Learning Linear Functions with Quadratic and Linear Multiplicative Updates

Tom Bylander BYLANDER@CS.UTSA.EDU
Department of Computer Science, University of Texas at San Antonio, San Antonio, TX 78249 USA

Abstract

We analyze variations of multiplicative updates for learning linear functions online. These can be described as substituting exponentiation in the Exponentiated Gradient (EG) algorithm with quadratic and linear functions. Both kinds of updates substitute exponentiation with simpler operations and reduce dependence on the parameter that specifies the sum of the weights during learning. In particular, the linear multiplicative update places no restrictions on the sum of the weights, and, under a wide range of conditions, achieves worst-case behavior close to the EG algorithm. We perform our analysis for square loss and absolute loss, and for regression and classification. We also describe some experiments showing that the performance of our algorithms are comparable to EG and the p-norm algorithm.

1. Introduction

We describe and analyze two online algorithms for learning linear functions. Our algorithms replace the update in the EG (Exponentiated Gradient) algorithm (Kivinen & Warmuth, 1997). One improvement is that exponentiation is replaced with simpler operations: addition and multiplication. A more important improvement is reduced parameter tuning while maintaining similar loss bounds. EG has a parameter that specifies the sum of the weights. In our algorithms, the sum of the weights are not restricted to a specific value. In particular for our linear multiplicative update, there is no restriction on the sum of the weights (our quadratic version has an upper bound). However, our algorithms still have initialization and learning rate parameters (as does EG), which still require tuning for our analytical results.

The issue of parameters is important because a search is needed to find the best parameter values for applying a learning algorithm to a task (Salzberg; Bengio). EG's restriction on the sum of the weights also restricts its hypothesis space, and so more search might be needed to optimize its parameters compared to an algorithm that covers a larger hypothesis space.

We analyze our algorithms both for square loss as well as absolute loss, and transform the absolute loss results into mistake bounds for binary classification. We also describe two experiments on synthetic data, one with an "easy" distribution and another with a "hard" distribution. In these experiments, the performance of our algorithms was comparable to EG and the p-norm algorithm (Grove et al., 2001; Gentile).

Considerable research has been performed on online algorithms for learning linear functions. We will note only the key papers that show the results that we build upon. The beginning of this topic within the machine learning community can perhaps be traced to (Littlestone), who developed the Winnow algorithm for learning linear functions.

Our results can be viewed as extensions of the EG algorithm (Kivinen & Warmuth, 1997) in the following ways. EG is based on relative entropy distance measures; we use unnormalized relative entropy as defined in that paper to analyze our results. Their approximated EG update is very close one of our updates; our linear multiplicative update substitutes $x_{t,i} - \widehat{y}$ in their paper with $x_{t,i}$. This update has also been studied by (Cesa-Bianchi et al., 2007), though only in the expert setting.

The style of our absolute loss bounds will be similar to (Bylander), which includes an additional constant loss per trial. However, for classification problems, this loss per trial will be eliminated using a technique similar to the clipping function in (Auer et al., 2002).

Appearing in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, 2011.

The p-norm algorithm (Grove et al., 2001; Gentile) is also relevant to our results. This algorithm does not have a sum of weights parameter and has loss bounds that are comparable to EG. These loss bounds also have the advantage over our results in that the learning rate and the initial weights do not need to be tuned to properties of the optimal weight vector. The advantage of our algorithms is that only basic arithmetic operations are required, while the p-norm algorithm requires two exponentiations for updating each weight.

2. Preliminaries

A *trial* is an ordered pair $(\mathbf{x}, y)$, consisting of a real vector $\mathbf{x} \in \Re^n$ (an *instance*, each value is an *input*) and a real number $y \in \Re$ (an *outcome*). A *prediction* $\widehat{y}$ on an instance $\mathbf{x}$ is made using a weight vector $\mathbf{w} \in \Re^n$ by computing the dot product $\widehat{y} = \mathbf{w} \cdot \mathbf{x} = \sum_{i=1}^n w_i x_i$. All of our results require that the weights are positive and do not include a bias weight; simple transformations to the instances can easily address these restrictions.

The square loss of a prediction $\widehat{y}$ on a trial $(\mathbf{x}, y)$ is $\text{Loss}_2(y, \widehat{y}) = (y - \widehat{y})^2$, and the absolute loss is $\text{Loss}_1(y, \widehat{y}) = |y - \widehat{y}|$. The square (absolute) loss of a weight vector on a *sequence* S of T trials, $S = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T))$, sums the square (absolute) loss over the trials. We use $\text{Loss}(\mathbf{w}, S)$ to denote the loss of weight vector on a trial sequence.

An *online algorithm* on a trial sequence begins with a weight vector $\mathbf{w}_1$, and then, for each trial t, makes a prediction $\mathbf{w}_t \cdot \mathbf{x}_t$ on the instance, receives the outcome y_t, and updates its weight vector to $\mathbf{w}_{t+1}$. The square (absolute) loss of the online algorithm on S sums the square (absolute) loss of the predictions. The parameters for the algorithms we are studying include the learning rate η and might include a constraint on the sum of weights W. That is, the sum of the updated weights might be normalized to equal W.

For an online algorithm A and a trial sequence S, we compare the loss of A to a *comparison class*, which we typically define as all nonnegative weight vectors $\mathbf{u}$ with a sum of weights equal to (or less than or equal to) a constant U. Our bounds are of the form $\text{Loss}(A, S) \leq (1 + c)\text{Loss}(\mathbf{u}, S) + a$ for all weight vectors $\mathbf{u}$ from the comparison class, where $a > 0$ and $c \geq 0$ are expressions based on A, S, U, and the type of loss. The relationship between U and W depends on the algorithm; in particular, our algorithms do not require $U = W$.

