Approximate Maximum Margin Algorithms with Rules Controlled by the Number of Mistakes

Petroula Tsampouka PT04R@ECS.SOTON.AC.UK
School of Electronics and Computer Science, University of Southampton, UK

John Shawe-Taylor J.SHAWE-TAYLOR@CS.UCL.AC.UK
Department of Computer Science, University College London, UK

Abstract

We present a family of incremental Perceptron-like algorithms (PLAs) with margin in which both the "effective" learning rate, defined as the ratio of the learning rate to the length of the weight vector, and the misclassification condition are entirely controlled by rules involving (powers of) the number of mistakes. We examine the convergence of such algorithms in a finite number of steps and show that under some rather mild conditions there exists a limit of the parameters involved in which convergence leads to classification with maximum margin. An experimental comparison of algorithms belonging to this family with other large margin PLAs and decomposition SVMs is also presented.

1. Introduction

Maximising the margin of the solution hyperplane, which plays an important role in the generalisation ability of a learning machine, is a central objective of Support Vector Machines (SVMs) (Vapnik, 1998; Cristianini & Shawe-Taylor, 2000). Their efficient implementation, however, is somewhat hindered by the fact that they require solving a quadratic programming problem.

The ambition to surpass the implementational difficulties associated with SVMs while retaining all the benefits of the large margin solutions led to a revival of the interest in alternative large margin classifiers which are able to operate directly on the primal maximal margin problem instead of its dual. Such algorithms include the standard Perceptron with margin (Duda & Hart, 1973), the aggressive ROMMA (Li & Long, 2002) and ALMA$_2$ (Gentile, 2001) which are all variants of the classical Perceptron algorithm (Rosenblatt, 1958). Here we address the maximal margin classification problem in an incremental setting within the context of Perceptron-like algorithms (PLAs) which, however, differ from the above variants in that the "effective" learning rate (Tsampouka & Shawe-Taylor, 2006) and the misclassification condition do not depend on the length of the weight vector at all but, instead, are entirely controlled by rules involving (powers of) the number of mistakes. This novel (class of) algorithm(s) will be called Mistake-Controlled Rule Algorithm(s) (MICRA). Under certain conditions MICRA converges in a finite number of steps to an approximation of the optimal solution which keeps improving as the parameters of the algorithm follow a specific limiting process.

An introductory discussion of Perceptron-like large margin classifiers leading to the construction of MICRA can be found in Section 2. MICRA is described in Section 3 together with an analysis regarding its convergence. Section 4 contains some experiments while Section 5 our conclusions.

2. Perceptron-Like Large Margin Classifiers

In what follows we assume that we are given a training set which, even if initially not linearly separable can, by an appropriate feature mapping into a space of a higher dimension (Vapnik, 1998; Cristianini & Shawe-Taylor, 2000), be classified into two categories by a linear classifier. This higher dimensional feature space in which the patterns are linearly separable will be the considered space. By adding one additional dimension and placing all patterns in the same position at a dis-

Appearing in *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, 2007. Copyright 2007 by the author(s)/owner(s).

tance ρ in that dimension we construct an embedding of our data into the so-called augmented space (Duda & Hart, 1973). The advantage of this embedding is that the linear hypothesis in the augmented space becomes homogeneous. Throughout our discussion a reflection with respect to the origin in the augmented space of the negatively labelled patterns is assumed in order to allow for a uniform treatment of both categories of patterns. Also, $R \equiv \max_k \|\boldsymbol{y}_k\|$, with $\boldsymbol{y}_k$ the k^{th} augmented pattern. Obviously, $R \geq \rho$.

The relation characterising optimally correct classification of the training patterns $\boldsymbol{y}_k$ by a weight vector $\boldsymbol{u}$ of unit norm in the augmented space is

$$\boldsymbol{u} \cdot \boldsymbol{y}_k \geq \gamma_{\mathrm{d}} \equiv \max_{\boldsymbol{u'}:\|\boldsymbol{u'}\|=1} \min_i \{\boldsymbol{u'} \cdot \boldsymbol{y}_i\} \quad \forall k \ . \tag{1}$$

The quantity γ_{d} will be referred to as the maximum directional margin. It coincides with the maximum margin in the augmented space with respect to hyperplanes passing through the origin if no reflection is assumed. Between γ_{d} and the maximum geometric margin γ in the original space the inequality

$$1 \leq \gamma/\gamma_{\mathrm{d}} \leq R/\rho \tag{2}$$

holds. In the limit $\rho \to \infty$, $R/\rho \to 1$ and from (2) $\gamma_{\mathrm{d}} \to \gamma$ (Tsampouka & Shawe-Taylor, 2005).

We concentrate on algorithms that update the augmented weight vector $\boldsymbol{a}_t$ by adding a suitable positive amount in the direction of the misclassified (according to an appropriate condition) training pattern $\boldsymbol{y}_k$. The general form of such an update rule is

$$\boldsymbol{a}_{t+1} = (\boldsymbol{a}_t + \eta_t f_t \boldsymbol{y}_k)\, N_{t+1}^{-1} \ , \tag{3}$$

where η_t is the learning rate which could depend (usually explicitly) on the number t of updates that took place so far and f_t an implicit positive and bounded function of the current step (update) t, possibly involving $\boldsymbol{a}_t$ and/or $\boldsymbol{y}_k$. We also allow for a normalisation of $\boldsymbol{a}_{t+1}$ through a factor N_{t+1}. For the Perceptron $\eta_t = \eta$ is constant, $f_t = 1$ and $N_{t+1} = 1$. Each time the misclassification condition is satisfied by a training pattern, that is a mistake occurs, the algorithm proceeds to the update of $\boldsymbol{a}_t$. We adopt the convention of initialising t from 1.

