ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

An Accelerated MDM Algorithm for SVM Training

Álvaro Barbero, Jorge López, José R. Dorronsoro *

Dpto. de Ingeniería Informática and Instituto de Ingeniería del Conocimiento
Universidad Autónoma de Madrid, 28049 Madrid, Spain

Abstract. In this work we will propose an acceleration procedure for the Mitchell–Demyanov–Malozemov (MDM) algorithm (a fast geometric algorithm for SVM construction) that may yield quite large training savings. While decomposition algorithms such as SVMLight or SMO are usually the SVM methods of choice, we shall show that there is a relationship between SMO and MDM that suggests that, at least in their simplest implementations, they should have similar training speeds. Thus, and although we will not discuss it here, the proposed MDM acceleration might be used as a starting point to new ways of accelerating SMO.

1 Introduction

The standard formulation of SVM training seeks to maximize the margin of a separating hyperplane by minimizing $\|W\|^2/2$ subject to $y_i(W \cdot X_i + b) \geq 1, i = 1, \ldots, N$; we assume a linearly separable training sample $\mathcal{S} = \{(X_i, y_i)\}$ with $y_i = \pm 1$. Any pair (W, b) verifying the previous restrictions is said to be in canonical form. However, in practice the problem actually solved is the simpler dual problem of maximizing $L_D(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j X_i \cdot X_j$ subject now to $\alpha_i \geq 0$, $\sum_i \alpha_i y_i = 0$. The optimal weight W^o can be then written in the so–called dual form as $W^o = \sum \alpha_i^o y_i X_i$ and patterns for which $\alpha_i^o > 0$ are called support vectors (SV). Among the many methods proposed to maximize $L_D(\alpha)$, Joachim's SVMLight [1] is probably the fastest one. It repeatedly solves simpler maximization problems working with a restricted set of q candidate SVs. When $q = 2$ these reduced problems can be solved analytically and SVMLight coincides then with Platt's SMO algorithm [2], also a popular, efficient and quite simple algorithm.

SVM training can also alternatively cast [3] as a Nearest Point Problem (NPP), where we want to find the closest points $W_{\pm}^*$ in the convex hulls of the positive (i.e., $y = 1$) and negative samples. More precisely we want to minimize $\|W_+ - W_-\|^2 = \sum_{i,j} \alpha_i \alpha_j y_i y_j X_i \cdot X_j$ where $\sum_{i=1}^N \alpha_i = 2$, $\sum_{i=1}^N y_i \alpha_i = 0$ $\alpha_i \geq 0$. If we work with a zero bias term b (which can be compensated working with extended vectors $X' = (X, 1)$), NPP can be further reduced [4] to a Minimum Norm Problem (MNP), where we want to find the vector W in the convex hull $C(\tilde{\mathcal{S}})$ of the set $\tilde{\mathcal{S}}$ with the smallest norm. More precisely, we want to minimize now $\|W\|^2 = \sum_{i,j} \alpha_i \alpha_j y_i y_j X_i \cdot X_j$ with $\sum_{i=1}^N \alpha_i = 1$, $\alpha_i \geq 0$.

*With partial support of Spain's TIN 2004–07676 and TIN 2007–66862 projects. The first author is kindly supported by FPU-MEC grant reference AP2006-02285.

ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

MNP is a much older problem than SVM training and two algorithms to solve it have been adapted to SVM construction, the Gilbert–Schlesinger–Kozinec (GSK) [5, 6, 4] and the Mitchell–Demyanov–Malozemov (MDM) [7, 6]. The MDM algorithm iteratively updates a weight vector $W_t = \sum \alpha_j^t y_j X_j$, selecting two vectors $X_L^t = \arg\min \{y_j W_t \cdot X_j\}$, $X_U^t = \arg\max \left\{y_j W_t \cdot X_j,\ \alpha_j^t > 0\right\}$, at each step and updating W_t to $W_{t+1} = W_t + \lambda_t(y_L^t X_L^t - y_U^t X_U^t) = W_t + \lambda_t D_t$, where

$$\lambda_t = \min\left\{\alpha_U^t, \frac{-W_t \cdot D_t}{\|D_t\|^2}\right\} = \min\left\{\alpha_U^t, \frac{\Delta_t}{\|D_t\|^2}\right\}.$$

