ONLINE INDEPENDENT SUPPORT VECTOR MACHINES

Francesco Orabona, Claudio Castellini and Giulio Sandini
DIST, University of Genoa, Italy

INTRODUCTION

Support Vector Machines (SVMs) are a machine learning method widely employed in, e.g., visual recognition, medical diagnosis and robotic control. One of their most interesting characteristics is that the solution achieved is *sparse:* a few samples (*support vectors*, SV) usually account for most of the complexity of the classification task. Both the training and testing time crucially depend on the number of SVs, and it is then very important to keep it small, trying at the same time to retain the accuracy of the solution. This is especially evident in online settings where a large amount of unknown data is expected to be available as the system adapts to the changing environment.

In fact, it is known that the solution to a SVM classification problem *can* be reduced without losing accuracy [1], but the proposed method cannot be used in an online setting, since the number of SVs is restored each time new samples are acquired. *Rank reduction approximations* have been proposed to further address this problem [2], but as well, they require prior knowledge about the training set.

We present a new algorithm, *Online Independent SVMs*, in which a reduced number of SVs is retained, exploiting their *linear independence* in the feature space. The algorithm breaks the well-known linear dependency between the number of SVs and the number of training points [3]. Sparseness can be further increased by weakening the concept of linear independence, trading accuracy for sparseness.

Experiments reveal that OISVMs achieve an excellent accuracy vs. compactness trade-off in general, while retaining the full accuracy of SVMs in case a finite-dimensional kernel.

SVMS IN A NUTSHELL

Given a set of *l* samples and labels $\{\mathbf{x}_i, y_i\}$, labels belonging to $\{-1,1\}$, the usual SV classification method finds a *separating hyperplane* $f(\mathbf{x})=\mathbf{w}\cdot\mathbf{x}+b$ which enforces the constraints $y_i(\mathbf{w}\cdot\mathbf{x}_i+b)-1+\xi_i\geq 0$ and has maximum distance from both groups of samples (*margin*). The problem is usually solved by minimizing the following expression:

$$\min_{\mathbf{w}}\left(\|\mathbf{w}\|^2 + C\sum_{i=1}^{l} L(\xi_i)\right)$$

where C is an error penalty and $L(\cdot)$ is an error functional. If the data set is not linearly separable, as is often the case, the *kernel trick* is applied: a nonlinear map $\Phi(\mathbf{x})$ is implicitly introduced using a kernel function $K(\mathbf{x},\mathbf{y})=\Phi(\mathbf{x})\cdot\Phi(\mathbf{y})$. The solution is then written as

$$f(\mathbf{x}) = \sum_{i=1}^{l} \alpha_i y_i \mathrm{K}(\mathbf{x}, \mathbf{x}_i)$$

that is, the decision surface is a sum of basic blocks obtained using the kernel function. Those samples for which the coefficients α are non-zero are called Support Vectors and play a crucial role in evaluating the solution. The training and testing time depend on their number, n_{SV}.

KEEPING IT SMALL

Downs et al. [1] pointed out that the SVs are not necessarily independent in the feature space (i.e., the space induced by the map Φ), hence n_{SV} can be reduced without losing accuracy. But in their approach this simplification occurs *after* the training phase, so this method cannot be applied to an online setting, were the solution must be updated after each sample. The idea is then to *decouple* the concept of "base" vectors, used to build the classification function, from the sample points used to find out the αs. If the base vectors span the same space of all the training points, the solution found will be equivalent to a normal training using all the points. Using the Representer Theorem [4] one can write

$$\mathbf{w} = \sum_{i=1}^{l} \beta_i \phi(\mathbf{x}_i)$$

Enforcing the Karush-Kuhn-Tucker conditions on *this*, more general version of the problem we obtain that β_i-αy_i must be in the null space of K. When a new sample is available we check whether it should be added to the set of base vectors, therefore limiting its growth in time. An effective test is that of checking whether the new sample is linearly independent from the already stored base vectors.

CHECKING IT ONLINE

In general, checking linear independence is done via decomposition or by looking at the eigenvalues of a matrix; but here we want to check whether a single vector is linearly independent from a set of vectors, as fast as possible. We do this by checking how well the vector can be approximated by a linear combination of the vectors in the set [5]:

$$\Delta = \min_{\mathbf{d}} \left\| \sum_{j\in \mathrm{B}} d_j \phi(\mathbf{x}_j) - \phi(\mathbf{x}_{t+1}) \right\|^2$$

If $\Delta > 0$ then $\mathbf{x}_{t+1}$ is linearly independent with respect to the basis. In practice we check whether $\Delta \leq \eta$ where $\eta > 0$ is a tolerance factor, and we expect that larger values of η lead to worse accuracy, but also to fewer bases vectors. As a matter of fact, if η is set at machine precision, this method retains the exact accuracy of SVMs.
Expanding the above equation and applying the kernel trick, we get

$$\Delta = \min_{\mathbf{d}} \left(\mathbf{d}^T \mathbf{K}_{BB}\mathbf{d} - 2\mathbf{d}^T\mathbf{k} + K(\mathbf{x}_{t+1},\mathbf{x}_{t+1})\right) = K(\mathbf{x}_{t+1},\mathbf{x}_{t+1}) - \mathbf{k}^T K_{BB}^{-1}\mathbf{k}$$

