ELSEVIER

Artificial Intelligence 106 (1998) 335–352

Artificial Intelligence

Worst-case analysis of the Perceptron and Exponentiated Update algorithms ☆

Tom Bylander

Division of Computer Science, The University of Texas at San Antonio, San Antonio, TX 78249, USA

Received 6 April 1998; received in revised form 21 September 1998

Abstract

The absolute loss is the absolute difference between the desired and predicted outcome. This paper demonstrates worst-case upper bounds on the absolute loss for the Perceptron learning algorithm and the Exponentiated Update learning algorithm, which is related to the Weighted Majority algorithm. The bounds characterize the behavior of the algorithms over any sequence of trials, where each trial consists of an example and a desired outcome interval (any value in the interval is an acceptable outcome). The worst-case absolute loss of both algorithms is bounded by: the absolute loss of the best linear function in a comparison class, plus a constant dependent on the initial weight vector, plus a per-trial loss. The per-trial loss can be eliminated if the learning algorithm is allowed a tolerance from the desired outcome. For concept learning, the worst-case bounds lead to mistake bounds that are comparable to past results.

Keywords: Learning algorithms; Absolute loss bounds; Mistake bounds; Randomized classification algorithms

1. Introduction

Linear and linear threshold functions are an important class of functions for machine learning. Although linear functions are limited in what they can represent, they often achieve good empirical results, e.g., [12,26], and they are standard components of neural networks.

For concept learning in which some linear threshold function is a perfect classifier, mistake bounds are known for the Perceptron algorithm [22,25], and the Winnow and

☆ This paper is a revised and extended version of Bylander [7].

[1] Email: bylander@cs.utsa.edu.

0004-3702/98/$ – see front matter
PII: S0004-3702(98)00098-8

Weighted Majority algorithms [18,19,21]. There are also results for these algorithms for various types of noise [3–6,10,20]. However, these previous results do not characterize the behavior of these algorithms over any sequence of examples.

This paper shows that minimizing the absolute loss characterizes the online behavior of two algorithms for learning linear threshold functions: the Perceptron algorithm and the Exponentiated Update algorithm (related to Weighted Majority), where the absolute loss is the sum of the absolute differences between the desired and predicted outcomes. The worst-case absolute loss of both algorithms is bounded by the sum of: the absolute loss of the best linear function in a comparison class, plus a constant dependent on the initial weight vector, plus a per-trial loss. The per-trial loss can be eliminated if the learning algorithm is allowed a tolerance from the desired outcome. In this latter case, the total additional loss is bounded by a constant over a sequence of any length.

The results of this paper hold for any sequence of examples and make no assumptions about the distribution of examples. Unfortunately, there is no direct relationship between absolute loss and the number of classification mistakes because a single misclassification could correspond to a small or a large absolute loss. Nevertheless, interesting mistake bounds can be derived in the linearly separable case.

A few previous results are also based on the absolute loss, though for specialized cases. Duda and Hart [11] derive the Perceptron update rule from the Perceptron criterion function, which is a specialization of the absolute loss. The Perceptron algorithm with a decreasing learning rate (harmonic series) on a stationary distribution of examples converges to a linear function with the minimum absolute loss [16]. A version of the Weighted Majority algorithm (WMC) has an absolute loss comparable to the best input [21]. Cesa-Bianchi [8] independently proved results similar to Theorems 2 and 3 of this paper; he also shows how to modify the algorithms for any loss function between the absolute loss and the square loss.

The analysis follows a pattern similar to worst-case analyses of online linear least-square algorithms [9,17]. The performance of an algorithm is compared to the best hypothesis in some comparison class. The bounds are based on how the distance from the online algorithm's current hypothesis to the target hypothesis changes in proportion to the algorithm's loss minus target's loss. The distance measure is chosen to facilitate the analysis.

The desired outcome for an example is allowed to be any real interval. Thus, concept learning can be implemented with a positive/negative outcome for positive/negative examples. In this case, the absolute loss bounds lead to mistake bounds for these algorithms that are similar to previous literature. Also, expected mistake bounds are obtained for randomized versions of the algorithms.

2. Preliminaries

A *trial* is an ordered pair $(\boldsymbol{x}, I)$, consisting of a real vector $\boldsymbol{x} \in \mathbb{R}^n$ (an *example*) and a real interval I (an *outcome*). A prediction $\widehat{y}$ on an example $\boldsymbol{x}$ is made using a weight

vector $\boldsymbol{w} \in \mathbb{R}^n$ by computing the dot product $\widehat{y} = \boldsymbol{w} \cdot \boldsymbol{x} = \sum_{i=1}^{n} w_i x_i$. The absolute loss of a weight vector $\boldsymbol{w}$ on a trial $(\boldsymbol{x}, I)$ is determined by:

$$\text{Abs-Loss}(\boldsymbol{w}, (\boldsymbol{x}, I)) = \begin{cases} y_{lo} - \widehat{y} & \text{if } \widehat{y} < I, \\ 0 & \text{if } \widehat{y} \in I, \\ \widehat{y} - y_{hi} & \text{if } \widehat{y} > I, \end{cases}$$

where $y_{lo} = \inf_{y \in I} y$ and $y_{hi} = \sup_{y \in I} y$. That is, it is desired for the prediction to be within the outcome interval. The Abs-Loss$(\cdot, \cdot)$ notation is also used to denote the absolute loss of a weight vector or algorithm (first argument) on a trial or sequence of trials (second argument).

For an online algorithm A, a comparison weight vector $\boldsymbol{u}$, and a trial sequence S, all of the bounds are of the form

$$\text{Abs-Loss}(A, S) \leqslant \text{Abs-Loss}(\boldsymbol{u}, S) + \zeta,$$

where ζ is an expression based on characteristics of the algorithm A and the trial sequence S. Before each trial S_t, the algorithm hypothesizes a weight vector $\boldsymbol{w}_t$. The bounds are based on demonstrating, for each trial S_t, that

$$\text{Abs-Loss}(\boldsymbol{w}_t, S_t) - \text{Abs-Loss}(\boldsymbol{u}, S_t) \leqslant \zeta_t,$$

and summing up the additional loss ζ_t over all the trials. When Abs-Loss$(\boldsymbol{w}_t, S_t) = 0$, obviously $\zeta_t = 0$ can be chosen. The other cases are covered by the following lemma.

Lemma 1. *When $\widehat{y} = \boldsymbol{w} \cdot \boldsymbol{x} < I$ for a given trial $S_t = (\boldsymbol{x}, I)$, then:*

$$\text{Abs-Loss}(\boldsymbol{w}, S_t) - \text{Abs-Loss}(\boldsymbol{u}, S_t) \leqslant \boldsymbol{u} \cdot \boldsymbol{x} - \widehat{y} = \boldsymbol{u} \cdot \boldsymbol{x} - \boldsymbol{w} \cdot \boldsymbol{x}. \tag{1}$$

When $\widehat{y} = \boldsymbol{w} \cdot \boldsymbol{x} > I$ for a given trial $S_t = (\boldsymbol{x}, I)$, then:

$$\text{Abs-Loss}(\boldsymbol{w}, S_t) - \text{Abs-Loss}(\boldsymbol{u}, S_t) \leqslant \widehat{y} - \boldsymbol{u} \cdot \boldsymbol{x} = \boldsymbol{w} \cdot \boldsymbol{x} - \boldsymbol{u} \cdot \boldsymbol{x}. \tag{2}$$

Proof. Let $y_{lo} = \inf_{y \in I} y$. When $\widehat{y} < I$, the first inequality follows from the fact that $y_{lo} - \widehat{y}$ is $\boldsymbol{w}$'s absolute loss and that $y_{lo} - \boldsymbol{u} \cdot \boldsymbol{x}$ is $\boldsymbol{u}$'s absolute loss when $\boldsymbol{u} \cdot \boldsymbol{x} \leqslant y_{lo}$, and that $y_{lo} - \boldsymbol{u} \cdot \boldsymbol{x}$ is less than $\boldsymbol{u}$'s absolute loss, otherwise. The proof for the second inequality is similar. □

3. Absolute loss bounds

Worst-case absolute loss bounds are derived for the Perceptron and Exponentiated Update algorithms, followed by a discussion.

