Simplifying SVM with Weighted LVQ Algorithm

Marcin Blachnik[1], Mirosław Kordos[2]

[1] Silesian University of Technology, Department of Management and Informatics, Katowice, Krasinskiego 8, Poland; marcin.blachnik@polsl.pl
[2] University of Bielsko-Biala, Department of Mathematics and Informatics, Bielsko-Biała, Willowa 2, Poland: mkordos@ath.bielsko.pl

Abstract. Reduced Set SVMs (RS-SVM) are a group of methods that simplify the internal structure of SVM models, while keeping the SVMs' decision boundaries as similar as possible to the original ones. RS-SVMs are very useful in reducing computational complexity of the original models. They accelerate the decision process by reducing the number of support vectors. They are especially important for large datasets, when lots of support vectors are selected. They also can be very useful for understanding the internal structure of SVM models by the use of prototype-based rules. This paper presents a new method based on the modified version of the LVQ algorithm called WLVQ, which combines both of the objectives: computational complexity reduction and generation of prototype-based rules.

Key words: SVM, Reduced set methods, LVQ, Prototype-based rules

1 Introduction

One of the problems faced by support vector machines (SVM) is the speed of the prediction process. This problem is especially important for large datasets and online prediction problems. In such cases usually a large number of support vectors (SV) is chosen, what increases the time required to calculate the kernel matrix and consequently the system response. This problem may be overcome by reduction the number of SV and at the same time preserving original SVM's decision borders.

Reducing the number of SVs may be also beneficial to understand the data properties reflected in the decision boundaries of the SVM model. As we have shown in [1] when the number of SVs gets dramatically reduced in such a way that the prototype positions represent groups of similar instances, the SVM model can be represented as a set of Prototype Based-Rules (P-Rules) [2]. In this approach each SV is treated as a prototype and associated with its similarity or distance function.

Summarizing, there are many benefits that can be obtained from reducing the number of support vectors. There are different techniques that can be used to achieve that goal:

1. Removing some of the original SVs that are linearly dependent leaving other SVs intact [3]
2. Applying the "Reduced Set" approach where new support vectors are selected or constructed anywhere in the input space (not necessarily as some of the training samples)[4, 5];

3. Modifying the cost function of the original SVM [6, 7].

The first approach is very useful, because it does not modify the decision boundary of the SVM classifier, but only reduces its computational complexity. This method usually allows reducing the number of support vector by few or more percents. However, this in many cases is not sufficient, so other techniques have to be used, and these two other groups are examples of the possible solutions.

The second method is based on constructing new set of SVs that is much smaller then the set obtained during the SVM training, but it preserves the shape of the decision boundary and keeps it as similar to the original one as possible. That approach would be further discussed in the next section (2). The last approach is based on reformulating the cost function of the SVM taking into account not only the width of the margin but also the number of SVs.

In this paper a new “Reduced Set" method is presented (the second approach). This method is based on the LVQ algorithm, which is modified to achieve the best possible reconstruction of the decision boundary. The discussed Weighted LVQ algorithm, embed in the prototype positions information about the shape of the decision boundary of the SVM model. The reduced set SVM (RS-SVM) method has an important advantage over other methods. One of the properties of the LVQ algorithm is a selection of prototypes which represent clusters of similar instances that preserves class labels. Such a property allows for better understanding of the model by the use of the P-rules concept of model comprehensibility.

The paper is organized as follows: the next section (2) introduces the SVM training process, and discusses the state of the art in the Reduced Set method. The section (3.1) presents the modified version of the LVQ algorithm called Weighted-LVQ (WLVQ), and an appropriate weighting procedure. The section (4) presents how to determine the appropriate number of SVs. Section (5) shows numerical examples of the new reduced set method (RS) on some artificial and real world problems. The last section (6) concludes the article and draws further research directions.

2 Simplifying SVM prediction model

2.1 Introduction to SVM

The SVM is a linear discrimination model defined in the feature space $\digamma$ after mapping data from the n-dimensional input space χ to this feature space $\phi(\mathbf{x})$. That can be defined as:

$$\mathbf{\Psi} = \sum_{i=1}^{m} \gamma_i \phi(\mathbf{x}_i) \tag{1}$$

According to the kernel trick, it is not necessary to directly map the data into the feature space $\digamma$ using the mapping function ($\phi(\cdot)$), but rather implicitly map the data using the property of the dot product using the kernel function. The decision function in this case is defined as:

$$f(\mathbf{x}) = \sum_{i=1}^{m} \gamma_i y_i K(\mathbf{x}, \mathbf{x}_i) + b \tag{2}$$

where $\mathbf{x}_i$ are the support vectors with non-zero γ_i coefficients (Lagrangian multipliers), $K(\mathbf{x}, \mathbf{x}_i)$ is the kernel function, and $y_i = C(\mathbf{x}_i) = \pm 1$ are the class labels.

