

# Fast Support Vector Machine Classification using linear SVMs

Karina Zapién Arreola  
INSA de Rouen

LITIS  
76801 St Etienne du Rouvray, France  
kzapien@insa-rouen.fr

Janis Fehr      Hans Burkhardt  
University of Freiburg

Chair of Pattern Recognition and Image Processing  
79110 Freiburg, Germany  
fehr@informatik.uni-freiburg.de

## Abstract

We propose a classification method based on a decision tree whose nodes consist of linear Support Vector Machines (SVMs). Each node defines a decision hyperplane that classifies part of the feature space. For large classification problems (with many Support Vectors (SVs)) it has the advantage that the classification time does not depend on the number of SVs. Here, the classification of a new sample can be calculated by the dot product with the orthogonal vector of each hyperplane. The number of nodes in the tree has shown to be much smaller than the number of SVs in a non-linear SVM, thus, a significant speedup in classification time can be achieved. For non-linear separable problems, the trivial solution (zero vector) of a linear SVM is analyzed and a new formulation of the optimization problem is given to avoid it.

## 1. Introduction

In terms of classification-speed, SVMs [13] are still outperformed by many standard classifiers when used in large problems. For non-linear kernels  $k$ , the classification function can be written as in Eq. (1), thus, the classification complexity lies in  $\Omega(n)$  for a problem with  $n$  SVs. However, for linear problems, the classification function has the form of Eq. (2), allowing classification in  $\Omega(1)$  by calculating the dot product with the normal vector  $\mathbf{w}$  of the hyperplane. In order to achieve a classification speedup we propose a classification method based on a tree whose nodes consist of linear SVM (Fig.(1)).

$$f(\mathbf{x}) = \text{sign} \left( \sum_{i=1}^m y_i \alpha_i k(\mathbf{x}_i, \mathbf{x}) + b \right) \quad (1)$$

$$f(\mathbf{x}) = \text{sign} (\langle \mathbf{w}, \mathbf{x} \rangle + b) \quad (2)$$

This paper is structured as follows: first we give a brief overview of related work. Section 2 describes our algo-

rithm in detail including a discussion of the zero solution problem. Experiments are presented in Section 3, followed by conclusions in Section 4.

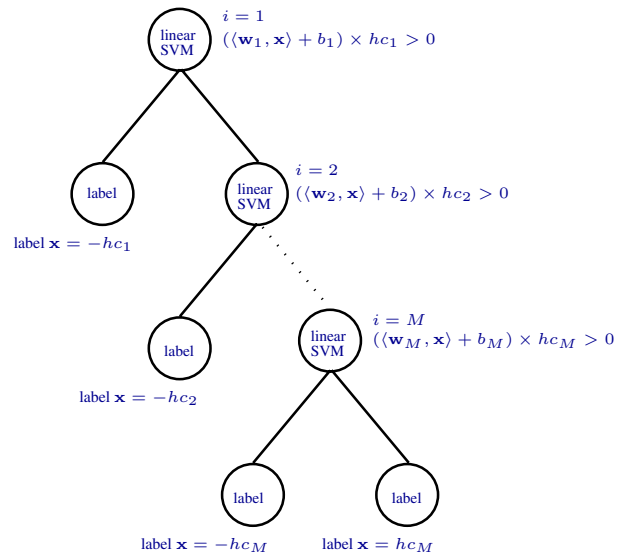


Figure 1. Decision tree with linear SVM

### 1.1. Related Work

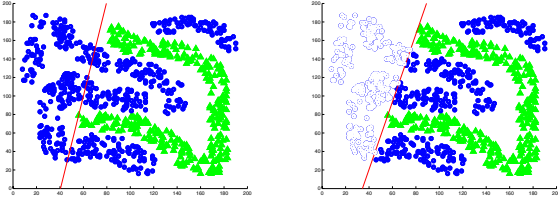
Recent work on SVM classification speedup mainly focused on the reduction of the decision problem: A method called *RSVM* (Reduced Support Vector Machines) was proposed by Lee [10], it preselects a subset of training samples as SVs and solves a smaller Quadratic Programming problem. Lei and Govindaraju [11] introduced a reduction of the feature space using principal component analysis and Recursive Feature Elimination. Burges and Schölkopf [3] proposed a method to approximate  $\mathbf{w}$  by a list of vectors associated with coefficients  $\alpha_i$ . All these methods yield good

speedup, but are fairly complex and computationally expensive. Our approach on the other hand, was endorsed by the work of Bennett et al. [1] who experimentally proved that inducing a large margin in decision trees with linear decision functions improved the generalization ability.

