Toward optimal SVM

Norbert Jankowski & Krzysztof Grąbczewski

Department of Informatics, Nicolaus Copernicus University
ul. Grudziądzka 5, 87-100 Toruń, Poland
{norbert|kgrabcze}@phys.uni.torun.pl, http://www.phys.uni.torun.pl/kmk

Abstract. Although it was proven by Vapnik that SVM is capable of finding optimal solutions, the full success can be achieved only if the adaptive process parameters (C and kernel functions parameters) are set to proper values. Manual parameters tuning is sometimes tricky. We propose a new way of automatic selection of the parameters. The criterion used to estimate C and kernel parameters aims in optimal accuracy on unseen data and simultaneously in small number of SVM's kernels. The results achieved show that it is possible to reduce the user interaction to just starting the learning procedure and obtain simple and accurate SVM models.

1 Introduction

Support Vector Machines [1,2,3] (SVMs) are getting more and more popular. One of the reasons is the guarantee of optimal margin, they offer. In fact the method very often yields high accuracy and efficiency. The idea is applicable to both data classification and function approximation tasks. The statement of the SVM optimization for classification problems may be the following:

$$\min_{\mathbf{w},p,\xi} \quad \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i \tag{1}$$

with constraints:

$$y_i(\mathbf{w}^\top \phi(\mathbf{x}_i) + p) \geq 1 - \xi_i \tag{2}$$

$$\xi_i \geq 0, \quad i = 1, \ldots, n \tag{3}$$

where n is the number of vectors, $\mathbf{x}_i$ is the i'th data vector and $\mathbf{y}_i$ is it's class label (1 or -1 – the binary classification). The dual problem definition is:

$$\min_{\alpha} \quad \frac{1}{2}\alpha^\top \mathbf{Q}\alpha + e^\top \alpha \tag{4}$$

with constraints:

$$0 \leq \alpha_i \leq C, \quad i = 1, \ldots, n \tag{5}$$

$$\mathbf{y}^\top \alpha = 0 \tag{6}$$

where e is the vector of 1s, C (a positive value) defines the upper bound on α_i factors, and Q is a matrix defined by:

$$Q_{ij} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j). \tag{7}$$

The K function is called a kernel and $K(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$.

Mostly used kernels are: gaussian, linear, sigmoidal and polynomial. With the exception of the linear kernel all the others have some parameters of free choice. The gaussian kernel free parameter is the bias b:

$$K(\mathbf{x}, \mathbf{y}) = \exp(-b||\mathbf{x} - \mathbf{y}||^2). \tag{8}$$

The sigmoidal kernel has also a b parameter which in this case plays the role of the slope:

$$K(\mathbf{x}, \mathbf{y}) = \tanh(b\, \mathbf{x}^\top \mathbf{y} + \theta). \tag{9}$$

The main problem with the original definition of SVM was that it's learning procedure (the quadratic programming (QP)) converged very slowly. Recent years have brought a few novel methods of acceleration of the QP procedure for SVM learning. Among the ones deserving the most attention are the methods proposed by Osuna et. al. [4], Joachims [5], Saunder et. al. [6], Platt [7,8] and Chang et. al. [9]. Platt's Sequential Minimal Optimization (SMO) algorithm for the QP problems is very fast and provides an analytical solution. Further improvements to the QP procedure were made by Keerthi et. al. [10].

The SMO algorithm augmented by the ideas presented in [10] yield very fast and accurate solution of SVM learning problem. We have used such a version of SVM in our research.

1.1 The goal

Despite the recent progress, the C parameter of Eq. 1 and the kernel's free parameters must be set manually by means of testing and experience, which can be tiresome. This is why we got interested in a *meta*-level of SVM learning which would be able to automatically determine nearly optimal values of the C and kernel's parameters - the ones leading to as high final accuracy (predicted, not the accuracy on the training set) as possible.

The goal of this paper is to show that it is possible to construct an algorithm for automatic selection of the parameters. The Meta-SVM learning procedure reduces the user task to just starting the process.

2 The Meta-SVM algorithm

Adaptive systems in their learning processes adjust their parameters to build a model for given data set. Adjusting parameters of the learning process is a meta-level learning, hence the name *Meta-SVM learning*.