The bounds are based on demonstrating single-trial loss bounds, showing for each trial S_t, that $\text{Loss}(\mathbf{w}_t, S_t) \leq (1 + c)\text{Loss}(\mathbf{u}, S_t) + a_t$, and summing up the additional loss a_t over all the trials. When $\mathbf{w}_t$ has a larger loss than $\mathbf{u}$, the updated $\mathbf{w}_{t+1}$ should be closer to the optimal $\mathbf{u}$. Our analysis is based on showing how $d(\mathbf{u}, \mathbf{w}_t) - d(\mathbf{u}, \mathbf{w}_{t+1})$ is related to the a_t term, where d is the unnormalized relative entropy distance measure:

$$d(\mathbf{u}, \mathbf{w}) = \sum_{i=1}^n \left(w_i - u_i + u_i \ln \frac{u_i}{w_i} \right)$$

For the case that $u_i = 0$, we define $0 \ln 0 = 0$. The case that $w_i = 0$ cannot happen with the algorithms.

Algorithm 1 QMU

Input: Learning rate $\eta > 0$,
initial weight vector $\mathbf{w}_1 \in \Re_+^n$,
maximum sum of weights W

for $t = 1, \ldots, T$ **do**
Get instance $\mathbf{x}_t \in \Re^n$
Predict $\widehat{y}_t = \mathbf{w}_t \cdot \mathbf{x}_t$
Get label $y_t \in \Re$.
Incur loss $\begin{cases} \text{Loss}_2(y_t, \widehat{y}_t) & \text{QMU}_2 \\ \text{Loss}_1(y_t, \widehat{y}_t) & \text{QMU}_1 \end{cases}$
Update weights:

$$\delta_t = \begin{cases} y_t - \widehat{y}_t & \text{QMU}_2 \\ \text{sign}(y_t - \widehat{y}_t) & \text{QMU}_1 \end{cases}$$

$$z_{t,i} = \eta \delta_t x_{t,i}$$

$$w'_{t+1,i} = w_{t,i} \left(1 + z_{t,i} + \frac{z_{t,i}^2}{3} \right)$$

$$w_{t+1,i} = \begin{cases} \dfrac{W w'_{t+1,i}}{\sum_{i=1}^n w'_{t+1,i}} & \text{if } W < \sum_{i=1}^n w'_{t+1,i} \\ w'_{t+1,i} & \text{otherwise} \end{cases}$$

end for

3. Quadratic Multiplicative Update

The Quadratic Multiplicative Update (QMU) algorithm (Algorithm 1) inputs a learning rate η, an initial weight vector $\mathbf{w}_1$, and a parameter W bounding the maximum sum of weights. The W parameter is not strictly needed. QMU could be run without any normalization; then the W in the analysis would correspond to the maximum sum of weights of the algorithm's weight vectors.

QMU differs from the EG algorithm (Kivinen & Warmuth, 1997) in two respects. One is that EG computes $w'_{t+1,i}$ by $w'_{t+1,i} = w_{t,i} \exp\{z_{t,i}\}$. The other is that EG always normalizes the weights, while QMU performs normalization only if the sum of weights exceeds W.

Let QMU_2 and QMU_1 respectively denote the square loss and absolute loss versions of the QMU algorithm. We have demonstrated the following results.

Theorem 1 *Let S be a sequence of T trials, where all instances $\mathbf{x}_t \in [-X, X]^n$, i.e., X bounds the absolute value of any input. Let $\mathbf{w}_1 = (W_1/n, \ldots, W_1/n)$, $0 < W_1 \leq U \leq W$, $V = (U + 2W)/3$, and $n \geq 3$. If $\eta = c/((1+c)VX^2)$ with $c > 0$, then, for any nonnegative weight vector $\mathbf{u}$ whose sum of weights is less than or equal to U, the following bound holds:*

$$\text{Loss}_2(\text{QMU}_2(\eta, \mathbf{w}_1, W), S) \leq (1+c)\text{Loss}_2(\mathbf{u}, S) + \frac{2(1+c)UVX^2}{c} \ln \frac{nU}{W_1}$$

For any $\eta > 0$ and for any nonnegative weight vector $\mathbf{u}$ whose sum of weights is less than or equal to U, the following bound holds:

$$\text{Loss}_1(\text{QMU}_1(\eta, \mathbf{w}_1, W), S) \leq \text{Loss}_1(\mathbf{u}, S) + \frac{U}{\eta} \ln \frac{nU}{W_1} + \frac{\eta TVX^2}{2}$$

If we think of U as fixed in advance, we can choose $U = W_1 = W = V$, which makes these loss bounds identical to EG, both for square loss (Kivinen & Warmuth, 1997) and absolute loss (Bylander). If U is treated more realistically as an unknown (for example, think of U as the sum of the unknown optimal weights), there are a number of considerations.

If the conditions $W_1 \leq U$ and $n \geq 3$ do not hold, then $U \ln(nU/W_1)$ should be replaced with $\max(W, W + U \ln(nU/(eW_1)))$ (this comes from Lemma 4).