These updates can be expressed in terms of the α multipliers as $\alpha_L^{t+1} = \alpha_L^t + \lambda_t$, $\alpha_U^{t+1} = \alpha_U^t - \lambda_t$, $\alpha_i^{t+1} = \alpha_i^t \ \ \forall i \neq L, U$. We note in passing that the above formulae allow an easy kernel extension of the MDM algorithm that only requires keeping track of the α_j^t coefficients, the approximate margins $d_j^t = y_j W_t \cdot X_j$ and the norms $\|W_t\|^2$. Observe that if $\lambda_t = \alpha_U^t$, $\alpha_U^{t+1} = 0$. Thus, the MDM algorithm can remove previous wrong SV choices (something that GSK cannot).

At first sight, one could look at the MDM algorithm as too far away from state–of–the–art SVM training methods such as SVMLight or SMO. However, notice that at each step MDM chooses two updating vectors X_L and X_U and adjusts two multipliers, just as SMO does. Thus, an alternative way of solving the MNP problem could be to proceed a la SMO. In fact, we shall show in section 2 that, after minor simplifications, doing so results in just the MDM algorithm. While much faster than GSK, in some problems MDM may also require too many iterations for wrong SV removal (see figure 1). In this work we propose a procedure to accelerate MDM training in which we detect the possible presence of training "cycles" and use them to construct a better updating vector. Although those updates have a higher computational cost than the standard ones, we shall illustrate in section 4 that significant training savings can be achieved. Moreover, the previously mentioned MDM-SMO relationship suggests that similar procedures could be successfully applied to accelerate SMO SVM training. We will not, however, discuss them here, and the paper will end with a short conclusion and some pointers to further work.

2 MDM and SMO

As just mentioned, one possibility when solving MNP could be to do it a la SMO; that is, to work at each step with two coefficients that we denote as $\alpha_{i_1}, \alpha_{i_2}$, fixing all the other α_j and arriving at an analytical solution for this restricted version of MNP. The new multipliers to be considered are thus $\alpha'_{i_1} = \alpha_{i_1} + \delta_{i_1}$, $\alpha'_{i_2} = \alpha_{i_2} + \delta_{i_2}$ and $\alpha'_j = \alpha_j$ for all other. In dual form, the new vector W' considered has thus the form $W' = W + \delta_{i_1} y_{i_1} X_{i_1} + \delta_{i_2} y_{i_2} X_{i_2}$. While several different heuristics can be used in SMO to choose the first coefficient α_{i_1} [8], the second multiplier α_{i_2} is selected so that there is a maximum decrease in $\|W'\|^2$. Taking into account the restriction $\sum_i \alpha'_i = 1$, we must have $\delta_{i_2} = -\delta_{i_1}$ and, therefore, we have $W' = W + \delta_{i_1}\left(y_{i_1} X_{i_1} - y_{i_2} X_{i_2}\right) = W + \delta_{i_1} D$ and $\|W'\|^2 = \|W\|^2 +$

ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

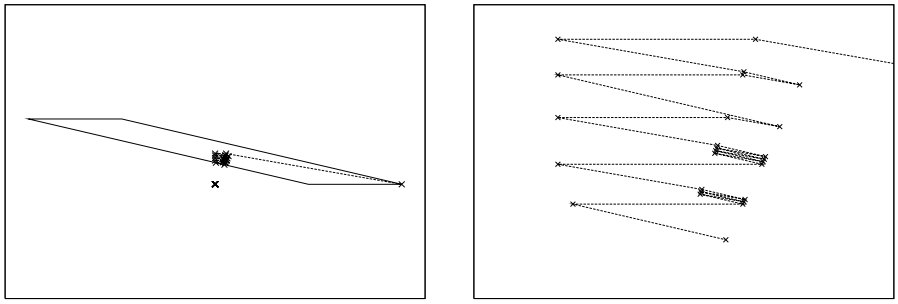

Fig. 1: The MDM algorithm may require many iterations even in simple problems. However (right) they may have a cyclical structure.

