Large Margin Classification for Moving Targets

Jyrki Kivinen, Alex J. Smola, and Robert C. Williamson

Research School of Information Sciences and Engineering
Australian National University
Canberra, ACT 0200, Australia
{Jyrki.Kivinen, Alex.Smola, Bob.Williamson}@anu.edu.au

Abstract. We consider using online large margin classification algorithms in a setting where the target classifier may change over time. The algorithms we consider are Gentile's ALMA, and an algorithm we call NORMA which performs a modified online gradient descent with respect to a regularised risk. The update rule of ALMA includes a projection-based regularisation step, whereas NORMA has a weight decay type of regularisation. For ALMA we can prove mistake bounds in terms of the total distance the target moves during the trial sequence. For NORMA, we need the additional assumption that the movement rate stays sufficiently low uniformly over time. In addition to the movement of the target, the mistake bounds for both algorithms depend on the hinge loss of the target. Both algorithms use a margin parameter which can be tuned to make them mistake-driven (update only when classification error occurs) or more aggressive (update when the confidence of the classification is below the margin). We get similar mistake bounds both for the mistake-driven and a suitable aggressive tuning. Experiments on artificial data confirm that an aggressive tuning is often useful even if the goal is just to minimise the number of mistakes.

1 Introduction

Consider the basic linear classification problem. We are given a set of data points $(\boldsymbol{x}_t, y_t)$ where $\boldsymbol{x}_t \in \mathbb{R}^n$ and $y_t \in \{-1, +1\}$ for $t = 1, \ldots, m$. The task is to find a coefficient vector $\boldsymbol{w} \in \mathbb{R}^n$ such that $\boldsymbol{w} \cdot \boldsymbol{x}_t > 0$ if $y_t = +1$, and $\boldsymbol{w} \cdot \boldsymbol{x}_t < 0$ if $y_t = -1$. In other words, we wish the *margin* $y_t \boldsymbol{w} \cdot \boldsymbol{x}_t$ to be positive for every example $(\boldsymbol{x}_t, y_t)$. Recently a lot of work has been done on large margin classification, where we do not just settle for any linear separator $\boldsymbol{w}$, but try to find one that achieves the largest possible separation [3]. Maximising the separation can be thought of as maximising the smallest margin over the example set, while keeping the norm of $\boldsymbol{w}$ bounded. This often leads to significant improvements in the generalisation ability of the resulting linear classifier [3].

The discussion above presumes a batch learning scenario: we obtain a set of examples $(\boldsymbol{x}_t, y_t)$ from a source, use that data to induce a classifier $\boldsymbol{w}$, and then use the classifier to predict the labels y for new instances $\boldsymbol{x}$ coming from the same source. Contrast this with the online scenario: at each time t, the algorithm receives an input $\boldsymbol{x}_t$, makes its prediction $\hat{y}_t$ using its *current hypothesis* $\boldsymbol{w}_t$, and

N. Cesa-Bianchi et al. (Eds.): ALT 2002, LNAI 2533, pp. 113–127, 2002.

upon seeing the correct outcome y_t updates its hypothesis to $\boldsymbol{w}_{t+1}$. Thus, the algorithm is interleaving predicting and learning. This goes on for some time, and the goal is to minimise the total number of prediction mistakes. The question now is: *can an analogue of large margin classification usefully be applied in an online setting?* In addition to the number of prediction mistakes, we might be interested in the convergence properties of the algorithm, the quality of the final hypothesis produced etc.

Theoretical analyses of mistake bounds of online algorithms are typically done in a worst-case setting. Then it is known that the best mistake bounds are achieved by mistake-driven algorithms, i.e., algorithms that update only after they made a mistake. For linear separation, this means having $y\boldsymbol{w} \cdot \boldsymbol{x} < 0$. The analogue of large margin classification would be to update whenever we have $y\boldsymbol{w} \cdot \boldsymbol{x} < \rho$ for some positive margin $\rho > 0$. If $0 < y\boldsymbol{w} \cdot \boldsymbol{x} < \rho$, we get updates that are not mistake-driven, and thus not useful at least in terms of worst-case mistake bounds. (Of course, if we wish our on-line algorithm to converge to a maximum-margin classifier, as in [12,6], we need to use a positive margin, but this is a somewhat different goal.) Since margins can be changed without affecting the classifications by simply multiplying the weight vector by a scalar, margin-based algorithms usually also employ some kind of regularisation to control the norm of the weight vector.

To get a better idea of the usefulness of large margins and regularisation in online learning, we consider the situation when the target classifier we are trying to learn is allowed to move over time. This is the setting analysed earlier for regression by Herbster and Warmuth [10] and Herbster [9] and for classification with disjunctions as targets by Auer and Warmuth [2]. This previous work also uses norm bounds on the hypothesis, or some other form of regularisation, to deal with having a moving target (even when no margins are involved). More recently, Mesterharm [14] has considered tracking arbitrary linear classifiers with a variant of Winnow [13].

In this paper we establish mistake bounds with moving targets for general linear classification algorithms. We have bounds for two algorithms, a simplified version of Gentile's Approximate Large Margin Algorithm (ALMA) [6], and a new Naive Online Regularised-risk Minimisation Algorithm (NORMA) which is motivated by gradient descent with respect to a regularised risk [11]. We have a special interest in whether using a nonzero margin may help here. As it turns out, the best bounds we can obtain for nonzero margins are identical to those for zero margin (i.e., mistake driven algorithms). This is not really conclusive in any sense, but it does give some evidence that mistake driven algorithms are not the only way to minimise mistakes. On the technical side, our analysis of ALMA is a rather standard application of known techniques. For analysing NORMA we need something a little different to handle the weight decay. The technique requires some additional assumptions, so our bounds for NORMA are less general than for ALMA. However, experiments on artificial data suggest that the actual performance of the algorithms is rather similar.

In Section 2, we describe more formally the online mistake-bounded model and what we mean by moving targets. Section 3 describes the algorithms we study here. The main theoretical results are given in Section 4. Some experiments, which use artificial data to study the actual behaviour of the algorithms, are described in Section 5.

2 Basic Setting

We consider linear classification problems. An example is a pair $(\boldsymbol{x}, y) \in \mathbb{R}^n \times \{-1, +1\}$. We interpret a weight vector $\boldsymbol{w} \in \mathbb{R}^n$ as a linear classifier, which gives for a vector $\boldsymbol{x}$ the classification $+1$ if $\boldsymbol{w} \cdot \boldsymbol{x} > 0$, and otherwise classification -1. We say that $\boldsymbol{w}$ makes a *mistake*, or *classification error*, on example $(\boldsymbol{x}, y)$, if $y\boldsymbol{w} \cdot \boldsymbol{x} \leq 0$. (Thus, we consider $\boldsymbol{w} \cdot \boldsymbol{x} = 0$ as an error.) We generalise this by saying that $\boldsymbol{w}$ makes a *margin error at scale* ρ if $y\boldsymbol{w} \cdot \boldsymbol{x} \leq \rho$. We also define the *hinge loss* as $L_\rho(\boldsymbol{w}, \boldsymbol{x}, y) = \max\{0, \rho - y\boldsymbol{w} \cdot \boldsymbol{x}\}$. The scale parameter ρ, or the *margin*, is usually nonnegative; we omit mentioning it when it is clear from the context. We are basically interested in finding weight vectors $\boldsymbol{w}$ that make few mistakes, or few margin errors, but the continuous-valued hinge loss turns out to be a useful tool in analysing the algorithms. Notice that $L_\rho(\boldsymbol{w}, \boldsymbol{x}, y) \geq \rho$ if and only if $\boldsymbol{w}$ made a mistake on $(\boldsymbol{x}, y)$.