3.1. Bounds for Perceptron

The Perceptron algorithm is given in Fig. 1. The Perceptron algorithm inputs an initial weight vector $\boldsymbol{s}$ (typically, the zero vector $\boldsymbol{0}$), and a learning rate η. The Perceptron update rule is applied if the prediction $\widehat{y}$ is outside the outcome interval, i.e., the current weight vector $\boldsymbol{w}$ is incremented (decremented) by $\eta \boldsymbol{x}$ if the prediction $\widehat{y}$ is too low (high). The use of any outcome interval generalizes the standard Perceptron algorithm.

Algorithm Perceptron(s,η)
Parameters:
s: the start vector, with $s \in \mathbb{R}^n$.
η: the learning rate, with $\eta > 0$.
Initialization:
Before the first trial, set $\boldsymbol{w}_1$ to s.
Prediction:
Upon receiving the tth example $\boldsymbol{x}_t$,
give the prediction $\widehat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$.
Update:
Upon receiving the tth outcome interval I_t,
update the weight vector using:

$$\boldsymbol{w}_{t+1} = \begin{cases} \boldsymbol{w}_t + \eta \boldsymbol{x}_t & \text{if } \widehat{y}_t < I_t, \\ \boldsymbol{w}_t & \text{if } \widehat{y}_t \in I_t, \\ \boldsymbol{w}_t - \eta \boldsymbol{x}_t & \text{if } \widehat{y}_t > I_t. \end{cases}$$

Fig. 1. Perceptron algorithm.

The behavior of the Perceptron algorithm is bounded by the following theorem, where $\|\boldsymbol{x}\| = \sqrt{\sum_{i=1}^n x_i^2}$ denotes the Euclidean norm of a vector $\boldsymbol{x}$.

Theorem 2. *Let S be a sequence of l trials. Let $X_P \geqslant \max_t \|\boldsymbol{x}_t\|$, the maximum vector length. Then for any comparison vector $\boldsymbol{u}$, where $\|\boldsymbol{u}\| \leqslant U_P$,*

$$\text{Abs-Loss}(\text{Perceptron}(\mathbf{0}, \eta), S) \leqslant \text{Abs-Loss}(\boldsymbol{u}, S) + \frac{U_P^2}{2\eta} + \frac{\eta l X_P^2}{2}.$$

Choosing $\eta = U_P/(X_P\sqrt{l})$ leads to:

$$\text{Abs-Loss}(\text{Perceptron}(\mathbf{0}, \eta), S) \leqslant \text{Abs-Loss}(\boldsymbol{u}, S) + U_P X_P \sqrt{l}.$$

Proof. Let $d(\boldsymbol{u}, \boldsymbol{w}) = \|\boldsymbol{u} - \boldsymbol{w}\|^2 = \sum_{i=1}^n (u_i - w_i)^2$. Consider the tth trial $S_t = (\boldsymbol{x}_t, I_t)$. Let $\widehat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$. If $\widehat{y}_t \in I_t$, then $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t$, and $d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1}) = 0$. If $\widehat{y}_t < I_t$, then $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \eta \boldsymbol{x}_t$, and it follows that:

$$\begin{aligned} d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1}) &= \sum_{i=1}^n (u_i - w_{t,i})^2 - \sum_{i=1}^n (u_i - w_{t+1,i})^2 \\ &= \sum_{i=1}^n (u_i - w_{t,i})^2 - \sum_{i=1}^n (u_i - w_{t,i} - \eta x_{t,i})^2 \\ &= 2\eta(\boldsymbol{u} \cdot \boldsymbol{x}_t - \boldsymbol{w}_t \cdot \boldsymbol{x}_t) - \eta^2 \|\boldsymbol{x}_t\|^2 \\ &\geqslant 2\eta(\boldsymbol{u} \cdot \boldsymbol{x}_t - \boldsymbol{w}_t \cdot \boldsymbol{x}_t) - \eta^2 X_P^2. \end{aligned}$$

From Lemma 1 and the fact that $\|\boldsymbol{x}_t\| \leqslant X_P$, it follows that:

$$\text{Abs-Loss}(\text{Perceptron}(\boldsymbol{w}_t, \eta), S_t) - \text{Abs-Loss}(\boldsymbol{u}, S_t)$$
$$\leqslant \boldsymbol{u} \cdot \boldsymbol{x}_t - \boldsymbol{w}_t \cdot \boldsymbol{x}_t \leqslant \frac{d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1})}{2\eta} + \frac{\eta X_P^2}{2}.$$

Similarly, if $\widehat{y}_t > I_t$, it follows that:

$$\text{Abs-Loss}(\text{Perceptron}(\boldsymbol{w}_t, \eta), S_t) - \text{Abs-Loss}(\boldsymbol{u}, S_t)$$
$$\leqslant \frac{d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1})}{2\eta} + \frac{\eta X_P^2}{2}.$$

By summing over all l trials:

$$\text{Abs-Loss}(\text{Perceptron}(\boldsymbol{0}, \eta), S) - \text{Abs-Loss}(\boldsymbol{u}, S)$$
$$= \sum_{t=1}^{l} \text{Abs-Loss}(\text{Perceptron}(\boldsymbol{w}_t, \eta), S_t) - \text{Abs-Loss}(\boldsymbol{u}, S_t)$$
$$\leqslant \sum_{t=1}^{l} \left(\frac{d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1})}{2\eta} + \frac{\eta X_P^2}{2} \right)$$
$$= \frac{d(\boldsymbol{u}, \boldsymbol{0}) - d(\boldsymbol{u}, \boldsymbol{w}_{l+1})}{2\eta} + \frac{\eta l X_P^2}{2}$$
$$\leqslant \frac{d(\boldsymbol{u}, \boldsymbol{0})}{2\eta} + \frac{\eta l X_P^2}{2} \leqslant \frac{U_P^2}{2\eta} + \frac{\eta l X_P^2}{2},$$

which proves the first inequality of the theorem. The second inequality follows immediately from the choice of η. □

3.2. *Bounds for Exponentiated Update*

The EU (Exponentiated Update) algorithm is given in Fig. 2. The EU algorithm inputs a start vector $\boldsymbol{s}$, a positive learning rate η, and a positive number U_E. Every weight vector consists of positive weights that sum to U_E. Normally, each weight in the start weight vector is set to U_E/n. For each trial, if the prediction $\widehat{y}$ is outside the outcome interval, then each weight w_i in the current weight vector $\boldsymbol{w}$ is multiplied (divided) by $e^{\eta x_i}$ if the prediction $\widehat{y}$ is too low (high). The updated weights are normalized so that they sum to U_E.