$2\delta_{i_1} W \cdot D + \delta_{i_1}^2 \|D\|^2 = \Phi(\delta_{i_1})$, where $D = y_{i_1} X_{i_1} - y_{i_2} X_{i_2}$. The last equation implies that the optimal $\delta_{i_1}^*$ is given by $\delta_{i_1}^* = -W \cdot D / \|D\|^2$. In turn, this yields $\Phi(\delta_{i_1}^*) = \|W + \delta_{i_1}^* D\|^2 = \|W\|^2 - (W \cdot D)^2 / \|D\|^2$. Thus, once the multiplier α_{i_1} has been chosen, we can select i_2 as $i_2 = \arg\max_j \left\{ (W \cdot D_j)^2 / \|D_j\|^2 \right\}$, where $D_j = y_{i_1} X_{i_1} - y_j X_j$. We can simplify the choice of i_2 if we drop the denominator $\|D_j\|^2$, selecting then i_2 as

$$i_2 = \arg\max_j \left\{ (W \cdot D_j)^2 \right\} = \arg\max_j \left\{ |W \cdot (y_{i_1} X_{i_1} - y_j X_j)| \right\}.$$

The usual choice in SMO for α_{i_1} is to take as the vector X_{i_1} the one that most violates the KKT conditions. It is not clear how to bring this to an MNP setting but a way of doing so is to observe that the pattern X_{i_1} for which the KKT conditions are violated most is the one that sets the margin of the current vector W and that, moreover, the SMO update will improve the margin of the new vector W' at the selected X_{i_1}. The same heuristic can be applied in the MNP setting if we choose $i_1 = \arg\min_i \{ y_i (W \cdot X_i) \}$. Now, since we also want to maximize $|W \cdot (y_{i_1} X_{i_1} - y_j X_j)|$, we can simply take $i_2 = \arg\max_j \{ y_j W \cdot X_j \}$. Furthermore, the optimal $\delta_{i_1}^*$ becomes now $\delta_{i_1}^* = -W \cdot D / \|D\|^2 > 0$, and, therefore, $\delta_{i_2}^* = -\delta_{i_1}^* < 0$. Since we must ensure that α'_{i_2} remains positive, α_{i_2} must be positive to begin with. Hence, we have to slightly refine our selection of i_2 to $i_2 = \arg\max_{j, \alpha_j > 0} \{ y_j W \cdot X_j \}$. Now, and as discussed in the previous section, the choices for i_1 and i_2 are exactly the updates that are used in the bias-free MDM problem, i.e., we have $L = i_1$, $U = i_2$. In other words, solving the MNP problem a la SMO yields the MDM algorithm.

3 Update Cycles in MDM Training

The main reason for the slow convergence of the GSK algorithm is that it requires many iterations to remove a wrong initial choice X_i of a support vector, i.e., to

ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

	BCW	HD	GCr	PID	Ban	Thy	Spl
$2\sigma^2$	10^4	$10^{3.5}$	10^3	10^2	1	1	$10^{1.5}$
C	10	10	1000	10	10	$10^{1.5}$	1

Table 1: SVM $2\sigma^2$ and C parameters used.

achieve $\alpha_i = 0$. On the other hand, we can remove a SV X_{i_2} in the MDM algorithm if we would have $\delta^*_{i_2} = W \cdot D/\|D\|^2 = -\alpha_{i_2}$. However, this is not guaranteed to happen and, in fact, wrong initial choices of support vectors may also be hard to correct by the MDM algorithm. This is shown in figure 1: it is clear that the optimal W^* must be placed on the rhomboid face closer to 0; however, if we start the MDM iterations on the lower right vector, it may take quite a few iterations before the MDM updates reach that face. In other words, the MDM will eventually reach the appropriate lower dimensional face removing thus the wrong initial vector choice, but quite often after many iterations as those described in figure 1, right.