In our algorithm searching for the most interesting SVM parameters is done by means of a cross-validation for learning performed inside the training set. The inner cross-validation algorithm looks for accurate and relatively small models. We are interested in relatively small models (not just the most accurate), because they are faster in final decision making and because it is known that smaller models (if only they represent the knowledge of given data set) have usually better accuracy [11,12,13] on unseen data (the empirical confirmation, that it is true also in the case of SVM models can be found in section 3).

Cross-validation for learning. The main loop of the algorithm performs a cross-validation (CV) as a learning technique. To run a p-fold CV the data set is divided into p equal (if possible) parts $\mathcal{S}_i,\ i = 1, \ldots, p$. In each iteration of the CV the SVM algorithm is used with different configurations (different values of the C and kernel parameters) to learn the classification of the set $\hat{\mathcal{S}}_i = \bigcup_{k=1,\ldots,p,\ k\neq i} \mathcal{S}_k$. The resulting model is validated with $\mathcal{S}_i$.

To check different settings of C (Eq. 1) and $bias$ (Eq. 8) we spread a grid over them, which explores the region $[C_{min}, C_{max}] \times [bias_{min}, bias_{max}]$, with steps C_{step} and $bias_{step}$ respectively. Observation of the variability of the results, while the C and $bias$ are being changed, clearly shows that the nature of the parameters is rater exponential than linear. Concluding, it is better if C_{min}, C_{max}, $bias_{min}$ and $bias_{max}$ represent the powers of 2 instead of linear values. Our tests let us state, that the most appropriate grid of C and $bias$ is $[-5, 5] \times [-10, 3]$ with step 1. It covers the most interesting regions of C and $bias$, what was observed on all the tested benchmarks. An assumption is that the data set is normalized.

Let $[C_1, \ldots, C_m]$, $[bias_1, \ldots, bias_n]$ be Cs and $bias$es which define the above grid. In each iteration of the cross-validation (for learning), for each C_i and $bias_j$ an independent SVM model is constructed. The k-th iteration stores the accuracies on the validation parts of the cross-validation in the matrix $Acc^k_{\mathcal{S}_k}[i, j]$, the numbers of support vectors composing the machine in $SV^k[i, j]$ and the numbers of unbounded support vectors in $bSV^k[i, j]$.

The matrices $Acc^k_{\mathcal{S}_k}$ let us find the pair $\langle i, j\rangle$ (in fact the C_i and $bias_j$) for which the predicted accuracy is the highest:

$$\langle p, q\rangle = \arg\max_{\langle i,j\rangle} \sum_k Acc^k_{\mathcal{S}_k}[i, j]. \tag{10}$$

The above model selection criterion may be extended by adding a term which puts some constraints on the numbers of support vectors and unbounded support vectors:

$$\langle p, q\rangle = \arg\max_{\langle i,j\rangle} \sum_k \left[Acc^k_{\mathcal{S}_k}[i, j] - \frac{\alpha \cdot SV^k[i, j] + \beta \cdot bSV^k[i, j]}{|\hat{\mathcal{S}}_k|} \right], \tag{11}$$

where $|\hat{\mathcal{S}}_k|$ is the number of vectors in $\hat{\mathcal{S}}_k$. If $\alpha = 0$ and $\beta = 0$ the two criteria are equivalent. In practice the sensible values for α and β are small real numbers (for example $\alpha = 0.15$ and $\beta = 0.05$).

The criterion of Eq. 11 facilitates reduction of the number of support vectors which means that it simplifies the structure of the final SVM model. It can be seen in section 3 that the aim of model simplification may be obtained without the cost of worse classification results (when compared to the results of the criterion defined by Eq. 10).

To reduce the computational cost i.e. the total number of SVMs which must be trained, it is possible to split the whole of algorithm into two phases. In the first phase the grid size may be 4 × 4 ($4C$'s × $4biases$). In the second one two grids of the same size may be composed with the middles in the best (according to Eq. 11) two pairs. Usually it will be enough to perform the second stage with the grids of size 3 × 3. The modified search technique requires 48 (or even 34) SVM models to be built while the search over the $[-5, 5] \times [-10, 3]$ grid with step 1 needs 154 models. Results obtained with these two schemes should be very similar.

3 Results

The results presented in table 1 were computed for the classification benchmarks from the UCI repository [14]. Data sets vary in the numbers of vectors from around hundred up to several thousands. The numbers of vectors, features and classes are placed in table 1. The precise descriptions of the sets may be obtained from [14].

