\min\left\{k_p m + 2\left(\alpha_p + \sqrt{\alpha n \ln k} + n \lg k\right)\right\}$$

mistakes in the worst case, where the minimum is taken over all partitions p of size at most k with noise at most α and k_p denotes the size and α_p the noise of partition p. For the above tuning we needed an upper bound for both the size and the noise of the partition. If an upper bound for only one the two is known, then a standard doubling trick can be used to guess other. This causes only a slight increase in the mistake bound (See Cesa-Bianchi et al. [CBFH+93].) Note that in the above mistake bound there is a subtle tradeoff between the noise α_p and size k_p of a partition p.

When using the second construction for learning non-pure relations, we show that our algorithm makes at most

$$\min\left\{k_p m + 3n\sqrt{\frac{m}{2}\lg k_p} + \sqrt{6\alpha_p(k_p mn - \alpha_p)}\right\}$$

mistakes in the worst case, where the minimum is taken over all partitions p and k_p denotes the size and α_p the noise of partition p.

It is surprising that the parallel application of on-line algorithms using multiplicative weight updates can be used to do some non-trivial clustering with provable performance. Are there other applications where the clustering capability can be exploited? For the problem of learning binary relations the mistake bound of the polynomial algorithm (second construction) which uses $\binom{n}{2}$ weights is still far away from the optimal mistake bound of the exponential algorithm (first construction) which uses k^n weights. There seems to be a tradeoff between efficiency (number of weights) and the quality of the mistake bound. One of the most fascinating open problem regarding this research is the following: Is it possible to significantly improve our mistake bound by using say $O(n^3)$ weights? Or can one prove, based on some reasonable complexity theoretic or cryptographic assumptions, that no polynomial-time algorithm can perform significantly better?

[4] We also can get rid of the factor 2 in the bound by switching to a probabilistic prediction algorithm [CBFH+93].

The remainder of this paper is organized as follows. In Section 2 we present our polynomial-time algorithm to learn both pure and non-pure relations and relate it to the generalized weighted majority algorithm (WMG) of Littlestone and Warmuth. Next, we analyze the mistake bound of this algorithm for the special case in which the relation is pure. We then analyze our algorithm's performance for learning non-pure relations.

2 CONSTRUCTION TWO: A POLYNOMIAL-TIME ALGORITHM

In this section we discuss the algorithm obtained by our second construction that uses one weighted majority node per row. Our complete algorithm is shown in Figure 3. We begin by giving an update that is equivalent to the one used in WMG [LW89]. Recall that in WMG if x is the prediction of an input with weight w, then if the feedback is the bit ρ then w is multiplied by $1-(1-\beta)|x-\rho|$ for $\beta \in [0,1)$. If $\beta = \gamma/(2-\gamma)$, then for our application the update of WMG can be summarized as follows

- If a node predicts correctly (so, $|x-\rho| = 0$) its weight is not changed.
- If a node makes a prediction of $1/2$ then its weight is multiplied by $1/(2-\gamma)$.
- If a node predicts incorrectly (so, $|x-\rho| = 1$) then its weight is multiplied by $\gamma/(2-\gamma)$.

In the new update all factors in the above construction are simply multiplied by $(2-\gamma)$. This update is used in our Algorithm *Learn-Relation(γ)* since it leads to simpler proofs. Because voting is performed by a weighted majority vote, the predictions made by the two schemes are identical. In order to use the analysis technique of Littlestone and Warmuth we must obtain a lower bound for the final weight in the system. However, using WMG the weight in the system is decreased by nodes that do not predict, and thus we would have to compute an upper bound on the total number of times that this occurs. Thus to simplify the analysis, we have modified the update scheme (i.e. at each step we multiplied all weights by $(2-\gamma)$) so that the weights of nodes that do not predict remain unchanged.