A sufficiently general form of the misclassification condition is

$$\boldsymbol{u}_t \cdot \boldsymbol{y}_k \leq C(t) \ , \tag{4}$$

where $\boldsymbol{u}_t \equiv \boldsymbol{a}_t / \|\boldsymbol{a}_t\|$ and $C(t) > 0$ if we require that the algorithm achieves a positive margin. If $\boldsymbol{a}_1 = \boldsymbol{0}$ we treat the first pattern in the sequence as misclassified. In the case that $C(t)$ is bounded from above by a strictly decreasing function of t which tends to zero the minimum directional margin required by such a condition becomes lower than any fixed value provided t is large enough. Such algorithms have the advantage of achieving some fraction of the unknown margin provided they converge. An example is the Perceptron with margin where $C(t) = b/\|\boldsymbol{a}_t\|$ (b is a positive constant) is suppressed due to the growth of $\|\boldsymbol{a}_t\|$.

Another important quantity characterising algorithms with the perceptron-like update rule (3) is the "effective" learning rate

$$\eta_{\mathrm{eff}\,t} \equiv \eta_t R \|\boldsymbol{a}_t\|^{-1}$$

which controls the impact that an update has on the direction $\boldsymbol{u}_t$ of the current weight vector

$$\boldsymbol{u}_{t+1} = \frac{\boldsymbol{u}_t + \eta_{\mathrm{eff}\,t} f_t \boldsymbol{y}_k / R}{\|\boldsymbol{u}_t + \eta_{\mathrm{eff}\,t} f_t \boldsymbol{y}_k / R\|} \ . \tag{5}$$

In the most well-known cases $\eta_{\mathrm{eff}\,t}$ is bounded from above by a strictly decreasing function of t which tends to zero like in the case of the Perceptron where $\eta_t = \eta$ and $\eta_{\mathrm{eff}\,t}$ is suppressed due to the growth of $\|\boldsymbol{a}_t\|$.

From the above discussion it becomes obvious that a PLA with the additive update (3) is uniquely determined by the functions $C(t)$, $\eta_{\mathrm{eff}\,t}$ and f_t. In particular, it does not depend on $\|\boldsymbol{a}_t\|$ as long as the above functions are $\|\boldsymbol{a}_t\|$-independent. If this is the case the update (3) of $\boldsymbol{a}_t$ can be replaced by the update (5) of $\boldsymbol{u}_t$. Our purpose here is to examine the sufficiently large subclass of such algorithms with $f_t = 1$ and $C(t)$, $\eta_{\mathrm{eff}\,t}$ inversely proportional to powers of the number of mistakes t and determine sufficient conditions under which algorithms in the above subclass converge asymptotically to the optimal solution. The rather special case of a constant η_{eff} is the CRAMMA algorithm of (Tsampouka & Shawe-Taylor, 2006).

3. The Mistake-Controlled Rule Algorithm MICRA$^{\epsilon,\zeta}$

We consider algorithms having an update rule given by (5) with $f_t = 1$, an effective learning rate

$$\eta_{\mathrm{eff}\,t} = \eta t^{-\zeta} \tag{6}$$

and a misclassification condition

$$\boldsymbol{u}_t \cdot \boldsymbol{y}_k \leq \beta t^{-\epsilon} \ . \tag{7}$$

Here η, ζ, β and ϵ are positive constants. We assume that the initial value $\boldsymbol{u}_1$ of $\boldsymbol{u}_t$ is in the direction of the first pattern. Then,

$$\boldsymbol{u}_t \cdot \boldsymbol{u} > 0 \ . \tag{8}$$

This is true given that, on account of (5), $\boldsymbol{u}_t$ is a linear combination with positive coefficients of the $\boldsymbol{y}_k$'s all of which satisfy $\boldsymbol{y}_k \cdot \boldsymbol{u} > 0$ because of (1). The above (family of) algorithm(s) parametrised in terms of the exponents ϵ and ζ will be called the Mistake-Controlled Rule Algorithm(s) MICRA$^{\epsilon,\zeta}$.

Theorem 1 *The* MICRA$^{\epsilon,\zeta}$ *algorithm converges in a finite number of steps provided* $\zeta \leq 1$. *Moreover, if* η *is given a dependence on* β *through the relation* $\eta = \eta_0 \left(\beta/R\right)^{-\delta}$ *the directional margin* γ'_{d} *that the algorithm achieves tends in the limit* $\beta/R \to \infty$ *to the maximum directional margin* γ_{d} *provided* $0 < \epsilon\delta + \zeta < 1$.

Proof Taking the inner product of (5) with the optimal direction $\boldsymbol{u}$, expanding $\left\|\boldsymbol{u}_t + \eta_{\mathrm{eff}\,t}\boldsymbol{y}_k/R\right\|^{-1}$ and using the inequality $(1+x)^{-\frac{1}{2}} \geq 1 - x/2$ we have

$$\boldsymbol{u}_{t+1} \cdot \boldsymbol{u} \geq \\ \left(\boldsymbol{u}_t \cdot \boldsymbol{u} + \eta_{\mathrm{eff}\,t}\frac{\boldsymbol{y}_k \cdot \boldsymbol{u}}{R}\right)\left(1 - \eta_{\mathrm{eff}\,t}\frac{\boldsymbol{y}_k \cdot \boldsymbol{u}_t}{R} - \eta^2_{\mathrm{eff}\,t}\frac{\left\|\boldsymbol{y}_k\right\|^2}{2R^2}\right).$$

Thus, we obtain for $\mathcal{D} \equiv \boldsymbol{u}_{t+1} \cdot \boldsymbol{u} - \boldsymbol{u}_t \cdot \boldsymbol{u}$

$$\frac{R}{\eta_{\mathrm{eff}\,t}}\mathcal{D} \geq \boldsymbol{y}_k \cdot \boldsymbol{u} - (\boldsymbol{u}_t \cdot \boldsymbol{u})(\boldsymbol{y}_k \cdot \boldsymbol{u}_t) - \eta_{\mathrm{eff}\,t}\Big(\boldsymbol{u}_t \cdot \boldsymbol{u}\left\|\boldsymbol{y}_k\right\|^2 \\ + 2(\boldsymbol{y}_k \cdot \boldsymbol{u})(\boldsymbol{y}_k \cdot \boldsymbol{u}_t)\Big)/2R - \eta^2_{\mathrm{eff}\,t}\left\|\boldsymbol{y}_k\right\|^2 \boldsymbol{y}_k \cdot \boldsymbol{u}/2R^2.$$