An online linear classification algorithm maintains as its current *hypothesis* a weight vector $\boldsymbol{w} \in \mathbb{R}^n$. We denote the hypothesis at time t by $\boldsymbol{w}_t$. The initial hypothesis $\boldsymbol{w}_1$ is typically $\mathbf{0}$, for lack of any other preference. At time t, for $t = 1, \ldots, T$, the algorithm receives an instance $\boldsymbol{x}_t \in \mathbb{R}^n$ and makes its prediction, which is $+1$ if $\boldsymbol{w}_t \cdot \boldsymbol{x}_t \geq 0$ and -1 otherwise. Then the algorithm receives the correct outcome $y_t \in \{-1, +1\}$ and updates its weight vector into $\boldsymbol{w}_{t+1}$ based on this new information.

Suppose A is some online algorithm. We write $\sigma_t = 1$ if the algorithm made a margin error at trial t ($y_t \boldsymbol{w}_t \cdot \boldsymbol{x}_t \leq \rho$) and $\sigma_t = 0$ otherwise. We denote the total number of margin errors made by A over a sequence of T examples by $\mathrm{ME}_\rho(A) = \sum_{t=1}^T \sigma_t$. Similarly, let $\mathrm{Mist}(A)$ be the number of mistakes. We will also use the *cumulative hinge loss* $\mathrm{Loss}_\rho(A) = \sum_{t=1}^T L_\rho(\boldsymbol{w}_t, \boldsymbol{x}_t, y_t)$. Notice that $\mathrm{Mist}(A) \leq \mathrm{Loss}_\rho(A)/\rho$.

To prove a mistake bound, we obviously need to assume something about the examples. For instance, the Perceptron Convergence Theorem [15] assumes that some weight vector $\boldsymbol{u} \in \mathbb{R}^n$ separates the examples with margin $\mu > 0$; the mistake bound is then proportional to $(||\boldsymbol{u}||_2/\mu)^2$. More generally, given a fixed *comparison vector* $\boldsymbol{u} \in \mathbb{R}^n$, let $\mathrm{Loss}_\mu(\boldsymbol{u}) = \sum_{t=1}^T L_\mu(\boldsymbol{u}, \boldsymbol{x}_t, y_t)$ be its cumulative hinge loss with respect to margin μ. Thus, $\boldsymbol{u}$ separates the examples with margin μ if and only if $\mathrm{Loss}_\mu(\boldsymbol{u}) = 0$. Our approach will be to bound the cumulative hinge loss $\mathrm{Loss}_\rho(A)$ of an online algorithm in terms of $\inf_{\boldsymbol{u} \in \mathcal{U}} \mathrm{Loss}_\mu(\boldsymbol{u})$ where $\mu > \rho$ and $\mathcal{U} \subset \mathbb{R}^n$ is some *comparison class* of vectors. Typical comparison classes consist of vectors of bounded q-norm for some $1 \leq q \leq 2$ [8,7], i.e., $\mathcal{U} = \{\boldsymbol{u} \mid ||\boldsymbol{u}||_q \leq B\}$ for some $B > 0$. Our loss bounds go to infinity as ρ approaches μ; in other words, we need to give the algorithm a slight advantage

by measuring its performance with respect to a smaller margin than that used for the comparison vectors. From a hinge loss bound we can rather easily derive a bound for the number of mistakes or margin errors.

We generalise the setting for moving comparison vectors by considering *comparison sequences* $\boldsymbol{U} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{T+1})$ [10]. The loss of such a comparison sequence is naturally defined as $\mathrm{Loss}_\mu(\boldsymbol{U}) = \sum_{t=1}^{T} L_\mu(\boldsymbol{u}_t, \boldsymbol{x}_t, y_t)$. As in the fixed target scenario, we assume some norm bound $||\boldsymbol{u}_t||_q \leq B$ for all the individual comparison vectors $\boldsymbol{u}_t$. We additionally restrict the amount of movement by the comparison vectors in terms of the total q-norm distance $\sum_t ||\boldsymbol{u}_t - \boldsymbol{u}_{t+1}||_q$ travelled by the comparison vectors. Thus, we define for parameters $1 \leq q \leq 2$, $B > 0$ and $D \geq 0$ the comparison class

$$\mathcal{U}_q(B, D) = \{ (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{T+1}) \mid \textstyle\sum_{t=1}^{T} ||\boldsymbol{u}_t - \boldsymbol{u}_{t+1}||_q \leq D, ||\boldsymbol{u}_t||_q \leq B \} \ .$$

For technical reasons, we also need to consider bounding the sum of *squared* distances, so we also define

$$\mathcal{U}'_q(B, D_1, D_2) = \Big\{ (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{T+1}) \mid \textstyle\sum_{t=1}^{T} ||\boldsymbol{u}_t - \boldsymbol{u}_{t+1}||_q \leq D_1, \\ \textstyle\sum_{t=1}^{T} ||\boldsymbol{u}_t - \boldsymbol{u}_{t+1}||_q^2 \leq D_2, ||\boldsymbol{u}_t||_q \leq B \Big\} \ .$$

The meaning of the parameter D_2 is perhaps a little non-intuitive; it will become clearer after Theorem 4 when we discuss how D_2 appears in the loss bounds.

3 The Algorithms

The algorithms we consider are based on the p-norm algorithms introduced by Grove et al. [8] and further studied, e.g., by Gentile and Littlestone [7]. Thus, for the rest of the paper we assume $p \geq 2$ and $2 \geq q > 1$ are such that $1/p + 1/q = 1$, and define

$$f_i(\boldsymbol{w}) = \frac{\mathrm{sign}(w_i)|w_i|^{q-1}}{||\boldsymbol{w}||_q^{q-2}} \ . \tag{1}$$

Notice that $\boldsymbol{f}$ is one-to-one from $\mathbb{R}^n$ onto $\mathbb{R}^n$ with the inverse given by $f_i^{-1}(\boldsymbol{\theta}) = \mathrm{sign}(\theta_i)|\theta_i|^{p-1}/||\boldsymbol{\theta}||_p^{p-2}$. The update of the p-norm Perceptron can be written as

$$\boldsymbol{w}_{t+1} = \boldsymbol{f}^{-1}(\boldsymbol{f}(\boldsymbol{w}_t) + \alpha \sigma_t y_t \boldsymbol{x}_t) \ ,$$

where $\alpha > 0$ is a learning rate parameter. The parameter p can adjusted to change the behaviour of the algorithm. For $p = 2$, the function $\boldsymbol{f}$ is the identity function, and the algorithm is the usual Perceptron algorithm. Setting $p = O(\log n)$ gives an algorithm with performance similar to Winnow [8,7].

The first algorithm we define here is called Naive Online Regularised-Risk Minimisation Algorithm, or NORMA. The algorithm is parameterised by a learning rate $\alpha > 0$, a weight decay parameter $0 \leq \lambda < 1/\alpha$, and a margin $\rho \geq 0$. The update is then

$$\boldsymbol{w}_{t+1} = \boldsymbol{f}^{-1}((1 - \alpha\lambda)\boldsymbol{f}(\boldsymbol{w}_t) + \alpha \sigma_t y_t \boldsymbol{x}_t) \ , \tag{2}$$

where again $\sigma_t = 1$ if $y_t \boldsymbol{w}_t \cdot \boldsymbol{x}_t \leq \rho$ and $\sigma_t = 0$ otherwise. For $p = 2$ the NORMA update can be seen as a gradient descent step with respect to the regularised risk $R(\boldsymbol{w}) = L_\rho(\boldsymbol{w}, \boldsymbol{x}_t, y_t) + \lambda ||\boldsymbol{w}||^2/2$; see [11] for additional discussion and applications.

