The Perceptron Algorithm with Uneven Margins

Yaoyong Li YAOYONG@CS.RHUL.AC.UK
Royal Holloway, University of London, Egham, Surrey, TW20 0EX, UK

Hugo Zaragoza HUGOZ@MICROSOFT.COM
Microsoft Research, 7 J J Thomson Avenue, CB3 0FB Cambridge, UK

Ralf Herbrich RHERB@MICROSOFT.COM
Microsoft Research, 7 J J Thomson Avenue, CB3 0FB Cambridge, UK

John Shawe-Taylor JOHN@CS.RHUL.AC.UK
Royal Holloway, University of London, Egham, Surrey, TW20 0EX, UK

Jaz Kandola JAZ@CS.RHUL.AC.UK
Royal Holloway, University of London, Egham, Surrey, TW20 0EX, UK

Abstract

The perceptron algorithm with margins is a simple, fast and effective learning algorithm for linear classifiers; it produces decision hyperplanes within some constant ratio of the maximal margin. In this paper we study this algorithm and a new variant: the perceptron algorithm with uneven margins, tailored for document categorisation problems (i.e. problems where classes are highly unbalanced and performance depends on the ranking of patterns). We discuss the interest of these algorithms from a theoretical point of view, provide a generalisation of Novikoff's theorem for uneven margins, give a geometrically description of these algorithms and show experimentally that both algorithms yield equal or better performances than support vector machines, while reducing training time and sparsity, in classification (USPS) and document categorisation (Reuters) problems.

1. Introduction

The support vector machine (SVM) is a well known learning algorithm for linear classifiers and has achieved state of the art results for many classification problems. Besides its high performance, the SVM algorithm is simple to use (i.e. few parameters need to be tuned prior to training). Furthermore, the *kernel-trick* provides an elegant and efficient way to deal with very high-dimensional feature spaces and to introduce domain knowledge into the learning algorithm.

The SVM algorithm requires solving a quadratic programming problem to find the linear classifier of maximal margin. Because generalisation error can be upper bounded by a function of the margin of a linear classifier, finding maximal margin classifiers is a sensible strategy. However, this is difficult to implement efficiently and, more importantly, often leads to very long training times.

Surprisingly, it has been shown theoretically that there are alternatives to finding the maximal margin hyperplane which often lead to algorithms that are simpler to implement, faster, and provide tighter upper-bounds on the generalisation error (e.g. Graepel et al. (2001)). Most of these alternative algorithms are based on Rosenblatt's original perceptron algorithm (PA) (Rosenblatt 1958), an on-line learning algorithm for linear classifiers.

In this paper we study three other learning algorithms for linear classifiers: the perceptron algorithm with margins (PAM) (Krauth and Mézard 1987), the ALMA algorithm (Gentile 2001) and the proposed perceptron algorithm with uneven margins (PAUM) . These three algorithms originate from the PA but add additional constraints on the margin of the resulting classifier. As such they cover the spectrum between the PA's *no-margin constraint* and the SVM's *maximum margin constraint.*

The PAUM is an extension of the PA specially designed to cope with two class problems where positive examples are very rare compared to negative ones. This occurs often in problems of information retrieval, detection, speech and face recognition, etc. We observe experimentally that the SVM clearly outperformed the PA for such problems. We will demonstrate that the new algorithm outperforms the SVM for the task of document categorisation. Furthermore we show that Novikoff's theorem on the number of updates of the PA can be generalised to the PAUM.

Despite the simplicity of the PAM and the PAUM, we observe empirically these algorithms yield classifiers equal or better than the SVM classifier, while reducing training time and sparsity. Algorithms are evaluated on the USPS standard classification task and the Reuters text categorisation tasks.

In Section 2 we describe the PAM and PAUM algorithms and present some theoretical results on their quality compared to that of the SVM, as well as a geometrical interpretation of these algorithms in version space. Section 3 presents several experimental comparisons in image classification (USPS) and document categorisation (Reuters).

2. The Perceptron Algorithm with Uneven Margins

In the following, we assume that we are given a training sample $\boldsymbol{z} = ((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \{-1, +1\})^m$ of size m together with a feature mapping $\boldsymbol{\phi} : \mathcal{X} \to \mathcal{K} \subseteq \mathbb{R}^n$ into an n −dimensional vector space $\mathcal{K}$. Our aim is to learn the parameters $\mathbf{w} \in \mathcal{K}$ and $b \in \mathbb{R}$ of a linear classifier

$$\begin{aligned} h_{\mathbf{w},b}(x) &:= \operatorname{sign}(f_{\mathbf{w},b}(x)) , \\ f_{\mathbf{w},b}(x) &:= \langle \mathbf{w}, \mathbf{x} \rangle + b , \end{aligned}$$

where $\mathbf{x} := \boldsymbol{\phi}(x)$ and $\langle \cdot, \cdot \rangle$ denotes the inner product in $\mathcal{K}$.