Then, by employing (1), (7) and (8) we get

$$\mathcal{D} \geq \eta_{\mathrm{eff}\,t}\left(\frac{\gamma_{\mathrm{d}}}{R} - \frac{\eta_{\mathrm{eff}\,t}}{2} - \frac{\eta^2_{\mathrm{eff}\,t}}{2}\right) - \eta_{\mathrm{eff}\,t}\left(1 + \eta_{\mathrm{eff}\,t}\right)\frac{\beta}{R}t^{-\epsilon} \quad . \tag{9}$$

From (7) it is obvious that convergence of the algorithm is impossible as long as $\beta t^{-\epsilon} > \gamma_{\mathrm{d}}$. Thus, we may assume that $t > t_0 \equiv \left(\beta/\gamma_{\mathrm{d}}\right)^{\frac{1}{\epsilon}}$. A repeated application of (9) $t - [t_0]$ times, where $[t_0]$ denotes the integer part of t_0, yields

$$\boldsymbol{u}_{t+1} \cdot \boldsymbol{u} - \boldsymbol{u}_{[t_0]+1} \cdot \boldsymbol{u} \geq \eta\frac{\gamma_{\mathrm{d}}}{R}\sum_{m=[t_0]+1}^{t} m^{-\zeta} \\ -\frac{\eta^2}{2}\sum_{m=[t_0]+1}^{t} m^{-2\zeta} - \frac{\eta^3}{2}\sum_{m=[t_0]+1}^{t} m^{-3\zeta} \\ -\eta\frac{\beta}{R}\sum_{m=[t_0]+1}^{t} m^{-(\zeta+\epsilon)} - \eta^2\frac{\beta}{R}\sum_{m=[t_0]+1}^{t} m^{-(2\zeta+\epsilon)}.$$

Noticing that $1 \geq \boldsymbol{u}_{t+1} \cdot \boldsymbol{u} - \boldsymbol{u}_{[t_0]+1} \cdot \boldsymbol{u}$ because of (8), employing

$$\int_{t_0+1}^{t} m^{-\theta}dm \leq \sum_{m=[t_0]+1}^{t} m^{-\theta} \leq \int_{t_0}^{t} m^{-\theta}dm + t_0^{-\theta}$$

for $\theta > 0$ and introducing $\tau \geq 0$ through the relation

$$t = t_0\left(1 + \tau\right) = \left(\beta/\gamma_{\mathrm{d}}\right)^{\frac{1}{\epsilon}}\left(1 + \tau\right) \quad , \tag{10}$$

we finally obtain

$$\left(\eta t_0^{1-\zeta}\right)^{-1}\left(\frac{\gamma_{\mathrm{d}}}{R}\right)^{-1}(1 + \omega) \geq g(\tau) \equiv \frac{\left(1+\tau\right)^{1-\zeta} - 1}{1 - \zeta} \\ -\frac{\left(1+\tau\right)^{1-(\zeta+\epsilon)} - 1}{1 - (\zeta + \epsilon)} - \frac{R}{2\gamma_{\mathrm{d}}}\eta t_0^{-\zeta}\frac{\left(1+\tau\right)^{1-2\zeta} - 1}{1 - 2\zeta} \\ -\frac{R}{2\gamma_{\mathrm{d}}}\eta^2 t_0^{-2\zeta}\frac{\left(1+\tau\right)^{1-3\zeta} - 1}{1 - 3\zeta} - \eta t_0^{-\zeta}\frac{\left(1+\tau\right)^{1-(2\zeta+\epsilon)} - 1}{1 - (2\zeta + \epsilon)}. \tag{11}$$

Here

$$\omega \equiv \frac{\gamma_{\mathrm{d}}}{R}\eta t_0^{-\zeta}\left(2 + \eta t_0^{-\zeta}\right) + \frac{1}{2}\eta^2 t_0^{-2\zeta}\left(1 + \eta t_0^{-\zeta}\right) > 0 \quad .$$

For $\zeta = 1$ the first term of $g(\tau)$ becomes $\ln(1+\tau)$. Since $0 < \zeta \leq 1$, $g(\tau)$ (with $\tau \geq 0$) is unbounded from above. Moreover, its derivative $g'(\tau)$ satisfies

$$(1+\tau)^{\zeta}g'(\tau) = 1 - (1+\tau)^{-\epsilon} - \frac{R}{2\gamma_{\mathrm{d}}}\eta t_0^{-\zeta}(1+\tau)^{-\zeta} \\ -\frac{R}{2\gamma_{\mathrm{d}}}\eta^2 t_0^{-2\zeta}(1+\tau)^{-2\zeta} - \eta t_0^{-\zeta}(1+\tau)^{-(\zeta+\epsilon)} \quad .$$

The r.h.s. of the above equation is a monotonically increasing function of τ which is negative at $\tau = 0$ and tends to 1 as $\tau \to \infty$. Therefore $g'(\tau)$ has a single root at $\tau = \tau_{\min}$ which corresponds to a minimum of $g(\tau)$ with $g(\tau_{\min}) < 0$. Moreover, the l.h.s. of (11) is positive. Thus, given that $g(0) = 0$, there is a single value τ_{b} of τ where (11) holds as an equality which provides an upper bound on τ satisfying $\tau_{\mathrm{b}} > \tau_{\min} > 0$. Then, from (10) using $\tau \leq \tau_{\mathrm{b}}$ we obtain the upper bound on the number of updates

$$t \leq t_{\mathrm{b}} \equiv \left(\beta/\gamma_{\mathrm{d}}\right)^{\frac{1}{\epsilon}}\left(1 + \tau_{\mathrm{b}}\right) \tag{12}$$

proving that the algorithm converges in a finite number of steps. From (12) and taking into account (7) the margin γ'_{d} achieved satisfies $\gamma'_{\mathrm{d}} \geq \beta(t_{\mathrm{b}} + 1)^{-\epsilon}$. Thus, for the fraction $f \equiv \gamma'_{\mathrm{d}}/\gamma_{\mathrm{d}}$ of γ_{d} achieved we have

$$1 \geq f \geq f_{\mathrm{b}} \equiv (\beta/\gamma_{\mathrm{d}})(t_{\mathrm{b}} + 1)^{-\epsilon} = \left(1 + \tau_{\mathrm{b}} + t_0^{-1}\right)^{-\epsilon} \quad . \tag{13}$$