We also consider a simplified version of Gentile's Approximate Large Margin Algorithm ALMA [6]. For simplicity, we call our algorithm just ALMA although it omits the parameter tuning method of Gentile's original algorithm. Our version of ALMA has a fixed learning rate parameter $\alpha > 0$, regularisation parameter $B > 0$ and margin $\rho \geq 0$. The update of ALMA has two steps:

Additive step $\boldsymbol{w}'_{t+1} = \boldsymbol{f}^{-1}(\boldsymbol{f}(\boldsymbol{w}_t) + \alpha \sigma_t y_t \boldsymbol{x}_t), \quad \sigma_t = 1_{y_t \boldsymbol{w}_t \cdot \boldsymbol{x}_t \leq \rho}$
Normalisation step $\boldsymbol{w}_{t+1} = \boldsymbol{w}'_{t+1}/\beta_t$ where $\beta_t = \max\{\, 1, ||\boldsymbol{w}'_{t+1}||_q / B \,\}$

Gentile's original ALMA also includes a method for tuning the parameters α and ρ during learning. The tuning method there has been carefully crafted so that assuming separable data, the algorithm converges to an approximate maximum margin classifier even without advance knowledge of the maximum margin. We use here a cruder version where α and ρ need to be fixed beforehand (and a poor choice may lead to bad performance), since we have not been able to generalise the dynamic tuning method to the moving target scenario.

In the case $p = 2$ (with $\boldsymbol{f}$ the identity function), we see that the hypotheses of NORMA and ALMA can be represented as $\boldsymbol{w}_{T+1} = \sum_{t=1}^{T} a_t \boldsymbol{x}_t$ for some scalar coefficients a_t. Thus, the algorithms allow the standard generalisation to non-linear classification by using kernels to compute dot products. Also the normalisation in the kernel version of ALMA can be accomplished with little computational overhead by keeping track of the changes in $||\boldsymbol{w}_t||$ [6].

Both NORMA and ALMA have been here represented as having three parameters: margin ρ, learning rate α, and a regularisation type parameter (λ or B). However, effectively there are only two parameters, as multiplying all the parameters (except for λ) by any constant will leave the predictions of the algorithm unchanged; also the scaled hinge losses $L_\rho(\boldsymbol{w}_t, \boldsymbol{x}_t, y_t)/\rho$ remain invariant. Thus, without loss of generality we could fix, e.g., $\rho = 1$, but we find it convenient to do the analysis with all the parameters explicitly written out. Although the parameterisations of NORMA and ALMA are quite similar, we find that of NORMA a little more intuitive. The underlying idea of gradient descent with respect to a regularised risk can be easily applied, e.g., in SVM regression using the ν parameterisation [11].

4 Worst-Case Mistake Bounds

We start with some general comments on the kind of bounds we are after. Fix some comparison class $\mathcal{U}$; say $\mathcal{U} = \mathcal{U}_q(B, D)$ for some $B > 0$ and $D \geq 0$. Let $K_* = \inf_{\boldsymbol{U} \in \mathcal{U}} \mathrm{Loss}_\mu(\boldsymbol{U})$ for some margin $\mu > 0$. Thus if some $\boldsymbol{U} \in \mathcal{U}$ separates the examples with margin μ, i.e., $y_t \boldsymbol{u}_t \cdot \boldsymbol{x}_t > \mu$ for all t, then $K_* = 0$. Otherwise K_* is a measure for the non-separability of the examples at scale μ. An alternative intuition is to think that the data have been obtained by corrupting some

(hypothetical) separable data by noise; then K_* would be a measure of the total amount of noise added.

In the case of non-moving targets, one can get bounds of the form $\mathrm{Mist}(A) \leq K_*/\mu + o(K_*)$ ([7]; see also [5]). Here $o(K_*)$ is a term that is sublinear in K_*; we would expect it to depend on the norms of the examples, the bound B, etc. Notice that K_*/μ is an upper bound for the number of mistakes made by the best comparison sequence from $\mathcal{U}$; of course, it may be a quite loose bound. We would expect target movement to result in an additional $O(D)$ term in the mistake bound, analogous to the regression case [10]. In other words, there should be a constant cost per unit target movement. It turns out that with the optimal choice of the parameters, bounds of exactly this form are attainable for ALMA. For NORMA there are some additional considerations about the nature of the target movement.

Choosing the parameters is an issue in the bounds we have. The bounds depend on the choice of the learning rate and margin parameters, and the optimal choices depend on quantities (such as K_*) that would not be available when the algorithm starts. In our bounds, we handle this by assuming an upper bound $K \geq K_*$ that can be used for tuning. By substituting $K = K_*$, we obtain the kind of bound we discussed above; otherwise the estimate K replaces K_* in the bound. In a practical application, we probably prefer to ignore the formal tuning results in the bounds and just tune the parameters by whatever empirical methods we prefer. Recently, online algorithms have been suggested that dynamically tune the parameters to almost optimal values as the algorithm runs [1,6]. Applying such techniques to our analysis remains an open problem.

We now turn to the actual bounds, starting with a margin error bound for ALMA. It will be convenient to give the parameter tunings in terms of the function

$$h(x, R, S) = \sqrt{\frac{S}{R}\left(x + \frac{S}{R}\right)} - \frac{S}{R} \ ,$$

where we assume x, R and S to be positive. Notice that $0 \leq h(x, R, S) \leq x$ holds, and $\lim_{R \to 0+} h(x, R, S) = x/2$. Accordingly, we define $h(x, 0, S) = x/2$.

Theorem 1. *Let $X > 0$ and suppose that $||\boldsymbol{x}_t||_p \leq X$ for all t. Fix $K \geq 0$, $D \geq 0$ and $B > 0$, and write*

$$C = \frac{p-1}{4} X^2 (B^2 + 2BD) \ . \tag{3}$$

Consider ALMA *with regularisation parameter B, margin parameter ρ and learning rate $\alpha = 2h(\mu - \rho, K, C)/((p-1)X^2)$ where $\mu > \rho \geq 0$. If we have $\mathrm{Loss}_\mu(\boldsymbol{U}) \leq K$ for some $\boldsymbol{U} \in \mathcal{U}_q(B, D)$, then*

$$\mathrm{ME}_\rho(\textsc{Alma}) \leq \frac{K}{\mu - \rho} + 2\frac{C}{(\mu - \rho)^2} + 2\left(\frac{K}{\mu - \rho} + \frac{C}{(\mu - \rho)^2}\right)^{1/2} \left(\frac{C}{(\mu - \rho)^2}\right)^{1/2} \ .$$

To prove Theorem 1, we apply Herbster and Warmuth's [10] technique of using a Bregman divergence [4] as a measure of progress. As first suggested

by Grove et al. [8], the p-norm family of algorithms is related to the *potential function* $F(\boldsymbol{w}) = ||\boldsymbol{w}||_q^2/2$. (Notice that $\nabla F = \boldsymbol{f}$ where $\boldsymbol{f}$ is as in (1).) Using this, we define the appropriate divergence for $\boldsymbol{u}, \boldsymbol{w} \in \mathbb{R}^n$ as

$$d_q(\boldsymbol{u}, \boldsymbol{w}) = F(\boldsymbol{u}) - F(\boldsymbol{w}) + \boldsymbol{f}(\boldsymbol{w}) \cdot (\boldsymbol{w} - \boldsymbol{u}) \ . \tag{4}$$

See [8,7,6] for basic properties of F and d_q.