2.1 LEARNING PURE RELATIONS

We now compute an upper bound on the number of mistakes made by *Learn-Relation* for the learning a pure relation. It is easily seen that when $k = 1$, that at most km mistakes are made. Thus for the remainder of this section we shall assume that $k \geq 2$. As when applying the weighted majority algorithm to a noise-free setting, we obtain the best performance by selecting $\gamma = 0$ (respectively $\beta = 0$).

We begin with some preliminaries. A function $f : \Re \rightarrow \Re$ is concave (respectively convex) over an interval D of $\Re$ if for all $x \in D$, $f''(x) \leq 0$ ($f''(x) \geq 0$). In our analysis we will repeatedly use the following variants of Jensen's inequality. Let f be a function from $\Re$ to $\Re$ that is concave over some interval D of $\Re$. Let $q \in \mathcal{N}$, and let $x_1, x_2, \ldots, x_q \in D$. Then

$$\sum_{i=1}^{q} x_i = S \Rightarrow \sum_{i=1}^{q} f(x_i) \leq q f(S/q).$$

Furthermore, if f is monotonically increasing over the interval D then the following holds:

$$\sum_{i=1}^{q} x_i \leq S \Rightarrow \sum_{i=1}^{q} f(x_i) \leq q f(S/q).$$

Likewise, let f be a function from $\Re$ to $\Re$ that is convex over some interval D of $\Re$. Let $q \in \mathcal{N}$, and let $x_1, x_2, \ldots, x_q \in D$. Then

$$\sum_{i=1}^{q} x_i = S \Rightarrow \sum_{i=1}^{q} f(x_i) \geq q f(S/q).$$

Let μ to denote the total number of mistakes made by the learner, and let μ_i will denote the number of mistakes that occur when the learner is predicting an entry in a row of type i. (Thus, $\sum_{i=1}^{k} \mu_i = \mu$.) Let n_i be the number of rows of type i. (So $n = \sum_{i=1}^{k} n_i$.) Let the set $\mathcal{E}$ contain all edges connecting two vertices of the same type. We further decompose $\mathcal{E}$ into $\mathcal{E}_1, \ldots, \mathcal{E}_k$ where $\mathcal{E}_i$ contains all edges between two rows of type i. Observe that $|\mathcal{E}_i| = \frac{n_i(n_i-1)}{2}$.

When making a prediction for M_{ij}, we define the *force* of a mistake to be the number of rows of the same type as row i for which column j was known when the mistake occurred. Let F_i be the sum of the forces of all mistakes made when predicting an entry in a row of type i. The following key lemma relates F_i to the weights on the edges in $\mathcal{E}_i$.

Lemma 1 *For each $1 \leq i \leq k$,*

$$F_i \leq \sum_{e \in \mathcal{E}_i} \lg w(e).$$

Proof: Observe that a mistake of force f that occurs when making a prediction for an entry of a row of type i, the weight on exactly f edges from $\mathcal{E}_i$ are doubled. Thus after all mistakes have occurred on rows of type i:

$$2^{F_i} \leq \prod_{e \in \mathcal{E}_i} w(e).$$

Taking logarithms of both sides we obtain the desired result. □

The basic structure of our proof, is to obtain an upper bound for the right hand side of the above equation. We then prove a lower bound for F_i and finally we combine these bounds to obtain an upper bound on the number of prediction mistakes made by our algorithm. A key observation used to prove an upper bound on $\sum_{e\in\mathcal{E}_i} \lg w(e)$ is that the overall weight in the system never increases. Therefore, since the initial weight in the system is $n(n-1)/2$ we obtain the following lemma.

Lemma 2 *Throughout the learning session,*

$$\sum_{e\in\mathcal{E}_i} w(e) \leq n(n-1)/2.$$

Proof: We use a proof by induction. Clearly the base case holds since there are $n(n-1)/2$ edges each with weight 1. For the inductive step observe that whenever a mistake occurs, the amount of weight set to 0 is at least as large as the amount that is doubled. □

In order to obtain a lower bound for F_i it is crucial to first obtain an upper bound on the number of mistakes for a given row type and given force. This characterizes the rate at which the weighted-majority nodes are gaining information.