The starting point of our analysis is the classical perceptron algorithm (Rosenblatt 1958). This is an on-line algorithm which proceeds by checking whether or not the current training examples $(x_i, y_i) \in \boldsymbol{z}$ is correctly classified by the current classifier $(y_i (\langle \mathbf{w}_t, \mathbf{x}_i \rangle + b_t) > 0$) and updating the weight vector $\mathbf{w}_t$ otherwise. In the update step, the weight vector $\mathbf{w}_t$ and the "bias" b_t are changed into $\mathbf{w}_{t+1} = \mathbf{w}_t + \eta y_i \mathbf{x}_i$ and $b_{t+1} = b_t + \eta$. This algorithm is guaranteed to converge whenever the training data is linearly separable in feature space (Novikoff 1962). The central quantity controlling the speed of convergence of the PA (i.e., an upper bound on the number of updates until convergence) is the *maximal margin* of the training data. The margin $\gamma(\mathbf{w}, b, \boldsymbol{z})$ of a classifier $f_{\mathbf{w},b}$ is minimal real-valued output on the training sample, that is,

$$\gamma(\mathbf{w}, b, \boldsymbol{z}) := \min_{(x_i, y_i) \in \boldsymbol{z}} \frac{y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b)}{\|\mathbf{w}\|} . \qquad (2.1)$$

In a nutshell, the larger the maximal margin $\gamma(\boldsymbol{z}) := \max_{\mathbf{w},b} \gamma(\mathbf{w}, b, \boldsymbol{z})$ for a particular training sample $\boldsymbol{z}$, the less updates the PA performs until convergence. Unfortunately, no lower bounds can be given for the margin $\gamma(\mathbf{w}_t, b_t, \boldsymbol{z})$ of the PA's solution $(\mathbf{w}_t, b_t)$.

It is well known that the SVM algorithm finds (up to a scaling factor) the parameters $(\mathbf{w}_{\mathrm{SVM}}, b_{\mathrm{SVM}})$ which maximise the margin $\gamma(\mathbf{w}, b, \boldsymbol{z})$. Maximum margin classifiers exhibit excellent generalisation performance in terms of misclassification error (Shawe-Taylor et al. 1998; Cristianini and Shawe-Taylor 2000). However, this maximisation requires solving a quadratic programming problem.

A generalisation of the PA was presented in Krauth and Mézard (1987). This algorithm, which is also known as the *perceptron algorithm with margins*, is more conservative in the test condition for updates. Rather than notifying only misclassified training examples, the PAM updates until $y_i (\langle \mathbf{w}_t, \mathbf{x}_i \rangle + b_t) > \tau$, where $\tau \in \mathbb{R}^+$ is a fixed parameter chosen before learning. The effect of τ is that the upper bound on the number of updates until convergence increases by a factor of approximately τ but, in return, the margin $\gamma(\mathbf{w}_t, b_t, \boldsymbol{z})$ can proven to be at least $\gamma(\boldsymbol{z}) \cdot \tau / (2\tau + R^2)$.

Our algorithm differs from the PAM insofar as it treats positive and negative examples differently. For example, in document categorisation problems it is much more important to correctly classify positive examples than to correctly classify negative examples, partly because their numbers differ by several orders of magnitude. An easy way to incorporate this idea into the PAM is to consider the positive and negative margin separately. The *positive (negative) margin* $\gamma_{\pm 1}(\mathbf{w}, b, \boldsymbol{z})$ is defined as

$$\gamma_{\pm 1}(\mathbf{w}, b, \boldsymbol{z}) := \min_{(x_i, \pm 1) \in \boldsymbol{z}} \frac{\pm (\langle \mathbf{w}, \mathbf{x}_i \rangle + b)}{\|\mathbf{w}\|} . \qquad (2.2)$$

The resulting algorithm — which is a direct generalisation of the PAM — is called *perceptron algorithm with uneven margins* and is given in Algorithm 1.

Recently, Gentile (2001) has presented ALMA, a variation of the PA which also aims at finding a large margin classifier. In a nutshell, after j mistakes ALMA uses $\tau_j \propto 1/\sqrt{j}$ and $\eta_j \propto 1/\sqrt{j}$ instead of a fixed margin parameter τ and learning rate η .