The EU algorithm can be used to implement the Weighted Majority algorithm [21]. Assuming that all $x_{t,i} \in [0, 1]$ and that β is the Weighted Majority's update parameter, set $\boldsymbol{s} = (1/n, \ldots, 1/n)$, $\eta = \ln 1/\beta$, and $U_E = 1$, and use outcome intervals of $[0, 1/2)$ of $(1/2, 1]$ for negative and positive examples, respectively. With these parameters, the EU algorithm makes the same classification decisions as the Weighted Majority algorithm. The only difference is that the weights are normalized to sum to U_E.

The EU algorithm is also closely related to the generalized EG algorithm [17]. If EG is instantiated using the absolute loss function, then one obtains the EU algorithm with $U_E = 1$ and real value outcomes (instead of real interval outcomes).[2]

[2] Kivinen and Warmuth [17] analyze the generalized EG algorithm using the square loss function.

Algorithm EU(s,η,U_E)

Parameters:

s: the start vector, with $\sum_{i=1}^{n} s_i = U_E$ and each $s_i > 0$.

η: the learning rate, with $\eta > 0$.

U_E: the sum of the weights for each weight vector, with $U_E > 0$.

Initialization:

Before the first trial, set each $w_{1,i}$ to s_i.

Prediction:

Upon receiving the tth example $\boldsymbol{x}_t$,
give the prediction $\widehat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$.

Update:

Upon receiving the tth outcome interval I_t,
update the weight vector using:

$$w_{t+1,i} = \begin{cases} \dfrac{U_E w_{t,i} e^{\eta x_{t,i}}}{\sum_{i=1}^{n} w_{t,i} e^{\eta x_{t,i}}} & \text{if } \widehat{y}_t < I_t, \\ w_{t,i} & \text{if } \widehat{y}_t \in I_t, \\ \dfrac{U_E w_{t,i} e^{-\eta x_{t,i}}}{\sum_{i=1}^{n} w_{t,i} e^{-\eta x_{t,i}}} & \text{if } \widehat{y}_t > I_t. \end{cases}$$

Fig. 2. Exponentiated Update algorithm.

This paper's analysis borrows two ideas from the analysis of the EG algorithm [17]: normalization of the weights so they always sum to U_E, and the relative entropy distance function. The behavior of the EU algorithm is bounded by the following theorem.

Theorem 3. *Let S be a sequence of l trials. Let $\boldsymbol{s} = (U_E/n, \ldots, U_E/n)$ be the start vector. Let $X_E \geqslant \max_{t,i} |x_{t,i}|$, the maximum magnitude of any value in an example. Then for any comparison vector $\boldsymbol{u}$, where $\sum_{i=1}^{n} u_i = U_E$ and where each $u_i \geqslant 0$:*

$$\text{Abs-Loss}\big(\text{EU}(\boldsymbol{s}, \eta, U_E), S\big) \leqslant \text{Abs-Loss}(\boldsymbol{u}, S) + \frac{U_E \ln n}{\eta} + \frac{\eta l U_E X_E^2}{2}.$$

Choosing $\eta = \sqrt{2 \ln n}/(X_E \sqrt{l})$ leads to:

$$\text{Abs-Loss}(\text{EU}(\boldsymbol{s}, \eta, U_E), S) \leqslant \text{Abs-Loss}(\boldsymbol{u}, S) + U_E X_E \sqrt{2l \ln n}.$$

Proof. Let S, l, $\boldsymbol{s}$, X_E, and U_E be defined as in the theorem. Let

$$d(\boldsymbol{u}, \boldsymbol{w}) = \sum_{i=1}^{n} u_i \ln(u_i/w_i),$$

where $0 \ln 0 = 0$ by definition. If the sum of $\boldsymbol{u}$'s weights is equal to the sum of $\boldsymbol{w}$'s weights, then $d(\boldsymbol{u}, \boldsymbol{w}) \geqslant 0$. Note that:

$$d(\boldsymbol{u}, \boldsymbol{s}) = \sum_{i=1}^{n} u_i \ln \frac{u_i n}{U_E} = \sum_{i=1}^{n} u_i \ln n - \sum_{i=1}^{n} u_i \ln \frac{U_E}{u_i} \leqslant U_E \ln n.$$

Consider the tth trial $S_t = (\boldsymbol{x}_t, I_t)$. Then $\widehat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$. Now if $\widehat{y}_t \in I_t$, then $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t$, and $d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1}) = 0$. If $\widehat{y}_t < I_t$, then

$$w_{t+1,i} = \frac{U_E w_{t,i} e^{\eta x_{t,i}}}{\sum_{j=1}^{n} w_{t,j} e^{\eta x_{t,j}}},$$

and it follows that:

$$\begin{aligned}
d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1}) &= \sum_{i=1}^{n} u_i \ln \frac{u_i}{w_{t,i}} - \sum_{i=1}^{n} u_i \ln \frac{u_i}{w_{t+1,i}} \\
&= \sum_{i=1}^{n} u_i \ln w_{t+1,i} - \sum_{i=1}^{n} u_i \ln w_{t,i} \\
&= \sum_{i=1}^{n} u_i \ln \frac{U_E e^{\eta x_{t,i}}}{\sum_{j=1}^{n} w_{t,j} e^{\eta x_{t,j}}} \\
&= \sum_{i=1}^{n} u_i \ln e^{\eta x_{t,i}} - \sum_{i=1}^{n} u_i \ln \sum_{j=1}^{n} \frac{w_{t,j} e^{\eta x_{t,j}}}{U_E} \\
&= \eta \sum_{i=1}^{n} u_i x_{t,i} - \sum_{i=1}^{n} u_i \ln \sum_{j=1}^{n} \frac{w_{t,j} e^{\eta x_{t,j}}}{U_E} \\
&= \eta \boldsymbol{u} \cdot \boldsymbol{x}_t - U_E \ln \sum_{i=1}^{n} \frac{w_{t,i} e^{\eta x_{t,i}}}{U_E}.
\end{aligned}$$

In Appendix A it is shown that:

$$\ln \sum_{i=1}^{n} \frac{w_{t,i} e^{\eta x_{t,i}}}{U_E} \leqslant \frac{\eta \boldsymbol{w}_t \cdot \boldsymbol{x}_t}{U_E} + \frac{\eta^2 X_E^2}{2}.$$

This implies that:

$$d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1}) \geqslant \eta \boldsymbol{u} \cdot \boldsymbol{x}_t - \eta \boldsymbol{w}_t \cdot \boldsymbol{x}_t - \frac{\eta^2 U_E X_E^2}{2}.$$

Using Lemma 1, it follows that:

$$\begin{aligned}
&\text{Abs-Loss}\big(\text{EU}(\boldsymbol{w}_t, \eta, U_E), S_t\big) - \text{Abs-Loss}(\boldsymbol{u}, S_t) \\
&\quad \leqslant \boldsymbol{u} \cdot \boldsymbol{x}_t - \boldsymbol{w}_t \cdot \boldsymbol{x}_t \leqslant \frac{d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1})}{\eta} + \frac{\eta U_E X_E^2}{2}.
\end{aligned}$$

Similarly, if $\widehat{y}_t > I_t$, it follows that:

$$\begin{aligned}
&\text{Abs-Loss}(\text{EU}(\boldsymbol{w}_t, \eta, U_E), S_t) - \text{Abs-Loss}(\boldsymbol{u}, S_t) \\
&\quad \leqslant \frac{d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1})}{\eta} + \frac{\eta U_E X_E^2}{2}.
\end{aligned}$$