Let us assume that $\beta/R \to \infty$ in which case from $\eta = \eta_0\left(\beta/R\right)^{-\delta}$ and given that $0 < \epsilon\delta + \zeta < 1$ we have $\eta t_0^{1-\zeta} \sim \left(\beta/R\right)^{\frac{1-\zeta-\epsilon\delta}{\epsilon}} \to \infty$ whereas $\eta t_0^{-\zeta} \sim \left(\beta/R\right)^{-\frac{\zeta+\epsilon\delta}{\epsilon}} \to 0$. Consequently, the l.h.s. of (11) vanishes in the limit $\beta/R \to \infty$ whereas $g(\tau)$ becomes a strictly increasing function for $\tau > 0$ (i.e. $\tau_{\min} \to 0$)

since $(1+\tau)^{\zeta} g'(\tau) = 1 - (1+\tau)^{-\epsilon} > 0$. Obviously, (11) holds as an equality only for $\tau = 0$. Therefore,

$$\tau_{\rm b} \to \tau_{\rm min} \to 0 \quad \text{as} \quad \beta/{\rm R} \to \infty \ . \qquad (14)$$

Combining (13) with (14) and noticing that $t_0^{-1} \to 0$ as $\beta/R \to \infty$ we conclude that $f \to 1$ or

$$\gamma'_{\rm d} \to \gamma_{\rm d} \quad \text{as} \quad \beta/{\rm R} \to \infty \ . \qquad \Box$$

Remark 1 In the case that $\zeta + 2\epsilon = 1$ with $\zeta > 1/2$ we may obtain explicitly an upper bound $t_{\rm b}$ on the number of updates and a lower bound $f_{\rm b}$ on the fraction f of the margin that the algorithm achieves. First we observe that since $1-2\zeta$, $1-3\zeta$ and $1-(2\zeta+\epsilon)$ are negative it is allowed to set the terms $(1+\tau)^{1-2\zeta}$, $(1+\tau)^{1-3\zeta}$ and $(1+\tau)^{1-(2\zeta+\epsilon)}$ to zero in the r.h.s. of (11). Then, the resulting inequality with $\zeta = 1-2\epsilon$ becomes

$$A^2 \geq \left((1+\tau)^{\epsilon} - 1\right)^2 \qquad (15)$$

where

$$A^2 = \frac{2\epsilon}{\eta}\left(\frac{R\gamma_{\rm d}}{\beta^2}\right)(1+\omega) + \frac{\epsilon\eta}{1-4\epsilon}\left(\frac{R}{\beta}\right)\left(\frac{\gamma_{\rm d}}{\beta}\right)^{\frac{1}{\epsilon}-3} + \frac{\epsilon\eta^2}{2-6\epsilon}\left(\frac{R}{\beta}\right)\left(\frac{\gamma_{\rm d}}{\beta}\right)^{\frac{2}{\epsilon}-5} + \frac{2\epsilon\eta}{1-3\epsilon}\left(\frac{\gamma_{\rm d}}{\beta}\right)^{\frac{1}{\epsilon}-2} .$$

Notice that $\epsilon < 1/4$ if $\zeta > 1/2$. By solving (15) as an equation we obtain explicitly the bounds $t_{\rm b}$ and $f_{\rm b}$. They are the ones of (12) and (13), respectively with

$$\tau_{\rm b} = (1+|A|)^{\frac{1}{\epsilon}} - 1 \ .$$

Here $0 < \epsilon\delta + \zeta < 1$ is equivalent to $2 - \frac{1}{\epsilon} < \delta < 2$. Then, with $\eta = \eta_0 \left(\beta/R\right)^{-\delta}$ as $\beta/R \to \infty$ we get $|A| \to 0$ leading to $\tau_{\rm b} \to 0$. This demonstrates explicitly the statement of Theorem 1. It is worth emphasising, however, that $|A|$ may be small even if β/R is not large if $\gamma_{\rm d}/R$ and ϵ are sufficiently small.

Example 1 If $\epsilon = \zeta = 1/2$ and moreover $\delta = 0$, i.e. η is β-independent, $\epsilon\delta + \zeta = 1/2$ and the condition of Theorem 1 is satisfied. Therefore, such an algorithm attains asymptotically as $\beta/R \to \infty$ the maximum directional margin. The above algorithm is a version of ALMA$_2$ in which the weight vector instead of being confined within a ball centered at the origin is normalised to a constant length which remains fixed during the asymptotic procedure. Thus, ALMA$_2$ can be thought of as belonging to the MICRA family. Then, the analysis of (Gentile, 2001) confirms our conclusion regarding asymptotic convergence to the optimal solution hyperplane in this special case. In the case, instead, that $\epsilon = \zeta = 1/2$ but $\delta = 1$, i.e. $\eta = \eta_0 \left(\beta/R\right)^{-1}$, $\epsilon\delta + \zeta = 1$ and the condition of Theorem 1 is violated. This case would correspond to a version of ALMA$_2$ with the function $C(t)$ entering the misclassification condition (4) given by $C(t) = \beta^2/\left(\|\boldsymbol{a}_t\|\sqrt{t}\right)$ and the weight vector normalised to the constant length β which, however, does not remain fixed during the asymptotic procedure $\beta/R \to \infty$. Since the condition of Theorem 1 is violated we are unable to prove asymptotic convergence of such an algorithm to the maximal margin solution. The same conclusion is reached if the technique of (Gentile, 2001) is employed which gives the lower bound $f_{\rm b} = \left(1 + \eta_0{}^{-1} + 2\eta_0 (R/\beta)^2\right)^{-1}$ on the fraction of $\gamma_{\rm d}$ achieved. As $\beta/R \to \infty$ we get $f_{\rm b} \to \eta_0/(1+\eta_0) < 1$. We see that a "slight" modification of the asymptotic procedure is able to affect the ability of a PLA to attain the solution with maximum margin. We believe that the inability in some cases of the Perceptron algorithm with margin, in contrast to ALMA$_2$, to approach the maximal margin solution is due to such "slight" differences between the two algorithms regarding the asymptotic procedure.