Before each data set was used in the computations it was normalized with 5% rejection ratio which means that each feature was scaled in such a way that all it's values but 5% of extreme minimal and maximal ones, fell into the $[-1, 1]$ interval.

In most of the cases the results come from 10-fold cross-validation tests. Each cross-validation was repeated ten times and average values of accuracy and standard deviation calculated and put into the table 1. We did not perform CV tests for the data sets which had their own test set. In such cases also the training and testing were repeated 10 times and the averaged results are presented.

In the case of more than two class problems, basing on the experience of [15], we decided to use *one-against-rest* method (one SVM model per class). The estimation of the C and kernel parameters was done independently for each of the submodels.

Each benchmark was started with the same configuration. No parameters of SVM learning were manually tuned.

For each benchmark from table 1 the average accuracy is presented for the testing set (TES column) and for the training set (TRS column). The *std* columns present the standard deviations of accuracies for testing and training sets respectively. The last column shows the average number of support vectors.

For each data set there is a table presenting the results of three ways of SVM learning (each in a separate row). The top row presents the results of Meta-SVM learning with $\alpha = 0.15, \beta = 0.05$ and the middle one with $\alpha = 0, \beta = 0$ (see Eq. 11). The bottom row presents the results obtained with the base SVM learning algorithm with the $C = 1$ and $bias = 0.1$ ($bias$ from Eq. 8). Such values of C

Data set	Correctness (%)				#SV	Data set	Correctness (%)				#SV
	TES	std	TRS	std			**TES**	std	TRS	std	
APPENDICITIS (2/106/7)						**HEPATITIS** (2/155/19)					
$(0.15, 0.05)$	**86.69**	0.09	89.69	0.009	32	$(0.15, 0.05)$	**82.73**	0.072	94	0.031	66
$(0, 0)$	**86.26**	0.09	90.89	0.15	46	$(0, 0)$	**80.8**	0.087	94.54	0.033	73
−	**87.71**	0.082	88.88	0.009	35	−	**79.38**	6.6	96	0.01	97
AUSTRALIAN CREDIT (2/690/14)						**IONOSPHERE** (2/200+151/34)					
$(0.15, 0.05)$	**84.97**	0.039	89.8	0.016	211	$(0.15, 0.05)$	**98**	0	99	0	73
$(0, 0)$	**85.68**	0.04	86.49	0.017	383	$(0, 0)$	**98**	0	99	0	125
−	**85.34**	0.04	89.79	0.005	260	−	**98**	0	95	0	100
DIABETES (2/768/8)						**KR-VS-KP** (2/3196/36)					
$(0.15, 0.05)$	**76.7**	0.046	79.2	0.014	367	$(0.15, 0.05)$	**99.65**	0.003	99.93	0.0006	302
$(0, 0)$	**77.15**	0.044	78.7	0.009	394	$(0, 0)$	**99.53**	0.046	99.95	0.0004	517
−	**76.6**	0.045	79.6	0.006	387	−	**96.76**	0.008	97.1	0.001	697
DNA (3/2000+1180/60)						**SONAR** (2/208/60)					
$(0.15, 0.05)$	**94.23**	0.0019	99.41	0.003	1268	$(0.15, 0.05)$	**87.02**	0.073	99.78	0.005	104
$(0, 0)$	**94.27**	0.0038	99.84	0.001	2379	$(0, 0)$	**87.04**	0.074	99.8	0.004	125
−	**64.58**	1e-16	1	0	5803	−	**78.12**	0.088	83.92	0.015	151
FLAG (8/193/28)						**WISCONSIN BREAST** (2/699/9)					
$(0.15, 0.05)$	**42.2**	0.098	71.2	0.074	405	$(0.15, 0.05)$	**96.54**	0.004	97.45	0.003	59
$(0, 0)$	**41.6**	0.11	77.7	0.063	513	$(0, 0)$	**96.68**	0.004	97.17	0.006	110
−	**32.8**	0.098	74.2	0.018	867	−	**96.81**	0.019	97.29	0.0021	76
GLASS (6/214/9)						**VOTES** (2/435/16)					
$(0.15, 0.05)$	**65.2**	0.09	87.34	0.036	293	$(0.15, 0.05)$	**94.25**	0.034	97.54	0.009	89
$(0, 0)$	**64.26**	0.09	87.55	0.038	424	$(0, 0)$	**94.5**	0.03	96.9	0.008	105
−	**56.30**	0.056	63.4	0.02	394	−	**90.6**	0.045	99.2	0.002	245
HEART (2/303/13)						**VOWEL** (6/871/3)					
$(0.15, 0.05)$	**82.54**	0.07	86.48	0.009	118	$(0.15, 0.05)$	**85.3**	0.036	90.1	0.009	579
$(0, 0)$	**82.5**	0.073	85.7	0.018	142	$(0, 0)$	**85.0**	0.034	90.4	0.01	969
−	**80.86**	0.076	89.2	0.005	146	−	**72.5**	0.042	73.1	0.007	1158