The key part of the analysis is the following lower bound on the *progress* the algorithm makes on its tth update.

Lemma 1. *Assume $||\boldsymbol{u}_t||_q \leq B$ and $||\boldsymbol{x}_t||_p \leq X$ for all t. Then at any trial t the update of* ALMA *with regularisation parameter B, margin parameter ρ and learning rate α satisfies*

$$\begin{aligned} &d_q(\boldsymbol{u}_t, \boldsymbol{w}_t) - d_q(\boldsymbol{u}_{t+1}, \boldsymbol{w}_{t+1}) \\ &\quad \geq \alpha L_\rho(\boldsymbol{w}_t, \boldsymbol{x}_t, y_t) - \alpha L_\mu(\boldsymbol{u}_t, \boldsymbol{x}_t, y_t) + \alpha\sigma_t \left(\mu - \rho - \alpha \frac{p-1}{2} X^2\right) \\ &\qquad + \frac{1}{2}||\boldsymbol{u}_t||_q^2 - \frac{1}{2}||\boldsymbol{u}_{t+1}||_q^2 - B||\boldsymbol{u}_{t+1} - \boldsymbol{u}_t||_q \ . \end{aligned}$$

Proof. We split the progress into three parts:

$$\begin{aligned} d_q(\boldsymbol{u}_t, \boldsymbol{w}_t) - d_q(\boldsymbol{u}_{t+1}, \boldsymbol{w}_{t+1}) = &(d_q(\boldsymbol{u}_t, \boldsymbol{w}_t) - d_q(\boldsymbol{u}_t, \boldsymbol{w}'_{t+1})) \\ &+ (d_q(\boldsymbol{u}_t, \boldsymbol{w}'_{t+1}) - d_q(\boldsymbol{u}_t, \boldsymbol{w}_{t+1})) \\ &+ (d_q(\boldsymbol{u}_t, \boldsymbol{w}_{t+1}) - d_q(\boldsymbol{u}_{t+1}, \boldsymbol{w}_{t+1})) \ . \end{aligned} \tag{5}$$

Grove et al. [8] have shown that $d_q(\boldsymbol{w}_t, \boldsymbol{w}'_{t+1}) \leq \frac{p-1}{2}\sigma_t\alpha^2||\boldsymbol{x}_t||_p^2$. Hence, for the first part of (5) we get

$$\begin{aligned} d_q(\boldsymbol{u}_t, \boldsymbol{w}_t) - d_q(\boldsymbol{u}_t, \boldsymbol{w}'_{t+1}) &= \alpha\sigma_t y_t \boldsymbol{x}_t \cdot (\boldsymbol{u}_t - \boldsymbol{w}_t) - d_q(\boldsymbol{w}_t, \boldsymbol{w}'_{t+1}) \\ &\geq \alpha\sigma_t y_t \boldsymbol{x}_t \cdot (\boldsymbol{u}_t - \boldsymbol{w}_t) - \frac{p-1}{2}\alpha^2\sigma_t||\boldsymbol{x}_t||_p^2 \\ &\geq \alpha(\sigma_t\mu - L_\mu(\boldsymbol{u}_t, \boldsymbol{x}_t, y_t)) - \alpha(\sigma_t\rho - L_\rho(\boldsymbol{w}_t, \boldsymbol{x}_t, y_t)) \\ &\quad - \frac{p-1}{2}\alpha^2\sigma_t||\boldsymbol{x}_t||_p^2 \ . \end{aligned}$$

It is easy to see that $\boldsymbol{w}_{t+1}$ satisfies

$$\boldsymbol{w}_{t+1} = \arg\min_{\boldsymbol{w} \in \mathcal{B}} d_q(\boldsymbol{w}, \boldsymbol{w}'_{t+1})$$

where $\mathcal{B} = \{\, \boldsymbol{w} \mid ||\boldsymbol{w}||_q \leq B \,\}$. Since $\boldsymbol{u}_t \in \mathcal{B}$ and $\mathcal{B}$ is convex, the well-known result about projections with respect to a Bregman divergence (see [10] for details) implies

$$d_q(\boldsymbol{u}_t, \boldsymbol{w}'_{t+1}) - d_q(\boldsymbol{u}_t, \boldsymbol{w}_{t+1}) \geq 0 \ .$$

For the third part we have

$$\begin{aligned} d_q(\boldsymbol{u}_t, \boldsymbol{w}_{t+1}) - d_q(\boldsymbol{u}_{t+1}, \boldsymbol{w}_{t+1}) &= \frac{1}{2}||\boldsymbol{u}_t||_q^2 - \frac{1}{2}||\boldsymbol{u}_{t+1}||_q^2 + (\boldsymbol{u}_{t+1} - \boldsymbol{u}_t) \cdot \boldsymbol{f}(\boldsymbol{w}_{t+1}) \\ &\geq \frac{1}{2}||\boldsymbol{u}_t||_q^2 - \frac{1}{2}||\boldsymbol{u}_{t+1}||_q^2 - B||\boldsymbol{u}_{t+1} - \boldsymbol{u}_t||_q \end{aligned}$$

by Hölder's inequality and the fact $||f(\boldsymbol{w}_{t+1})||_p = ||\boldsymbol{w}_{t+1}||_q \leq B$. Substituting the above three estimates to the right-hand side of (5) gives the claim.

The following technical lemma, which is proved by a simple differentiation, is used for choosing the optimal parameters.

Lemma 2. *Given $R > 0$, $S > 0$ and $\gamma > 0$ define $f(z) = R/(\gamma - z) + S/(z(\gamma - z))$ for $0 < z < \gamma$. Then $f(z)$ is maximised for $z = h(\gamma, R, S)$, and the maximum value is*

$$f(h(\gamma, R, S)) = \frac{R}{\gamma} + \frac{2S}{\gamma^2} + 2\left(\frac{R}{\gamma} + \frac{S}{\gamma^2}\right)^{1/2}\left(\frac{S}{\gamma^2}\right)^{1/2} \ .$$

Proof of Theorem 1. By summing the bound of Lemma 1 over $t = 1, \ldots, T$ we get

$$\begin{aligned} &d_q(\boldsymbol{u}_1, \boldsymbol{w}_1) - d_q(\boldsymbol{u}_{T+1}, \boldsymbol{w}_{T+1}) \\ &\quad \geq \alpha \mathrm{Loss}_\rho(\textsc{Alma}) - \alpha \mathrm{Loss}_\mu(\boldsymbol{U}) + \alpha \mathrm{ME}_\rho(\textsc{Alma})\left(\mu - \rho - \alpha\frac{p-1}{2}X^2\right) \\ &\qquad + \frac{1}{2}||\boldsymbol{u}_1||_q^2 - \frac{1}{2}||\boldsymbol{u}_{T+1}||_q^2 - B\sum_{t=1}^{T}||\boldsymbol{u}_{t+1} - \boldsymbol{u}_t||_q \ . \end{aligned}$$

We take $\boldsymbol{w}_1 = \boldsymbol{0}$, so $d_q(\boldsymbol{u}_1, \boldsymbol{w}_1) = ||\boldsymbol{u}_1||_q^2/2$, and clearly $-d_q(\boldsymbol{u}_{T+1}, \boldsymbol{w}_{T+1}) \leq 0$. On the right-hand side, we use the assumptions about $\boldsymbol{U}$. We get

$$\mathrm{ME}_\rho(\textsc{Alma})(\mu - \rho - \alpha(p-1)X^2/2) \leq K - \mathrm{Loss}_\rho(\textsc{Alma}) + \frac{1}{\alpha}(BD + B^2/2) \ . \quad (6)$$

We can of course drop the non-positive term $-\mathrm{Loss}_\rho(\textsc{Alma})$. For the value α given in the theorem, we have $\mu - \rho - \alpha(p-1)X^2/2 > 0$, so we get

$$\mathrm{ME}_\rho(\textsc{Alma}) \leq \frac{K}{\mu - \rho - \alpha(p-1)X^2/2} + \frac{BD + B^2/2}{\alpha(\mu - \rho - \alpha(p-1)X^2/2)} \ .$$

The claim follows by applying Lemma 2 with $z = \alpha(p-1)X^2/2$, $\gamma = \mu - \rho$, $R = K$ and $S = C$.

