SoftDoubleMinOver: A Simple Procedure for Maximum Margin Classification

Thomas Martinetz, Kai Labusch, and Daniel Schneegaß

Institute for Neuro- and Bioinformatics
University of Lübeck
D-23538 Lübeck, Germany
martinetz@informatik.uni-luebeck.de
http://www.inb.uni-luebeck.de

Abstract. The well-known MinOver algorithm is a simple modification of the perceptron algorithm and provides the maximum margin classifier without a bias in linearly separable two class classification problems. DoubleMinOver as a slight modification of MinOver is introduced, which now includes a bias. It is shown how this simple and iterative procedure can be extended to SoftDoubleMinOver for classification with soft margins. On benchmarks the extremely simple Soft-DoubleMinOver algorithm achieves the same classification performance with the same computational effort as sophisticated Support-Vector-Machine software.

1 Introduction

The Support-Vector-Machine (SVM) [1], [2] has been applied very successfully and become a standard tool in classification and regression tasks (e.g. [3], [4], [5]). A major drawback, particularly for industrial applications where easy and robust implementation is an issue, is the large Quadratic-Optimization problem which has to be solved. The users have to rely on existing software packages, which are hardly comprehensive and, in some cases at least, error-free. This is in contrast to most Neural Network approaches, where learning has to be simple and incremental almost by definition. The pattern-by-pattern nature of learning in Neural Networks usually leads to simple training procedures which can easily be implemented. It is desirable to have similar training procedures also for the SVM.

Several approaches for obtaining more or less simple incremental training procedures for the SVM have been introduced so far [6], [7], [8], [9]. We want to mention in particular the Kernel-Adatron by Friess, Cristianini, and Campbell [6] and the Sequential-Minimal-Optimization algorithm (SMO) by Platt [7]. The SMO algorithm by Platt is the most widespread iterative training procedure for SVMs. It is fast and robust, but it is still not yet of a pattern-by-pattern nature. It is not yet as easy to implement as one is used from Neural Network approaches.

Therefore, in this paper we will revisit and extend the MinOver algorithm which was introduced by Krauth and Mézard [10] for constructing synaptic weight matrices of optimal stability in spin-glass models of Neural Networks. It is well-known that MinOver also yields the maximum margin hyperplane without a bias in linearly separable classification problems. We will reformulate MinOver. In this reformulation the MinOver algorithm is a slight modification of the perceptron learning rule and could hardly be simpler. Then we extend MinOver to DoubleMinOver which remains as simple as MinOver but now provides the maximum margin solution also for linear classifiers with a bias. Then we will show how DoubleMinOver can be extended to SoftDoubleMinOver for classification with soft margins. SoftDoubleMinOver yields solutions also in case the classification problem is not linearly separable. Finally, we will present results and comparisons on standard benchmark problems.

2 The DoubleMinOver Algorithm

Given a linearly separable set of patterns $\mathbf{x}_i \in \mathbb{R}^D$, $i = 1, \ldots, N$ with corresponding class labels $y_i \in \{-1, 1\}$. We want to find the hyperplane which separates the patterns of these two classes with maximum margin. The hyperplane for classification is determined by its normal vector $\mathbf{w} \in \mathbb{R}^D$ and its bias $b \in \mathbb{R}$. It achieves a separation of the two classes, if

$$y_i(\mathbf{w}^T\mathbf{x}_i - b) > 0 \qquad \text{for all} \qquad i = 1, \ldots, N$$

is valid. The margin Δ of this separation is given by

$$\Delta(\mathbf{w}, b) = \min_i [y_i(\mathbf{w}^T\mathbf{x}_i - b)/||\mathbf{w}||].$$

Maximum margin classification is given by the $\mathbf{w}_*$, $||\mathbf{w}_*|| = 1$ and b_* for which $\Delta(\mathbf{w}_*, b_*) = \Delta_*$ becomes maximal.

A simple and iterative algorithm which provides the maximum margin classification in linearly separable cases is the well-known MinOver algorithm introduced by [10] in the context of constructing synaptic weight matrices of optimal stability in spin-glass models of Neural Networks. However, it only provides the maximum margin solution if no bias b is included. The MinOver algorithm yields a vector $\mathbf{w}_t$ which converges against the maximum margin solution with increasing number of iterations t. This is valid as long as a full separation, i.e. a $\Delta_* > 0$, exists. The MinOver algorithm works like the perceptron algorithm, with the slight modification that with each training step t the pattern $\mathbf{x}_{\min}(t)$ out of the training set $\mathcal{T} = \{\mathbf{x}_i | i = 1, \ldots, N\}$ with the worst, i.e. the minimum distance (overlap) $y_i\mathbf{w}^T\mathbf{x}_i$ is chosen ($b = 0$). Hence, the name MinOver.