Lemma 3 *For each row type r and force f there are at most m mistakes of force f.*

Proof: We use a proof by contradiction. Suppose that for row type r the learner makes $m+1$ force f mistakes. Then there must be two mistakes that occur for the same column. Suppose the first of these mistakes occurs when predicting M_{ij} and the second occurs when predicting $M_{i'j}$ where both rows i and i' are of type r. However, after making a force f mistake when predicting M_{ij} that entry is known and thus the force of the $M_{i'j}$ mistake must be at least $f+1$ giving the desired contradiction. □

We are now ready to analyze our algorithm performance when learning pure relations.

Theorem 4 *The algorithm Learn-Relation with $\gamma = 0$ learns a k-binary-relation making at most*

$$km + \min\left\{\frac{n^2}{2}\lg k, n\sqrt{3m\lg k}\right\}$$

mistakes in the worst-case.

Proof: From Lemma 1 we now that for $1 \leq i \leq k$ that $F_i \leq \sum_{e\in\mathcal{E}_i} \lg w(e)$. Thus by obtaining an upper bound on the sum of the log of the weights and a lower bound on F_i, we can combine them to obtain an upper bound on the total number of prediction mistakes.

Observe that the log function is concave and monotonically increasing. Thus by Jensen's inequality and Lemma 2 we get that

$$F_i \leq \sum_{e\in\mathcal{E}_i} \lg w(e) \leq |\mathcal{E}_i| \lg\left(\frac{n(n-1)}{2|\mathcal{E}_i|}\right). \qquad (1)$$

We can now compute a lower bound on F_i. Let σ_i denote the sum of the first μ_i elements of the sequence $\langle 0\rangle^m\langle 1\rangle^m\langle 2\rangle^m\cdots$. From Lemma 3 it follows that $F_i \geq \sigma_i$. Clearly

$$F_i \geq \mu_i - m \qquad (2)$$

since all but m mistakes have force at least one. Observe that if $n_i = 1$ then $F_i = 0$, and thus by Inequality (2) it follows that $\mu_i \leq m$. Thus, without loss of generality, we can assume that $n_i \geq 2$ for all i. Combining Lemma 1 with Inequality (2) we get that

$$\mu_i - m \leq F_i \leq |\mathcal{E}_i| \lg\left(\frac{n(n-1)}{2|\mathcal{E}_i|}\right).$$

Solving for μ_i and summing over k yields

$$\mu = \sum_{i=1}^{k} \mu_i \leq km + \sum_{i=1}^{k} |\mathcal{E}_i| \lg\left(\frac{n(n-1)}{2|\mathcal{E}_i|}\right).$$

It is easily verified that the function $|\mathcal{E}_i| \lg \frac{n(n-1)}{2|\mathcal{E}_i|}$ is concave and monotonically increasing for $1 \leq |\mathcal{E}_i| \leq n(n-1)/4$. Therefore, if $|\mathcal{E}_i| \leq n(n-1)/4$ for $1 \leq i \leq k$, then since $\sum_{i=1}^{k} |\mathcal{E}_i| \leq n(n-1)/2$, we can apply Jensen's inequality to obtain:

$$\mu \leq km + \frac{n^2}{2}\lg k. \qquad (3)$$

Observe that since $\sum_{i=1}^{k} |\mathcal{E}_i| \leq n(n-1)/2$, at most one of these terms can be greater than $n(n-1)/4$. Without loss of generality, consider when $|\mathcal{E}_k| > n(n-1)/4$. Since, $|\mathcal{E}_i| \lg \frac{n(n-1)}{2|\mathcal{E}_i|}$ is maximized when $|\mathcal{E}_i| = n(n-1)/4$, it follows that

$$\sum_{i=1}^{k} |\mathcal{E}_i| \lg\left(\frac{n(n-1)}{2|\mathcal{E}_i|}\right) \leq \sum_{i=1}^{k-1} |\mathcal{E}_i| \lg\left(\frac{n(n-1)}{2|\mathcal{E}_i|}\right) + \frac{n(n-1)}{4}.$$