W and η are parameters of QMU, and X is a constant, so the value of c in the bound for QMU_2 can be treated as a function of U. Suppose, for example, that U is between 0 and W, and $\eta = 1/(2WX^2)$. This gives a value of c between 1 (at $U = W$) and 1/2 (at $U = 0$), much better for smaller values of U. The distance measure will primarily vary in the $U \ln(nU)$ term. Thus, we can expect faster and better convergence for smaller values of U.

Finally, we should explain the magic number 1/3 in the update (used in the computation of $w'_{t+1,i}$). The 1/3 factor in the update can be traced to Lemma 7 and can be replaced by any number greater than 1/4 and less than 1/3. The choice of 1/3 represents a tradeoff between dependence on W versus U, indicated by $V = (U+2W)/3$ in the theorem. Specifically, replacing 1/3 with a would lead to $V = 2a^2U/(8a-2) + 2aW$.

The update has some unusual characteristics when $\eta\delta_t x_{t,i}$ is negative. Using z in place of $\eta\delta_t x_{t,i}$, we note that $1 + z + z^2/3$ has a minimum at $z = -1.5$ and becomes greater than 1 when $z < -3$. In this case, the weight can increase even though that would increase error on the instance. This indicates that the analysis has a lot of "slack" for larger magnitudes.

Algorithm 2 LMU

Input: Learning rate $\eta > 0$,
initial weight vector $\mathbf{w}_1 \in \Re^n_+$
for $t = 1, \ldots, T$ **do**
Get instance $\mathbf{x}_t \in \Re^n$
Predict $\widehat{y}_t = \mathbf{w}_t \cdot \mathbf{x}_t$
Get label $y_t \in \Re$.
Incur loss $\begin{cases} \text{Loss}_2(y_t, \widehat{y}_t) & \text{LMU}_2 \\ \text{Loss}_1(y_t, \widehat{y}_t) & \text{LMU}_1 \end{cases}$
Update weights:
$\delta_t = \begin{cases} y_t - \widehat{y}_t & \text{LMU}_2 \\ \text{sign}(y_t - \widehat{y}_t) & \text{LMU}_1 \end{cases}$
$z_{t,i} = \eta\delta_t x_{t,i}$
$w_{t+1,i} = w_{t,i}(1 + z_{t,i})$
end for

4. Linear Multiplicative Update

The Linear Multiplicative Update (LMU) algorithm (Algorithm 2) inputs a learning rate η and an initial weight vector $\mathbf{w}_1$. LMU replaces the exponential function in the EG update step with a linear function and does not require any normalization, so the W parameter is eliminated. However, the algorithm requires $z_{t,i} > -1$, which is potentially a problem with the square loss version of LMU.

Let LMU_2 and LMU_1 respectively denote the square loss and absolute loss versions of the LMU algorithm. We have demonstrated the following results.

Theorem 2 *Let S be a sequence of T trials, where all instances $\mathbf{x}_t \in [-X, X]^n$, i.e., X bounds the absolute value of any input. Let $\mathbf{w}_1 = (W_1/n, \ldots, W_1/n)$, $0 < W_1 \leq U \leq W$, and $n \geq 3$.*

If $z_{t,i} \geq -1/7$ for $1 \leq t \leq T$ and $1 \leq i \leq n$, and if $\eta = 9c/(10(1+c)UX^2)$ with $c > 0$, then, for any nonnegative weight vector $\mathbf{u}$ whose sum of weights is less than or equal to U, the following bounds hold:

$$\text{Loss}_2(\text{LMU}_2(\eta, W_1), S) \leq (1+c)\text{Loss}_2(\mathbf{u}, S) + \frac{20(1+c)U^2X^2}{9c} \ln \frac{nU}{W_1}$$

If $z_{t,i} \geq -1/7$ for $1 \leq t \leq T$ and $1 \leq i \leq n$, then, for any $\eta > 0$ and for any nonnegative weight vector $\mathbf{u}$ whose sum of weights is less than or equal to U, the following bound holds:

$$\text{Loss}_1(\text{LMU}_1(\eta, W_1), S) \leq \text{Loss}_1(\mathbf{u}, S) + \frac{U}{\eta} \ln \frac{nU}{W_1} + \frac{5\eta TUX^2}{9}$$

The extra losses for the LMU algorithm are similar to the QMU algorithm (and so also to EG), with slightly higher constants (20/9 and 5/9 instead of 2 and 1/2, respectively). The condition $z_{t,i} \geq -1/7$, which comes from Lemma 8, is tightly related to these constants. For example, if a stronger condition $z_{t,i} \geq -1/15$ were true, that would lead to the constants 44/21 and 11/21, respectively. The more general condition $z_{t,i} > -1$ does not lead to useful constants.

Because these bounds require $z_{t,i} \geq -1/7$, this places an implicit constraint on U. If LMU_2's weight vector becomes large, it is more likely that $\widehat{y}$ become large, which could lead to $\eta(y-\widehat{y})x_{t,i} < -1/7$. With LMU_1, this is not a serious issue because ensuring $\eta\,\text{sign}(y-\widehat{y})x_{t,i} \geq -1/7$ can be achieved by satisfying $\eta \leq X/7$.

The η parameter is also implicitly constrained by U. For example, suppose we set η with some value for W in mind, though W is not a parameter of LMU. Let $\eta = 1/(2WX^2)$ for LMU_2, and consider different values for U. From $U = 0$ to $U = W$, c ranges from 0 to 5/4. As U approaches $9W/5$, c approaches infinity. Thus, although W is not a parameter of LMU, the choice for η needs to take W into account.