Efficient Implementation: A completely equivalent formulation of MICRA is obtained if the update rule (3) with $f_t = N_{t+1} = 1$ and $\eta_t = \|\boldsymbol{a}_t\| \eta_{{\rm eff}\,t}/R$ is employed and the misclassification condition (7) is reexpressed as $\boldsymbol{a}_t \cdot \boldsymbol{y}_k \leq \|\boldsymbol{a}_t\| \beta t^{-\epsilon}$. Such a formulation apart from bearing a close resemblance to the Perceptron algorithm has the additional advantage of being computationally more efficient. A pseudocode implementing this formulation is given below.

Algorithm 1 MICRA$^{\epsilon,\zeta}$

```
    Input: A linearly separable augmented set with
    reflection assumed $S = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_k, \ldots, \boldsymbol{y}_m)$
    Fix: $\eta$, $\beta$
    Define: $R = \max_k \|\boldsymbol{y}_k\|$, $q_k = \|\boldsymbol{y}_k\|^2$, $\bar{\eta} = \eta/R$
    Initialise: $t = 1$, $\boldsymbol{a}_1 = \boldsymbol{y}_1$, $\|\boldsymbol{a}_1\| = \|\boldsymbol{y}_1\|$,
    $\eta_1 = \|\boldsymbol{a}_1\| \bar{\eta}$, $\beta_1 = \|\boldsymbol{a}_1\| \beta$
    repeat
      for $k = 1$ to $m$ do
        $p_{tk} = \boldsymbol{a}_t \cdot \boldsymbol{y}_k$
        if $p_{tk} \leq \beta_t$ then
          $\boldsymbol{a}_{t+1} = \boldsymbol{a}_t + \eta_t \boldsymbol{y}_k$
          $\|\boldsymbol{a}_{t+1}\| = \sqrt{\|\boldsymbol{a}_t\|^2 + \eta_t (2p_{tk} + \eta_t q_k)}$
          $t \leftarrow t + 1$
          $\eta_t = \|\boldsymbol{a}_t\| \bar{\eta} t^{-\zeta}$, $\beta_t = \|\boldsymbol{a}_t\| \beta t^{-\epsilon}$
        end if
      end for
    until no update made within the for loop
```

In order to further reduce the computational cost we may form a reduced "active set" of patterns consisting of the ones found misclassified during each epoch

Table 1. Results for the sonar dataset. The directional margin γ'_{d} achieved and the number of updates (upds) are given for the Perceptron, agg-ROMMA and MICRA$^{0.05,0.9}$. For MICRA we choose $\eta = 50$.

Perceptron			agg-ROMMA			MICRA$^{0.05,0.9}$		
$\frac{b}{\eta R^2}$	$10^3\gamma'_{\mathrm{d}}$	upds	δ	$10^3\gamma'_{\mathrm{d}}$	upds	$10^3\frac{\beta}{R}$	$10^3\gamma'_{\mathrm{d}}$	upds
3.9	7.27	820,261	0.2	7.28	778,412	3.59	7.29	327,468
30	7.85	5,930,214	0.1	7.85	1,546,595	4.04	7.86	706,274
100	7.91	19,599,882	0.05	8.19	2,716,711	4.43	8.19	1,932,165
500	7.93	97,717,549	0.01	8.37	14,079,715	4.95	8.37	11,610,899

Table 2. The number of updates (upds) required to achieve $\gamma'_{\mathrm{d}} \simeq 0.00819$ in the sonar dataset with MICRA and ALMA$_2$. For MICRA various ϵ, ζ values are considered and the η employed is given.

ϵ, ζ	0.005, 0.99	0.05, 0.9	0.1, 0.8	0.15, 0.7	0.2, 0.6	0.2, 0.5	0.5, 0.5	ALMA$_2$
η	190	60	17	4.4	1.2	0.28	0.35	
upds/10^6	1.53	1.86	2.32	2.89	3.57	3.74	7.54	53.4

which are then cyclically presented to the algorithm for N mini-epochs unless no update occurs during a mini-epoch. Subsequently, a new full epoch involving all the patterns takes place giving rise to a new active set. The algorithm terminates only if no mistake occurs during a full epoch. This procedure clearly amounts to a different way of sequentially presenting the patterns to the algorithm and does not affect the applicability of Theorem 1. The MICRA algorithm incorporating the above procedure will be referred to as the "reduced" MICRA or red-MICRA.

4. Experiments

A comparison of MICRA with other classifiers will rely on their ability to achieve fast convergence to a certain approximation of the "optimal" hyperplane in the feature space where the patterns are linearly separable. Although it is straightforward to formulate MICRA in dual space we will treat it here, unless otherwise specified, as a primal space algorithm. For linearly separable data our feature space will be the initial instance space. For linearly inseparable data, instead, a space extended by as many dimensions as the instances will be considered where each instance is placed at a distance Δ from the origin in the corresponding dimension. The justification for this construction relies on the well-known fact that the hard margin optimisation in this extended space is equivalent to the soft margin optimisation in the original instance space with objective function $\|\boldsymbol{w}\|^2 + \Delta^{-2}\sum_i {\xi_i}^2$ involving the weight vector $\boldsymbol{w}$ and the 2-norm of the slacks ξ_i (Cristianini & Shawe-Taylor, 2000). To obtain a meaningful comparison we follow the above approach, i.e. linear kernels and 2-norm soft margin, for both PLAs and SVMs.