2.2 State of art of reduced set methods

The idea and the methodology of the reduced set methods was proposed by Burges in [5]. His idea is based on reducing the number of support vectors by minimizing the distance between the original SVM hyperplane $\mathbf{\Psi}$ and the hyperplane $\mathbf{\Psi}'$ obtained with the reduced set model:

$$d = \min ||\mathbf{\Psi} - \mathbf{\Psi}'||^2 \tag{3}$$

To preserve the original decision boundary the distance should be minimized so that the approximation of the new decision function $\mathbf{\Psi}'$

$$\mathbf{\Psi}' = \sum_{i=1}^{m'} \beta_i \phi(\mathbf{z}_i) \tag{4}$$

is as close to the original $\mathbf{\Psi}$ as possible, satisfying the inequality $m' \ll m$, with scalar coefficients β_i, where m' - is the reduced number of SVs.

According to the above statement, in the reduced set model the value of β_i and the positon of $\mathbf{z}_i$ have to be determined and it can be achieved by minimization of (3) over β and $\mathbf{z}$ that can be written as:

$$\begin{aligned} \min_{\beta,\mathbf{z}}(d) = \sum_{i,j=1}^{m} \gamma_i\gamma_j K(\mathbf{x}_i, \mathbf{x}_j) + \sum_{i,j=1}^{m'} \beta_i\beta_j K(\mathbf{z}_i, \mathbf{z}_j) \\ -2 \sum_{i=1}^{m} \sum_{j=1}^{m'} \beta_j\gamma_i K(\mathbf{x}_i, \mathbf{z}_j) \end{aligned} \tag{5}$$

As it was shown in [5], taking the matrix notation $K^{zx}\gamma = K^{zz}\beta$ where $\gamma = [\gamma_1, \gamma_2, \ldots, \gamma_m]^T$, $\beta = [\beta_1, \beta_2, \ldots, \beta_{m'}]^T$, and K^{zx} is matrix of the $m' \times m$ dimensions containing $K(\mathbf{z}_i, \mathbf{x}_j)$ values, the solution of minimization of (5) can be written as:

$$\beta = \left(K^{zz}\right)^{-1} K^{zx}\gamma \tag{6}$$

Now the goal, which is to determine the position of vectors $\mathbf{z}_j$, can be solved in two different ways. In the first solution vectors $\mathbf{z}_j$ can be selected from the vectors $\mathbf{x}_j$ using one of instance selection techniques like ENN or CNN algorithms [8] or by any other systematic search strategy. This is beneficial in terms of interpretation and comprehensibility of the solutions extracted from the SVM model by using prototype based rules. In that case each SV represents an input instance, what in many applications allows for further in depth investigation of these selected cases. On the other hand $\mathbf{z}_j$ vectors can be constructed anywhere in the input space. This approach is used by the majority of already invented algorithms. For example Schölkopf et. al. has proposed a strategy based on clustering in the feature space [4] that can be interpreted as EM iteration for the determination of the center of a Gaussian cluster representing similar vectors that match the sign of y_i and γ_i. Another approach has been proposed by Burges [5] where the author claims that the highest drop in the distance between hyperplanes d can be achieved for vectors $\mathbf{z}$ that are the eigenvectors with the highest absolute eigenvalues $\lambda = \beta\mathbf{z}^2$. Another interesting method has been proposed by Kwok and Tsang [9]. The method is based on the Multidimensional Scaling (MDS) algorithm, which transforms images of the feature space vectors back into the input space. Prototypes derived from these algorithms have no direct counterparts in the instances of the training set. However, the methods based on the construction of the new SVs enable much greater reduction of the number of the original SVs, while preserving a small distance d between hyperplanes. Unfortunately non of described prototypes construction methods allows for data understanding. The prototypes obtained by these methods are not informative, not providing any knowledge of the data structure. That deficiency may be overcome by the use of WLVQ algorithm described in this paper.

3 Reducing number of support vectors with LVQ algorithm

3.1 Weighted LVQ algorithm

The LVQ algorithm is one of the most popular and simple neural networks that is based on optimization of the position of codebook vectors, which are also called prototypes. The LVQ algorithm has been applied to RBF neural networks training with very good results [10], so it was also considered as a tool for reduced set methods.