5. Classification

To apply our analysis to binary classification problems, we make the following revisions. First, we allow the outcome to be a real interval rather than a real number. The loss functions are adjusted so that if the prediction is in the interval, the loss is zero. If the prediction is less/greater than the interval, the loss is calculated using the minimum/maximum of the interval. The previous theorems hold for real intervals and the modified loss calculations.

Second, for binary classification trials, we use the intervals $[1,\infty)$ for $(-\infty,-1]$ (labels $y = 1$ and $y = -1$) for positive and negative instances, respectively. With these intervals, the absolute loss is equivalent to the hinge loss $\max(0, 1-y\widehat{y})$, which has a convenient correspondence to 0-1 loss. With deterministic prediction (positive if $\widehat{y} > 0$, else negative), the hinge loss is an upper bound on the number of mistakes, i.e., the hinge loss must be at least 1 if the predicted class differs from the outcome. With probabilistic prediction (if $\widehat{y}$ is between -1 and 1, positive is predicted with probability $(\widehat{y}-1)/2$), the hinge loss divided by two is an upper bound on the expected number of mistakes.

The following theorem describes the performance of LMU_1 on classification problems. Similar theorems could be provided for EG and QMU_1, but LMU_1 has the advantage of not requiring any U or W parameters.

Theorem 3 *Let S be a sequence of T trials, where all instances $\mathbf{x}_t \in [-X, X]^n$, i.e., X bounds the absolute value of any input, and where all outcome intervals are either $[1,\infty)$ or $(-\infty,-1]$. Let $W_1 \leq U$ and $n \geq 3$. If $z_{t,i} \geq -1/7$ for $1 \leq t \leq T$ and $1 \leq i \leq n$, and if $\eta = 9c/(5(1+c)UX^2)$, then, for any nonnegative weight vector $\mathbf{u}$ whose sum of weights is less than or equal to U, the following bounds hold:*

$$\text{Loss}_1(\text{LMU}_1(\eta,1),S) \leq$$
$$(1+c)\text{Loss}_1(\mathbf{u},S) + \frac{5(1+c)^2U^2X^2}{9c}\ln\frac{n(1+c)U}{W_1}$$

The restriction on the outcome intervals allows to us to remove the extra loss per trial, but at the expense of a larger constant times the loss for $\mathbf{u}$ and the distance measure. The extra loss is $O(L + U^2X^2\ln n)$ provided η is a fraction of $9/(5UX^2)$ and $1/\eta$ is $\Theta(UX^2)$.

While the LMU algorithm does not require U or W as a parameter, the η parameter is implicitly constrained by U. For example, if $\eta = 1/100$ and $X = 1$, then the theorem does not provide any useful bounds when $U \geq 900/5$, though, on the positive side, the theorem applies to a wide range of values for U.

6. Experiments

We tested QMU and LMU along with the EG and p-norm algorithm on two types of artificially generated sequences of classification trials. Each sequence consisted of 100,000 trials, where each instance consisted of 1000 inputs (called attributes from this point on), where each attribute was either $+1$ or -1. No bias weights were employed in our experiments.

The target hypotheses were of the type $\lceil k/2 \rceil$ out of k relevant attributes, where k was an odd integer varying from 1 to 19 over the trial sequences. A relevant attribute might have a negative or positive weight. For each trial, $\lceil k/2 \rceil$ relevant attributes were selected to match the noise-free outcome; this ensures that the (near) optimal weight is either $+1$ or -1 for a relevant attribute and 0 for an irrelevant attribute. We also varied the noise from 0 to 0.4 (40% noise) in steps of 0.1 (10%) over the trial sequences; note that the hinge loss of the target hypothesis will be either 0 for a noise-free trial or 2 for a noisy trial.

For each number of relevant attributes and noise level, 10 trial sequences were generated, resulting in a total of 500 trial sequences (for each experiment below).

For all algorithms, we used a learning rate of 0.01. We used 1000 weights for the p-norm algorithm, and 2000 weights for the other algorithms because they need an additional 1000 weights to encode negative weights.