Remark. In order to generalise these algorithms to the application of kernels, that is, inner product functions

Algorithm 1 PAUM (τ_{-1}, τ_{+1})

```
Require: A linearly separable training sample $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{y}) \in (\mathcal{X} \times \{-1,+1\})^m$
Require: A learning rate $\eta \in R^+$
Require: Two margin parameters $\tau_{-1},\ \tau_{+1} \in \mathbb{R}^+$

  $\mathbf{w}_0 = \mathbf{0}$ ; $b_0 = 0$ ; $t = 0$ ; $R = \max_{x_i \in \boldsymbol{x}} \|\mathbf{x}_i\|$
  repeat
    for $i = 1$ to $m$ do
      if $y_i (\langle \mathbf{w}_t, \mathbf{x}_i \rangle + b_t) \leq \tau_{y_i}$ then
        $\mathbf{w}_{t+1} = \mathbf{w}_t + \eta y_i \mathbf{x}_i$
        $b_{t+1} = b_t + \eta y_i R^2$
        $t \leftarrow t + 1$
      end if
    end for
  until no updates made within the for loop
  return $(\mathbf{w}_t, b_t)$
```

$k(x, \widetilde{x}) := \langle \mathbf{x}, \widetilde{\mathbf{x}} \rangle$, we observe that the each weight vector must be expressible as $\mathbf{w}_t = \sum_{j=1}^{m} \alpha_j \mathbf{x}_j$ because in the update step training examples are only added to the initial weight vector $\mathbf{w}_0 = \mathbf{0}$. Inserting the linear expansion into the inner products $\langle \mathbf{w}, \mathbf{x}_i \rangle$, we see that Algorithm 1 can alternatively be written in terms of the expansion coefficients $\boldsymbol{\alpha} \in \mathbb{R}^m$. This, however, is only computationally advantageous if $n \gg m$.

2.1. An Extension of Novikoff's Theorem

We analyse the PAUM by giving an upper bound on the numbers of updates as well as a lower bound on the positive and negative margin of the resulting classifier. This theorem is an extension of Novikoff's theorem (Novikoff 1962) as well as of the result of Krauth and Mézard (1987).

Theorem 2.1. *Let $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{y}) \in (\mathcal{X} \times \{-1,+1\})^m$ be a given training sample, and let $R := \max_{x \in \boldsymbol{x}} \|\mathbf{x}_i\|$.*

1. *Suppose there exists $(\mathbf{w}_{\text{opt}}, b_{\text{opt}}) \in (\mathcal{K} \times \mathbb{R})$ such that $\|\mathbf{w}_{\text{opt}}\| = 1$, $|b_{\text{opt}}| \leq R$ and*
$$\gamma(\mathbf{w}_{\text{opt}}, b_{\text{opt}}, \boldsymbol{z}) \geq \Gamma . \tag{2.3}$$
Then the number of updates made by the PAUM(τ_{-1}, τ_{+1}) on $\boldsymbol{z}$ is bounded by
$$4 \left(\left(\frac{R}{\Gamma} \right)^2 + \frac{\max\{\tau_{+1}, \tau_{-1}\}}{\eta \Gamma^2} \right) . \tag{2.4}$$

2. *Fix a learning rate $\eta \in \mathbb{R}^+$ and margin parameters $\tau_{-1}, \tau_{+1} \in \mathbb{R}^+$. Let $\psi := \frac{\eta R^2 + \tau_{+1}}{\eta R^2 + \tau_{-1}}$ and suppose there exists $(\widetilde{\mathbf{w}}_{\text{opt}}, \widetilde{b}_{\text{opt}}) \in (\mathcal{K} \times \mathbb{R})$ such that $\|\widetilde{\mathbf{w}}_{\text{opt}}\| = 1$, $|\widetilde{b}_{\text{opt}}| \leq R$ and, for $j \in \{-1,+1\}$,*
$$\gamma_j \left(\widetilde{\mathbf{w}}_{\text{opt}}, \widetilde{b}_{\text{opt}}, \boldsymbol{z} \right) \geq \Gamma_j , \tag{2.5}$$
where $\Gamma_{-1} \in \mathbb{R}^+$ and $\Gamma_{+1} := \psi \cdot \Gamma_{-1}$. Then, for the solution $(\mathbf{w}_t, b_t)$ of the PAUM(τ_{-1}, τ_{+1}) , for $j \in \{-1,+1\}$, we know
$$\gamma_j (\mathbf{w}_t, b_t, \boldsymbol{z}) > \Gamma_j \cdot \frac{\tau_j}{\sqrt{8}\,(\eta R^2 + \tau_j)} . \tag{2.6}$$

The proof can be found in Appendix A. First, (2.4) is a direct generalisation of Novikoff's theorem because setting $\tau_{+1} = \tau_{-1} = 0$, we retain the original result. Furthermore, choosing $\tau_{+1} = \tau_{-1}$ recovers the results of Krauth and Mézard (1987). Most interestingly, we observe that η defines a trade-off between (guaranteed) convergence time and approximation w.r.t. both positive and negative margins: For $\eta \to 0$ the PAUM finds a solution with maximal margins (up to a factor of $\sqrt{1/8}$) but the algorithm is no longer guaranteed to converge (see (2.6) and (2.4)). On the other hand, for $\eta \to \infty$ the algorithm converges as quickly as the original PA but we are no longer able to guarantee any positive (negative) margin of the resulting solution. Finally, note that the constants 4 and $\sqrt{1/8}$ in the two bounds can be further optimised by fixing the bias to 0 .