Next, we use the margin error result of Theorem 1 to obtain mistake bounds. It turns out that two ways of choosing the parameter pair (α, ρ) result in the same mistake bound. In particular, the same bound we get for the mistake driven algorithm with $\rho = 0$ also holds for certain positive $\rho > 0$, assuming the learning rate is chosen appropriately.

Theorem 2. *Let $X > 0$ and suppose that $||\boldsymbol{x}_t||_p \leq X$ for all t. Fix $K \geq 0$, $B > 0$ and $D \geq 0$. Define C as in (3), and given $\mu > 0$ let $r = h(\mu, K, C)$. Consider* Alma *with regularisation parameter B, learning rate $\alpha = 2r/((p-1)X^2)$ and margin set to either $\rho = 0$ or $\rho = \mu - r$. Then for both of these margin settings, if there exists a comparison sequence $\boldsymbol{U} \in \mathcal{U}_q(B, D)$ such that $\mathrm{Loss}_\mu(\boldsymbol{U}) \leq K$, we have*

$$\mathrm{Mist}(\textsc{Alma}) \leq \frac{K}{\mu} + \frac{2C}{\mu^2} + 2\left(\frac{C}{\mu^2}\right)^{1/2}\left(\frac{K}{\mu} + \frac{C}{\mu^2}\right)^{1/2} \ .$$

Proof. For $\rho = 0$ this is a direct corollary of Theorem 1. To get non-zero ρ, we set $\alpha = 2(\mu - \rho)/((p-1)X^2)$ so that the coefficient in front of $\mathrm{ME}_\rho(\textsc{Alma})$ in (6) becomes zero. We then exploit $\mathrm{Mist}(\textsc{Alma}) \le \mathrm{Loss}_\rho(\textsc{Alma})/\rho$ to get

$$\mathrm{Mist}(\textsc{Alma}) \le \frac{K}{\rho} + \frac{(p-1)X^2(BD + B^2/2)}{2\rho(\mu - \rho)} \quad .$$

The claim follows by applying Lemma 2 with $\gamma = \mu$ and $z = \mu - \rho$.

To interpret Theorem 2, let us start with a fixed target ($D = 0$) and $p = 2$. In the noise-free case $K = 0$, we recover the familiar Perceptron mistake bound X^2B^2/μ^2. Notice that by Theorem 2 we can get this mistake bound also using the positive margin $\rho = \mu/2$ with suitable α. However, a positive margin obviously leads to a larger number of updates; the margin error bound we get from Theorem 1 with this tuning is worse by a factor of 4 compared to $\rho = 0$.

In the noisy case $K > 0$, we get additional terms $K/\mu + O(\sqrt{K/\mu})$ as expected. For a discussion of how different choices of p affect this kind of a bound, see [8] and [7]. If the target movement bound D is non-zero, it will appear linearly in the mistake bound as expected.

Our bounds generalise those of Gentile [6] in that we allow a moving target. Also, Gentile was concerned only with margin error bounds and not obtaining mistake bounds using a nonzero margin. However, in the case of no target movement ($D = 0$), Gentile gets better bounds than ours by using special techniques we have not been able to apply to the moving target case ($D > 0$). Also, Gentile's algorithm includes a dynamical tuning of the parameters, unlike the simplified version we here call Alma.

We now go to bounds for Norma. Since Norma does not maintain a bound on the norm of the weight vector, the meaning of margin errors is not as clear as for Alma. However, the number of margin errors, i.e., updates, is still interesting as a measure of the complexity of the hypothesis produced by the algorithm.

Theorem 3. *Let $X > 0$ and suppose that $||\boldsymbol{x}_t||_p \le X$ for all t. Fix $K \ge 0$, $B > 0$, $D_1 \ge 0$ and $D_2 \ge 0$. Write*

$$C = \frac{p-1}{4} X^2 \left(B^2 + B\left(\sqrt{TD_2} + D_1 \right) \right) \tag{7}$$

and, given parameters $\mu > \rho \ge 0$, let $\alpha' = 2h(\mu - \rho, K, C)/((p-1)X^2)$. Consider Norma *with weight decay parameter*

$$\lambda = \frac{1}{B\alpha'} \sqrt{\frac{D_2}{T}} \quad , \tag{8}$$

learning rate parameter $\alpha = \alpha'/(1+\alpha'\lambda)$ and margin ρ. If we have $\mathrm{Loss}_\mu(\boldsymbol{U}) \le K$ for some $\boldsymbol{U} \in \mathcal{U}'_q(B, D_1, D_2)$, then

$$\mathrm{ME}_\rho(\textsc{Norma}) \le \frac{K}{\mu - \rho} + \frac{2C}{(\mu - \rho)^2} + 2\left(\frac{C}{(\mu - \rho)^2} \right)^{1/2} \left(\frac{K}{\mu - \rho} + \frac{C}{(\mu - \rho)^2} \right)^{1/2} \quad .$$

Proof. It will be convenient to write $\boldsymbol{\theta}_t = \boldsymbol{f}(\boldsymbol{w}_t)$. We also define $\boldsymbol{\theta}'_{t+1} = \boldsymbol{\theta}_t + \alpha'\sigma_t y_t \boldsymbol{x}_t$, so $\boldsymbol{\theta}_{t+1} = (1-\alpha\lambda)\boldsymbol{\theta}'_{t+1}$, and let $\boldsymbol{w}'_{t+1}$ be such that $\boldsymbol{\theta}'_{t+1} = \boldsymbol{f}(\boldsymbol{w}'_{t+1})$. As in the proof of Lemma 1, we split the progress into three parts:

$$\begin{aligned} d_q(\boldsymbol{u}_t, \boldsymbol{w}_t) - d_q(\boldsymbol{u}_{t+1}, \boldsymbol{w}_{t+1}) = {} & (d_q(\boldsymbol{u}_t, \boldsymbol{w}_t) - d_q(\boldsymbol{u}_t, \boldsymbol{w}'_{t+1})) \\ & + (d_q(\boldsymbol{u}_t, \boldsymbol{w}'_{t+1}) - d_q(\boldsymbol{u}_t, \boldsymbol{w}_{t+1})) \\ & + (d_q(\boldsymbol{u}_t, \boldsymbol{w}_{t+1}) - d_q(\boldsymbol{u}_{t+1}, \boldsymbol{w}_{t+1})) \ . \end{aligned} \tag{9}$$

For the first part we have

$$\begin{aligned} d_q(\boldsymbol{u}_t, \boldsymbol{w}_t) - d_q(\boldsymbol{u}_t, \boldsymbol{w}'_{t+1}) \geq {} & \alpha'(\sigma_t\mu - L_\mu(\boldsymbol{u}_t, \boldsymbol{x}_t, y_t)) - \alpha'(\sigma_t\rho - L_\rho(\boldsymbol{w}_t, \boldsymbol{x}_t, y_t)) \\ & - \frac{p-1}{2}\alpha'^2\sigma_t X^2 \ . \end{aligned} \tag{10}$$

as in the proof of Lemma 1.