By summing over all l trials:

$$
\begin{aligned}
&\text{Abs-Loss}\big(\text{EU}(\boldsymbol{s}, \eta, U_E), S\big) - \text{Abs-Loss}(\boldsymbol{u}, S) \\
&= \sum_{t=1}^{l} \text{Abs-Loss}\big(\text{EU}(\boldsymbol{w}_t, \eta, U_E), S_t\big) - \text{Abs-Loss}(\boldsymbol{u}, S_t) \\
&\leqslant \sum_{t=1}^{l} \left(\frac{d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1})}{\eta} + \frac{\eta U_E X_E^2}{2} \right) \\
&= \frac{d(\boldsymbol{u}, \boldsymbol{s}) - d(\boldsymbol{u}, \boldsymbol{w}_{l+1})}{\eta} + \frac{\eta l U_E X_E^2}{2} \\
&\leqslant \frac{d(\boldsymbol{u}, \boldsymbol{s})}{\eta} + \frac{\eta l U_E X_E^2}{2} \\
&\leqslant \frac{U_E \ln n}{\eta} + \frac{\eta l U_E X_E^2}{2},
\end{aligned}
$$

which proves the first inequality of the theorem. The second inequality follows immediately from the choice of η. □

3.3. Discussion

Theorems 2 and 3 provide similar results. They both have the form:

$$\text{Abs-Loss}(A, S) \leqslant \text{Abs-Loss}(\boldsymbol{u}, S) + \text{O}(l),$$

where l, the length of the trial sequence, is allowed to vary, and other parameters are fixed. If l is known in advance, then a good choice for the learning rate η leads to:

$$\text{Abs-Loss}(A, S) \leqslant \text{Abs-Loss}(\boldsymbol{u}, S) + \text{O}(\sqrt{l}).$$

Because there can be a small absolute loss for each trial no matter the length of the sequence, all the bounds depend on l. It is not hard to generate trial sequences that approach these bounds.

The bound for the Perceptron algorithm depends on U_P and X_P, which bound the respective lengths (two-norms) of the best weight vector and the example vectors. The bound for the EU algorithm depends on U_E, the one-norm of the best weight vector (the sum of the weights); X_E, the infinity-norm of the example vectors (the maximum magnitude of any value in any example); and a $\ln n$ term. Thus, similar to the square loss case [9, 17] and previous mistake bound analyses [19], the EU algorithm should outperform the Perceptron algorithm when the best comparison weight vector has many small weights and the example vectors have few small values.

The bound for the EU algorithm appears restrictive because the weights of the comparison vector must be nonnegative and must sum to U_E. However, a simple transformation can expand the comparison class to include negative weights with U_E as the upper bound on the sum of the weight's absolute values [17]. Specifically, the length of each example $\boldsymbol{x}$ is doubled by appending the values of $-\boldsymbol{x}$ to the example. This transformation doubles the number of weights, which would change the $\ln n$ term to $\ln 2n$.

4. Mistake bounds

To analyze concept learning, consider trial sequences that consist of *classification trials*, in which the outcome for each trial is either a positive or negative label. The classification version of an online algorithm is distinguished from the absolute loss version.

A *classification algorithm* classifies an example as positive if $\widehat{y} > 0$, and negative if $\widehat{y} < 0$, making no classification if $\widehat{y} = 0$. That is, an outcome interval of $(0, \infty)$ is used for positive examples, and an outcome interval of $(-\infty, 0)$ is used for negative examples. No updating is performed if the example is classified correctly. The choice of 0 for a classification threshold is convenient for the analysis; note that because Theorems 2 and 3 apply to any outcome intervals, any classification threshold could be used.

To relate the 0-1 loss on classification trials to absolute loss, slightly different outcome intervals are useful. An *absolute loss algorithm* uses the outcome interval $[1, \infty)$ for positive examples and the outcome interval $(-\infty, -1]$ for negative examples. An absolute loss algorithm performs updating if $\widehat{y}$ is not in the correct interval. As a result, the absolute loss of the absolute loss algorithm on a given trial is greater than or equal to the 0-1 loss of the classification algorithm using the same weight vector (the 0-1 loss for a trial is 1 if the classification algorithm is incorrect, and 0 if correct). For the following observation, a subsequence of a trial sequence omits zero or more trials, but does not change the ordering of the remaining trials.

Observation 4. *Let S be a classification trial sequence. If a classification algorithm makes m mistakes on S, then there is a subsequence of S of length m, where the corresponding absolute loss algorithm has an absolute loss of at least m. Equivalently, if there is no subsequence of S of length m, where the absolute loss algorithm has an absolute loss of m or more, then the classification algorithm must make fewer than m mistakes on S.*

Based on this observation, mistake bounds for the Perceptron and EU algorithms are derived. The notation Abs-Loss$(\cdot, \cdot)$ is used for the absolute loss of the absolute loss algorithm, and 0-1-Loss$(\cdot, \cdot)$ for the 0-1 loss of the classification algorithm.

Theorem 5. *Let S be a sequence of l classification trials. Let $X_P \geqslant \max_t \|\boldsymbol{x}_t\|$. Suppose there exists a vector $\boldsymbol{u}$ with $\|\boldsymbol{u}\| \leqslant U_P$ and* Abs-Loss$(\boldsymbol{u}, S) = 0$. *Let S' be any subsequence of S of length m. Then $m > U_P^2 X_P^2$ implies*

$$\text{Abs-Loss}\big(\text{Perceptron}\big(\mathbf{0}, 1/X_P^2\big), S'\big) < m,$$

which implies

$$\text{0-1-Loss}\big(\text{Perceptron}\big(\mathbf{0}, 1/X_P^2\big), S\big) < m.$$

Proof. Using Theorem 2, Abs-Loss$(\boldsymbol{u}, S) = 0$, $\eta = 1/X_P^2$, and $m > U_P^2 X_P^2$:

$$\begin{aligned}\text{Abs-Loss}\big(\text{Perceptron}(\mathbf{0}, \eta), S'\big) &\leqslant \text{Abs-Loss}(\boldsymbol{u}, S') + \frac{U_P^2}{2\eta} + \frac{\eta m X_P^2}{2} \\ &\leqslant \frac{U_P^2 X_P^2}{2} + \frac{m}{2} < \frac{m}{2} + \frac{m}{2} = m.\end{aligned}$$

Because every subsequence of length m has an absolute loss less than m, then Observation 4 implies 0-1-Loss(Perceptron($\mathbf{0}$, η), S) $< m$. □

Actually, the value of the learning rate does not affect the mistake bound when the start vector is the zero vector and 0 is the classification threshold. It only affects the relative length of the current weight vector. This is because the weight vector is the learning rate η times the sum of a subset of example vectors. η is always positive, so it cannot affect the sign of the dot product.

The mistake bound corresponds to previous mistake bounds in the literature. For example, suppose there exists a vector $\boldsymbol{u}$ with $\|\boldsymbol{u}\| = U_P$ and Abs-Loss($\boldsymbol{u}$, S) = 0. This means that the outcome of each positive or negative example is at least 1 or at most -1, respectively. This corresponds to a "separation" of 1 from the 0 classification threshold.