2.2. A Graphical Illustration of the PAUM

In order to enhance the understanding of the PAUM, we recall that for every training sample $\boldsymbol{z}$, there must exist the so called *version space* $V(\boldsymbol{z})$ defined as

$$V(\boldsymbol{z}) := \{(\mathbf{w}, b) \in (\mathcal{K} \times \mathbb{R}) \mid \gamma(\mathbf{w}, b, \boldsymbol{z}) \geq 0\} .$$

Note that $V(\boldsymbol{z})$ is empty if $\boldsymbol{z}$ is not linearly separable in feature space. The version space is a convex region because it is the intersection of m halfspaces. Every solution of the PAUM(τ_{-1}, τ_{+1}) as well as the SVM solution must be a point in the set $V(\boldsymbol{z})$. Since, in general, $\dim(\mathcal{K}) \gg 1$ it is impossible to visualise version space for real-world datasets. Hence, we project version space onto a plane spanned by three points within version space; the SVM solution, the PA solution and the PAM(0.5) solution. In Figure 2.1 we have depicted such a "slice" for a typical learning problem (for details on the dataset, see Section 3). As expected, if we use the PA the resulting classifier ("0.0") is very close to the bounding training examples (the margin of this classifier is very small). Increasing τ in the PAM(τ) finds solutions (e.g. "1.0") which are visibly close to the support vector solution. However, this can also be achieved with less conservative updates on the

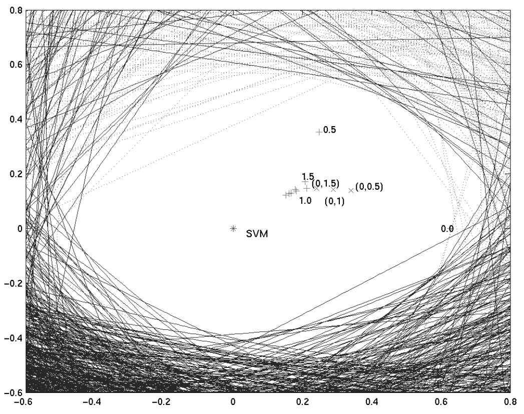

Figure 2.1. A slice through version space for the Reuters-21578 categorisation problem 'earn'. Solid/dotted lines correspond to negative/positive training examples. Some solutions of PAUM(τ_{-1}, τ_{+1}) for different (τ_{-1}, τ_{+1}) parameters, along with the SVM solution, are displayed. Points which have only one number were obtained by using $\tau_{-1} = \tau_{+1}$.

negative examples (of which there are approximately three times more than positive) using, for example, PAUM$(0, 1)$.

2.3. The λ Trick for Linearly Inseparable Training Samples

Theorem 2.1 shows that the PAUM will stop after some finite number of updates for any linearly separable training sample. In order to deal with training samples which are linearly inseparable in feature space, we use the so-called "λ trick". In the current formulation, this amounts to augmenting each feature vector $\mathbf{x}_i$ by the m –dimensional unit vector $\sqrt{\lambda}\mathbf{e}_i$, $\lambda \in \mathbb{R}^+$, where $\mathbf{e}_i \in \mathbb{R}^m$ has all components zero except for the i th component which equals one. Since every training examples spans a new dimension in the augmented feature space, the training sample is necessarily linearly separable. Intuitively, if $\alpha_i \propto \eta y_i$ denotes the cumulated updates of the i th training example then the real-valued output of the augmented example $\mathbf{x}_j$ is given by

$$\sum_{i \neq j} \alpha_i \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle + \alpha_j \underbrace{\left(\lambda + \|\mathbf{x}_j\|^2\right)}_{>0} .$$

Now, the second term can dominate the sum by just adjusting α_j which, by definition, has the same sign as y_j . Though the augmented training sample becomes linearly separable in feature space, the final classifier will commit some training errors in the original feature space, the number of which is controlled by λ .

Remark. If we are using a kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ rather than an explicit feature mapping, the λ trick amounts to a simple change of the kernel function during training. More formally,

$$k_\lambda \left(x, \widetilde{x}\right) := k \left(x, \widetilde{x}\right) + \lambda \cdot \mathbb{I}_{x = \widetilde{x}} .$$

This trick is discussed in greater detail in Herbrich (2002). It is known that the λ trick for linear classification learning algorithms can not only deal with inseparable training samples, but also tolerate noise and outliers. Hence, we expect that the PAUM combined with the λ trick can have better performance even for linearly separable training samples, as confirmed by our experiments (see Section 3).

3. Experimental Results

We now provide experimental comparisons of the algorithms presented above. In Section 3.1 we will evaluate the SVM, ALMA, PA and PAM on a well known classification problem, the USPS dataset which is a benchmark for optical character recognition systems. In Section 3.2 we evaluate the SVM, PAM and PAUM on two standard datasets for document categorisation, the Reuters collections.