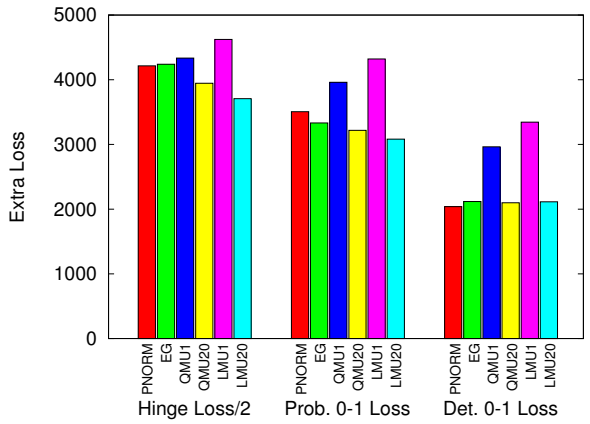

Figure 1. Average Extra Loss for Random Irrelevant Attributes

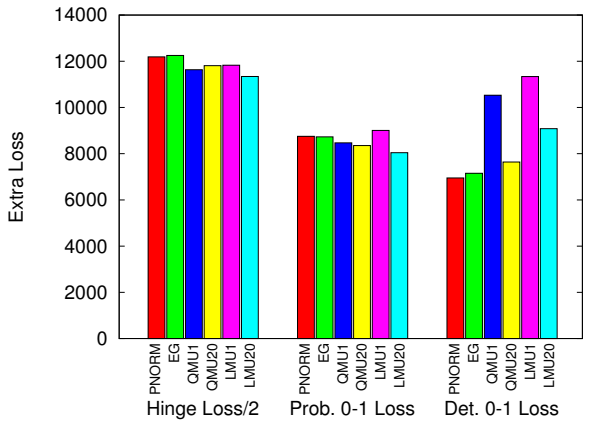

Figure 2. Average Extra Loss on More Difficult Trial Sequences

For EG, the sum of weights was set to 20. For QMU, the maximum sum of weights was set to 20. LMU and p-norm do not limit the sum of weights. The initial weights were set to 0 for the p-norm algorithm and 20/2000 for EG. For QMU and LMU, we used two different initial weights: 1/2000 (the initial sum is 1) and 20/2000 (an initial sum of 20); the combinations are referred to by QMU1, QMU20, LMU1, and LMU20.

6.1. Random Irrelevant Attributes

Figure 1 displays the overall results when the values of the irrelevant attributes are generated randomly. Each bar shows, over the trial sequences, the average extra loss of an algorithm compared to the target hypothesis. The three groups are the hinge loss divided by 2, the probabilistic 0-1 loss, and the deterministic 0-1 loss.

All differences between algorithms in Figure 1 are statistically significant ($p < 0.05$) except for p-norm vs. EG and EG vs. QMU1 for hinge loss, and EG vs. LMU20 and QMU vs. LMU20 for deterministic 0-1 loss. Note that a difference of 1000 corresponds to 1% error for the 0-1 losses.

LMU1 and QMU1 had higher losses than p-norm and EG with LMU1 worse than QMU1. However, LMU20 and QMU20 had lower hinge and probabilistic 0-1 losses than p-norm and EG, while the deterministic 0-1 losses were close. This indicates that experimental fine-tuning of LMU and QMU is necessary, but can have significant benefits. Note that EG with a sum of weights equal to 1 would be unable to converge to most of the target hypotheses. p-norm's performance is good, but required over 10 times as much processing time as LMU and QMU.

6.2. Less Random Irrelevant Attributes

We generate trial sequences just as before, except that, for each trial, one selected irrelevant attribute is set to agree with the outcome (after any noise is applied). Each irrelevant attribute is selected in turn for 100 trials. That is, a selected attribute will agree 100% with the outcome for 100 trials, then it will return to being randomly generated while the next irrelevant attribute is selected for 100 trials.

The overall results are shown in Figure 2, which displays the average extra loss of each algorithm, the difference between the algorithm's loss and the target hypothesis's loss. All differences between algorithms are statistically significant ($p < 0.05$) except for QMU20 vs. LMU1 for hinge loss and for p-norm vs. EG for probabilistic 0-1 loss. Note that the extra loss for all algorithms were two to three times higher for this type of trial sequence even though we are only changing, on average, one irrelevant attribute per two trials.

LMU and QMU had slightly lower hinge loss than p-norm and EG, but LMU1 had higher probabilistic 0-1 loss, and all LMU and QMU variations had higher deterministic 0-1 loss, with LMU1 and QMU1 performing much worse. We have no explanation for this behavior yet; absolute loss results do not bound deterministic 0-1 loss very well.

7. Conclusion

We have presented two new linear learning algorithms, QMU and LMU. With both theoretical and empirical results, we have shown that the new algorithms have comparable performance and some advantages compared to EG and p-norm. EG requires a parameter

that specifies the sum of the weights, while QMU only requires an upper bound on the weights, and LMU eliminates this parameter.

While the EG and p-norm algorithms performed significantly better empirically in some cases with respect to 0-1 loss, QMU and LMU are significantly faster. In our experiments, EG used over twice as much time compared to either QMU or LMU, while p-norm used over 10 times as much processing time. In future work, we would like to more clearly describe the relationship between these learning algorithms, both theoretically and empirically.

A. Appendix: Proofs

We use unnormalized relative entropy (Kivinen & Warmuth, 1997) to measure the distance between two nonnegative weight vectors:

$$d(\mathbf{u}, \mathbf{w}) = \sum_{i=1}^{n} \left(w_i - u_i + u_i \ln \frac{u_i}{w_i} \right)$$

By definition, $0 \ln 0 = 0$.

A.1. Lemmas

Lemma 4 is used to bound the distance from the initial weight vector to any weight vector whose sum of weights is less than or equal to U.

Lemma 4 *Let* $\mathbf{u}$ *and* $\mathbf{w}$ *be nonnegative weight vectors of length* n. *Let* $w_i = W_1/n$ *for* $1 \le i \le n$. *If* $U \ge \sum_{i=1}^{n} u_i$ *and* $W_1 > 0$, *then:*

$$d(\mathbf{u}, \mathbf{w}) \le \max(W_1, W_1 + U \ln \frac{nU}{eW_1})$$

If, in addition, $U \ge W_1$ *and* $n \ge 3$, *then*

$$d(\mathbf{u}, \mathbf{w}) \le U \ln \frac{nU}{W_1}$$

Proof: Let $W_1 > 0$, $w_i = W_1/n$ for $1 \le i \le n$ and $U' = \sum_{i=1}^{n} u_i \le U$. It follows that:

$$\begin{aligned} d(\mathbf{u}, \mathbf{w}) &= \sum_{i=1}^{n} \left(w_i - u_i + u_i \ln \frac{u_i}{w_i} \right) \\ &= W_1 - U' + \sum_{i=1}^{n} u_i \ln \frac{nu_i}{W_1} \\ &\le W_1 - U' + \sum_{i=1}^{n} u_i \ln \frac{nU'}{W_1} \\ &= W_1 - U' + U' \ln \frac{nU'}{W_1} \end{aligned}$$

This expression has a positive second derivative with respect to U'. Because $U' \in [0, U]$, the maximum of the expression must be either at $U' = 0$, which results in W_1, or at $U' = U$, which results in $W_1 + U \ln(nU/eW_1)$. This proves the first inequality.