Comparison with PLAs: We begin with experiments on several UCI datasets aiming at verifying our analysis and evaluating the performance of MICRA relative to the Perceptron with margin and aggressive ROMMA. For MICRA we use a β-independent η ($\delta = 0$) and ϵ, ζ values for which, in most cases, the analysis of Remark 1 applies. Our goal in this comparison involving only PLAs will be to obtain a given value of the margin in as few updates as possible.

First we analyse the training dataset of the sonar classification problem (104 instances, 60 attributes) as originally selected for the aspect-angle dependent experiment. Here the augmented space parameter is set to the value $\rho = 1$ leading to $R \simeq 3.8121$ and $\gamma_{\mathrm{d}} \simeq 0.00841$. The results of our comparative study of the Perceptron, agg-ROMMA[1] and MICRA$^{0.05,0.9}$ algorithms are presented in Table 1. We observe that MICRA is certainly the fastest. Moreover, the Perceptron does not seem able to approach the maximum margin arbitrarily close. We also present in Table 2 the number of updates required to achieve a margin $\gamma'_{\mathrm{d}} \simeq 0.00819$ using MICRA with several ϵ, ζ values and ALMA$_2$. For ALMA$_2$ the accuracy parameter α was set to $\alpha = 0.1527$ with the remaining parameters chosen to correspond to the ones of the Theorem in (Gentile, 2001) if the data are normalised such that the longest pattern has unit length. From Table 2 it becomes clear that small ϵ's combined with relatively large ζ's lead to faster convergence.

We additionally analyse the linearly separable dataset WBC$_{-11}$ (672 instances, 9 attributes). It is con-

[1] The parameter $\delta \in (0, 1]$ in agg-ROMMA controls the accuracy to which the maximum margin is approximated. It should not be confused with δ in Theorem 1.

Table 3. Results for the WBC_{-11} dataset. For MICRA the choice $\eta = 2.3$ is made.

Perceptron			agg-ROMMA			$\mathrm{MICRA}^{0.1,0.8}$		
$\frac{b}{\eta R^2}$	$10^2\gamma'_{\mathrm{d}}$	upds	δ	$10^2\gamma'_{\mathrm{d}}$	upds	$10^3\frac{\beta}{R}$	$10^2\gamma'_{\mathrm{d}}$	upds
1.8	2.197	4,980,423	0.2	2.195	5,784,868	1.85	2.198	267,145
4.1	2.321	10,761,773	0.1	2.318	13,931,792	2.07	2.324	467,369
45	2.415	113,406,210	0.01	2.415	174,388,827	2.70	2.415	4,533,155

Table 4. Results for the WBC dataset (extended with $\Delta = 1$). For MICRA we choose $\eta = 20$.

Perceptron			agg-ROMMA			$\mathrm{MICRA}^{0.05,0.9}$		
$\frac{b}{\eta R^2}$	$10^2\gamma'_{\mathrm{d}}$	upds	δ	$10^2\gamma'_{\mathrm{d}}$	upds	$10^3\frac{\beta}{R}$	$10^2\gamma'_{\mathrm{d}}$	upds
3.5	11.905	206,469	0.1	11.916	169,588	7.02	11.957	105,964
8.1	12.462	457,334	0.05	12.468	409,956	7.54	12.470	183,643
700	12.837	38,336,601	0.01	12.928	1,554,492	8.40	12.949	734,629

structed from the Wisconsin Breast Cancer (WBC) dataset by first omitting the 16 instances with missing attributes and subsequently removing from the dataset containing the remaining 683 instances the 11 instances having the positions 2, 4, 191, 217, 227, 245, 252, 286, 307, 420 and 475. The value $\rho = 30$ is chosen for the augmented space parameter ρ leading to $R = \sqrt{1716}$ and $\gamma_{\mathrm{d}} \simeq 0.0243$. The results of our comparative study of the Perceptron, agg-ROMMA and $\mathrm{MICRA}^{0.1,0.8}$ are presented in Table 3. The superiority of the performance of MICRA is remarkable.

Finally, we turn to the linearly inseparable full WBC dataset which, after ignoring the 16 instances with missing attributes, has 683 instances each with 9 attributes. For the extended space parameter Δ and the augmented space parameter ρ we choose the values $\Delta = 1$ and $\rho = 10$, respectively. This leads to $R = \sqrt{917}$ and to a maximum margin $\gamma_{\mathrm{d}} \simeq 0.13033$ with respect to zero-threshold hyperplanes in the extended (and augmented) space. Table 4 contains the results of our comparative study. We observe that the Perceptron shows again some difficulty in approaching γ_{d} and that once again MICRA is the fastest.

To conclude our comparative study of PLAs we point out that, from the experiments of (Tsampouka & Shawe-Taylor, 2006) on the same datasets, MICRA with $\epsilon \ll 1$ and $\zeta \simeq 1$ is much faster than CRAMMA.

Comparison with SVMs: A comparison of MICRA with SVMs, unlike PLAs, could only involve the CPU-time required to achieve a certain approximation of the hyperplane giving rise to the maximum geometric margin γ in the feature space where the patterns are linearly separable. PLAs like MICRA become extremely slow in the vicinity of the maximum directional margin γ_{d} which is attainable only asymptotically. Moreover, γ_{d} approaches γ only in the limit $\rho \to \infty$. As a consequence, MICRA could converge faster than SVMs only to a solution with geometric margin γ' slightly lower than γ. We choose to compare red-MICRA with SVMs at a margin larger than 99% of γ.