3.1. Classification Experiments (USPS)

The USPS dataset consists of vectors corresponding to images of hand-written digits, labelled with the digit they represent (0,1,...,9). There are a total of 7291 training patterns and 2007 test patterns. We made no special preprocessing of the images and used a Gaussian kernel: $k\left(\vec{x}_i, \vec{x}_j\right) = \exp(-(2\sigma)^{-2} \left\|\vec{x}_i - \vec{x}_j\right\|^2)$ with $\sigma = 3.5$ as in Gentile (2001).

For this problem we adhere to the standard approach of learning independently 10 binary classifiers (one for each digit) on the training sample and then, for each pattern in the test sample, choosing the class of the classifier that produces the highest output. For PA, PAM and ALMA, which depend on the ordering of the training sample, we repeat the entire process ten times (permuting randomly the training sample every time) and we average results over the ten runs.

Table 1 reports the percentage of test examples misclassified on average (as described above) for the PA, PAM (with τ equal to 0.2 and 0.4) and ALMA (with τ equal to 0.9 and 0.95). Performance of the on-line algorithms is shown after 1 and 3 training epochs as well as after convergence (≈ 8 epochs). Results for the SVM are taken from Platt et al. (2000) and for ALMA from Gentile (2001).

Algorithm	% Misclassification		
	1 epoch	3 epochs	Convergence
PA	6.20%	5.50%	5.10%
PAM (0.2)	4.84%	4.72%	4.69%
PAM (0.4)	**4.71%**	**4.64%**	**4.56%**
ALMA (0.9)	5.43%	4.90%	—
ALMA (0.95)	5.72%	4.85%	—
SVM			4.58%

Table 1. Classification experiments on the USPS dataset.

First note that after one epoch of training all algorithms yield reasonable results, while SVM outperforms PA and ALMA, and only slightly the PAM. This is remarkable given the simplicity of the PA and PAM, and the fact that each training example has been used only once for each class. In this case, the training time as compared to SVMs was reduced by an order of magnitude. After 3 epochs all algorithms improved, specially ALMA, although the SVM continues to outperform slightly. None of the algorithms dramatically improves in performance after convergence, but it must be noted that eventually the PAM with $\tau = 0.4$ slightly outperforms the SVM (not statistically significant).

We see from these results that PAM offer a good compromise between the simplicity of PA and the accuracy of SVMs. Despite ALMA's theoretical motivation, it does not seem to improve on the simple PAM.

3.2. Document Categorisation Experiments (Reuters)

In document categorisation we need to rank a set of documents with respect to their relevance to a particular topic, and evaluate the quality of the resulting ranked list of documents. Topics are not mutually exclusive and their size (i.e. the number of documents relevant to a topic) can vary widely.

Performance measures for document categorisation differ from usual machine learning performance measures due to the fact that there are very few positive examples and a range of misclassification costs need to be considered. Performance is often measured by some average function of the *precision* of a classifier measured at different *recall* values. After training a classifier on a particular topic, the resulting function $f_{\mathbf{w},b}$ can be used to order any sample of documents. Then, for a given sample $\boldsymbol{z}$, a given classification function f and any threshold θ on this function, we can compute the precision and recall as:

$$p_{\boldsymbol{z},f}(\theta) := \frac{|\{(x_i, y_i) \in \boldsymbol{z} \mid (f(x_i) > \theta) \wedge (y_i = +1)\}|}{|\{(x_i, y_i) \in \boldsymbol{z} \mid f(x_i) > \theta\}|}$$

	ALL	**TOP10**	**LAST30**
Macro-Average Precision:			
PA	0.700	0.917	0.539
PAM(1)	0.714	0.920	0.543
PAUM(-1,1)	0.716	0.921	0.582
PAUM(1,50)	**0.751**	**0.924**	**0.636**
SVM	0.746	0.918	0.634
Average Sparsity:			
PA	91	442	9
PAM(1)	132	650	12
PAUM(-1,1)	89	443	8
PAUM(1,50)	**462**	**1872**	**82**
SVM	269	933	72

Table 2. Experiments on Reuters-21578 dataset. We have indicated in bold face the results for the PAUM model with best performance over the training sample using 10-fold cross-validation.

and

$$r_{\boldsymbol{z},f}(\theta) := \frac{|\{(x_i, y_i) \in \boldsymbol{z} \mid (f(x_i) > \theta) \wedge (y_i = +1)\}|}{|\{(x_i, y_i) \in \boldsymbol{z} \mid y_i = +1\}|} .$$

By plotting precision vs. recall for all values of θ we obtain the so called *precision-recall curve,* from which most performance measures in information retrieval originate. Here, we use the macro-averaged precision (MAP) measure, which approximates the area under a precision-recall curve by averaging the precision values obtained at each positive point:

$$\text{MAP}_{\boldsymbol{z},f} := \frac{1}{|\{(x_i, +1) \in \boldsymbol{z}\}|} \sum_{\{(x_i,+1)\in\boldsymbol{z}\}} p_{\boldsymbol{z},f}(f(x_i)) .$$

In order to gain a better insight into the behaviour of the algorithms with respect to topic size, we report three different averages, the average over all topics (ALL), the 10 largest (TOP10) and the 30 smallest (LAST30).