In the original LVQ algorithm only the distance between an instance $\mathbf{x}_i$ and the nearest prototypes $\mathbf{p}_k$ are taken into account when updating prototypes position. Therefore in [11] we have introduced a new cost function that also applies external knowledge of the data distribution:

$$\begin{aligned} E(\mathbf{P}) = & \frac{1}{2}\sum_{k=1}^{m'}\sum_{i=1}^{m}\mathbf{1}\left(\mathbf{x}_i \in R_k\right)\mathbf{1}\left(c(\mathbf{x}_i) = c(\mathbf{p}_k)\right)g(\mathbf{x}_i)\left|\left|\mathbf{x}_i - \mathbf{p}_k\right|\right|^2 \\ & -\frac{1}{2}\sum_{k=1}^{m'}\sum_{i=1}^{m}\mathbf{1}\left(\mathbf{x}_i \in R_k\right)\mathbf{1}\left(c(\mathbf{x}_i) \neq c(\mathbf{p}_k)\right)g(\mathbf{x}_i)\left|\left|\mathbf{x}_i - \mathbf{p}_k\right|\right|^2 \end{aligned} \tag{7}$$

where $\mathbf{1}(L)$ is the switching function, which returns 1 when condition L is *true*, and 0 otherwise, $c(\mathbf{x})$ returns class label of vector $\mathbf{x}$, and R_k is the Voronoi area defined for prototype $\mathbf{p}_k$. This cost function can be minimized according to $\mathbf{p}$ by iteratively updating codebook positions:

$$\begin{aligned} \mathbf{p}_k = \mathbf{p}_k + \alpha(j)g(\mathbf{x}_i)\mathbf{1}\left(c(\mathbf{x}_i) = c(\mathbf{p}_k)\right)(\mathbf{x}_i - \mathbf{p}_k) \\ \mathbf{p}_k = \mathbf{p}_k - \alpha(j)g(\mathbf{x}_i)\mathbf{1}\left(c(\mathbf{x}_i) \neq c(\mathbf{p}_k)\right)(\mathbf{x}_i - \mathbf{p}_k) \end{aligned} \tag{8}$$

where the context factor $g(\mathbf{x}_i)$ describes the external knowledge provided in order to achieve certain properties during the training. The $g(\mathbf{x}_i)$ value can be understood as an instance weight that describes the significance of the instance during the training process.

3.2 Determining weights coefficient

As described above the $g(\mathbf{x}_i)$ function can be used to introduce external dependencies that impose additional restrictions on the optimization process. Such dependencies may be defined according to the shape of the decision boundary of the SVM classifier. To achieve that aim all the vectors that are situated close to the decision boundary (2) should have higher weight values and vectors that are far from the boundary should be less significant to the optimization process. According to that we have defined $g(\mathbf{x}_i)$ as:

$$g(\mathbf{x}_i) = 1 - \left|\tanh\left(\sigma \cdot t\left(x_i\right)\right)\right| \tag{9}$$

or

$$g(\mathbf{x}_i) = \exp\left(-\sigma \cdot t\left(x_i\right)^2\right) \tag{10}$$

where σ is a user defined constant, and $t\left(x_i\right)$ is a normalized SVM decision $f\left(x_i\right)$ such that

$$t\left(x_i\right) = \begin{cases} f\left(x_i\right)/std_P & \text{if } f\left(x_i\right) > 0 \\ f\left(x_i\right)/std_N & \text{if } f\left(x_i\right) < 0 \end{cases} \tag{11}$$

where std_P and std_N are normalization factors defined as $std_P = std\left(\forall x_i : \left(f\left(x_i\right) > 0\right) x_i\right)$ and respectively $std_N = std\left(\forall x_i : \left(f\left(x_i\right) < 0\right) x_i\right)$