## 2. SVM Trees

The algorithm is described for binary problems, an extension to multiple-class problems can be realized with different techniques like one vs. one or one vs. rest [7].

**Basic Algorithm** At each node  $i$  of the tree, a hyperplane is found that correctly classifies all samples in one class (this class will be called the “hard” class, denoted  $hc_i$ ). Then, all correctly classified samples of the other class (the “soft” class) are removed from the problem, Fig. (2). The



**Figure 2. Problem fourclass [6]. Left: hyperplane for the first node. Right: Problem after first node (“hard” class = triangles).**

decision which class is to be assigned “hard” is taken in a greedy manner for every node (see Section 2.2). The algorithm terminates when the remaining samples all belong to the same class. Fig.(3) shows a training sequence. Section 2.3 further improves the approach, but first we give a formalization of the basic approach.

**Definition 1** Let  $m, m_1$  and  $m_{-1} \in \mathbb{N}$ ,  $\mathcal{C} = \{1, \dots, m\}$  and  $m = m_1 + m_{-1}$ . Without loss of generality we can define samples  $\mathbf{x}_i \in \mathbb{R}^n$  and labels  $y_i, i \in \mathcal{C}$ , such that:

**Class 1 (Positive Class)**  $\mathcal{C}_1 = \{1, \dots, m_1\}$ ,  $y_i = 1$  for all  $i \in \mathcal{C}_1$ , center of gravity  $\mathbf{s}_1 = \frac{1}{m_1} \sum_{i \in \mathcal{C}_1} \mathbf{x}_i$ , with a global penalization value  $D_1$  and individual penalization values  $C_i = D_1$  for all  $i \in \mathcal{C}_1$ .

**Class -1 (Negative Class)**  $\mathcal{C}_{-1} = \{m_1 + 1, \dots, m_1 + m_{-1}\}$ ,  $y_i = -1$  for all  $i \in \mathcal{C}_{-1}$ , center of gravity  $\mathbf{s}_{-1} = \frac{1}{m_{-1}} \sum_{i \in \mathcal{C}_{-1}} \mathbf{x}_i$ , with a global penalization value  $D_{-1}$  and individual penalization values  $C_i = D_{-1}$  for all  $i \in \mathcal{C}_{-1}$ .

**Training** In order to train a SVM that classifies perfectly one class, different weights of the slack-variables  $C_i$  are set for each class in Equations (3) and (7). We define a “hard” class  $\mathcal{C}_k$  with  $D_k \rightarrow \infty$  and one “soft” class  $\mathcal{C}_{\bar{k}}$  with  $D_{\bar{k}} \ll D_k$  in each node, where  $k = 1$  and  $\bar{k} = -1$ , or  $k = -1$  and  $\bar{k} = 1$ . For large  $D_k$ , most samples of the

class  $k$  will be correctly classified, while the errors of the “soft” class will be minimized.

### Linear SVM Primal Problem

$$\min_{\mathbf{w} \in \mathbb{R}^n, b \in \mathbb{R}, \xi \in \mathbb{R}^m} \frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^m C_i \xi_i \quad (3)$$

$$\text{subject to } y_i (\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \geq 1 - \xi_i, \quad i \in \mathcal{C}, \quad (4)$$

$$\xi_i \geq 0, \quad i \in \mathcal{C}. \quad (5)$$

### Linear SVM Dual Formulation

$$\max_{\alpha \in \mathbb{R}^m} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \quad (6)$$

$$\text{subject to } 0 \leq \alpha_i \leq C_i, \quad i \in \mathcal{C}, \quad (7)$$

$$\sum_{i=1}^m \alpha_i y_i = 0. \quad (8)$$

## 2.1. Zero Solution

The described algorithm works fine for many problems, but in some cases the optimization process converges to a trivial solution: the zero vector. In order to get a deeper understanding of the zero solution problem, we take advantage of the convex hull interpretation of SVMs [1], where the solution of a SVM is interpreted as the orthogonal hyperplane to the segment between the two closest points of a reduced convex hull of the classes.