For the second part, the definition of d_q gives

$$d_q(\boldsymbol{u}_t, \boldsymbol{w}'_{t+1}) - d_q(\boldsymbol{u}_t, \boldsymbol{w}_{t+1}) = d_q(\boldsymbol{w}_{t+1}, \boldsymbol{w}'_{t+1}) + (\boldsymbol{\theta}'_{t+1} - \boldsymbol{\theta}_{t+1}) \cdot (\boldsymbol{w}_{t+1} - \boldsymbol{u}_t) \ .$$

By using $\boldsymbol{w}_{t+1} = (1-\alpha\lambda)\boldsymbol{w}'_{t+1}$ and the fact $\boldsymbol{w} \cdot \boldsymbol{f}(\boldsymbol{w}) = ||\boldsymbol{w}||_q^2$ we get

$$\begin{aligned} d_q(\boldsymbol{w}_{t+1}, \boldsymbol{w}'_{t+1}) &= \frac{1}{2}||(1-\alpha\lambda)\boldsymbol{w}'_{t+1}||_q^2 - \frac{1}{2}||\boldsymbol{w}'_{t+1}||_q^2 + \alpha\lambda\boldsymbol{w}'_{t+1} \cdot \boldsymbol{\theta}'_{t+1} \\ &= \frac{1}{2}\left(\frac{\alpha\lambda}{1-\alpha\lambda}\right)^2 ||\boldsymbol{w}_{t+1}||_q^2 \ . \end{aligned}$$

Also, since $\boldsymbol{\theta}'_{t+1} - \boldsymbol{\theta}_{t+1} = \alpha\lambda\boldsymbol{\theta}'_{t+1} = \alpha\lambda\boldsymbol{\theta}_{t+1}/(1-\alpha\lambda)$, we have

$$\begin{aligned} (\boldsymbol{\theta}'_{t+1} - \boldsymbol{\theta}_{t+1}) \cdot (\boldsymbol{w}_{t+1} - \boldsymbol{u}_t) &= \frac{\alpha\lambda}{1-\alpha\lambda}(\boldsymbol{\theta}_{t+1} \cdot \boldsymbol{w}_{t+1} - \boldsymbol{\theta}_{t+1} \cdot \boldsymbol{u}_t) \\ &= \frac{\alpha\lambda}{1-\alpha\lambda}(||\boldsymbol{w}_{t+1}||_q^2 - \boldsymbol{\theta}_{t+1} \cdot \boldsymbol{u}_t) \ . \end{aligned}$$

Hence, recalling the definition of α' and using the fact $||\boldsymbol{w}||_q = ||\boldsymbol{f}(\boldsymbol{w})||_p$, we get

$$\begin{aligned} & d_q(\boldsymbol{u}_t, \boldsymbol{w}'_{t+1}) - d_q(\boldsymbol{u}_t, \boldsymbol{w}_{t+1}) \\ &\quad = \left(\alpha'\lambda + \frac{\alpha'^2\lambda^2}{2}\right) ||\boldsymbol{w}_{t+1}||_q^2 - \alpha'\lambda\boldsymbol{\theta}_{t+1} \cdot \boldsymbol{u}_t \\ &\quad = \left(\alpha'\lambda + \frac{\alpha'^2\lambda^2}{2}\right) ||\boldsymbol{\theta}_{t+1}||_p^2 - \alpha'\lambda\boldsymbol{\theta}_{t+1} \cdot \boldsymbol{u}_t \ . \end{aligned} \tag{11}$$

For the third part of (9) the definition of d_q directly gives

$$d_q(\boldsymbol{u}_t, \boldsymbol{w}_{t+1}) - d_q(\boldsymbol{u}_{t+1}, \boldsymbol{w}_{t+1}) = \frac{1}{2}||\boldsymbol{u}_t||_q^2 - \frac{1}{2}||\boldsymbol{u}_{t+1}||_q^2 + (\boldsymbol{u}_{t+1} - \boldsymbol{u}_t) \cdot \boldsymbol{\theta}_{t+1} \ . \tag{12}$$

Substituting (10), (11) and (12) into (9) gives us

$$\begin{aligned} & d_q(\boldsymbol{u}_t, \boldsymbol{w}_t) - d_q(\boldsymbol{u}_{t+1}, \boldsymbol{w}_{t+1}) \\ &\quad \geq \alpha'(\sigma_t\mu - L_\mu(\boldsymbol{u}_t, \boldsymbol{x}_t, y_t)) - \alpha'(\sigma_t\rho - L_\rho(\boldsymbol{w}_t, \boldsymbol{x}_t, y_t)) \\ &\qquad - \frac{p-1}{2}\alpha'^2\sigma_t||\boldsymbol{x}_t||_p^2 + \frac{1}{2}||\boldsymbol{u}_t||_q^2 - \frac{1}{2}||\boldsymbol{u}_{t+1}||_q^2 + R(\boldsymbol{\theta}_{t+1}) \end{aligned} \tag{13}$$

where

$$R(\boldsymbol{\theta}) = \left(\alpha'\lambda + \frac{\alpha'^2\lambda^2}{2}\right)||\boldsymbol{\theta}||_p^2 - \alpha'\lambda\boldsymbol{\theta}\cdot\boldsymbol{u}_t + (\boldsymbol{u}_{t+1} - \boldsymbol{u}_t)\cdot\boldsymbol{\theta} \ .$$

To bound $R(\boldsymbol{\theta}_{t+1})$ from below, we notice that R is convex. Its gradient is given by

$$\nabla R(\boldsymbol{\theta}) = \left(2\alpha'\lambda + \alpha'^2\lambda^2\right)\boldsymbol{f}^{-1}(\boldsymbol{\theta}) + \boldsymbol{u}_{t+1} - (1+\alpha'\lambda)\boldsymbol{u}_t$$

where $\boldsymbol{f}^{-1}$ is the inverse of $\boldsymbol{f}$. Therefore, $R(\boldsymbol{\theta}_{t+1}) \geq R(\boldsymbol{\theta}_*)$ where

$$\boldsymbol{f}^{-1}(\boldsymbol{\theta}_*) = \frac{\boldsymbol{u}_t - \boldsymbol{u}_{t+1} + \alpha'\lambda\boldsymbol{u}_t}{2\alpha'\lambda + \alpha'^2\lambda^2} \ .$$