We now modify MinOver such that a bias b can be included. For this purpose we divide $\mathcal{T}$ into the set $\mathcal{T}^+$ of patterns with class label $y_i = +1$ and the set $\mathcal{T}^-$ of patterns with class label $y_i = -1$. Then, instead of looking for the pattern with minimum distance on $\mathcal{T}$, we look for the pattern $\mathbf{x}_{\min^+}(t)$ with minimum distance $y_i(\mathbf{w}^T\mathbf{x}_i - b)$ on $\mathcal{T}^+$ and for the pattern $\mathbf{x}_{\min^-}(t)$ with minimum distance $y_i(\mathbf{w}^T\mathbf{x}_i - b)$ on $\mathcal{T}^-$. Hence, the name DoubleMinOver.

With t_{max} as the number of desired iterations, DoubleMinOver works like follows:

```
0. Set $t = 0$, choose a $t_{max}$, and set $\mathbf{w}_{t=0} = 0$.
1. Determine the $\mathbf{x}_{\min^+}(t)$ out of $\mathcal{T}^+$ and the $\mathbf{x}_{\min^-}(t)$ out of $\mathcal{T}^-$ which
   minimize $h(\mathbf{x}_i) = y_i\mathbf{w}_t^T\mathbf{x}_i$, respectively.
2. Set $\mathbf{w}_{t+1} = \mathbf{w}_t + \mathbf{x}_{\min^+}(t) - \mathbf{x}_{\min^-}(t)$.
3. Set $t = t + 1$ and go to 1.) if $t < t_{max}$.
4. Determine $\mathbf{x}_{\min^+}$ and $\mathbf{x}_{\min^-}$ according to 1. and set $b = \frac{1}{2}\left(h\left(\mathbf{x}_{\min^+}\right) - h\left(\mathbf{x}_{\min^-}\right)\right)$.
```

2.1 On the Convergence of DoubleMinOver

For a given $\mathbf{w}$, the margin $\Delta(\mathbf{w}, b)$ is maximized with $b(\mathbf{w})$ for which the margin to both classes is equal, i.e., for which

$$\min_{\mathbf{x}_i \in \mathcal{T}^+} [y_i(\mathbf{w}^T \mathbf{x}_i - b(\mathbf{w}))] = \min_{\mathbf{x}_i \in \mathcal{T}^-} [y_i(\mathbf{w}^T \mathbf{x}_i - b(\mathbf{w}))]$$

is valid. This leads to the expression of step 4. for the bias

$$b(\mathbf{w}) = \frac{1}{2} \left(\min_{\mathbf{x}_i \in \mathcal{T}^+} y_i \mathbf{w}^T \mathbf{x}_i - \min_{\mathbf{x}_i \in \mathcal{T}^-} y_i \mathbf{w}^T \mathbf{x}_i \right) \ .$$

We now have to look for the $\mathbf{w}$ which maximizes $\Delta(\mathbf{w}) = \Delta(\mathbf{w}, b(\mathbf{w}))$. We obtain

$$\begin{aligned} \Delta(\mathbf{w}) &= \min_{\mathbf{x}_i \in \mathcal{T}} \frac{y_i(\mathbf{w}^T \mathbf{x}_i - b(\mathbf{w}))}{||\mathbf{w}||} = \min_{\mathbf{x}_i \in \mathcal{T}^+} \frac{y_i \mathbf{w}^T \mathbf{x}_i - b(\mathbf{w})}{||\mathbf{w}||} = \min_{\mathbf{x}_i \in \mathcal{T}^-} \frac{y_i \mathbf{w}^T \mathbf{x}_i + b(\mathbf{w})}{||\mathbf{w}||} \\ &= \frac{1}{2} \left(\min_{\mathbf{x}_i \in \mathcal{T}^+} \frac{y_i \mathbf{w}^T \mathbf{x}_i}{||\mathbf{w}||} + \min_{\mathbf{x}_i \in \mathcal{T}^-} \frac{y_i \mathbf{w}^T \mathbf{x}_i}{||\mathbf{w}||} \right) . \end{aligned}$$