Now since $\sum_{i=1}^{k-1} |\mathcal{E}_i| \leq n(n-1)/4$, we can apply Jensen's inequality to obtain:

$$\begin{aligned}\sum_{i=1}^{k} |\mathcal{E}_i| \lg\left(\frac{n(n-1)}{2|\mathcal{E}_i|}\right) &\leq \\ \frac{n(n-1)}{4}\lg(2(k-1)) + \frac{n(n-1)}{4} &= \\ \frac{n(n-1)}{2} + \frac{n(n-1)}{4}\lg(k-1) &\leq \\ \frac{n^2}{2}\lg k.\end{aligned}$$

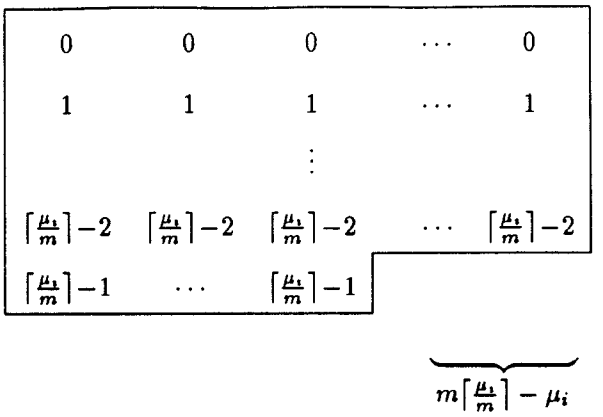

Figure 4: First μ_i elements of the sequence $\langle 0 \rangle^m \langle 1 \rangle^m \langle 2 \rangle^m \ldots$.

Thus Inequality (3) holds in all cases, proving our first bound on μ.

We now compute a more sophisticated lower bound on σ_i. Let $S(x) = \sum_{i=1}^{x} i$. Using the structure illustrated in Figure 4 it is easily seen that

$$\begin{aligned} F_i &\geq mS\left(\left\lceil \frac{\mu_i}{m} \right\rceil - 1\right) - \left(m\left\lceil \frac{\mu_i}{m} \right\rceil - \mu_i\right)\left(\left\lceil \frac{\mu_i}{m} \right\rceil - 1\right) \\ &= \frac{m}{2}\left\lceil \frac{\mu_i}{m} \right\rceil \left(\left\lceil \frac{\mu_i}{m} \right\rceil - 1\right) - m\left\lceil \frac{\mu_i}{m} \right\rceil^2 + m\left\lceil \frac{\mu_i}{m} \right\rceil \\ &\quad + \mu_i\left\lceil \frac{\mu_i}{m} \right\rceil - \mu_i \\ &\geq \left(\frac{\mu_i}{m} - 1\right)\left(\mu_i - \frac{m}{2}\left(\frac{\mu_i}{m} + 1\right)\right) \\ &= \frac{\mu_i^2}{2m} - \mu_i + \frac{m}{2} \end{aligned} \tag{4}$$

Combining Lemma (1) with Inequality (4) we get that

$$\frac{\mu_i^2}{2m} - \mu_i + \frac{m}{2} \leq F_i \leq |\mathcal{E}_i| \lg\left(\frac{n(n-1)}{2|\mathcal{E}_i|}\right).$$

Solving for μ_i yields

$$\mu_i \leq m + \sqrt{2m|\mathcal{E}_i| \lg\left(\frac{n(n-1)}{2|\mathcal{E}_i|}\right)}.$$