Table 1. Results comparison for meta-SVM learning. On the right side of the database name the "(C/V/F)" values inform about the numer of classes (C), the number of vectors (V) and the number of features (F) respectively. The top row for each database presents the results of Meta-SVM learning with $\alpha = 0.15, \beta = 0.05$, the middle one with $\alpha = 0, \beta = 0$ (see Eq. 11), and the last row presents the results obtained with the base SVM learning algorithm with $C = 1$ and $bias = 0.1$.

and *bias* are not accidental – they are selected as the setting which yields good results in average for most of the benchmarks.

The search grid described in section 2 had the same parameters for all the data sets: $C \in [-5, 5]$ with step 1 and $bias \in [-10, 3]$ with step 1. Each Meta-SVM learning was based on 3-fold inner cross-validation.

As it can be seen some results are better for const $C = 1$ and $bias = 0.1$ than for Meta-SVM. The reason is, that $C = 1$ and $bias = 0.1$ are really good in average and it happens that these values are very close to optimal, while finding them with the presented estimation algorithm (especially for small data sets) is not simple. Sometimes it is possible to find values of the parameters which perform well in cross-validation test, however it must be pointed out, that one should not use the whole data to find the best parameters and then run tests with the chosen parameters, because in such case the parameters selection exploits the information contained within the test data (which should be unseen). For example our estimation algorithm run on the whole appendicitis data set gives C and *bias* which then in CV test are 88.82% accurate. Of course, the results, we present in the table are not obtained this way.

Figures 1 and 2 present densities for two benchmark databases: wisconsin breast cancer (fig. 1) and glass (fig. 2). For each database there are four density plots. The first (upper left) presents the density for the Meta-SVM criterion (Eq. 11), the second (upper right) presents the predicted accuracy, and 3-rd and 4-th (lower left and right) show the densities of the numbers of support vectors and unbounded support vectors respectively. The red square represents the optimal values of the $\log_2 C$ and the $\log_2 bias$.

4 Conclusions

It is known that SVM is an optimal margin classifier, however the setting of the learning process parameters plays crucial role in accurate SVM construction. The advantages of the scheme of automated parameters choosing presented here are confirmed by the results of several benchmarks. The method usually enlarges the test accuracy (improves the generalization abilities) of the final model (except accidental lucks as presented in section 3. The structure of the final SVM is significantly smaller which reduces the computation costs (typical SVMs have more complex structure than RBF networks trained on the same data).

We do not compare the results with any other kind of systems, because the goal of this paper is not to prove that SVM is the best method for all data sets, or that it is able to win all the competitions. The aim is to show an efficient method which automatically selects the SVM learning parameters and finds simple and accurate SVM models for given data set without user interaction.

References

1. V. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, New York, 1995.

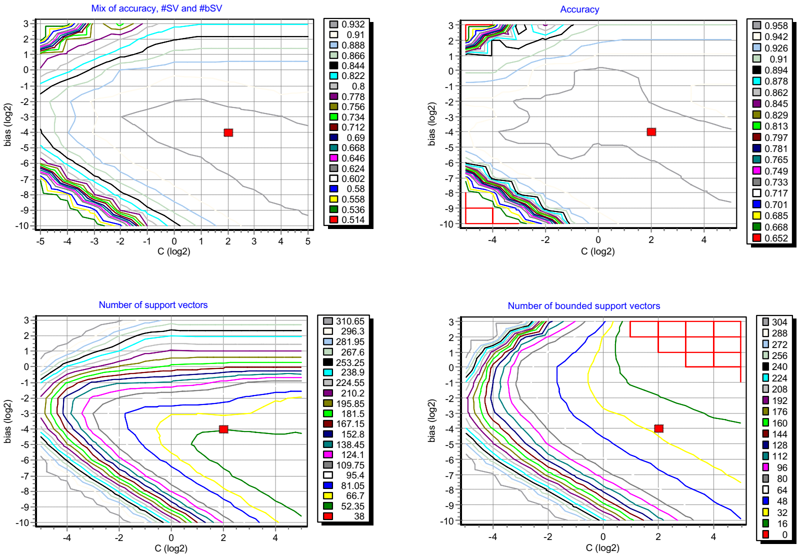

Fig. 1. Densities of Meta-SVM criterion, accuracy, number of support vectors and number of unbounded support vectors for wisconsin breast cancer benchmark.