**Theorem 1** If the convex hull of the “hard” class  $\mathcal{C}_1$  intersects the convex hull of the “soft” class  $\mathcal{C}_{-1}$ , then  $\mathbf{w} = \mathbf{0}$  is a feasible point for the primal Problem (3) if  $D_{-1} \geq \max_{i \in \mathcal{C}_1} \{\lambda_i\} \cdot D_1$ , where

$$\mathbf{p} = \sum_{i \in \mathcal{C}_1} \lambda_i \mathbf{x}_i,$$

is a convex combination for a point  $\mathbf{p}$  that belongs to both convex hulls.

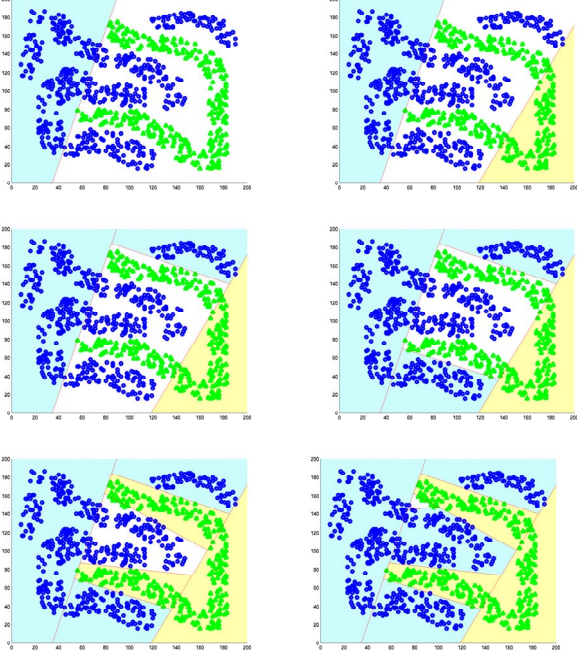
**Proof Sketch** This can be easily shown by setting  $\forall i \in \mathcal{C}_1 : \alpha_i = \lambda_i D_1 \leq D_1$  and  $\forall j \in \mathcal{C}_{-1} : \alpha_j = \lambda_j D_1 \leq \max_{j \in \mathcal{C}_1} \{\lambda_j\} D_1 \leq D_{-1}$  by assumption. It can be checked that these values produce a feasible point for the dual problem. Calculation of the orthogonal vector  $\mathbf{w}$  of the primal problem with these values leads to the zero solution.

**Theorem 2** If the center of gravity  $\mathbf{s}_{-1}$  of class  $\mathcal{C}_{-1}$  is inside of the convex hull of class  $\mathcal{C}_1$ , then it can be written as

$$\mathbf{s}_{-1} = \sum_{i \in \mathcal{C}_1} \lambda_i \mathbf{x}_i \quad \text{and} \quad \mathbf{s}_{-1} = \sum_{j \in \mathcal{C}_{-1}} \frac{1}{m_{-1}} \mathbf{x}_j$$

with  $\lambda_i \geq 0$  for all  $i \in \mathcal{C}_1$  and  $\sum_{i \in \mathcal{C}_1} \lambda_i = 1$ . If additionally,  $D_1 \geq \lambda_{\max} D_{-1} m_{-1}$ , where  $\lambda_{\max} = \max_{i \in \mathcal{C}_1} \{\lambda_i\}$ , then  $\mathbf{w} = \mathbf{0}$  is a feasible point for the primal Problem (3).