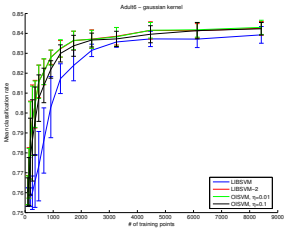
The inverse of $\mathbf{K}_{BB}$ is efficiently updated using the incremental formula of the inverse.
The training method largely follows Keerthi et al. [6]. Let D ∈ {1,…,*l*}; then the unconstrained primal problem is

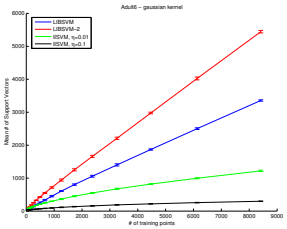
$$\min_{\beta}\left(\frac{1}{2}\beta^T \mathbf{K}_{DD}\beta + \frac{1}{2}C\sum_{i=1}^{l}\max\left(0, 1 - y_i K_{i,D}\beta\right)^2\right)$$

Then, we explicitly set D = B, assuring thus that the solution to the problem is unique, since $\mathbf{K}_{BB}$ is full rank by construction. Newton's method **[…]** can then be used.

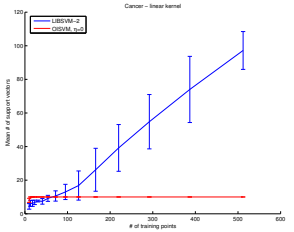
EXPERIMENTAL RESULTS

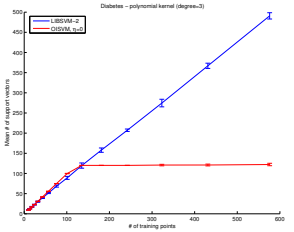
We have used the UCI *adult6* database (11220 samples, 123 features), and other smaller common datasets. For each benchmark, we display the mean number of retained support vectors on 7 random 75%/25% train/test runs (y-axis), as more training samples are acquired (x-axis). We compare against LIBSVM [7], a standard SVM implementation, and LIBSVM-2 which is the same library modified as suggested by the Authors in order to use the squared norm of the errors in the cost function. We used also the norm-1 version to compare the sparseness. In the case of finite-dimensional kernels, we only show the performance of LIBSVM-2 against OISVMs with at machine precision; in the case of the infinite-dimensional kernel, we show curves for various values of η. Bigger η mean bigger sparseness and more loss of accuracy.





Database	Mean difference with LIBSVM-2 test error	Mean % of the SV retained vs norm-2	Mean % of the SV retained vs norm-1
Cancer	-0.11% (ni=0.9)	10%	21.5%
Diabetes	-0.19% (ni=0.2)	26.4%	40.5%
German	0.057% (ni=0.1)	6.1%	9.3%





In the case of finite-dimensional kernels, we only show the mean of number support vectors obtained by LIBSVM-2 against OISVMs at machine precision. The classification rate is not shown because it is *exactly* equal to the one of LIBSVM-2. As one can see, as opposed to the linear behavior of LIBSVM-2, OISVMs quickly attain a constant number of support vectors and then stop adding new ones to the base set. The numbers of bases match exactly the dimensions of the related feature spaces, given by

$$\binom{m+deg-1}{deg}$$

where *m* is the dimension of the input space and *deg* is the degree of the polynomial kernel (obviously 1 in the case of linear kernels).

CONCLUSIONS

A new method is presented to keep Support Vector Machines "small" and "fast". The method works by avoiding using support vectors which are linearly dependent of previous ones in the feature space. The primal SVM problem is directly minimized via an incremental algorithm which benefits of the small size of the kernel matrix.

Experimental results show that

1. in the case of finite-dimensional kernels, the method attains the theoretical limit of linearly independent support vectors allowed by the feature space, without losing any accuracy;

2. in the case of infinite-dimensional kernels, it dramatically reduces the number of support vectors at the price of a negligible degradation in the accuracy.

Notice that, in this latter case also, it can be used to obtain full precision, choosing the tolerance threshold to be equal to machine precision.

Acknowledgments: the work here presented is partially supported by the EU projects Neurobotics (IST-FP6-001917) and RobotCub (IST-2004-004370).

REFERENCES

[1] T. Downs, K. E. Gates, and A. Masters. Exact simplification of support vectors solutions. *Journal of Machine Learning Research*, 2:293–297, 2001.

[2] F. R. Bach, M. I. Jordan. Predictive low-rank decomposition for kernel methods. *In Proc. of the Twenty-second International Conference on Machine Learning (ICML)*, 2005.

[3] I. Steinwart. Sparseness of support vector machines. *Journal of Machine Learning Research*, 4:1071–1105, 2003.

[4] D. Cox and F. O'Sullivan. Asymptotic analysis of penalized likelihood and related estimators. *Ann. Statist.*, 18:1676–1695, 1990.

[5] Y. Engel, S.Mannor, and R.Meir. Sparse online greedy support vector regression. *In Proc. 13th European Conference on Machine Learning*, 2002.

[6] S. S. Keerthi, O. Chapelle, and D. DeCoste. Building support vector machines with reduced classifier complexity. *Journal of Machine Learning Research*, 8:1–22, 2006.

[7] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.