With

$$\mathcal{Z} = \left\{ \mathbf{z}_{ij} = \mathbf{x}_i - \mathbf{x}_j \mid \forall\, (i,j) : x_i \in \mathcal{T}^+, x_j \in \mathcal{T}^- \right\}$$

we obtain

$$\Delta(\mathbf{w}) = \frac{1}{2} \min_{\mathbf{z}_{ij}} \frac{\mathbf{w}^T \mathbf{z}_{ij}}{||\mathbf{w}||} \ .$$

In this formulation we can directly apply the $\mathcal{O}(t^{-1/2})$ convergence proofs for MinOver in [10] or [11]. For both methods in fact even a $\mathcal{O}(t^{-1})$ convergence is given [11].

2.2 DoubleMinOver in its dual formulation and with kernels

The vector $\mathbf{w}_t$ which determines the dividing hyperplane is given by

$$\mathbf{w}_t = \sum_{\tau=0}^{t-1} (\mathbf{x}_{\min^+}(\tau) - \mathbf{x}_{\min^-}(\tau)) = \sum_{\mathbf{x}_i \in \mathcal{T}} y_i n_i(t) \mathbf{x}_i$$

with $n_i(t) \in \mathbb{N}_0$ as the number of times each $\mathbf{x}_i$ has been used for training up to step t. $\sum_{\mathbf{x}_i \in \mathcal{T}} n_i(t) = 2t$ is valid. In this dual formulation the training step of the DoubleMinOver algorithm simply consists of searching for $\mathbf{x}_{\min^+}(t)$ and $\mathbf{x}_{\min^-}(t)$ and increasing their corresponding n_i by one.

In the dual representation the inner product $\mathbf{w}^T \mathbf{x}$ can be written as

$$\mathbf{w}^T \mathbf{x} = \sum_{\mathbf{x}_i \in \mathcal{T}} y_i n_i \mathbf{x}_i^T \mathbf{x} \ . \tag{1}$$

If the input patterns $\mathbf{x} \in \mathbb{R}^D$ are transformed into another (usually higher dimensional) feature space by a transformation $\mathbf{\Phi}(\mathbf{x})$, DoubleMinOver can work with the Kernel $K(\mathbf{x}, \mathbf{x}') = \mathbf{\Phi}(\mathbf{x})^T \mathbf{\Phi}(\mathbf{x}')$ instead of the usual inner product. At each step of the algorithm where $\mathbf{w}^T \mathbf{x}_i$ occures one then uses

$$\mathbf{w}^T \mathbf{\Phi}(\mathbf{x}_i) = \sum_{\mathbf{x}_j \in \mathcal{T}} y_j n_j K(\mathbf{x}_j, \mathbf{x}_i) = y_i h\left(\mathbf{x}\right) \ . \tag{2}$$

3 SoftDoubleMinOver

So far linear separability of the patterns was required. Since this is not always the case, the concept of a "soft margin" was introduced in [1], [2]. With a soft margin training patterns are allowed to be misclassified for a certain cost. With DoubleMinOver we can easily realize a 2-norm soft margin.

In Cristianini and Shawe-Taylor [12] it is shown that solving the 2-norm soft margin classification problem within a feature space implicitly defined by a kernel $K(\mathbf{x}, \mathbf{x}')$ is equivalent to solving the hard margin problem within a feature space defined by a kernel $\hat{K}(\mathbf{x}, \mathbf{x}')$ for which $\hat{K}(\mathbf{x}_i, \mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j) + C^{-1}\delta_{ij}$ is valid for each $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{T}$, with δ_{ij} as the Kronecker δ which is 1 for $i = j$ and 0 otherwise. Within the feature space defined by $\hat{K}(\mathbf{x}, \mathbf{x}')$ the training data are linearly separable by construction. The scalar parameter C determines the "hardness" of the margin. The smaller C, the softer the margin. For $C \to \infty$ we obtain the dual formulation of DoubleMinOver (hard margin).