Thus,

$$\begin{aligned} \sum_{i=1}^{k} \mu_i &\leq km + \sqrt{2m}\sum_{i=1}^{k}\sqrt{|\mathcal{E}_i| \lg\left(\frac{n(n-1)}{2|\mathcal{E}_i|}\right)} \\ &\leq km + \sqrt{m}\sum_{i=1}^{k} n_i \sqrt{\lg\left(\frac{n(n-1)}{n_i(n_i-1)}\right)} \end{aligned}$$

where the final inequality uses the fact that

$$|\mathcal{E}_i| = n_i(n_i-1)/2 < n_i^2/2.$$

Noting that $\sum_{i=1}^{k} n_i = n$ and that $n_i\sqrt{\lg \frac{n(n-1)}{n_i(n_i-1)}}$ is concave for $n_i \geq 2$ we have

$$\begin{aligned} \mu = \sum_{i=1}^{k} \mu_i &\leq km + \sqrt{m}k\frac{n}{k}\sqrt{\lg\left(\frac{n(n-1)}{n/k(n/k-1)}\right)} \\ &\leq km + n\sqrt{m}\sqrt{\lg\frac{k^2(n-1)}{n-k}} \\ &\leq km + n\sqrt{m}\sqrt{\lg(k^3)} \\ &= km + n\sqrt{3m\lg k} \end{aligned} \tag{5}$$

where the last inequality comes from the fact that $k \geq \frac{n-1}{n-k}$ for $n \geq k+1$. Inequality (5) gives the second upperbound[5] on μ completing the proof. □

In addition to presenting their algorithm to make at most $km + n\sqrt{(k-1)m}$ mistakes, Goldman et al. [GRS89] present an information-theoretic lower bound for a class of algorithms that they call row-filter algorithms. They say that an algorithm A is a *row-filter algorithm* if A makes its prediction for M_{ij} strictly as a function of j and all entries in the set of rows consistent with row i and defined in column j. For this class of algorithms they show a lower bound to $\Omega(n\sqrt{m})$ for $m \geq n$ on the number of mistakes that any algorithm must make. Recently, William Chen [Che92] has extended their proof to obtain a lower bound of $\Omega(n\sqrt{m \lg k})$ for $m \geq n \lg k$. Observe that *Learn-Relation* is *not* a row-filter algorithm since the weights stored on the edges between the rows allows it to use the outcome of previous predictions to aid in its prediction for the current trial. Nevertheless, a simple modification of the projective geometry lower bound of Goldman et al. [GRS89] can be used to show an $\Omega(n\sqrt{m})$ lower bound for $m \geq n$ on the number of prediction mistakes

[5] Note that $n \geq k$. Furthermore, if $n = k$ then nm mistakes can occur, thus the restriction that $n \geq k+1$ does not limit the applicability of this result.

by our algorithm. Chen's extension of the projective geometry argument to incorporate k does not extend in such a straightforward manner, however, we conjecture that his lower bound can be generalized to prove that the mistake-bound we obtained for *Learn-Relation* is asymptotically tight. Thus to obtain a better algorithm, more than pairwise information between rows may be needed in making predictions.

2.2 LEARNING NON-PURE RELATIONS

In this section we show how we can take advantage of the robust nature of *Learn-Relation* to obtain an algorithm to learn non-pure relations.

Theorem 5 *For all $\beta \in [0,1)$, Algorithm Learn-Relation when using the parameter $\gamma = \frac{2\beta}{1+\beta}$ makes at most*

$$\min\left\{k_p m + \frac{n\sqrt{3m\lg k_p} + \sqrt{2m(\alpha_p k_p n - \frac{\alpha_p^2}{m})\lg\frac{1}{\beta}}}{\sqrt{\lg\frac{2}{1+\beta}}}\right\}$$

mistakes[6] in learning a binary-relation where the minimum is taken over all partitions p and k_p denotes the size and α_p the noise of partition p.

Proof: We now analyze the number of mistakes made by our algorithm using the basic technique established in the previous section. Let p be *any* partition of size k_p and noise α_p. We begin by noting that for any target matrix, there exists a partition p with $k_p = 1$ for which $\alpha_p \leq nm/2$. Namely, use the point obtained by selecting for each column the bit that occurs most often in that column (breaking ties arbitrarily) as the center. Clearly, using this center point, $\alpha_p \leq nm/2$. Thus, without loss of generality, we shall assume that we only consider partitions p for which $\alpha_p \leq nm/2$.