Write $\boldsymbol{w}_* = \boldsymbol{f}^{-1}(\boldsymbol{\theta}_*)$. First using $||\boldsymbol{w}||_q = ||\boldsymbol{f}(\boldsymbol{w})||_p$ and $\boldsymbol{w}\cdot\boldsymbol{f}(\boldsymbol{w}) = ||\boldsymbol{w}||_q^2$ and then observing that $||\boldsymbol{u}_t - \boldsymbol{u}_{t+1} + \alpha'\lambda\boldsymbol{u}_t||_q \leq ||\boldsymbol{u}_t - \boldsymbol{u}_{t+1}||_q + \alpha'\lambda||\boldsymbol{u}_t||_q$ gives us

$$\begin{aligned}
R(\boldsymbol{\theta}_*) &= (\alpha'\lambda + \frac{\alpha'^2\lambda^2}{2})||\boldsymbol{\theta}_*||_p^2 - (2\alpha'\lambda + \alpha'^2\lambda^2)\boldsymbol{w}_*\cdot\boldsymbol{\theta}_* \\
&= -\frac{1}{2}(2\alpha'\lambda + \alpha'^2\lambda^2)||\boldsymbol{w}_*||_q^2 \\
&\leq -\frac{1}{2}\frac{1}{2\alpha'\lambda + \alpha'^2\lambda^2}\left(||\boldsymbol{u}_t - \boldsymbol{u}_{t+1}||_q + \alpha'\lambda||\boldsymbol{u}_t||_q\right)^2 \\
&= -\frac{1}{2}\frac{1}{2+\alpha'\lambda}\left(\frac{||\boldsymbol{u}_t - \boldsymbol{u}_{t+1}||_q^2}{\alpha'\lambda} + 2||\boldsymbol{u}_t - \boldsymbol{u}_{t+1}||_q||\boldsymbol{u}_t||_q + \alpha'\lambda||\boldsymbol{u}_t||_q^2\right) \ .
\end{aligned}$$

By applying $R(\boldsymbol{\theta}_{t+1}) \geq R(\boldsymbol{\theta}_*)$ in (13) and noticing that $-1/(2+\alpha'\lambda) > -1/2$, we get

$$\begin{aligned}
&d_q(\boldsymbol{u}_t, \boldsymbol{w}_t) - d_q(\boldsymbol{u}_{t+1}, \boldsymbol{w}_{t+1}) \\
&\quad\geq -\alpha'(\sigma_t\rho - L_\rho(\boldsymbol{w}_t, \boldsymbol{x}_t, y_t)) + \alpha'(\sigma_t\mu - L_\mu(\boldsymbol{u}_t, \boldsymbol{x}_t, y_t)) \\
&\qquad - \alpha'^2\sigma_t\frac{p-1}{2}||\boldsymbol{x}_t||_p^2 + \frac{1}{2}||\boldsymbol{u}_t||_q^2 - \frac{1}{2}||\boldsymbol{u}_{t+1}||_q^2 \\
&\qquad - \frac{1}{4}\left(\frac{||\boldsymbol{u}_{t+1} - \boldsymbol{u}_t||_q^2}{\alpha'\lambda} + 2||\boldsymbol{u}_t||_q||\boldsymbol{u}_{t+1} - \boldsymbol{u}_t||_q + \alpha'\lambda||\boldsymbol{u}_t||_q^2\right) \ . \quad (14)
\end{aligned}$$

By summing over $t = 1, \ldots, T$ and using the assumption that $\boldsymbol{U} \in \mathcal{U}_q'(B, D_1, D_2)$ we get

$$\begin{aligned}
&d_q(\boldsymbol{u}_1, \boldsymbol{w}_1) - d_q(\boldsymbol{u}_{T+1}, \boldsymbol{w}_{T+1}) \\
&\quad\geq \alpha'\mathrm{Loss}_\rho(\textsc{Norma}) - \alpha'\mathrm{Loss}_\mu(\boldsymbol{U}) \\
&\qquad + \alpha'\mathrm{ME}_\rho(\textsc{Norma})\left(\mu - \rho - \alpha'\frac{p-1}{2}X^2\right) \\
&\qquad + \frac{1}{2}||\boldsymbol{u}_1||_q^2 - \frac{1}{2}||\boldsymbol{u}_{T+1}||_q^2 \\
&\qquad - \frac{1}{4}\left(\frac{D_2}{\alpha'\lambda} + 2BD_1 + T\alpha'\lambda B^2\right) \ .
\end{aligned}$$

Now λ appears only in a subexpression $S(\alpha'\lambda)$ where

$$S(z) = -\frac{D_2}{z} - zTB^2 \ .$$

Since $S(z)$ is maximized for $z = \sqrt{D_2/(TB^2)}$, we choose λ as in (8) which gives $S(\alpha'\lambda) = -2B\sqrt{TD_2}$. We assume $\boldsymbol{w}_1 = \boldsymbol{0}$, so $d_q(\boldsymbol{u}_1, \boldsymbol{w}_1) - d_q(\boldsymbol{u}_{T+1}, \boldsymbol{w}_{T+1}) \leq d_q(\boldsymbol{u}_1, \boldsymbol{w}_1) = ||\boldsymbol{u}_1||_q^2/2$. By moving some terms around and estimating $||\boldsymbol{u}_{T+1}||_q \leq B$ and $\mathrm{Loss}_\mu(\boldsymbol{U}) \leq K$ we get

$$\begin{aligned} &\mathrm{Loss}_\rho(\textsc{Norma}) + \mathrm{ME}_\rho(\textsc{Norma}) \left(\mu - \rho - \alpha' \frac{p-1}{2} X^2\right) \\ &\quad \leq K + \frac{B^2 + B(\sqrt{TD_2} + D_1)}{2\alpha'} \ . \end{aligned} \tag{15}$$

To get a bound for margin errors, notice that the value α' given in the theorem satisfies $\mu - \rho - \alpha'(p-1)X^2/2 > 0$. We make the trivial estimate $\mathrm{Loss}_\rho(\textsc{Norma}) \geq 0$, which gives us

$$\mathrm{ME}_\rho(\textsc{Norma}) \leq \frac{K}{\mu - \rho - \alpha'(p-1)X^2/2} + \frac{B^2 + B(\sqrt{TD_2} + D_1)}{2\alpha'(\mu - \rho - \alpha'(p-1)X^2/2)} \ .$$

The bound follows by applying Lemma 2 with $\gamma = \mu - \rho$ and $z = \alpha'(p-1)X^2/2$.

As with Alma, we can get a mistake bound either by setting $\rho = 0$ in the margin error bound or doing a slightly different analysis that leads to a non-zero margin.

Theorem 4. *Let $X > 0$ and suppose that $||\boldsymbol{x}_t||_p \leq X$ for all t. Fix $K \geq 0$, $B > 0$, $D_1 \geq 0$ and $D_2 \geq 0$. Define C as in (7), and given $\mu > 0$ let $\alpha' = 2r/((p-1)X^2)$ where $r = h(\mu, K, C)$. Consider* Norma *with weight decay parameter as in (8), learning rate $\alpha = \alpha'/(1-\alpha'\lambda)$, and margin set to either $\rho = 0$ or $\rho = \mu - r$. Then for both of these margin settings, if there exists a comparison sequence $\boldsymbol{U} \in \mathcal{U}_q'(B, D_1, D_2)$ such that $\mathrm{Loss}_\mu(\boldsymbol{U}) \leq K$, we have*

$$\mathrm{Mist}(\textsc{Norma}) \leq \frac{K}{\mu} + \frac{2C}{\mu^2} + 2\left(\frac{C}{\mu^2}\right)^{1/2} \left(\frac{K}{\mu} + \frac{C}{\mu^2}\right)^{1/2} \ .$$

Proof of Theorem 4 from Theorem 3 is completely analogous with the proof of Theorem 2 from Theorem 1. We omit the details.

Comparing the bounds for the algorithms, we notice that the Norma bound has a term $\sqrt{TD_2}$ replacing D in the Alma bound. Suppose the parameters here have been chosen optimally: $D = \sum_{t=1}^T ||\boldsymbol{u}_t - \boldsymbol{u}_{t+1}||_q$ and $D_2 = \sum_{t=1}^T ||\boldsymbol{u}_t - \boldsymbol{u}_{t+1}||_q^2$. Then it is easy to see that $\sqrt{TD_2} \geq D$ always holds, with equality if the target speed is uniform ($||\boldsymbol{u}_t - \boldsymbol{u}_{t+1}||_q = ||\boldsymbol{u}_{t'} - \boldsymbol{u}_{t'+1}||_q$ for all t, t'). Thus, the bound for Norma gets worse if the target speed changes a lot. We believe that this may be due to our proof techniques, since the experiments reported in Section 5 do not show such differences between Alma and Norma.