**Proof Sketch** By setting  $\alpha_i = \lambda_i D_1$  for all  $i \in \mathcal{C}_1$  and  $\alpha_j = \lambda_j D_1$  for all  $j \in \mathcal{C}_{-1}$ , checking if this is a feasible solution for the dual problem and calculating the orthogonal vector  $\mathbf{w}$  of the primal problem with these values, the zero solution will be obtained.



**Figure 3. Sequence of hyperplanes for nodes 1-6 of the tree.**

## 2.2. H1-SVM Problem Formulation

To avoid the zero vector, the penalization value for the hard class can be reduced. Another solution is proposed by the modification of the original SVM optimization problem:

### H1-SVM Primal Problem

$$\min_{\mathbf{w} \in \mathbb{R}^n, b \in \mathbb{R}} \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i \in \mathcal{C}_k} y_i (\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \quad (9)$$

$$\text{subject to } y_i (\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \geq 1 \text{ for all } i \in \mathcal{C}_k, \quad (10)$$

where  $k = 1$  and  $\bar{k} = -1$ , or  $k = -1$  and  $\bar{k} = 1$ .

This new formulation constrains Eq. (10) to classify all samples in the class  $\mathcal{C}_k$  perfectly, forcing a ‘‘hard’’ convex hull (H1) for  $\mathcal{C}_k$ . The number of misclassification on the other class  $\mathcal{C}_{\bar{k}}$  is added to the objective function, so, the solution is a trade off between a maximal margin and a minimal of misclassification in the ‘‘soft’’ class  $\mathcal{C}_{\bar{k}}$ .

### H1-SVM Dual Formulation

$$\max_{\alpha \in \mathbb{R}^m} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \quad (11)$$

$$\text{subject to } 0 \leq \alpha_i \leq C_i, i \in \mathcal{C}_k, \quad (12)$$

$$\alpha_j = 1, j \in \mathcal{C}_{\bar{k}}, \quad (13)$$

$$\sum_{i=1}^m \alpha_i y_i = 0, \quad (14)$$

where  $k = 1$  and  $\bar{k} = -1$ , or  $k = -1$  and  $\bar{k} = 1$ .

This problem can be similarly solved as Problem (6) by using the same algorithm (SMO [12]), and adding some modifications to force  $\alpha_i = 1 \forall i \in \mathcal{C}_{\bar{k}}$ .

**Theorem 3** For the H1-SVM the zero solution can only occur if  $|\mathcal{C}_k| \geq (n-1)$  and there exists a linear combination of the sample vectors in the ‘‘hard’’ class  $\mathbf{x}_i \in \mathcal{C}_k$  and the sum of the sample vectors in the ‘‘soft’’ class,  $\sum_{i \in \mathcal{C}_{\bar{k}}} \mathbf{x}_i$ .

**Proof** Without loss of generality, let the ‘‘hard’’ class be class  $\mathcal{C}_1$ . Then,

$$\begin{aligned} \mathbf{w} &= \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i = \sum_{i \in \mathcal{C}_1} \alpha_i \mathbf{x}_i - \sum_{i \in \mathcal{C}_{-1}} \alpha_i \mathbf{x}_i \\ &= \sum_{i \in \mathcal{C}_1} \alpha_i \mathbf{x}_i - \sum_{i \in \mathcal{C}_{-1}} \mathbf{x}_i. \end{aligned} \quad (15)$$

If we define  $\mathbf{z}_i = \sum_{i \in \mathcal{C}_{-1}} \mathbf{x}_i$  and  $|\mathcal{C}_1| \geq (n-1) = \dim(\mathbf{z}_i) - 1$ , there exist  $\{\alpha_i\}, i \in \mathcal{C}_1, \alpha_i \neq 0$  such that

$$\mathbf{w} = \sum_{i \in \mathcal{C}_1} \alpha_i \mathbf{x}_i - \mathbf{z}_i = \mathbf{0}. \quad \square$$