In our experiments SVMs are represented by algorithms based on decomposition methods which are many orders of magnitude faster than standard SVMs. More specifically, red-MICRA is compared with LIBSVM (Chang & Lin, 2001), an improved version of SMO (Platt, 1998), and SVM^{light} (Joachims, 1999). For both algorithms we choose $m = 400$MB for the memory parameter and $C = 10^5$ (approximating $C = \infty$) for the 1-norm soft margin parameter since we are dealing with a hard margin problem in the appropriate feature space. Also, the working set size parameter q of SVM^{light} is fixed to the default value $q = 10$. For each dataset we obtain values of the geometric margin γ' corresponding to two different values of the accuracy parameter ϵ both for LIBSVM and SVM^{light}. The larger value of the margin obtained by these algorithms corresponds to $\epsilon = 0.001$ and is regarded as a good approximation to the maximum geometric margin γ. We require that the margin γ' achieved by red-MICRA be larger than 99% of the larger margins (corresponding to $\epsilon = 0.001$) and larger than the lower margins (corresponding to $\epsilon > 0.001$) obtained by both LIBSVM and SVM^{light}. Unless otherwise specified we take advantage of the sparsity in the attributes of the initial space only if these attributes are binary. Also, we always exploit the enormous sparsity in the attributes associated with the additional dimensions of the extended instance space. The experiments were conducted on a 1.8 GHz Intel Pentium M processor with 504 MB RAM running Windows XP. The codes written in C++ were run using Microsoft's

Table 5. Results of a comparative study of LIBSVM, SVM^{light} and red-MICRA on several UCI datasets.

data set	Δ	LIBSVM				SVM^{light}				$\mathrm{red-MICRA}^{0.05,0.9}$					
		$10^2\gamma'$	Secs	$10^2\gamma'$	Secs	$10^2\gamma'$	Secs	$10^2\gamma'$	Secs	ρ	η	N	$10^5\frac{\beta}{R}$	$10^2\gamma'$	Secs
sonar	0	0.8451	0.17	0.8405	$\boxed{0.10}$	0.8460	6.85	0.8388	4.84	1	45	80	462.2	0.8406	$3.60^\star$
ionosphere	1	10.554	0.06	10.389	$\boxed{0.05}$	10.551	0.30	10.448	0.19	1.5	10	10	2929	10.449	0.07
votes	1	16.846	0.02	16.708	$\boxed{0.02}$	16.841	0.18	16.690	0.11	1	5	20	6385	16.718	$\boxed{0.02}$
WBC	1	13.034	0.12	12.848	$\boxed{0.09}$	13.033	0.81	12.929	0.45	2	25	20	837.6	12.932	0.35
tic-tac-toe	1	10.300	0.47	10.183	0.27	10.295	3.35	10.185	1.35	0.5	8	20	5334	10.203	$\boxed{0.05}$
german	25	95.361	0.62	94.055	0.45	95.332	2.96	94.217	1.82	8	30	50	908.9	94.415	$\boxed{0.36}$
mushroom	0	36.551	0.58	35.988	0.33	36.538	0.17	36.103	0.11	0	4.5	50	12535	36.212	$\boxed{0.10}$

Table 6. Results of a comparative study of LIBSVM, SVM^{light} and red-MICRA on several subsets of the Adult dataset.

subset size	LIBSVM				SVM^{light}				$\mathrm{red-MICRA}^{0.05,0.9}$				
	$10^2\gamma'$	Secs	$10^2\gamma'$	Secs	$10^2\gamma'$	Secs	$10^2\gamma'$	Secs	η	N	$10^2\frac{\beta}{R}$	$10^2\gamma'$	Secs
1605	3.9383	1.41	3.9022	1.07	3.9375	3.02	3.8877	1.58	20	100	1.918	3.9038	$\boxed{0.63}$
3185	2.7437	5.55	2.7187	4.29	2.7434	11.3	2.7093	6.23	25	100	1.400	2.7187	$\boxed{1.73}$
6414	1.9292	22.5	1.9094	17.6	1.9290	71.3	1.9097	37.7	45	300	1.025	1.9111	$\boxed{5.83}$
11220	1.4499	73.2	1.4348	58.6	1.4497	283.4	1.4342	141.7	65	300	0.798	1.4356	$\boxed{14.7}$
16100	1.2069	389.7	1.1927	312.3	1.2068	638.2	1.1923	318.6	80	500	0.673	1.1950	$\boxed{28.7}$
32561	0.8526	3902.3	0.8424	2484.5	0.8525	2733.8	0.8432	1439.4	105	600	0.492	0.8441	$\boxed{75.0}$

Visual C++ 5.0 compiler.

Table 5 contains the results of our comparative study of LIBSVM, SVM^{light} and red-MICRA on several UCI datasets with I/O excluded from the CPU-times reported. The value of the accuracy parameter ϵ corresponding to the lower value of the margin is set to $\epsilon = 0.03$ for LIBSVM and $\epsilon = 0.015$ for SVM^{light}. The sonar and WBC datasets are described already. The ionosphere dataset consists of 351 instances each with 34 attributes. The House votes dataset consists of 435 instances each with 16 attributes taking values from the set $\{\mathrm{y}, \mathrm{n}, ?\}$ represented here as $\{1, -1, 0\}$. The tic-tac-toe dataset consists of 958 instances each with 9 attributes taking values from the set $\{\mathrm{x}, \mathrm{o}, \mathrm{b}\}$ represented as $\{1, -1, 0\}$. The german dataset consists of 1000 instances each with 24 attributes. Finally, the linearly separable mushroom dataset consists of 5644 instances after removing the ones with missing attributes. Each instance has 22 categorical attributes replaced here by 125 binary ones out of which exactly 22 are true. We believe that from Table 5 it is fair to conclude that, roughly speaking, red-MICRA is of speed comparable to that of decomposition SVMs.