If also $U \ge W_1$ and $n \ge 3$, then because $U/W_1 \ge 1$ and $\ln n > 1$, it follows that $U \ln(nU/W_1) \ge U \ln n \ge U \ge W_1$. Also, $U \ln(nU/W_1) \ge W_1 + U \ln(nU/eW_1) = W_1 - U + U \ln(nU/W_1)$ follows from $W_1 \le U$. This proves the second inequality. ∎

If $\mathbf{w}$ has a sum of weights that is greater than $\mathbf{u}$, then Lemma 5 shows that decreasing $\mathbf{w}$'s weights decreases the distance to $\mathbf{u}$.

Lemma 5 *Let* $\mathbf{u}$ *and* $\mathbf{w}$ *be nonnegative weight vectors of length* n. *Let* $w_i > 0$ *for* $1 \le i \le n$. *Let* $U = \sum_{i=1}^{n} u_i$ *and* $W = \sum_{i=1}^{n} w_i$. *If* $W > U$ *and* a *is a positive value such that* $U \le aW < W$, *then* $d(\mathbf{u}, \mathbf{w}) > d(\mathbf{u}, a\mathbf{w})$.

Proof: Under the conditions of the lemma, consider:

$$d(\mathbf{u}, a\mathbf{w}) = \sum_{i=1}^{n} \left(aw_i - u_i + u_i \ln \frac{u_i}{aw_i} \right)$$

The derivative of this expression with respect to a is $\sum_{i=1}^{n} (w_i - u_i/a) = W - U/a$, which is positive if $a > U/W$. Hence, this expression decreases as a decreases from 1 to U/W, proving the lemma. ∎

Lemma 6 bounds the change in the distance measure when a weight vector is updated.

Lemma 6 *Let* $\mathbf{u}$ *and* $\mathbf{w}$ *be nonnegative weight vectors of length* n. *Let* $w_i > 0$ *for* $1 \le i \le n$. *Let* $\mathbf{x}$ *be a weight vector of length* n. *Let* $\eta > 0$. *Let* $f(x)$ *and* $g(x)$ *be functions such that* $1 + f(x) \ge e^{g(x)}$. *Let* $\mathbf{w}'$ *satisfy* $w_i e^{g(x)} \le w'_i \le w_i(1 + f(x_i))$. *Then the following inequality holds:*

$$d(\mathbf{u}, \mathbf{w}) - d(\mathbf{u}, \mathbf{w}') \ge \sum_{i=1}^{n} u_i g(x_i) - \sum_{i=1}^{n} w_i f(x_i)$$

Proof: Using the assumptions in the lemma, then

$$\begin{aligned} &d(\mathbf{u}, \mathbf{w}) - d(\mathbf{u}, \mathbf{w}') \\ &= \sum_{i=1}^{n} \left(w_i - u_i + u_i \ln \frac{u_i}{w_i} \right) - \sum_{i=1}^{n} \left(w'_i - u_i + u_i \ln \frac{u_i}{w'_i} \right) \\ &= \sum_{i=1}^{n} (w_i - w'_i) + \sum_{i=1}^{n} u_i \ln \frac{w'_i}{w_i} \end{aligned}$$

$$\geq \sum_{i=1}^{n}(w_i - w_i(1+f(x_i)) + \sum_{i=1}^{n} u_i \ln \frac{w_i e^{g(x)}}{w_i}$$
$$= -\sum_{i=1}^{n} w_i f(x_i) + \sum_{i=1}^{n} u_i g(x_i) \quad \blacksquare$$

Lemmas 7 and 8 provide lower bounds for quadratic and linear functions in terms of an exponential function. These lemmas ensure that the inequality $1 + f(x) \geq e^{g(x)}$ in Lemma 6 is satisfied.

Lemma 7 *For all $a > 1/4$ and $z \in \Re$:*

$$1 + z + az^2 \geq \exp\left\{z - \frac{a^2 z^2}{8a-2}\right\}$$

In particular:

$$1 + z + \frac{z^2}{3} \geq \exp\left\{z - \frac{z^2}{6}\right\}$$

Proof: The inequality is equivalent to:

$$f(a,z) = \ln(1 + z + az^2) - \left(z - \frac{a^2 z^2}{8a-2}\right) \geq 0$$

Assuming $a > 1/4$, $f(a,z)$ and its first derivative with respect to z are zero when $z = 0$. The second derivative with respect to z is nonnegative, which proves $f(a,z) \geq 0$ and the first inequality of the lemma. The second inequality substitutes $1/3$ for a. $\blacksquare$

Lemma 8 *For all $a > 1/2$, if $z \geq -\frac{6a-3}{6a-1}$, then:*

$$1 + z \geq \exp\{z - az^2\}$$

In particular, for $a = 5/9$, if $z \geq -1/7$, then:

$$1 + z \geq \exp\{z - \frac{5z^2}{9}\}$$

Proof Sketch: The inequality is equivalent to:

$$f(a,z) = \ln(1+z) - (z - az^2) \geq 0$$

$f(a,z)$ and its first derivative with respect to z are zero when $z = 0$. The second derivative with respect to z is positive for $z \geq 0$ and $a > 1/2$, so $f(a,z) \geq 0$ when $z \geq 0$.