Let $\mathcal{E}_i$ contain all edges connecting two vertices corresponding to rows in the same cluster i of p. Let F_i be the sum of the forces of all mistakes made when predicting an entry in a row of cluster i. Let J_i be the number of times a weight in $\mathcal{E}_i$ predicts incorrectly (with all its weight) during the learning session. Thus it follows that:

$$\prod_{e\in\mathcal{E}_i} w(e) \geq (2-\gamma)^{F_i - J_i}(\gamma)^{J_i} = (2-\gamma)^{F_i}\left(\frac{\gamma}{2-\gamma}\right)^{J_i}.$$

Thus taking logarithms of both sides we obtain:

$$F_i \lg(2-\gamma) - J_i \lg\left(\frac{2-\gamma}{\gamma}\right) \leq \sum_{e\in\mathcal{E}_i} \lg w(e). \quad (6)$$

[6] Observe that when the optimal partition p is one for which $\alpha_p = 0$ and β is chosen as 0 then we obtain the same result as in the previous section.

For ease of exposition, let $a = \lg(2-\gamma) = \lg\frac{2}{1+\beta}$ and $b = \lg\frac{2-\gamma}{\gamma} = \lg\frac{1}{\beta}$.

We now proceed as in the analysis of the basic algorithm. Observe that Inequalities (1) and (4) apply here. Namely, we have that

$$\sum_{e\in\mathcal{E}_i} \lg w(e) \leq |\mathcal{E}_i| \lg\left(\frac{n(n-1)}{2|\mathcal{E}_i|}\right)$$

$$F_i \geq \frac{\mu_i^2}{2m} - \mu_i + \frac{m}{2}$$

Thus, we need just compute an upper bound for J_i. In the worst case, J is increased by one whenever $M_{ij} \neq M_{i'j}$ when rows i and i' are in the same cluster. For each row in S_i let $\delta_{i,j}$ be the number of elements of J_i in column j. Then

$$J_i = \sum_{j=1}^{m} \delta_{i,j}(n - \delta_{i,j}) = \delta_i n - \sum_{j=1}^{m} \delta_{i,j}^2$$

where $\sum_{j=1}^{m} \delta_{i,j} = \delta_i$. Since x^2 is convex, it follows that $\sum_{j=1}^{m} \delta_{i,j}^2 \geq \frac{\delta_i^2}{m}$. Thus,

$$J_i \leq \delta_i n - \frac{\delta_i^2}{m}.$$

Combining all of the above we obtain that:

$$a\left(\frac{\mu_i^2}{2m} - \mu_i + \frac{m}{2}\right) - b\left(n\delta_i - \frac{\delta_i^2}{m}\right) \leq |\mathcal{E}_i| \lg\left(\frac{n(n-1)}{2|\mathcal{E}_i|}\right).$$

Solving for μ_i yields

$$\mu_i \leq m + \frac{m}{a}\sqrt{\frac{2abn}{m}\delta_i - \frac{2ab}{m^2}\delta_i^2 + \frac{2a}{m}|\mathcal{E}_i| \lg\frac{n(n-1)}{2|\mathcal{E}_i|}}$$

$$\leq m + \sqrt{\frac{2m}{a}|\mathcal{E}_i| \lg\frac{n(n-1)}{2|\mathcal{E}_i|}} + \sqrt{\frac{2bm}{a}\delta_i\left(n - \frac{\delta_i}{m}\right)}.$$

Summing over the k_p partitions and using the inequality $|\mathcal{E}_i| = n_i(n_i-1)/2 \leq n_i^2/2$, we obtain

$$\mu \leq k_p m + \sqrt{\frac{m}{a}}\sum_{i=1}^{k_p}\left(n_i\sqrt{\lg\frac{n(n-1)}{n_i(n_i-1)}}\right) + \sqrt{\frac{2bm}{a}}\sum_{i=1}^{k_p}\sqrt{\delta_i n - \frac{\delta_i^2}{m}}.$$