We conducted experiments on two document collections, the well known 'Mod-Apte' split of the Reuters-21578 collection, and a subset of the new Reuters-Vol1 collection. The Mod-Apte sample has 9603 and 3299 training and test documents respectively, and 90 classes (ranging from 1 to $\approx$ 1000 relevant training documents). For the new Reuters-Vol1 collection we chose the following split: All the 12807 documents in the week starting the 26/08/1996 for training, and all 12800 documents of the following week for testing. We considered only the 99 categories for which there was at least one training document and one test document in these two periods. The usual preprocessing of documents was carried out leading to 20 000 features (i.e. distinct words or *terms*) for Reuters-21578

	ALL	**TOP10**	**LAST30**
Macro-Average Precision:			
PA	0.535	0.891	0.269
PAM(1)	0.561	0.899	0.303
PAUM(-1,1)	0.538	0.890	0.275
PAUM(1,50)	**0.589**	**0.904**	**0.345**
SVM	0.574	0.897	0.325
Average Sparsity:			
PA	454	1811	42
PAM(1)	667	2659	65
PAUM(-1,1)	455	1830	41
PAUM(1,50)	**1811**	**5737**	**375**
SVM	980	2841	266

Table 3. Experiments on the Reuters-Vol1 dataset. We have indicated in bold face the results for the PAUM model with best performance over the training sample using 10-fold cross-validation.

and 120 000 features for Reuters-Vol1. Vectors were constructed from documents in the usual bag-of-words way using *tf-idf* weighting and normalising vectors. In other words, the document x is represented as an $n-$ dimensional vector $\mathbf{x}$ where $x_i := \mathrm{tf}_i \cdot \log(m/\mathrm{df}_i)$, tf_i is the number of times term i appears in document x and df_i is the number of documents in which the term i appears. Note that df_i and n refer to the training sample.

Table 2 reports results on the Reuters-21578 collections for the SVM and the PAUM with a number of (τ_{-1}, τ_{+1}) settings. First we note that the PA provides results which are close to those of the SVM[1]. Indeed, when considering only the 10 largest topics (TOP10) the PA is as good as the SVM, and is twice as sparse (indeed, for these topics we observe training times for the PA orders of magnitude smaller than for the SVM).

Second, note that although the PAM increases performance slightly over the PA, the price paid in sparsity[2] and training time does not seem to be worthwhile. The real gain in performance is obtained when uneven margins are used. The PAUM$(-1,1)$ succeeds in achieving near-SVM performance with low sparsity. Note that negative values for $\tau_{\pm 1}$ allow for misclassification errors though Theorem 2.1 remains valid (for small magnitudes). Indicated in bold is the performance of the PAUM$(1,50)$ which achieved the best performance on the training sample using 10-fold cross-validation on classes with at least 40 positive examples[3]. This model performs better than the SVM and reduced the training time of the SVM by a factor of two.

[1]Note that uneven margins in SVMs only lead to a change in the bias b . This, however, does not change the macro-averaged precision.

[2]In this paper *sparsity* is defined as the number of zero components, $\alpha_i = 0$, of the vector $\boldsymbol{\alpha} \in \mathbb{R}^m$ of expansion coefficients.

	$\lambda = 0$	$\lambda = 1$
MAP	0.689	0.694
Sparsity	76	67

Table 4. The macro-average precision and sparsity results of the PA on the Reuters-21578 collection, for the 77 linearly separable classes, without the noise parameter ($\lambda = 0$) and with $\lambda = 1$.

Note that probably better results would be obtained by further increasing τ_{+1} at the price of less sparsity. This quantity acts as a trade-off between sparsity and performance. We did not conduct further experiments to keep the sparsity comparable to that of the SVM.

Results on the Reuters-Vol1 dataset are presented in Table 3. The overall behaviour of the algorithms is similar: Again the PA performs as well as the rest of algorithms for the largest topics (TOP10), and the SVM and the PAUM perform similarly for similar sparsity values.

Finally, we evaluate experimentally the effect of the noise parameter λ needed to deal with linearly inseparable problems. Out of the 90 topics in the Reuters-21578 collection, and using the document encoding described earlier, there are 77 topics with linearly-separable training samples. On these 77 topics we can set the noise parameter λ to 0 and observe how performance and sparsity is affected; we show the results of this comparison in Table 4. For this particular problem, MAP performance increased by 0.005 and sparsity decreased when using $\lambda = 1$.