Assuming $a > 1/2$, we can show that $f(a,z) \geq 0$ in the interval $z \in [-(6a-3)/(6a-1), 0]$ by demonstrating both of the following:

$$f_1(a,z) = \ln(1+z) - (z - z^2/2 + z^3(6a-1)/6) \geq 0$$

$$f_2(a,z) = (z - z^2/2 + z^3(6a-1)/6) - (z - az^2) \geq 0$$

and noting that $f(a,z) = f_1(a,z) + f_2(a,z)$. $\blacksquare$

A.2. Proof of Theorem 1: QMU Bounds

The QMU_2 algorithm has the single trial loss bound:

$$\text{Loss}_2(\mathbf{w}_t, S_t) \leq (1+c)\text{Loss}_2(\mathbf{u}, S_t) + \frac{2(1+c)VX^2}{c}(d(\mathbf{u},\mathbf{w}_t) - d(\mathbf{u},\mathbf{w}_{t+1}))$$

using $\eta = c/((1+c)VX^2)$. This is based as follows.

Let $z_t = \eta(y_t - \mathbf{w}_t \cdot \mathbf{x}_t)$, and let $z_{t,i} = z_t x_{t,i}$. This implies $w'_{t+1,i} = w_{t,i}(1 + z_{t,i} + z_{t,i}^2/3)$. Then:

$$d(\mathbf{u},\mathbf{w}_t) - d(\mathbf{u},\mathbf{w}_{t+1}) \geq d(\mathbf{u},\mathbf{w}_t) - d(\mathbf{u},\mathbf{w}'_{t+1})$$
$$\geq \sum_{i=1}^{n} u_i \left(z_{t,i} - \frac{z_{t,i}^2}{6}\right) - \sum_{i=1}^{n} w_{t,i}\left(z_{t,i} + \frac{z_{t,i}^2}{3}\right)$$
$$\geq \sum_{i=1}^{n} u_i \left(z_{t,i}x_{t,i} - \frac{z_t^2 X^2}{6}\right) - \sum_{i=1}^{n} w_{t,i}\left(z_t x_{t,i} + \frac{z_t^2 X^2}{3}\right)$$
$$\geq z_t(\mathbf{u}\cdot\mathbf{x} - \mathbf{w}\cdot\mathbf{x}) - \frac{z_t^2 V X^2}{2}$$

The first inequality follows from Lemma 5. The second inequality follows from Lemmas 6 and 7. The third inequality follows from $|x_{t,i}| \leq X$. The last inequality follows from $\sum_{i=1}^{n} u_i \leq U$, $\sum_{i=1}^{n} w_i \leq W$, and $V = (2W + U)/3$.

Now to finish showing the single trial loss bound, we need to find a and c such that

$$a\,\text{Loss}_2(\mathbf{w}_t, S_t) - a(c+1)\text{Loss}_2(\mathbf{u}, S_t)$$
$$= a(y_t - \mathbf{w}_t\cdot\mathbf{x}_t)^2 - a(c+1)(y_t - \mathbf{u}\cdot\mathbf{x}_t)^2$$
$$\leq z_t(\mathbf{u}\cdot\mathbf{x} - \mathbf{w}\cdot\mathbf{x}) - \frac{z_t^2 V X^2}{2}$$

Using standard techniques (Kivinen & Warmuth, 1997), it can be shown that $a = \eta/2$ and $\eta = c/((1+c)VX^2)$ satisfy the inequality.

Adding up all the single trial losses and applying Lemma 4 implies the QMU_2 bound of the theorem.

For QMU_1, if we use instead $z_t = \eta\,\text{sign}(y_t - \mathbf{w}_t\cdot\mathbf{x}_t)$, we can borrow part of the previous derivation to show:

$$d(\mathbf{u},\mathbf{w}_t) - d(\mathbf{u},\mathbf{w}_{t+1}) \geq z_t(\mathbf{u}\cdot\mathbf{x}_t - \mathbf{w}_t\cdot\mathbf{x}_t) - z_t^2 X^2 V/2$$

Now to show a single trial loss bound, we find a and c such that

$$a\,\text{Loss}_1(\mathbf{w}_t, S_t) - a(c+1)\text{Loss}_1(\mathbf{u}, S_t)$$
$$= a|y_t - \mathbf{w}_t\cdot\mathbf{x}_t| - a(c+1)|y_t - \mathbf{u}\cdot\mathbf{x}_t|$$
$$\leq z_t(\mathbf{u}\cdot\mathbf{x}_t - \mathbf{w}_t\cdot\mathbf{x}_t) - z_t^2 X^2 V/2 + \frac{\eta^2 X^2 V}{2}$$

where the last fraction is the extra loss per trial included in the bound. Using standard techniques, it can be shown that $a = \eta$ and $c = 0$ satisfy the inequality for any positive value for η. The QMU_1 bound of the theorem follows from Lemma 4 and adding up all the single trial losses. ∎

A.3. Proof Sketch Theorem 2: LMU Bounds

The LMU_2 algorithm has the single trial loss bound:

$$\text{Loss}_2(\mathbf{w}_t, S_t) \leq (1+c)\text{Loss}_2(\mathbf{u}, S_t) + \frac{20(1+c)UX^2}{9c}(d(\mathbf{u}, \mathbf{w}_t) - d(\mathbf{u}, \mathbf{w}_{t+1})$$

using $\eta = 9c/(10(1+c)UX^2)$. This is based on Lemmas 6 and 8. Adding up all the single trial losses and applying Lemma 4 implies the LMU_2 bound of the theorem.