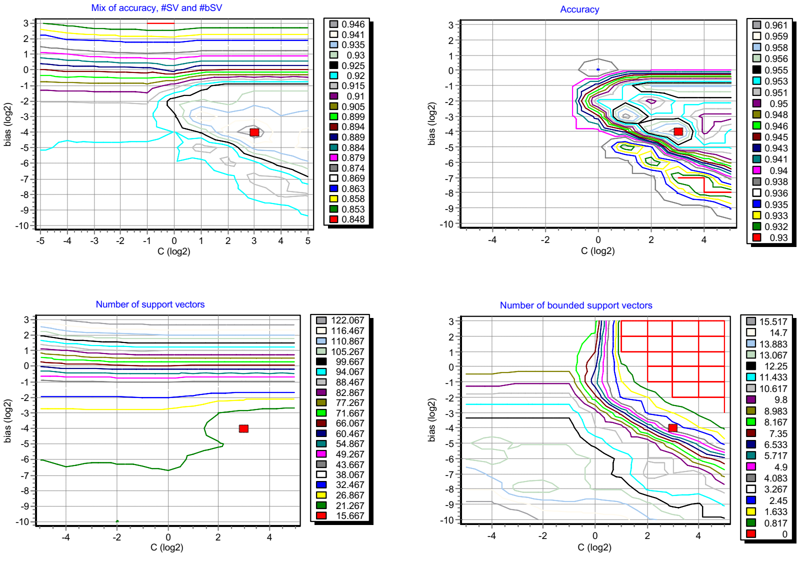

Fig. 2. Densities of Meta-SVM criterion, accuracy, number of support vectors and number of unbounded support vectors for glass benchmark.

2. V. Vapnik. The support vector method of function estimation. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, pages 239–268. Springer-Verlag, 1998.
3. V. Vapnik. *Statistical Learning Theory.* Wiley, New York, NY, 1998.
4. E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *In Proceedings of CVPR'97*, pages 130–136, New York, NY.
5. T. Joachims. *Advances in Kernel Methods - Support Vector Learning*, chapter Making large-scale SVM learning practical. MIT Press, Cambridge, MA., 1998.
6. C. Saunders, M. O. Stitson, J. Weston, L. Bottou, B. Schoelkopf, and A. Smola. Support vector machine reference manual. Technical Report CSD-TR-98-03, Royal Holloway, University of London, Egham, UK, 1998.
7. J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Sch.olkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning.* MIT Press, Cambridge, MA.
8. J. C. Platt. Using analytic QP and sparseness to speed training of support vector machines. In *Advances in Neural Information Processing Systems 11.* MIT, 1999.
9. C.-C. Chang, C.-W. Hsu, and C.-J. Lin. The analysis of decomposition methods for support vector machines. *IEEE Transactions on Neural Networks*, 11(4):1003–1008, 2000.
10. S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation*, 13:637–649, 2001.
11. F. Girosi, M. Jones, and T. Poggio. Regularization theory and neural networks. *Neural Computation*, 7:219–269, 1995.
12. P. Niyogi and F. Girosi. On the relationship between generalization error, hypothesis complexity, and sample complexity for radial basis functions. *Neural Computation*, 8(4):819–842, 1996.
13. N. Jankowski and V. Kadirkamanathan. Statistical control of RBF-like networks for classification. In *7th International Conference on Artificial Neural Networks*, pages 385–390, Lausanne, Switzerland, October 1997. Springer-Verlag.
14. C. J. Merz and P. M. Murphy. UCI repository of machine learning databases, 1998. http://www.ics.uci.edu/∼mlearn/MLRepository.html, (UCI).
15. C.-W. Hsu and C.-J. Lin. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13:415–425, 2002.