4. Conclusions

We have shown that the perceptron algorithm with margins is a very efficient learning algorithm for linear classifiers. We have generalised this algorithm to allow uneven margins, proved a generalisation of Novikoff's for this algorithm, and provided a geometrical picture of this family of linear learning algorithms. Uneven margins are specially appropriate for problems were class sizes are highly unbalanced. We demonstrated this on a standard classification problem and a document categorisation problem, where the use of uneven margins yields classifier which are sparser and more

[3]The best parameters where chosen from $\tau_{-1} \in \{-1.5, -1, -0.5, 0, 0.1, 0.5, 1.0\}$ and $\tau_{+1} \in \{-1, -0.5, 0, 0.1, 0.5, 1, 2, 5, 10, 50\}$, respectively.

performant than SVMs.

There are several directions that we will pursue in the future. Firstly, it seems intuitive to adapt the margin parameters to the size of each topic individually. Promising results were obtained in preliminary experiments were we fixed τ_{-1} to a small negative number and used $\tau_{+1} \propto (1+\exp(-\kappa \cdot c))^{-1}$ with c being the fraction of positive examples in a topic. Secondly, in the context of document categorisation, it seems that performance measures are so decorrelated from misclassification error that performance on the training sample (in terms of MAP) can be used for model selection. Since PAUM's are fast to train, one could afford to try many values of τ_{+1} and τ_{-1} .

Acknowledgements

We are grateful to Huma Lodhi for technical assistance and the three reviewers for their valuable comments. We thank Xerox Research Centre Europe for providing the experimental dataset of the new Reuters corpus.

A. Proof of Theorem 2.1

Proof. Throughout the proof we will use the shorthand notation $\widehat{\mathbf{x}} := (\mathbf{x}', R)'$ and $\widehat{\mathbf{w}} := \left(\mathbf{w}', \frac{b}{R}\right)'$ to denote augmented training examples (mapped into feature space) and weight vectors.

1. From Algorithm 1 we know that $\widehat{\mathbf{w}}_t = \widehat{\mathbf{w}}_{t-1} + \eta y_i \widehat{\mathbf{x}}_i$ whenever $y_i\left(\langle \widehat{\mathbf{w}}_{t-1}, \widehat{\mathbf{x}}_i\rangle\right) \le \tau_{y_i}$. Thus,

$$\begin{aligned} \|\widehat{\mathbf{w}}_t\|^2 &= \|\widehat{\mathbf{w}}_{t-1}\|^2 + 2\eta y_i \langle \widehat{\mathbf{w}}_{t-1}, \widehat{\mathbf{x}}_i\rangle + \eta^2 \|\widehat{\mathbf{x}}_i\|^2 \\ &\le \|\widehat{\mathbf{w}}_{t-1}\|^2 + 2\eta\tau_{y_i} + 2\eta^2 R^2 , \end{aligned}$$

because $\|\widehat{\mathbf{x}}_i\|^2 = \|\mathbf{x}_i\|^2 + R^2 \le 2R^2$. A repeated application of this inequality implies that

$$\begin{aligned} \|\widehat{\mathbf{w}}_t\|^2 &\le 2t\eta^2R^2 + 2\eta\left(t_{+1}\tau_{+1} + t_{-1}\tau_{-1}\right) &\text{(A.1)} \\ &\le 2t\eta\left(\eta R^2 + \tau_{\max}\right) , &\text{(A.2)} \end{aligned}$$

where t_{+1} is the number of updates of positive examples, $t_{-1} := t - t_{+1}$ and $\tau_{\max} := \max\{\tau_{+1}, \tau_{-1}\}$. Similarly, from (2.3) we have

$$\begin{aligned} \langle \widehat{\mathbf{w}}_{\text{opt}}, \widehat{\mathbf{w}}_t\rangle &= \langle \widehat{\mathbf{w}}_{\text{opt}}, \widehat{\mathbf{w}}_{t-1}\rangle + \eta y_i \langle \widehat{\mathbf{w}}_{\text{opt}}, \widehat{\mathbf{x}}_i\rangle \\ &\ge \langle \widehat{\mathbf{w}}_{\text{opt}}, \widehat{\mathbf{w}}_{t-1}\rangle + \eta\Gamma \ge t\eta\Gamma . \quad \text{(A.3)} \end{aligned}$$

Combining (A.2) and (A.3) and using the Cauchy-Schwarz inequality gives the relation

$$\begin{aligned} t^2(\eta\Gamma)^2 &\le \left(\langle \widehat{\mathbf{w}}_{\text{opt}}, \widehat{\mathbf{w}}_t\rangle\right)^2 \le \|\widehat{\mathbf{w}}_{\text{opt}}\|^2 \|\widehat{\mathbf{w}}_t\|^2 \\ &\le \|\widehat{\mathbf{w}}_{\text{opt}}\|^2 2t\eta\left(\eta R^2 + \tau_{\max}\right) . \quad \text{(A.4)} \end{aligned}$$

Since $b_{\text{opt}} \le R$ by assumption, $\|\widehat{\mathbf{w}}_{\text{opt}}\|^2 \le \|\mathbf{w}_{\text{opt}}\|^2 + 1 = 2$ which, inserted into (A.4), implies the result (2.4).