We also analysed several subsets of the Adult (32561 instances, 123 binary attributes) and of the Web (49749 instances, 300 binary attributes) datasets in the version of (Platt, 1998) with results presented in Tables 6 and 7, respectively. Here $\Delta = 1$. Also, in both tables the lower value of the margin for LIBSVM corresponds to $\epsilon = 0.03$. For the Adult dataset no augmentation is required and the lower value of the margin for SVM^{light} corresponds to $\epsilon = 0.025$. For the Web dataset, instead, we do perform an augmentation for red-MICRA with parameter $\rho = 0.25$. Also, the lower value of the margin for SVM^{light} in Table 7 is obtained with $\epsilon = 0.02$. We observe that the CPU-time required for red-MICRA to converge is shorter and exhibits a better scaling behaviour with the size of the dataset. Moreover, the shortage of memory as the dataset size grows apparently slows down LIBSVM. In contrast, SVM^{light} and red-MICRA are not affected.

Finally, we conducted an experiment with the multiclass Covertype dataset (581012 instances, 54 attributes) obtainable from UCI and studied the binary classification problem of the first class versus all the others using again the whole dataset for training. Due to the memory difficulties encountered by LIBSVM we compared red-MICRA only with SVM^{light} for which we obtained only one margin value corresponding to an accuracy parameter $\epsilon = 0.01$. Such a value of ϵ is sufficiently small to guarantee a margin γ' larger than 0.99γ. The dataset was rescaled by multiplying all the

*Value obtained using the dual space formulation.

Table 7. Results of a comparative study of LIBSVM, SVM^{light} and red-MICRA on several subsets of the Web dataset.

subset size	LIBSVM				SVM^{light}				red − MICRA$^{0.05,0.9}$				
	$10^2\gamma'$	Secs	$10^2\gamma'$	Secs	$10^2\gamma'$	Secs	$10^2\gamma'$	Secs	η	N	$10^2\frac{\beta}{R}$	$10^2\gamma'$	Secs
2477	10.448	0.57	10.292	0.51	10.445	0.30	10.312	0.18	25	10	1.681	10.344	$\boxed{0.07}$
4912	7.0079	2.07	6.8967	1.83	7.0067	1.10	6.8909	0.61	25	10	1.212	6.9393	$\boxed{0.20}$
9888	4.8784	8.95	4.7970	7.82	4.8772	5.45	4.8072	3.22	30	10	0.868	4.8316	$\boxed{0.86}$
24692	2.9555	115.5	2.9066	90.2	2.9549	66.9	2.9111	32.1	50	10	0.535	2.9265	$\boxed{4.82}$
49749	2.1094	725.0	2.0723	635.8	2.1089	360.2	2.0771	176.4	70	10	0.405	2.0894	$\boxed{18.3}$

Table 8. Results of a comparative study of SVM^{light} and red-MICRA on the Covertype dataset.

data size	SVM^{light}		red − MICRA$^{0.05,0.9}$				
	$10^3\gamma'$	Secs	η	N	$10^5\frac{\beta}{R}$	$10^3\gamma'$	Secs
581012	15.774	47987.7	70	400	336	15.789	$\boxed{4728.0}$

attributes with 0.001 and their sparsity was fully exploited. Moreover, for the rescaled data the parameter values $\Delta = 10$ and $\rho = 2$ were chosen. From the results desplayed in Table 8 red-MICRA appears about 10 times faster.

Very recently SVM-Perf, a cutting-plane algorithm for training linear SVMs, was presented and empirically proved much faster than SVM^{light} (Joachims, 2006). From the results reported, however, no direct meaningful comparison with red-MICRA is possible since SVM-Perf implements the 1-norm soft margin.

5. Conclusions

We presented MICRA, a family of Perceptron-like large margin classifiers completely independent of the length of the weight vector. Our theoretical approach proved sufficiently powerful in establishing asymptotic convergence to the optimal hyperplane for a whole class of such algorithms in which the misclassification condition and the effective learning rate $\eta_{\mathrm{eff}\,t}$ are entirely controlled by rules involving arbitrary powers of the number of mistakes. Moreover, we provided experimental evidence in support of our theoretical analysis. The experimental results also suggest that algorithms belonging to the MICRA family with slow relaxation of the misclassification condition and relatively fast suppression of $\eta_{\mathrm{eff}\,t}$ with the number of mistakes are very powerful tools in the hands of a skillful practitioner. Of course, this does not diminish at all the value and usefulness of established algorithms like LIBSVM or SVM^{light} which only need fixing the accuracy parameter ϵ. It is remarkable, however, that simple extensions of the old Perceptron algorithm can be so competitive.

References

Chang, C.-C., & Lin, C.-J. (2001). *LIBSVM: a library for support vector machines.* Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Cristianini, N., & Shawe-Taylor, J. (2000). *An introduction to support vector machines.* Cambridge University Press.

Duda, R. O., & Hart, P. E. (1973). *Pattern classsification and scene analysis.* Wiley.

Gentile, C. (2001). A new approximate maximal margin classification algorithm. *Machine Learning Research*, *2*, 213–242.

Joachims, T. (1999). Making large-scale svm learning practical. In *Advances in kernel methods-support vector learning.* MIT Press.

Joachims, T. (2006). Training linear svms in linear time. *KDD'06* (pp. 217–226). ACM Press.

Li, Y., & Long, Y. (2002). The relaxed online maximum margin algorithm. *Machine Learning*, *46*, 361–387.

Platt, J. C. (1998). *Sequential minimal optimization: A fast algorithm for training support vector machines* (Technical Report MSR-TR-98-14). Microsoft Research.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65 (6)*, 386–408.

Tsampouka, P., & Shawe-Taylor, J. (2005). Analysis of generic perceptron-like large margin classifiers. *ECML 2005* (pp. 750–758). Springer-Verlag.

Tsampouka, P., & Shawe-Taylor, J. (2006). Constant rate approximate maximum margin algorithms. *ECML 2006* (pp. 437–448). Springer-Verlag.

Vapnik, V. (1998). *Statistical learning theory.* Wiley.