The SoftDoubleMinOver algorithm in its dual formulation then works like follows:

0. Set $t = 0$, choose a t_{max}, and set $n_i = 0$ for $i = 1, \ldots, N$.
1. Determine $\mathbf{x}_{\min^+}(t) \in \mathcal{T}^+$ and $\mathbf{x}_{\min^-}(t) \in \mathcal{T}^-$ which minimize

$$\hat{h}\left(\mathbf{x}_i\right) = y_i \sum_{\mathbf{x}_j \in \mathcal{T}} y_j n_j \left(K(\mathbf{x}_j, \mathbf{x}_i) + \frac{\delta_{ij}}{C} \right) = \frac{n_i}{C} + h\left(\mathbf{x}_i\right) \ .$$

2. Increase $n_{\min^+}$ and $n_{\min^-}$ by one, respectively.
3. Set $t = t + 1$ and go to 1.) if $t < t_{max}$.
4. Determine $\mathbf{x}_{\min^+}$ and $\mathbf{x}_{\min^-}$ according to 1. and set $b = \frac{1}{2}\left(\hat{h}\left(\mathbf{x}_{\min^+}\right) - \hat{h}\left(\mathbf{x}_{\min^-}\right)\right)$.

Having determined the n_i and b via SoftDoubleMinOver, the class assignment of a new pattern $\mathbf{x}$ takes place, of course, based on the original kernel. The decision depends on whether

$$\sum_{\mathbf{x}_i \in \mathcal{T}} y_i n_i K(\mathbf{x}_i, \mathbf{x}) - b$$

is larger or smaller than zero.

4 Experimental results on benchmark problems

To validate and compare the performance of SoftDoubleMinOver[1] we tested it on a number of common classification benchmark problems. The classification benchmarks stem from the UCI[2], DELVE[3] and STATLOG[4] [13] collection. We compare our results

[1] A SoftDoubleMinOver package is available at `http://www.inb.uni-luebeck.de/maxminover`

[2] UCI Repository: `http://www.ics.uci.edu/~mlearn/MLRepository.html`

[3] DELVE Datasets: `http://www.cs.utoronto.ca/~delve/index.html`

[4] STATLOG Datasets: `http://www.niaad.liacc.up.pt/statlog/index.html`

Table 1. Classification results obtained with SoftDoubleMinOver on standard benchmarks. For comparison the results obtained with the C-SVM Implementation of the OSU-SVM Toolbox and those reported in the Fraunhofer benchmark repository are listed.

			C2-SDMO		OSU-SVM		Reference
Benchmark	#TR	#TE	Seconds/Iter.	ERR	Seconds	ERR	ERR_{REF}
banana	400	4900	$0.030/200$	11.6 ± 0.83	0.031	10.4 ± 0.46	12.0 ± 0.66
br-cancer	200	77	$0.019/100$	27.1 ± 4.96	0.012	28.2 ± 4.62	26.0 ± 4.74
diabetis	468	300	$0.060/300$	23.3 ± 1.78	0.065	23.1 ± 1.82	24.0 ± 1.73
fl-solar	666	400	$0.148/300$	32.4 ± 1.80	0.229	32.3 ± 1.82	32.0 ± 1.82
german	700	300	$0.142/200$	24.1 ± 2.67	0.177	24.0 ± 2.17	24.0 ± 2.07
heart	170	100	$0.010/100$	15.5 ± 3.22	0.006	15.2 ± 3.21	16.0 ± 3.26
image	1300	1010	$0.811/2000$	13.1 ± 4.33	0.812	9.8 ± 0.62	3.0 ± 0.60
ringnorm	400	7000	$0.030/300$	2.6 ± 0.41	0.021	2.5 ± 0.38	1.7 ± 0.12
splice	1000	2175	$0.615/500$	16.1 ± 0.65	0.654	14.9 ± 0.78	11.0 ± 0.66
titanic	150	2051	$0.034/1500$	22.4 ± 0.96	0.013	22.3 ± 1.04	22.0 ± 1.02
waveform	400	4600	$0.047/300$	11.4 ± 0.59	0.045	10.7 ± 0.53	10.0 ± 0.43
thyroid	140	75	$0.004/200$	4.2 ± 2.40	0.003	4.1 ± 2.42	4.8 ± 2.19
twonorm	400	7000	$0.057/200$	2.4 ± 0.13	0.033	2.4 ± 0.14	3.0 ± 0.23

#Tr : number of training data, #Te : number of test data

with those reported in the SVM-benchmark repository of the Fraunhofer Institute[5] and results we obtained with the C-SVM of the OSU-SVM Matlab Toolbox[6] that is based on SMO [7].