4 Finding optimal number of support vectors

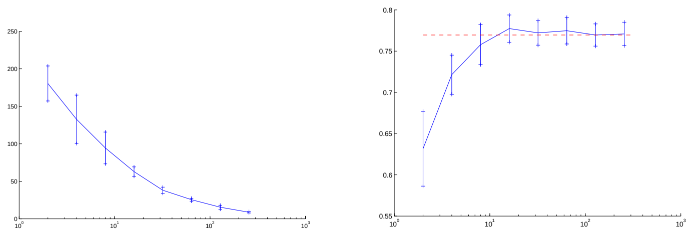
RS-SVM approach allows for significant reduction of the number of SVs. However, the problem of determining the correct number of reduced set of SVs remains open. The most natural solution seems to be the optimization of the distance between separating hyperplanes (3):

$$E_1(m') = \|\mathbf{\Psi} - \mathbf{\Psi}'\| = \tag{12}$$
$$\left(\sum_{i,j=1}^{m} \gamma_i\gamma_j K(\mathbf{x}_i, \mathbf{x}_j) + \sum_{i,j=1}^{m'} \beta_i\beta_j K(\mathbf{z}_i, \mathbf{z}_j) - 2\sum_{i=1}^{m}\sum_{j=1}^{m'} \beta_j\gamma_i K(\mathbf{x}_i, \mathbf{z}_j)\right)^2$$

When facing the comprehensibility problem of the SVM model by the use of P-rules, the cost function can be extended with an additional term $\alpha m'/m$, which represents the model complexity as a ratio of a reduced number of SVs (m') to the original number of SVs (m) multiplied by some constant α. This introduces a punishment into the cost function and promotes the solution with a smaller number of SVs. Because the distance $\|\mathbf{\Psi} - \mathbf{\Psi}'\|$ may take very high values, α may be rescaled by $1/\|\mathbf{\Psi} - \mathbf{\Psi}'_1\|$, where $\mathbf{\Psi}'_1$ is $\mathbf{\Psi}'$ defined with just one SV.

Visualization of the relation between distance $\|\mathbf{\Psi} - \mathbf{\Psi}'\|$ and the number of SVs is presented in figure (1). The same figure shows the relation between the accuracy and the number of SVs.

The analysis of results presented in figure (1).b shows that the performance of the RS model can overcome the performance of the SVM model. Such situation may happen when the SVM is over-fitted, so the RS-SVM is able to detect the over-fitting condition and propose a simpler model. This leads to two other cost functions defined as a



(a) Dependence of the cost function E_1 on the logarithm of the number of SVs

(b) Dependence of the mean accuracy (cost function E_3) on the logarithm of the number of SVs. Dashed line represents mean accuracy of the original SVM

Fig. 1. Comparison of the distance (Eq. (13)) and accuracy (Eq. (14)) based cost functions for Pima Indians diabetes data.

difference between the accuracy of the SVM and RS-SVM model,

$$E_2(m') = \text{acc(SVM)} - \text{acc(RSSVM(m'))} \tag{13}$$

where $\text{acc}()$ is the classification accuracy measured using some loss function. Because the $acc(SVM)$ remain constant during optimization of the number of SVs, the function (13) can be simplified omitting the first component:

$$E_3(m') = \text{acc(RSSVM(m'))} \tag{14}$$

5 Numerical examples

To verify the proposed algorithm we performed a set of numerical tests on real world problems.

In the comparison four algorithms were compared, the original SVM model (libSVM implementation), Schölkopf and Burges algorithm, both implementations were based on the Spider toolbox for Matlab, and the WLVQ algorithm also implemented as an operator of Spider toolbox `http://www.p-rules.eu`. The datasets taken for the comparison were obtained from the UCI repository [12]. For that purpose the most popular datasets were selected, like *wisconsin brest cancer, heart disease, pima indiens diabetes* and *spam base*. In this experiment the number of SVs of all reduced set methods was fixed to 20. All results were obtained with a 10-fold cross-validation test. At the beginning the hyperparameters of the SVM model were optimized and after the selection of the best set of parameters the RS models were generated. The obtained results are presented in table (1)

As it can be seen, the quality of the RS-SVM method is comparable to the others. In almost all cases the Burges algorithm achieved the smallest distance d (3) between hyperplanes. However, this did not correlates with the best accuracy, where usually the

Table 1. Empirical comparison of SVM and three reduced set methods

Dataset	SVM		Burges		Schölkopf		WLVQ	
	Accuracy	$\#SVs$	Accuracy	d	Accuracy	d	Accuracy	d
heart disease	84.50±5.54	130	82.84±5.95	0.500	82.50±5.59	14.39	84.18±4.43	17.02
diabetes	77.61±3.02	430	73.44±4.47	0.078	73.05±4.50	2.80	73.17±5.10	0.825
wbc	97.07±2.19	74	97.21±1.47	0.033	97.21±1.76	0.453	97.07±1.70	0.348
spam	93.84±1.09	734	93.31±1.12	17018	82.92±5.12	41810	90.02±1.64	39421

proposed algorithm based on WLVQ networks obtained the best accuracy - the closest to the SVM model.