The usual threshold calculation ([9] and [12]) can no longer be used to define the hyperplane since this will give some mistakes in the ‘‘hard’’ class. For this approach, letting the ‘‘hard’’ class = 1, the threshold is calculated as in Eq. (16).

$$b = \frac{\min_{i \in \mathcal{C}_1} \langle \mathbf{w}, \mathbf{x}_i \rangle + \max_{\{j \in \mathcal{C}_{-1}, \langle \mathbf{w}, \mathbf{x}_j \rangle < 0\}} \langle \mathbf{w}, \mathbf{x}_j \rangle}{2} \quad (16)$$

If the ‘‘hard’’ class = -1,  $b$  is calculated as in Eq. (17).

$$b = \frac{\max_{i \in \mathcal{C}_1} \langle \mathbf{w}, \mathbf{x}_i \rangle + \min_{\{j \in \mathcal{C}_{-1}, \langle \mathbf{w}, \mathbf{x}_j \rangle > 0\}} \langle \mathbf{w}, \mathbf{x}_j \rangle}{2} \quad (17)$$

Four hyperplanes are calculated at each node  $i$ : two with the solution of the *Linear-SVM* Problem (one with  $hc_i = \mathcal{C}_1$ , other with  $hc_i = \mathcal{C}_{-1}$ ) and similarly, two more with the *H1-SVM* Problem. The hyperplane that reduces the problem most is added to the tree. Further analysis has to be done to avoid calculating unnecessary hyperplanes since each one has a computational cost equivalent to one SMO resolution.

## 2.3. Tree Optimization Heuristics

The basic algorithm can be improved with some heuristics. First, a greedy heuristic (*Gr-Heu*) of choosing the ‘‘hard’’

class for each node can be extended with hyperplanes orthogonal to the solution of the previous node. Experiments (Table 1) have shown that this can lead out of local optimization minimum. Within these orthogonal hyperplanes and the four previous ones, the solution that reduces the problem the most is added to the tree as node.

Second, after training, redundant nodes are pruned from the tree to avoid some overfitting. Further pruning is not done since important information can lie in the middle of the space (area corresponding to the last leaves).

### 3. Experiments

The experiments were conducted on several problems<sup>1</sup>. *DNA sequences* [2] (1330 training samples, 1446 test samples, 180 features); *Faces* [4] (9172 training samples, 4262 test samples, 576 features); *Heart disease* [5] (90 training samples, 190 test samples, 13 features); *Isolated Letter Speech* [5] (155950 training samples, 1559 test samples, 617 features); *USPS* [8] (18063 training samples, 7291 test samples, 256 features). The data was split into training and test sets and normalized to minimum and maximum feature values (Min-Max) or standard deviation (Std-Dev).

DNA (Min-Max)	RBF Kernel	H1-SVM	H1-SVM Gr-Heu	RBF/H1	RBF/H1 Gr-Heu
Nr. SVs or Hyperplanes	798	3	3	266	266
Training Time	00:02.35	00:01.84	00:03.74	1.28	0.63
Classification Time	00:06.70	00:01.86	00:01.81	<b>3.6</b>	<b>3.7</b>
Classif. Accuracy %	93.64 %	90.25 %	90.25 %	1.04	1.04

Faces (Min-Max)	RBF Kernel	H1-SVM	H1-SVM Gr-Heu	RBF/H1	RBF/H1 Gr-Heu
Nr. SVs or Hyperplanes	2206	4	4	551.5	551.5
Training Time	14:55.23	10:55.70	14:21.99	1.37	1.04
Classification Time	03:13.60	00:14.73	00:14.63	<b>13.14</b>	<b>13.23</b>
Classif. Accuracy %	95.78 %	91.01 %	91.01 %	1.05	1.05

Heart (Std-Dev)	RBF Kernel	H1-SVM	H1-SVM Gr-Heu	RBF/H1	RBF/H1 Gr-Heu
Nr. SVs or Hyperplanes	51	13	8	9.38	18.75
Training Time	00:00.04	00:00.07	00:00.03	0.57	1.33
Classification Time	00:00.06	00:00.05	00:00.09	<b>1.6</b>	<b>1.6</b>
Classif. Accuracy %	81.05 %	70.53 %	72.11 %	1.15	1.12