Each result reported in the benchmark repository of the Fraunhofer Institute is based on 100 different partitionings of the respective benchmark problem data into training and test sets (Except for the splice and image benchmark which consist of 20 partitionings). For classification they used the standard C-SVM with RBF-kernels. The reported classification result is the average over all 100 realizations. Each partitioning is available from this repository.

Table 1 lists the average classification errors we obtained with SoftDoubleMinOver and the OSU-SVM on the different benchmark problems. We used the default parameter settings of the OSU-SVM Toolbox. As the Fraunhofer Institute we used RBF-kernels, and we took their kernel widths γ. The C values in SoftDoubleMinOver and the OSU-SVM where chosen such that the minimum error is obtained. On all benchmarks the simple SoftDoubleMinOver is as fast as and achieves results comparable to those of the OSU-SVM and those reported in the Fraunhofer benchmark repository after only a few training steps. On the ”ringnorm”, the ”image” and the ”splice” benchmark both the OSU-SVM as well as SoftDoubleMinOver are significantly worse than the Fraunhofer reference. By either performing more iterations for SoftDoubleMinOver or tweeking the parameters of the OSU-SVM one can obtain comparable results for these benchmarks, too.

[5] Benchmark Repository: `http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm`

[6] OSU SVM Classifier Toolbox: `http://www.ece.osu.edu/~maj/osu_svm/`

5 Conclusions

The main purpose of this paper is to present a very simple, incremental algorithm which solves the maximum margin classification problem with or without kernels and with or without a soft margin. SoftDoubleMinOver as an extension of MinOver learns by simply iteratively selecting patterns from the training set. Based on previous work it can be shown that SoftDoubleMinOver converges like $\mathcal{O}(t^{-1})$ to the exact solution, with t as the number of iteration steps. The incremental nature of the algorithm allows one to trade-off the computational time and the precision of the obtained hyperplane. The computational effort increases linearly with the number of training patterns N. In experiments on standard benchmark problems SoftDoubleMinOver achieves a performance comparable to the widespread OSU-SVM which is based on the SMO-algorithm. However, SoftDoubleMinOver as a ”three-liner” is much easier to implement, and with its pattern-by-pattern nature it might be a good starting point for a real on-line learning procedure for maximum margin classification.

References

1. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning **20** (1995) 273–297
2. Vapnik, V.: The Nature of Statistical Learning Theory. Springer-Verlag, New York (1995)
3. LeCun, Y., Jackel, L., Bottou, L., Brunot, A., Cortes, C., Denker, J., Drucker, H., Guyon, I., Muller, U., Sackinger, E., Simard, P., Vapnik, V.: Comparison of learning algorithms for handwritten digit recognition. Int.Conf.on Artificial Neural Networks (1995) 53–60
4. Osuna, E., Freund, R., Girosi, F.: Training support vector machines:an application to face detection. CVPR’97 (1997) 130–136
5. Schölkopf, B.: Support vector learning (1997)
6. Friess, T., Cristianini, N., Campbell, C.: The kernel adatron algorithm: a fast and simple learning procedure for support vector machine. Proc. 15th International Conference on Machine Learning (1998)
7. Platt, J.: Fast Training of Support Vector Machines using Sequential Minimal Optimization. In: Advances in Kernel Methods - Support Vector Learning. MIT Press (1999) 185–208
8. Keerthi, S.S., Shevade, S.K., Bhattacharyya, C., Murthy, K.R.K.: A fast iterative nearest point algorithm for support vector machine classifier design. IEEE-NN **11** (2000) 124–136
9. Li, Y., Long, P.: The relaxed online maximum margin algorithm. Machine Learning **46(1-3)** (2002) 361–387
10. Krauth, W., Mezard, M.: Learning algorithms with optimal stability in neural networks. J.Phys.A **20** (1987) 745–752
11. Martinetz, T.: Minover revisited for incremental support-vector-classification. Lecture Notes in Computer Science **3175** (2004) 187–194
12. Cristianini, N., Shawe-Taylor, J.: Support Vector Machines (and other kernel-based learning methods). Cambridge University Press, Cambridge (2000)
13. King, R., Feng, C., Shutherland, A.: Statlog: comparison of classification algorithms on large real-world problems. Applied Artificial Intelligence **9** (1995) 259–287