Now if $\boldsymbol{u}$ is transformed into a unit vector, the separation becomes $\delta = 1/U_P$. If each example $\boldsymbol{x}$ is also a unit vector, i.e., $X_P = 1$, then the mistake bound is $U_P^2 = 1/\delta^2$, which is identical to the bound of Minsky and Papert [22].[3]

Now consider the EU algorithm.

Theorem 6. *Let S be a sequence of l classification trials. Let $X_E \geqslant \max_{t,i} |x_{t,i}|$. Suppose there exists a vector $\boldsymbol{u}$ with nonnegative weights such that $\sum_{i=1}^{n} u_i = U_E$ and* Abs-Loss($\boldsymbol{u}$, S) = 0. *Let $\boldsymbol{s} = (U_E/n, \ldots, U_E/n)$. Let S' be any subsequence of S of length m. Then $m > 2U_E^2 X_E^2 \ln n$ implies*

$$\text{Abs-Loss}\big(\text{EU}\big(\boldsymbol{s}, 1/\big(U_E X_E^2\big)\big), S'\big) < m,$$

which implies

$$\text{0-1-Loss}\big(\text{EU}\big(\boldsymbol{s}, 1/\big(U_E X_E^2\big)\big), S\big) < m.$$

Proof. Using Theorem 3, Abs-Loss($\boldsymbol{u}$, S) = 0, $\eta = 1/(U_E X_E^2)$, and $m > 2U_E^2 X_E^2 \ln n$:

$$\begin{aligned}\text{Abs-Loss}\big(\text{EU}(\boldsymbol{s}, \eta, U_E), S'\big) &\leqslant \text{Abs-Loss}\big(\boldsymbol{u}, S'\big) + \frac{U_E \ln n}{\eta} + \frac{\eta m U_E X_E^2}{2} \\ &\leqslant U_E^2 X_E^2 \ln n + \frac{m}{2} < m.\end{aligned}$$

Because every subsequence of length m has an absolute loss less than m, then Observation 4 implies 0-1-Loss(EU($\boldsymbol{s}$, η, U_E), S) $< m$. □

While the learning rate is important for the EU classification algorithm, the normalization by U_E is unnecessary. The normalization affects the sum of the weights, but not their relative sizes.

This mistake bound corresponds to mistake bounds for the Weighted Majority algorithm and the Balanced algorithm in Littlestone [19].[4] Demonstrating the equivalence of the

[3] Block [2], Novikoff [23], and Papert [24] are generally credited with providing the first proofs of this mistake bound.

[4] In Littlestone [19], the Weighted Majority algorithm is also analyzed as a general linear threshold learning algorithm in addition to an analysis as a "master" algorithm as in Littlestone and Warmuth [21].

bounds is somewhat tedious because of superficial differences among the algorithms. However, a big-oh equivalence is easily shown. In Littlestone's analysis, $X_E = 1$ and comparison vectors have a separation of δ with weights that sum to 1. To get a separation of 1, the sum of the weights needs to be $U_E = 1/\delta$. Under these conditions, the bounds of this paper are $2U_E^2 X_E^2 \ln n = 2 \ln n/\delta^2$. The $O(\ln n/\delta^2)$ mistake bound agrees with Littlestone.

Mistake bounds can also be derived for when the best comparison vector also makes mistakes. Note that if a comparison vector makes a mistake on a classification trial, it can deviate from the threshold by as much as $U_E X_E$, which implies an absolute loss of up to $U_E X_E + 1$ for the absolute loss algorithm. This leads to the following theorem for the EU algorithm.

Theorem 7. *Let S be a sequence of l classification trials. Let $X_E \geqslant \max_{t,i} |x_{t,i}|$. Suppose there exists a vector $\boldsymbol{u}$ with nonnegative weights such that $\sum_{i=1}^{n} u_i = U_E$ and* 0-1-Loss$(\boldsymbol{u}, S) = k$. *Suppose also that* Abs-Loss$(\boldsymbol{u}, S_t) = 0$ *for all trials other than the k mistakes. Let $\boldsymbol{s} = (U_E/n, \ldots, U_E/n)$. Let S' be any subsequence of S of length m. Let η be any learning rate such that $\eta < 2/(U_E X_E^2)$. Then*

$$m > \frac{(U_E X_E + 1)k + \frac{U_E \ln n}{\eta}}{1 - \frac{\eta U_E X_E^2}{2}}$$

implies Abs-Loss(EU$(\boldsymbol{s}, \eta, U_E), S') < m$, *which implies* 0-1-Loss(EU$(\boldsymbol{s}, \eta, U_E), S) < m$.

Proof. If 0-1-Loss$(\boldsymbol{u}, S) = k$ and Abs-Loss$(\boldsymbol{u}, S_t) = 0$ for all trials other than the k mistakes, then Abs-Loss$(\boldsymbol{u}, S) \leqslant (U_E X_E + 1)k$ because each mistake can have a corresponding absolute loss of up to $U_E X_E + 1$. To use Theorem 3, we want to obtain:

$$\begin{aligned}\text{Abs-Loss}\big(\text{EU}(\boldsymbol{s}, \eta, U_E), S'\big) &\leqslant \text{Abs-Loss}\big(\boldsymbol{u}, S'\big) + \frac{U_E \ln n}{\eta} + \frac{\eta m U_E X_E^2}{2} \\ &\leqslant (U_E X_E + 1)k + \frac{U_E \ln n}{\eta} + \frac{\eta m U_E X_E^2}{2}.\end{aligned}$$

The last expression is less than m when $\eta < 2/(U_E X_E^2)$ and

$$m > \frac{(U_E X_E + 1)k + \frac{U_E \ln n}{\eta}}{1 - \frac{\eta U_E X_E^2}{2}}.$$

Because every subsequence of length m has an absolute loss less than m, then Observation 4 implies 0-1-Loss(EU$(\boldsymbol{s}, \eta, U_E), S) < m$. □

One special case of interest is when $U_E = 1$ and $X_E = 1$. This corresponds to using the EU algorithm as a master algorithm [19,21]. That is, the inputs to the EU algorithm are produced by the outputs of other learning algorithms, which in turn are being trained on the same sequence of observations. Suppose one of EU's inputs is produced by an algorithm that makes k or fewer mistakes (using -1 and 1 for encoding negative and positive predictions, respectively). Then, the mistake bound $2.67k + 2.67 \ln n$ can be obtained when

$\eta = 0.5$. This is close to the Weighted Majority mistake bound of $2.64k + 2.64 \ln n$ using $\beta = e^{-1}$ [21].[5]

5. Toleranced absolute loss

The above analysis leads to a per-trial loss for both algorithms, so consider an extension in which the goal is come within a tolerance τ of each outcome interval rather than directly hitting the interval itself. The notation Abs-Loss$(\cdot, S, \tau)$, where the tolerance τ is nonnegative, indicates that every outcome interval I of each trial in the trial sequence S is modified to $I' = I \pm \tau$ where $y' \in I'$ if and only if $y - \tau \leqslant y' \leqslant y + \tau$ for some $y \in I$. The absolute loss is calculated in accordance with the modified outcome intervals.