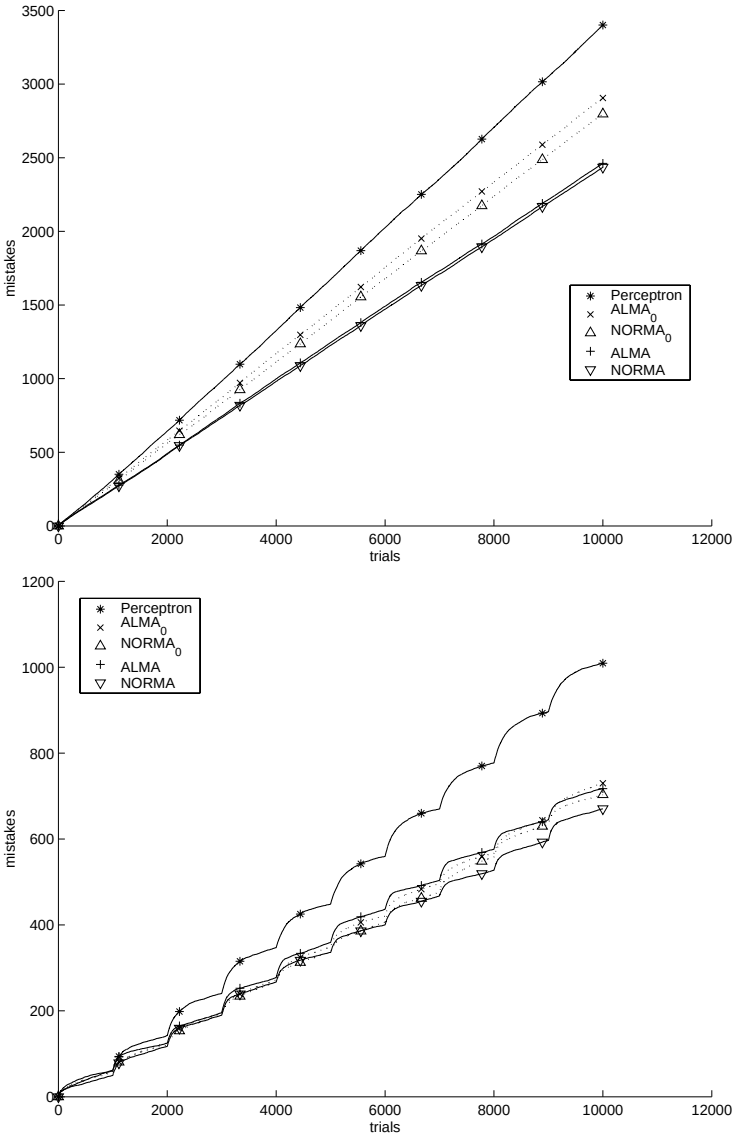

Fig. 1. Mistakes made by the algorithms on drifting (above) and switching (below) data

5 Experiments

The mistake bounds in Section 4 are of course only worst-case upper bounds, and even as such maybe not very tight. Hence, we have performed some preliminary experiments on artificial data to see qualitatively how the bounds relate to the actual performance of the algorithms. Our bounds would suggest that some

form of regularisation is useful when the target is moving, and forcing a positive margin may give an additional benefit. Further, the difference in the assumptions of Theorems 1 and 3 suggests that NORMA might not perform so well when the movement rate of the target varies a lot.

To generate the examples, we use one mixture of 2-dimensional Gaussians for the positive examples and another for negative ones. We remove all examples that would be misclassified by the Bayes-optimal classifier (which is based on the actual distribution known to us) or are close to its decision boundary. This gives us data that are cleanly separable using a Gaussian kernel. Target movement takes place as random changes in the parameters of the Gaussians. We use two movement schedules: In the *drifting* case, there is a relatively small parameter change after every ten trials. In the *switching* case, there is a very large parameter change after every 1000 trials. Thus, other things being equal, our bound for NORMA would be much better in the drifting than in the switching case. In either case, we ran each algorithm for 10000 trials and cumulatively summed up the mistakes made by them.

In our experiments we compare NORMA and ALMA with $p = 2$ and the basic Perceptron algorithm (which is the same as NORMA with the margin and weight decay parameters set to zero). We also consider variants NORMA_0 and ALMA_0 where we fix the margin to zero but keep the weight decay (or regularisation) parameter. We used Gaussian kernels to handle the non-linearity of the data. For these experiments, the parameters of the algorithms were tuned by hand optimally for each example distribution.

Figure 1 shows the cumulative mistake counts for the algorithms. There does not seem to be any decisive differences between the algorithms. In particular, NORMA seems to work quite well also on switching data. In general, it does seem that using a positive margin is better than fixing the margin to zero, and regularisation even with zero margin is better than the basic Perceptron algorithm.

Acknowledgments. This work was support by the Australian Research Council.

References

[1] P. Auer, N. Cesa-Bianchi and C. Gentile. Adaptive and self-confident on-line learning algorithms. Technical Report NC-TR-00-083, NeuroCOLT, 2000.

[2] P. Auer and M. K. Warmuth. Tracking the best disjunction. *Machine Learning*, 32(2):127–150, August 1998.

[3] P. Bartlett and J. Shawe-Taylor. Generalization performance of support vector machines and other pattern classifiers. In B. Schölkopf, C. J. C. Burges and A. J. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, pages 43–54. MIT Press, 1999.

[4] L. M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Physics*, 7:200–217, 1967.

[5] Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296, 1999.
[6] C. Gentile. A new approximate maximal margin classification algorithm. *Journal of Machine Learning Research*, 2:213–242, December 2001.
[7] C. Gentile and N. Littlestone. The robustness of the p-norm algorithms. In *Proc. 12th Annu. Conf. on Comput. Learning Theory*, pages 1–11. ACM Press, New York, NY, 1999.
[8] A. J. Grove, N. Littlestone and D. Schuurmans. General convergence results for linear discriminant updates. *Machine Learning*, 43(3):173–210, 2001.
[9] M. Herbster. Learning additive models online with fast evaluating kernels. In D. Helmbold and B. Williamson, editors, *Proc. 14th Annu. Conf. on Comput. Learning Theory*, pages 444–460. Springer LNAI 2111, Berlin, July 2001.
[10] M. Herbster and M. K. Warmuth. Tracking the best linear predictor. *Journal of Machine Learning Research*, 1:281–309, September 2001.
[11] J. Kivinen, A. J. Smola and R. C. Williamson. Online learning with kernels. In T. G. Dietterich, S. Becker and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 785–792. MIT Press, Cambridge, MA, 2002.
[12] Y. Li and P. M. Long. The relaxed online maximum margin algorithm. *Machine Learning*, 46(1):361–387, January 2002.
[13] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1988.
[14] C. Mesterharm. Tracking linear-threshold concepts with Winnow. In J. Kivinen and B. Sloan, editors, *Proc. 15th Annu. Conf. on Comput. Learning Theory*, pages 138–152. Springer LNAI 2375, Berlin, July 2002.
[15] A. B. J. Novikoff. On convergence proofs on perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata*, volume 12, pages 615–622. Polytechnic Institute of Brooklyn, 1962.