The LMU_1 algorithm has the single trial loss bound:

$$\text{Loss}_1(\mathbf{w}_t, S_t) \leq \text{Loss}_1(\mathbf{u}, S_t) + \frac{d(\mathbf{u}, \mathbf{w}_t) - d(\mathbf{u}, \mathbf{w}_{t+1})}{\eta} + \frac{5\eta VX^2}{9}$$

Again, this is based on Lemmas 6 and 8. Adding up all the single trial losses and applying Lemma 4 implies the LMU_1 bound of the theorem. ∎

A.4. Proof of LMU_1 Classification Bound

For convenience, we use the labels $y = 1$ and $y = -1$, respectively, for positive and negative instances.

For the LMU_1 algorithm for classification, we obtain the single trial loss bound:

$$\text{Loss}_1(\mathbf{w}_t, S_t) \leq (1+c)\text{Loss}_1(\mathbf{u}, S_t) + \frac{5(1+c)UX^2}{9c}(d((1+c)\mathbf{u}, \mathbf{w}_t) - d((1+c)\mathbf{u}, \mathbf{w}_{t+1})$$

This is based on the the following.

The interesting case is when $\widehat{y} < 1$ for $y_t = 1$ or when $\widehat{y} > -1$ for $y_t = -1$. In this case, $\mathbf{w}_t$'s loss is equal to $1 - y_t\widehat{y}$, and $\mathbf{u}$'s loss is equal to $\max(0, 1 - y_t\mathbf{u} \cdot \mathbf{x}_t)$.

Let $z_t = \eta\,\text{sign}(y_t - \mathbf{w}_t \cdot \mathbf{x}_t) = \eta y_t$ and $z_{t,i} = z_t x_{t,i}$. This implies $w_{t+1,i} = w_{t,i}(1 + z_{t,i})$. Instead of $\mathbf{u}$, we compare $(1+c)\mathbf{u}$ to the weight vectors. This is equivalent to the clipping function in (Auer et al., 2002).

$$\begin{aligned} &d((1+c)\mathbf{u}, \mathbf{w}_t) - d((1+c)\mathbf{u}, \mathbf{w}_{t+1}) \\ &\geq \sum_{i=1}^{n}(1+c)u_i\left(z_{t,i} - \frac{5z_{t,i}^2}{9}\right) - \sum_{i=1}^{n} w_{t,i}z_{t,i} \\ &\geq z_t((1+c)\mathbf{u} \cdot \mathbf{x} - \mathbf{w} \cdot \mathbf{x}) - \frac{5(1+c)z_t^2UX^2}{9} \end{aligned}$$

The first inequality follows from Lemmas 6 and 8. The second inequality follows from $|x_{t,i}| \leq X$ and $\sum_{i=1}^{n} u_i \leq U$.

Now to show the single trial loss bound for LMU_1, we find a and c such that

$$\begin{aligned} &a\text{Loss}_1(\mathbf{w}_t, S_t) - a(c+1)\text{Loss}_1(\mathbf{u}, S_t) \\ &= a(1 - y_t\mathbf{w}_t \cdot \mathbf{x}_t) - a(c+1)\max(0, 1 - y_t\mathbf{u} \cdot \mathbf{x}_t) \\ &\leq a(1 - y_t\mathbf{w}_t \cdot \mathbf{x}_t) - a(c+1)(1 - y_t\mathbf{u} \cdot \mathbf{x}_t) \\ &= ay_t((1+c)\mathbf{u} \cdot \mathbf{x}_t - \mathbf{w}_t \cdot \mathbf{x}_t) - ac \\ &\leq \eta y_t((1+c)\mathbf{u} \cdot \mathbf{x} - \mathbf{w} \cdot \mathbf{x}) - \frac{5(1+c)\eta^2 y_t^2 UX^2}{9} \end{aligned}$$

Setting $a = \eta$ and $c = 5(1+c)\eta UX^2/9$ implies $\eta = 9c/(5(1+c)UX^2)$.

The bound of the theorem follows from Lemma 4 using $(1+c)\mathbf{u}$ instead of $\mathbf{u}$ and adding up all the single trial losses. ∎

References

Auer, P., Cesa-Bianchi, N., and Gentile, C. Adaptive and self-confident on-line algorithms. *Journal of Computer and System Sciences*, 64:48–75, 2002.

Bengio, Y. Gradient-based optimization of hyperparameters. *Neural Computation*, 12:1889–1900, 2000.

Bylander, T. Worst-case analysis of the perceptron and exponentiated update algorithms. *Artificial Intelligence*, 106:335–352, 1998.

Cesa-Bianchi, N., Mansour, Y., and Stoltz, G. Improved second order bounds for prediction with expert advice. *Machine Learning*, 66:321–352, 2007.

Gentile, C. The robustness of the p-norm algorithms. *Machine Learning*, 53:265–299, 2003.

Grove, A. J., Littlestone, N., and Schuurmans, D. General convergence results for linear discriminant updates. *Machine Learning*, 43:173–210, 2001.

Kivinen, J. and Warmuth, M. K. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132:1–63, 1997.

Littlestone, N. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

Salzberg, S. L. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery*, 1:317–328, 1997.