For the Perceptron and EU algorithms, the above analysis leads to an additional per-trial loss of $\eta X_P^2/2$ and $\eta U_E X_E^2/2$, respectively. If τ is equal to these values, then it turns out that the per-trial loss can be eliminated, leaving a constant additional loss over the sequence in the worst-case, independent of the length of the sequence. The proofs for Theorems 2 and 3 can be generalized to obtain the following theorems:

Theorem 8. *Let S be a sequence of l trials and τ be a positive real number. Let $X_P \geqslant \max_t \|\boldsymbol{x}_t\|$ and $\eta = 2\tau / X_P^2$. Then for any comparison vector $\boldsymbol{u}$, where $\|\boldsymbol{u}\| \leqslant U_P$*

$$\text{Abs-Loss}\big(\text{Perceptron}(\mathbf{0}, \eta), S, \tau\big) \leqslant \text{Abs-Loss}(\boldsymbol{u}, S) + \frac{U_P^2 X_P^2}{4\tau}.$$

Proof. Let

$$d(\boldsymbol{u}, \boldsymbol{w}) = \|\boldsymbol{u} - \boldsymbol{w}\|^2 = \sum_{i=1}^{n} (u_i - w_i)^2.$$

Consider the tth trial $S_t = (\boldsymbol{x}_t, I_t)$. Let $\widehat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$. If $\widehat{y}_t \in I_t \pm \tau$, then $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t$, and

$$d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1}) = 0.$$

If $\widehat{y}_t < I_t \pm \tau$, then $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \eta \boldsymbol{x}_t$. In the proof of Theorem 2, it was shown that

$$d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1}) \geqslant 2\eta(\boldsymbol{u} \cdot \boldsymbol{x}_t - \boldsymbol{w}_t \cdot \boldsymbol{x}_t) - \eta^2 X_P^2.$$

From Lemma 1 and the fact that $\|\boldsymbol{x}_t\| \leqslant X_P$, it follows that:

$$\begin{aligned}
&\text{Abs-Loss}\big(\text{Perceptron}(\boldsymbol{w}_t, \eta), S_t, \tau\big) - \text{Abs-Loss}(\boldsymbol{u}, S_t) \\
&\quad = \text{Abs-Loss}\big(\text{Perceptron}(\boldsymbol{w}_t, \eta), S_t\big) - \tau - \text{Abs-Loss}(\boldsymbol{u}, S_t) \\
&\quad \leqslant \boldsymbol{u} \cdot \boldsymbol{x}_t - \boldsymbol{w}_t \cdot \boldsymbol{x}_t - \tau \leqslant \frac{d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1})}{2\eta} + \frac{\eta X_P^2}{2} - \tau.
\end{aligned}$$

Similarly, if $\widehat{y}_t > I_t \pm \tau$, it follows that:

[5] One can obtain an analogue of Theorem 7 for the Perceptron algorithm, but the bounds for the master algorithm case are $O(k\sqrt{n})$, which is much worse than $O(k + \ln n)$.

$$\begin{aligned}&\text{Abs-Loss}\big(\text{Perceptron}(\boldsymbol{w}_t, \eta), S_t, \tau\big) - \text{Abs-Loss}(\boldsymbol{u}, S_t)\\ &\leqslant \frac{d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1})}{2\eta} + \frac{\eta X_P^2}{2} - \tau.\end{aligned}$$

By letting $\tau = \eta X_P^2/2$ and summing over all l trials:

$$\begin{aligned}&\text{Abs-Loss}\big(\text{Perceptron}(\boldsymbol{0}, \eta), S, \tau\big) - \text{Abs-Loss}(\boldsymbol{u}, S)\\ &= \sum_{t=1}^{l} \text{Abs-Loss}\big(\text{Perceptron}(\boldsymbol{w}_t, \eta), S_t, \tau\big) - \text{Abs-Loss}(\boldsymbol{u}, S_t)\\ &\leqslant \sum_{t=1}^{l} \frac{d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1})}{2\eta} \leqslant \frac{U_P^2}{2\eta} = \frac{U_P^2 X_P^2}{4\tau},\end{aligned}$$

which proves the inequality of the theorem. □

Theorem 9. *Let S be a sequence of l trials and τ be a positive real number. Let $\boldsymbol{s} = (U_E/n, \ldots, U_E/n)$ be the start vector. Let $X_E \geqslant \max_{t,i} |x_{t,i}|$ and $\eta = 2\tau/(U_E X_E^2)$. Then for any comparison vector $\boldsymbol{u}$, where $\sum_{i=1}^{n} u_i = U_E$ and where each $u_i \geqslant 0$:*

$$\text{Abs-Loss}\big(\text{EU}(\boldsymbol{s}, \eta, U_E), S, \tau\big) \leqslant \text{Abs-Loss}(\boldsymbol{u}, S) + \frac{U_E^2 X_E^2 \ln n}{2\tau}.$$

Proof. Let S, l, $\boldsymbol{s}$, X_E, and U_E be defined as in the theorem. Let

$$d(\boldsymbol{u}, \boldsymbol{w}) = \sum_{i=1}^{n} u_i \ln(u_i/w_i),$$

where $0 \ln 0 = 0$ by definition. If the sum of $\boldsymbol{u}$'s weights is equal to the sum of $\boldsymbol{w}$'s weights, then $d(\boldsymbol{u}, \boldsymbol{w}) \geqslant 0$. Recall from the proof of Theorem 3 that

$$d(\boldsymbol{u}, \boldsymbol{s}) \leqslant U_E \ln n.$$

Consider the tth trial $S_t = (\boldsymbol{x}_t, I_t)$. Then $\widehat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$. Now if $\widehat{y}_t \in I_t \pm \tau$, then $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t$, and $d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1}) = 0$. If $\widehat{y}_t < I_t \pm \tau$, then:

$$w_{t+1,i} = \frac{U_E w_{t,i} e^{\eta x_{t,i}}}{\sum_{j=1}^{n} w_{t,j} e^{\eta x_{t,j}}}.$$

In the proof for Theorem 3, it is shown that:

$$d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1}) \geqslant \eta \boldsymbol{u} \cdot \boldsymbol{x}_t - \eta \boldsymbol{w}_t \cdot \boldsymbol{x}_t - \frac{\eta^2 U_E X_E^2}{2}.$$

Using Lemma 1, it follows that:

$$\begin{aligned}&\text{Abs-Loss}\big(\text{EU}(\boldsymbol{x}_t, \eta, U_E), S_t, \tau\big) - \text{Abs-Loss}(\boldsymbol{u}, S_t)\\ &= \text{Abs-Loss}\big(\text{EU}(\boldsymbol{x}_t, \eta, U_E), S_t\big) - \tau - \text{Abs-Loss}(\boldsymbol{u}, S_t)\\ &\leqslant \boldsymbol{u} \cdot \boldsymbol{x}_t - \boldsymbol{w}_t \cdot \boldsymbol{x}_t - \tau \leqslant \frac{d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1})}{\eta} + \frac{\eta U_E X_E^2}{2} - \tau.\end{aligned}$$

Similarly, if $\widehat{y}_t > l_t \pm \tau$, it follows that

$$\begin{aligned}&\text{Abs-Loss}\big(\text{EU}(\boldsymbol{w}_t, \eta, U_E), S_t, \tau\big) - \text{Abs-Loss}(\boldsymbol{u}, S_t)\\ &\leqslant \frac{d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1})}{\eta} + \frac{\eta U_E X_E^2}{2} - \tau.\end{aligned}$$

By letting $\tau = \eta U_E X_E^2/2$ and summing over all l trials:

$$\begin{aligned}&\text{Abs-Loss}\big(\text{EU}(\boldsymbol{s}, \eta, U_E), S, \tau\big) - \text{Abs-Loss}(\boldsymbol{u}, S)\\ &= \sum_{t=1}^{l} \text{Abs-Loss}\big(\text{EU}(\boldsymbol{w}_t, \eta, U_E), S_t, \tau\big) - \text{Abs-Loss}(\boldsymbol{u}, S_t)\\ &\leqslant \sum_{t=1}^{l} \frac{d(\boldsymbol{u}, \boldsymbol{w}_t) - d(\boldsymbol{u}, \boldsymbol{w}_{t+1})}{\eta} \leqslant \frac{U_E \ln n}{\eta} = \frac{U_E^2 X_E^2 \ln n}{2\tau},\end{aligned}$$

which proves the inequality of the theorem. □

For both algorithms, the toleranced absolute loss of each algorithm exceeds the (nontoleranced) absolute loss of the best comparison vector by a constant over the whole sequence, no matter how long the sequence is. If the best comparison vector has a zero absolute loss, then the toleranced absolute loss is bounded by a constant over the whole sequence. These results strongly support the claim that the Perceptron and EU algorithms are online algorithms for minimizing absolute loss.