2. By the update rule $\widehat{\mathbf{w}}_t = \widehat{\mathbf{w}}_{t-1} + \eta y_i \widehat{\mathbf{x}}_i$ and (2.5), just as in the derivation of (A.3), we have

$$\begin{aligned} \left\langle \widehat{\widetilde{\mathbf{w}}}_{\text{opt}}, \widehat{\mathbf{w}}_t \right\rangle &\ge \eta\left(t_{+1}\Gamma_{+1} + t_{-1}\Gamma_{-1}\right) \\ &= \eta\Gamma_{-1}\left(\psi t_{+1} + t_{-1}\right) , \quad \text{(A.5)} \end{aligned}$$

where the relationship $\Gamma_{+1} := \psi \cdot \Gamma_{-1}$ is used. On the other hand, from the inequality (A.1) we have

$$\begin{aligned} \|\widehat{\mathbf{w}}_t\|^2 &\le 2t\eta^2R^2 + 2\eta\left(t_{+1}\tau_{+1} + t_{-1}\tau_{-1}\right) \\ &= 2\eta\left[t_{+1}\left(\eta R^2 + \tau_{+1}\right) + t_{-1}\left(\eta R^2 + \tau_{-1}\right)\right] \\ &= 2\eta\left[\left(\psi t_{+1} + t_{-1}\right)\left(\eta R^2 + \tau_{-1}\right)\right] , \quad \text{(A.6)} \end{aligned}$$

where the relationship $t = t_{+1} + t_{-1}$ and the definition of ψ are used. The two inequalities (A.5) and (A.6) combined together with the Cauchy-Schwarz inequality give the relations

$$\begin{aligned} \eta^2\Gamma_{-1}^2\left(\psi t_{+1} + t_{-1}\right)^2 &\le \left\|\widehat{\widetilde{\mathbf{w}}}_{\text{opt}}\right\|^2 \|\widehat{\mathbf{w}}_t\|^2 \\ &\le 4\eta\left[\left(\psi t_{+1} + t_{-1}\right)\left(\eta R^2 + \tau_{-1}\right)\right] , \end{aligned}$$

where $\left\|\widehat{\widetilde{\mathbf{w}}}_{\text{opt}}\right\|^2 \le \|\widetilde{\mathbf{w}}_{\text{opt}}\|^2 + 1 = 2$ is used. This inequality implies the bound

$$\psi t_{+1} + t_{-1} \le 4\frac{\left(\eta R^2 + \tau_{-1}\right)}{\eta\Gamma_{-1}^2} \quad \text{(A.7)}$$

By substituting (A.7) into (A.6), we obtain

$$\|\widehat{\mathbf{w}}_t\|^2 \le 8\frac{\left(\eta R^2 + \tau_{-1}\right)^2}{\Gamma_{-1}^2} = 8\frac{\left(\eta R^2 + \tau_{+1}\right)^2}{\Gamma_{+1}^2} .$$

The result (2.6) follows by combining the last inequality with (2.2) and observing that, at termination, $\min_{(x_i,\pm1)\in z} \pm\langle \widehat{\mathbf{w}}_t, \widehat{\mathbf{x}}_i\rangle > \tau_{\pm1}$.

□

References

Cristianini, N. and J. Shawe-Taylor (2000). *An Introduction to Support Vector Machines.* Cambridge, UK: Cambridge University Press.

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

Graepel, T., R. Herbrich, and R. C. Williamson (2001). From margin to sparsity. In T. K. Leen, T. G. Dietterich, and V. Tresp (Eds.), *Advances in Neural Information Processing Systems 13*, Cambridge, MA, pp. 210–216. MIT Press.

Herbrich, R. (2002). *Learning Kernel Classifiers: Theory and Algorithms*. MIT Press.

Krauth, W. and M. Mézard (1987). Learning algorithms with optimal stability in neural networks. *Journal of Physics A 20*, 745–752.

Novikoff, A. B. J. (1962). On convergence proofs on perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata*, Volume 12, pp. 615–622. Polytechnic Institute of Brooklyn.

Platt, J. C., N. Cristianini, and J. Shawe-Taylor (2000). Large margin DAGs for multiclass classification. In S. A. Solla, T. K. Leen, and K.-R. Müller (Eds.), *Advances in Neural Information Processing Systems 12*, Cambridge, MA, pp. 547–553. MIT Press.

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

Shawe-Taylor, J., P. L. Bartlett, R. C. Williamson, and M. Anthony (1998). Structural risk minimization over data-dependent hierarchies. *IEEE Transactions on Information Theory 44*(5), 1926–1940.