We have that $\sum_{i=1}^{k_p} n_i = n$ and $\sum_{i=1}^{k_p} \delta_i \leq \alpha_p$. Now by the concavity of $n_i\sqrt{\lg\frac{n(n-1)}{n_i(n_i-1)}}$ and the observation that $\sqrt{n\delta_i - \delta_i^2/m}$ is concave and increasing for $0 \leq \delta_i \leq \frac{nm}{2}$ it follows that

$$\mu \leq k_p m + \sqrt{\frac{m}{a}} n\sqrt{\lg\frac{k_p^2(n-1)}{n-k_p}} + \sqrt{\frac{2bm}{a}\left(\alpha_p k_p n - \frac{\alpha_p^2}{m}\right)}.$$

Since $k_p \geq \frac{n-1}{k_p-1}$,

$$\mu \leq k_p m + n\sqrt{\frac{3}{a} m \lg k_p} + \sqrt{\frac{2bm}{a}\left(\alpha_p k_p n - \frac{\alpha_p^2}{m}\right)}$$

$$= k_p m + \frac{n\sqrt{3m \lg k_p} + \sqrt{2m(\alpha_p k_p n - \frac{\alpha_p^2}{m}) \lg \frac{1}{\beta}}}{\sqrt{\lg \frac{2}{1+\beta}}}$$

The above analysis was performed for any partition $p \in P$. Thus taking the minimum over all partitions in P we get the desired result. □

So for example, setting $\beta = 1/4$ gives that the number of mistakes is at most

$$\min\left\{k_p m + 3n\sqrt{\frac{m}{2} \lg k_p} + \sqrt{6\alpha_p(k_p mn - \alpha_p)} : p \in P\right\}.$$

For given upper bounds k and α on the size and noise of a partition, the techniques of Cesa-Bianchi et al. [?] can be used to obtain a near optimal selection for β.

Acknowledgements

We thank William Chen for his help in locating and correcting a bug in an earlier version of the analysis for our basic algorithm, and for the insight he provided us. We also thank the anonymous referees for their comments.

Sally Goldman is partially supported by NSF grant CCR-91110108. Manfred Warmuth is supported by ONR grant NO0014-91-J-1162 and NSF grant IRI-9123692.

References

[Ang88] Dana Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1988.

[BF72] J. Barzdin and R. Freivald. On the prediction of general recursive functions. *Soviet Mathematics Doklady*, 13:1224–1228, 1972.

[CBFH+93] Nicolò Cesa-Bianchi, Yoav Freund, David P. Helmbold, David Haussler, Robert E. Schapire, and Manfred K. Warmuth. How to use expert advice. In *Proceedings of the Twenty Fifth Annual ACM Symposium on Theory of Computing*, May 1993. To appear.

[Che92] William Chen, 1992. Personal communication.

[GRS89] Sally A. Goldman, Ronald L. Rivest, and Robert E. Schapire. Learning binary relations and total orders. In *30th Annual Symposium on Foundations of Computer Science*, pages 46–51, October 1989. To appear in *SIAM Journal of Computing*. Also a full version is available as Technical Report MIT/LCS/TM-413, MIT Laboratory for Computer Science, May 1990.

[Lit88] Nick Littlestone. Learning when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

[Lit89] Nicholas Littlestone. *Mistake Bounds and Logarithmic Linear-threshold Learning algorithms*. PhD thesis, U. C. Santa Cruz, March 1989.

[LLW91] Nicholas Littlestone, Philip M. Long, and Manfred K. Warmuth. On-line learning of linear functions. In *Proceedings of the Twenty Third Annual ACM Symposium on Theory of Computing*, pages 465–475, May 1991.

[LW89] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. In *30th Annual Symposium on Foundations of Computer Science*, pages 256–261, October 1989. To appear in *Information and Computation*.

[Vov90] Volodimir G. Vovk. Aggregating strategies. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 371–383. Morgan Kaufmann, August 1990.