6. Randomized classification algorithms

To apply Theorems 8 and 9, again consider concept learning and classification trial sequences.[6] A *randomized classification algorithm* for a classification trial sequence is defined as follows. The prediction $\widehat{y}$ is converted into a classification prediction by predicting positive if $\widehat{y} \geqslant 1/2$, and negative if $\widehat{y} \leqslant -1/2$. If $-1/2 < \widehat{y} < 1/2$, then predict positive with probability $\widehat{y} + 1/2$, otherwise predict negative. It is assumed that the method for randomizing this prediction is independent of the outcome intervals, e.g., the outcome is fixed before the randomized prediction.

Under randomized prediction, classification outcomes are converted to outcome intervals by using $[1, \infty)$ and $(-\infty, -1]$ for positive and negative classification trials, respectively, just as was done above. However, a tolerance of $\tau = 1/2$ is added so that the toleranced absolute loss is determined based on outcome intervals of $[1/2, \infty)$ and $(-\infty, -1/2]$. Note that when $-1/2 < \widehat{y} < 1/2$, updating is performed regardless of whether the classification prediction is correct or not.

The idea of a randomized algorithm is borrowed from [21], which analyzes a randomized version of the Weighted Majority algorithm. This paper's randomization differs in that there are ranges of $\widehat{y}$, where positive and negative predictions are deterministic.

[6] Refer to Section 4 for the definition of classification trial sequence.

Note that the toleranced absolute loss of the randomized classification algorithm on a classification trial (referring to the $\widehat{y}$ prediction) is equal to the probability of an incorrect classification prediction if $-1/2 < \widehat{y} < 1/2$. Otherwise, the toleranced absolute loss is 0 for correct classification predictions and at least 1 for incorrect predictions. In all cases, the toleranced absolute loss is greater than or equal to the expected value of the 0-1 loss. This supports the following observation.

Observation 10. *Let S be a classification trial sequence. Then, the toleranced absolute loss of a randomized classification algorithm on S is greater than or equal to the expected value of the algorithm's* 0-1 *loss on S.*

The notation Abs-Loss$(\cdot, \cdot, 1/2)$ is used for the toleranced absolute loss of the randomized classification algorithm, and 0-1-Loss$(\cdot, \cdot, 1/2)$ for its 0-1 loss.

Theorem 11. *Let S be a sequence of l classification trials. Let $X_P \geqslant \max_t \|\boldsymbol{x}_t\|$. Suppose there exists a vector $\boldsymbol{u}$ with $\|\boldsymbol{u}\| \leqslant U_P$ and* Abs-Loss$(\boldsymbol{u}, S) = 0$. *Then*

$$\text{Abs-Loss}\left(\text{Perceptron}\left(\mathbf{0}, \frac{1}{X_P^2}\right), S, \frac{1}{2}\right) \leqslant \frac{U_P^2 X_P^2}{2},$$

which implies

$$E\left[\text{0-1-Loss}\left(\text{Perceptron}\left(\mathbf{0}, \frac{1}{X_P^2}\right), S, \frac{1}{2}\right)\right] \leqslant \frac{U_P^2 X_P^2}{2}.$$

Proof. Using Theorem 8, Abs-Loss$(\boldsymbol{u}, S) = 0$, $\eta = 1/X_P^2$, and $\tau = 1/2$:

$$\text{Abs-Loss}\big(\text{Perceptron}(\mathbf{0}, \eta), S, \tau\big) \leqslant \text{Abs-Loss}(\boldsymbol{u}, S) + \frac{U_P^2 X_P^2}{4\tau} = \frac{U_P^2 X_P^2}{2}.$$

Observation 10 implies $E[\text{0-1-Loss}(\text{Perceptron}(\mathbf{0}, \eta), S, \tau)] \leqslant U_P^2 X_P^2/2$. □

Theorem 12. *Let S be a sequence of l classification trials. Let $X_E \geqslant \max_{t,i} |x_{t,i}|$. Suppose there exists a vector $\boldsymbol{u}$ of nonnegative weights with $\sum_{i=1}^n u_i \leqslant U_E$ and* Abs-Loss$(\boldsymbol{u}, S) = 0$. *Let $\boldsymbol{s} = (U_E/n, \ldots, U_E/n)$. Then*

$$\text{Abs-Loss}\left(\text{EU}\left(\boldsymbol{s}, \frac{1}{U_E X_E^2}\right), S, \frac{1}{2}\right) \leqslant U_E^2 X_E^2 \ln n,$$

which implies

$$E\left[\text{0-1-Loss}\left(\text{EU}\left(\boldsymbol{s}, \frac{1}{U_E X_E^2}\right), S, \frac{1}{2}\right)\right] \leqslant U_E^2 X_E^2 \ln n.$$

Proof. Using Theorem 9, Abs-Loss$(\boldsymbol{u}, S) = 0$, $\eta = 1/(U_E X_E^2)$, and $\tau = 1/2$:

$$\text{Abs-Loss}\big(\text{EU}(\boldsymbol{s}, \eta), S, \tau\big) \leqslant \text{Abs-Loss}(\boldsymbol{u}, S) + \frac{U_E^2 X_E^2 \ln n}{2\tau} = U_E^2 X_E^2 \ln n.$$

Observation 10 implies $E[\text{0-1-Loss}(\text{EU}(\boldsymbol{s}, \eta), S, \tau)] \leqslant U_E^2 X_E^2 \ln n$. □

For both randomized algorithms, the worst-case bounds on the expected 0-1 loss is half of the worst-case mistake bounds of the deterministic algorithms. Roughly, randomization can improve the worse-case bounds by a factor of 2 because a value of $\widehat{y}$ close to 0 has a 0-1 loss of 1 in the deterministic worst case, while the expected 0-1 loss is close to $1/2$ for the randomized algorithms.

7. Conclusion

This paper has presented an analysis of the Perceptron and Exponentiated Update algorithms that shows that they are online algorithms for minimizing the absolute loss over a sequence of trials (examples). Specifically, this paper shows that the worst-case absolute loss of the online algorithms is comparable to the optimal weight vector from a class of comparison vectors.

The analysis is fully general. No assumptions about the linear separability or the probability distribution of the trials are made. The Perceptron analysis only refers to the maximum vector length of a example and the maximum vector length of a comparison vector. The Exponentiated Update analysis only refers to the maximum magnitude of a value in an example and the sum of weights of a comparison vector.