The zigzags in that figure illustrate the more general presence of cycles within MDM training, where if a certain update vector D_T has reappeared after K steps from a former instance D_{T-K}, it is likely that, somewhere later in training, the intermediate updates $D_{T-j}, 1 \leq j \leq K-1$, will also be repeated, that is $D_{t-j} = D_{T-j}$ for some $t > T$. For instance, this is what happens in figure 1, that also suggests a possible way out of this. Notice that if we used $V = \lambda_1 D_1 + \lambda_2 D_2$, with D_1, D_2 the first updates in figure 1, right, as an updating direction, we could reach the right face in a single update minimizing $\|W_2 + \lambda V\|^2$. Once there, just another step brings us to the optimal W^*. For a more general cycle $D_{T-K}, D_{T-K+1}, \ldots, D_{T-1}, D_T = D_{T-K}$, we could define $V = \sum_{j=1}^{K} \lambda_{T-j} D_{T-j}$ and, instead of the MDM standard update, consider one of the form $W_{T+1} = W_T + \lambda_T V$, where λ_T minimizes the norm $\|W_T + \lambda V\|^2$. It can be easily seen that the optimal λ_T is given by $\lambda_T = -W_T \cdot V/\|V\|^2$. These new updates require the efficient computation of $W_{T+1} \cdot V$ and $\|V\|^2$, as well as that of the new coefficients α_j^{T+1} and approximate margins $d_j^{T+1} = y_j W_{T+1} \cdot X_j$.

Let $\mathcal{I} = \{i_1, \ldots, i_M\}$ be the index set of those patterns X_{i_h} that appear as either X_L or X_U in V. We have then $V = \sum_{j=1}^{K} \lambda_{T-j} \left(y_L^{T-j} X_L^{T-j} - y_U^{T-j} X_U^{T-j} \right) = \sum_{h=1}^{M} \mu_h y_{i_h} X_{i_h}$, where for $T - K \leq p, q \leq T-1$ we use the notation $\mu_h = \sum_{X_{i_h} = X_L^p} \lambda_p - \sum_{X_{i_h} = X_U^q} \lambda_q$. It is clear now that $\|V\|^2 = \sum_{m,n} \mu_m \mu_n y_{i_m} y_{i_n} X_{i_m} \cdot X_{i_n}$. As for the new d_j^{T+1}, we have $d_j^{T+1} = y_j W_{T+1} \cdot X_j$ and, hence,

$$d_j^{T+1} = y_j W_T \cdot X_j + y_j \lambda_T \sum_{h=1}^{M} \mu_h y_{i_h} X_{i_h} \cdot X_j. = d_j^T + y_j \lambda_T \sum_{h=1}^{M} \mu_h y_{i_h} X_{i_h} \cdot X_j.$$

Finally, $\sum_j \alpha_j^{T+1} y_j X_j = W_{T+1} = W_T + \lambda_T V = \sum_j \alpha_j^T y_j X_j + \lambda_T \sum_{h=1}^{M} \mu_h y_{i_h} X_{i_h}$, and it follows that $\alpha_j^{T+1} = \alpha_j^T$ if $j \notin \mathcal{I}$, $\alpha_{i_h}^{T+1} = \alpha_{i_h}^T + \lambda_T \mu_{i_h}$, where the last

	Std. MDM		Accel. MDM		
Dataset	# KOs	# iters.	# KOs	# iters.	# KO reduct.
BCW	1,354.01	2.76	862.94	1.41	36.27 %
HD	2,699.47	5.51	919.43	1.45	65.94 %
GCr	4,549,640.18	2,523.37	508,670.08	215.961	88.82 %
PID	24,564.49	17.91	18,053.01	10.67	26.51 %
Ban	622,856.43	65.26	610,072.00	63.76	2.05 %
Thy	930.80	2.44	487.34	1.00	47.64 %
Spl	50,501.98	8.84	50,337.28	8.81	0.33 %

Table 2: Average number of kernel operations and iterations (both in thousands) for the standard and accelerated MDM algorithm and % reduction in kernel operations achieved by the second method.

updates must also verify $0 \leq \alpha_{i_h}^{T+1} = \alpha_{i_h}^T + \lambda_T \mu_{i_h} \leq 1$. If $\mu_{i_h} > 0$, the relevant bound is the upper one, while the lower one has to be checked if $\mu_{i_h} < 0$. This implies that for these special cycle updates we must clip λ_T from above as $\lambda_T \leq \min\left\{(1 - \alpha_{i_h}^T)/\mu_{i_h} : \mu_{i_h} > 0\right\}$ and also as $\lambda_T \leq \min\left\{-\alpha_{i_h}^T/\mu_{i_h} : \mu_{i_h} < 0\right\}$. We will numerically illustrate next the proposed procedure.