Isolat (Std-Dev)	RBF Kernel	H1-SVM	H1-SVM Gr-Heu	RBF/H1	RBF/H1 Gr-Heu
Nr. SVs or Hyperplanes	35340	344	344	102.73	102.73
Training Time	07:13.75	18:51.98	1:04:11.3	0.38	0.11
Classification Time	03:01.99	00:32.85	00:36.43	<b>5.54</b>	<b>5</b>
Classif. Accuracy %	96.15 %	94.42 %	94.42 %	1.02	1.02

USPS (Min-Max)	RBF Kernel	H1-SVM	H1-SVM Gr-Heu	RBF/H1	RBF/H1 Gr-Heu
Nr. SVs or Hyperplanes	3597	49	49	73.41	73.41
Training Time	00:44.74	00:22.70	02:09.58	1.97	0.35
Classification Time	01:58.59	00:19.99	00:20.07	<b>5.93</b>	<b>5.91</b>
Classif. Accuracy %	95.82 %	93.76 %	93.76 %	1.02	1.02

**Table 1. Experiments with different datasets. Last two columns: ratio between methods.**

<sup>1</sup>These experiments were run in a computer with a P4, 2.8 GHz and 1G in Ram.

Speedup comparison with similar works is difficult to state since most publications ([1], [10]) used datasets with less than 1000 samples, where the training and testing time are negligibles.

### 4. Conclusion

We have presented a new method for fast SVM classification. Compared to non-linear SVM and speedup methods our experiments showed a competitive speedup (up to factor 13 - see *Faces* Table 1) while achieving reasonable classification results (loosing about 2% compared to non-linear methods). The advantages of this approach clearly lies in its simplicity since no parameter has to be tuned. A possible extension would be the introduction of non-linear kernels at a lower level of the tree to improve the classification rate.

### References

- [1] K. P. Bennett, N. Cristianini, J. Shawe-Taylor, and D. Wu. Enlarging the margins in perceptron decision trees. *Machine Learning*, 41(3):295–313, 2000.
- [2] P. Brazdil and J.Gama. Statlog datasets. <http://www.liacc.up.pt/ML/statlog/datasets.html>.
- [3] C. Burges and B. Schölkopf. Improving speed and accuracy of support vector learning machines. In Mozer, Jordan, and Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 375–381. MIT Press, MA, 1997.
- [4] P. Carbonetto. Faces data. <http://www.cs.ubc.ca/~pcarbo/>.
- [5] C. B. D.J. Newman, S. Hettich and C. Merz. UCI repository of machine learning databases. <http://www.ics.uci.edu/~mllearn/MLRepository.html>.
- [6] T. K. Ho and E. M. Kleinberg. Building projectable classifiers of arbitrary complexity. In *Proceedings of the 13th International Conference on Pattern Recognition*. Vienna, Austria, 1996.
- [7] C. Hsu and C. Lin. *A Comparison of Methods for Multi-Class Support Vector Machines*. Technical report, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, 2001. 19, 2001.
- [8] J. J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994.
- [9] S. Keerthi, S. Shevade, C. Bhattacharyya, and K. Murthy. *Improvements to Platt’s SMO Algorithm for SVM Classifier Design*. Technical report, Dept. of CSA, Bangalore, India, 1999.
- [10] Y. Lee and O. Mangasarian, editors. *RSVM: Reduced Support Vector Machines*. 2001 SIAM International Conference, Chicago, Philadelphia, 2001.
- [11] S. R. Lin H. and A. L., editors. *Speeding Up Multi-class SVM Evaluation by PCA and Feature Selection*. 2005 SIAM Workshop, Newport Beach, CA, 2005.
- [12] B. Schölkopf and A. Smola. *Learning with Kernels*. The MIT Press, Cambridge, MA, USA, 2002.
- [13] V. Vapnik. *The Nature of Statistical Learning Theory*. New York: Springer Verlag, 1995.