When a classification trial sequence is linearly separable, this paper has also shown that the absolute loss bounds are closely related to the known mistake bounds for both deterministic and randomized versions of these algorithms. Additional research is needed to study the classification behavior of these algorithms when the target comparison vector is allowed to drift, for both the linearly separable and nonseparable case. [7]

Based on minimizing absolute loss, it is possible to derive a backpropagation learning algorithm for multiple layers of linear threshold units. It would be interesting to determine suitable initial conditions and parameters that lead to good performance.

Acknowledgements

Thanks to Manfred Warmuth and anonymous reviewers for comments on this paper. This material is based in part upon work supported by the Texas Advanced Research Program under Grant No. 1997-010115-225.

Appendix A. Inequality for exponentiated update

A more general version of Lemma A.1 is shown in Hoeffding [15, p. 22]. It is presented here for completeness.

Lemma A.1. *Let $\boldsymbol{w} \in \mathbb{R}^n$ consist of nonnegative weights with $\sum_{i=1}^n w_i = U_E$. Let $\boldsymbol{x} \in \mathbb{R}^n$ such that $X_E \geqslant \max_i |x_i|$. Let η be any real number. Then the following inequality holds:*

$$\ln \sum_{i=1}^{n} \frac{w_i e^{\eta x_i}}{U_E} \leqslant \frac{\eta \boldsymbol{w} \cdot \boldsymbol{x}}{U_E} + \frac{\eta^2 X_E^2}{2}.$$

[7] See [1,13,14] for some interesting research along these lines.

Proof. Define f as

$$f(\eta, \boldsymbol{w}, \boldsymbol{x}) = \ln \sum_{i=1}^{n} \frac{w_i e^{\eta x_i}}{U_E}.$$

Now differentiate f twice with respect to η.

$$\frac{\partial f}{\partial \eta} = \frac{\sum_{i=1}^{n} w_i x_i e^{\eta x_i}}{\sum_{i=1}^{n} w_i e^{\eta x_i}}, \qquad \frac{\partial^2 f}{\partial \eta^2} = \frac{\sum_{i=1}^{n} w_i x_i^2 e^{\eta x_i}}{\sum_{i=1}^{n} w_i e^{\eta x_i}} - \left(\frac{\sum_{i=1}^{n} w_i x_i e^{\eta x_i}}{\sum_{i=1}^{n} w_i e^{\eta x_i}} \right)^2.$$

When $\eta = 0$, $f(\eta, \boldsymbol{w}, \boldsymbol{x}) = 0$ and $\partial f / \partial \eta = \boldsymbol{w} \cdot \boldsymbol{x} / U_E$. With regard to the second partial derivative, the following bound holds for the second partial derivative:

$$\frac{\partial^2 f}{\partial \eta^2} \leqslant \frac{\sum_{i=1}^{n} w_i x_i^2 e^{\eta x_i}}{\sum_{i=1}^{n} w_i e^{\eta x_i}} \leqslant \frac{X_E^2 \sum_{i=1}^{n} w_i e^{\eta x_i}}{\sum_{i=1}^{n} w_i e^{\eta x_i}} = X_E^2.$$

Hence, by Taylor's theorem:

$$f(\eta, \boldsymbol{w}, \boldsymbol{x}) \leqslant \frac{\eta \boldsymbol{w} \cdot \boldsymbol{x}}{U_E} + \frac{\eta^2 X_E^2}{2},$$

which is the inequality of the lemma. □

References

[1] P. Auer, M. Warmuth, Tracking the best disjunction, Machine Learning, to appear.

[2] H.D. Block, The Perceptron: a model for brain functioning, Reviews of Modern Physics 34 (1) (1962) 123–135.

[3] A. Blum, A. Frieze, R. Kannan, S. Vempala, A polynomial-time algorithm for learning noisy linear threshold functions, in: Proceedings 37th IEEE Annual Symposium on Foundations of Computer Science, 1996.

[4] T. Bylander, Learning linear-threshold functions in the presence of classification noise, in: Proceedings 7th Annual ACM Conference on Computational Learning Theory, 1994, pp. 340–347.

[5] T. Bylander, Learning linear threshold approximations using Perceptrons, Neural Computation 7 (1995) 370–379.

[6] T. Bylander, Learning probabilistically consistent linear threshold functions, in: Proceedings 10th Annual Conference on Computational Learning Theory, 1997, pp. 485–490.

[7] T. Bylander, Worst-case absolute loss bounds for linear learning algorithms, in: Proceedings AAAI-97, Providence, RI, 1997, pp. 485–490.

[8] N. Cesa-Bianchi, Analysis of two gradient-based algorithms for on-line regression, in: Proceedings 10th Annual Conference on Computational Learning Theory, 1997, pp. 163–170.

[9] N. Cesa-Bianchi, P.M. Long, M.K. Warmuth, Worst-case quadratic loss bounds for a generalization of the Widrow–Hoff rule, IEEE Trans. Neural Networks 7 (1996) 604–619.

[10] E. Cohen, Learning noisy Perceptrons by a Perceptron in polynomial time, in: Proceedings 38th IEEE Annual Symposium on Foundations of Computer Science, 1997.

[11] R.O. Duda, P.E. Hart, Pattern Classification and Scene Analysis, Wiley, New York, 1973.

[12] S.I. Gallant, Perceptron-based learning algorithms, IEEE Trans. Neural Networks 1 (1990) 179–191.

[13] M. Herbster, M. Warmuth, Tracking the best expert, Machine Learning, to appear.

[14] M. Herbster, M. Warmuth, Tracking the best regressor, in: Proceedings 11th Annual Conference on Computational Learning Theory, 1998.

[15] W. Hoeffding, Probability inequalities for sums of bounded variables, J. Amer. Statist. Assoc. 58 (1963) 13–30.

[16] R.L. Kashyap, Algorithms for pattern classification, in: J.M. Mendel, K.S. Fu (Eds.), Adaptive, Learning and Pattern Recognition Systems: Theory and Applications, Academic Press, New York, 1970, pp. 81–113.

[17] J. Kivinen, M.K. Warmuth, Exponentiated gradient versus gradient descent for linear predictors, Information and Computation 132 (1997) 1–63.
[18] N. Littlestone, Learning quickly when irrelevant attributes abound: a new linear-threshold algorithm, Machine Learning 2 (1988) 285–318.
[19] N. Littlestone, Mistake bounds and logarithmic linear-threshold learning algorithms, Ph.D. Thesis, University of California, Santa Cruz, CA, 1989.
[20] N. Littlestone, Redundant noisy attributes, attribute errors, and linear-threshold learning using Winnow, in: Proceedings 4th Annual Workshop on Computational Learning Theory, 1991, pp. 147–156.
[21] N. Littlestone, M.K. Warmuth, The weighted majority algorithm, Information and Computation 108 (1994) 212–261.
[22] M.L. Minsky, S.A. Papert, Perceptrons, MIT Press, Cambridge, MA, 1969.
[23] A.B.J. Novikoff, On convergence proofs for Perceptrons, in: Proceedings Symposium on the Mathematical Theory of the Automata, Vol. XII, 1962, pp. 615–622.
[24] S. Papert, Some mathematical models of learning, in: Proceedings 4th London Symposium on Information Theory, 1961.
[25] F. Rosenblatt, Principles of Neurodynamics, Spartan Books, New York, 1962.
[26] J. Shavlik, R.J. Mooney, G. Towell, Symbolic and neural learning programs: an experimental comparison, Machine Learning 6 (1991) 111–143.