4 Numerical Experiments

We shall illustrate the previous procedure over the datasets breast cancer–Wisconsin (BCW), heart disease (HD), Pima Indians' diabetes (PID), German credit (GCr), banana (Ban), thyroid (Thy) and splice (Spl). While originally in the UCI database, we shall work here with their versions in G. Rätsch's Benchmark Repository [9]. These problems are not linearly separable and, as usual, we will consider margin slacks ξ_i and use a square penalty term $C \sum_i \xi_i^2$, where C is an appropriately chosen parameter. An advantage of this is that the linear theory extends straightforwardly to the square penaly setting by considering extended vectors and kernels [8]. The original patterns have 0 mean and 1 variance componentwise and work with the Gaussian kernel $k(x, x') = \exp\left(-\|x - x'\|^2/2\sigma^2\right)$. Table 1 summarizes the $2\sigma^2$ and C values used. These last values have been obtained by an optimal grid search. In the experiments we have not used the train–test splits in [9], performing instead a random 10×10 cross validation procedure. Notice that at the optimal W^* we must have for all support vectors $y_j W^* \cdot X_j = m^*$, where m^* denotes the optimal margin. We have thus stopped training at the first iteration t at which $0 \leq y_U^t W_t \cdot X_U^t - y_L^t W_t \cdot X_L^t \leq \gamma \| W_t\|^2$, for some precision parameter γ.

We have compared standard MDM training with the accelerated procedure described in section 3 in terms of both the average number of iterations and of kernel operations required. We report them in table 2 for $\gamma = 0.001$. The table also shows as a percentage the reduction of the number of kernel operations

achieved. As it can be seen, while the efficiency gain is small for the splice and banana datasets, there are large savings for the other five, that can go above 65 % for the heart disease problem and above 88% for German credit datasets (similar results are obtained for the less precise $\gamma = 0.01$). These gains come without any loss in accuracy efficiency. We do not report here by space limitations, but they are essentially the same for standard and accelerated MDM, and mostly coincide with those reported in [10] (see also [9]). Notice however that accuracies are not really comparable: remember that our train–test splits are different from those used in [10] and we use square penalties instead of the linear ones in [10].

5 Conclusions and Further Work

In this work we have shown that we can take advantage of the presence of cycles in the updating sequence D_j used by the MDM algorithm by collapsing these vectors in a single update that gives a better minimizing direction. As we have seen, very large savings in the number of iterations and kernel operations can be obtained for some problems (although they can be more modest in some others).

While initially designed to solve the Minimum Norm Problem, we have shown that the MDM algorithm can be related to SMO, one of the most efficient and popular SVM training methods. This suggests that it may be worthwhile to study in more detail the relationship between the SMO and MDM algorithms in the general non zero bias case and to try to derive direct acceleration methods for SMO. This, the relationship between the MDM and the $q = 2$ SVMLight algorithm and the characterization of those problems for which the proposed procedure is likely to produce larger training savings, are being studied.

References

[1] T. Joachims. Making large-scale support vector machine learning practical. *Advances in Kernel Methods - Support Vector Machines*, pages 169–184, 1999.

[2] J.C. Platt. Fast training of support vector machines using sequential minimal optimization. *Advances in Kernel Methods - Support Vector Machines*, pages 185–208, 1999.

[3] K.P. Bennett and E.J. Bredensteiner. Duality and geometry in svm classifiers. *Proc. 17th Int. Conf. Machine Learning*, pages 57–64, 2000.

[4] V. Franc and V. Hlavac. An iterative algorithm learning the maximal margin classiffier. *Pattern Recognition*, 36:1985–1996, 2003.

[5] E.G. Gilbert. Minimizing the quadratic form on a convex set. *SIAM J. Contr.*, 4:61–79, 1966.

[6] S.S. Keerthi, S.K. Shevade, C. Bhattacharyya, and K.R.K. Murphy. A fast iterative nearest point algorithm for support vector machine classifier design. *IEEE Transactions on Neural Networks*, 11(1):124–136, 2000.

[7] B.F. Mitchell, V.F. Dem'yanov, and V.N. Malozemov. Finding the point of a polyhedron closest to the origin. *SIAM J. Contr.*, 12:19–26, 1974.

[8] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press, Cambridge, 2000.

[9] G. Rätsch. Benchmark repository. ida.first.fraunhofer.de/projects/bench/benchmarks.htm.

[10] G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for AdaBoost. *Machine Learning*, 42(3):287–320, March 2001.