A very interesting results were obtained in the case when the SVM model was incorrectly optimized and tended to over-fit the data. The results are presented in table (2).

Table 2. Empirical comparison of over-fitting SVM and 3 reduced set methods

Dataset	SVM		Burges	Schölkopf	WLVQ
	Accuracy	$\#SVs$	Accuracy	Accuracy	Accuracy
heart disease	75.77±8.60	107	73.05±6.36	72.07±8.72	80.47±5.43
diabetes	73.71±5.57	332	64.98±4.03	68.23±5.47	73.70±4.77
wbc	94.15±1.68	58	95.31±2.38	93.55±3.47	95.75±2.63
spam	94.15±1.68	58	95.31±2.38	93.55±3.47	95.75±2.63

From these results we can see that the accuracy of Burges and Schölkopf algorithms have much smaller values of distance d, however their accuracy is often much worse. That is because these two algorithms that minimize the distance d are concentrating on the most complex and bent part of the separating hyperplane $\boldsymbol{\Psi}$. This results with even a stronger over-fitting. While in the WLVQ-based model the prototypes are representing the centers of instance groups from different classes, which avoids the problem of over-fitting.

6 Conclusions

The proposed RS-SVM algorithm based on weighted LVQ algorithm has proven highly effective. In comparison to other methods the accuracy is of the same level. Similar results were obtained for the $||\boldsymbol{\Psi} - \boldsymbol{\Psi}'||$ distance (except Burges algorithm which outperforms other methods). There are also other advantages of the WLVQ-based SV. During the optimization process the LVQ codebooks are attracted to areas of high density of input vectors. This makes the reduced set of SVs meaningful and useful for understanding of relations hidden in the data, especially for the P-Rules.

An important advantage of the RS-SVM algorithm is the short time required to recalculate the RS-SVM model. In comparison to other methods codebooks position (SVs) are determined independently of the weights β_i, so the process consists of two serial subprocesses. In the first step the WLVQ algorithm is trained and then in the second step the appropriate weights β_i for each codebook are determined.

Our future plans include the investigation of the relation between the number of codebooks and the accuracy of the pure WLVQ algorithm and the $\|\mathbf{\Psi} - \mathbf{\Psi}'\|$ distance. If such relation appears, what seems to be a correct assumption, it could be interesting to apply the dynamic LVQ algorithm (DLVQ) to automatically optimize the number of SVs without recalculating the β coefficients.

Acknowledgement The work was sponsored by the Polish Ministry of Science and Higher Education, project No. 4421/B/T02/2010/38 (N516 442138).

References

1. Blachnik, M., Duch, W.: Prototype rules from SVM. In: Rule Extraction from Support Vector Machines. Volume 80 of Studies in Computational Intelligence Series. Springer (2008)
2. Duch, W., Grudziński, K.: Prototype based rules - new way to understand the data. In: IEEE International Joint Conference on Neural Networks, Washington D.C, IEEE Press (2001) 1858–1863
3. Lin, K., Lin, C.: A study on reduced support vector machines. IEEE Transactions on Neural Networks **14** (2003) 1449–1459
4. Schölkopf, B., Knirsch, P., Smola, A., Burges, C.: Fast approximation of support vector kernel expansions. Informatik Aktuell, Mustererkennung (1998)
5. Burges, C.: Simplified support vector decision rules. In: ICML. (1996) 71–77
6. Downs, T., Gates, K., Masters, A.: Exact simplification of support vector solutions. The JMLR **2** (2001) 293–297
7. Wu, M., Schölkopf, B., Bakur, G.: A direct method for building sparse kernel learning. The JMLR **4** (2006) 603–624
8. Jankowski, N., Grochowski, M.: Comparison of instance selection algorithms. i. algorithms survey. LNCS **3070** (2004) 598–603
9. Kwok, J., Tsang, I.: The pre-image problem in kernel methods. IEEE Transactions on Neural Networks **15** (2003) 408–415
10. Palm, F.S.H.K.G.: Three learning phases for radial-basis-function networks. Neural Networks **14** (2001) 439–458
11. Blachnik, M., Duch, W.: Improving accuracy of lvq algorithm by instance weighting. LNCS **6354** (2010)
12. Merz, C., Murphy, P.: UCI repository of machine learning databases (1998-2004) http://www.ics.uci.edu/∼mlearn/MLRepository.html.