A Fast Revised Simplex Method for SVM Training

Christopher Sentelle*
University of Central Florida
csentelle@cfl.rr.com

Georgios C. Anagnostopoulos
Florida Institute of Technology
georgio@fit.edu

Michael Georgiopoulos
University of Central Florida
michaelg@mail.ucf.edu

Abstract

Active set methods for training the Support Vector Machines (SVM) are advantageous since they enable incremental training and, as we show in this research, do not exhibit exponentially increasing training times commonly associated with the decomposition methods as the SVM training parameter, C, is increased or the classification difficulty increases. Previous implementations of the active set method must contend with singularities, especially associated with the linear kernel, and must compute infinite descent directions, which may be inefficient, especially as C is increased. In this research, we propose a revised simplex method for quadratic programming, which has a guarantee of non-singularity for the sub-problem, and show how this can be adapted to SVM training.

1. Introduction

The support vector machine (SVM) is a popular classifier due to its excellent generalization performance. The SVM classifier, introduced by Vapnik [15], employs structural risk minimization whereby a bound on the risk is minimized by maximizing the margin between the separating hyperplane and the closest data point to the hyperplane. The problem associated with the 1-norm soft margin classifier [3] is formulated as a quadratic programming problem, in dual form, as follows

$$\begin{aligned} \min\ & \frac{1}{2}\alpha^T Q\alpha - 1^T\alpha \\ \text{s.t. } & y^T\alpha = 0 \\ & 0 \le \alpha_i \le C\ \forall i \end{aligned}$$

where α is a $l \times 1$ vector of Lagrange multipliers associated with l constraints in the primal problem and $Q_{ij} = y_i y_j K_{ij}$. K_{ij} is a short-hand notation for $K(x_i, x_j)$ where $K(,)$ is a kernel function, which forms a dot product in a nonlinearly mapped space, $x_i \in \Re^n$ is a training sample, and $y_i \in \{-1, +1\}$ is the label attached to the i^{th} data point.

This QP problem can be solved with many methodologies available within the numerical optimization community. However, these techniques often require storage of the entire Hessian matrix, Q, creating a memory requirement on the order of the square of the number of data points. This makes the problem intractable on limited-memory systems. To address this issue, chunking [14] and decomposition methods [9] introduced the concept of iteratively solving smaller sub-problems of the original problem until the original problem is solved. The problem with these methods is that the convergence rate is dominated by the method for selecting the subset of data points to work with (working set) at each iteration [7]. However, cleverly implemented decomposition methods as introduced by Platt [10] and Joachims [6] have remained the benchmark when comparing training times for new algorithms.

More recently, active set methods have been introduced [12], [13]. The advantage of active set methods is that they allow incremental training through "warm starts" and, as we show, they are more stable as the regularization parameter, C, is

*Christopher Sentelle gratefully acknowledges the support of the College of Engineering & Computer Science and the I2Lab at the University of Central Florida. This work was supported in part by NSF grants: 0341601, 0647018, 0717674, 0717680, 0647120, 0525429, 0203446.

adjusted. However, in both Scheinberg and Shilton *et al.*, the implementation of the active set method is complicated by the fact that the sub-problem (equality constrained problem that is solved at each iteration of the active set method) can be singular resulting in the need to choose a descent direction out of infinite possibilities.

In our implementation, we chose to create an efficient implementation of the revised simplex method (SVM-RSQP) introduced by Rusin [11], which has the property of guaranteeing non-singularity of the inner subproblem at each step. Like Scheinberg's implementation, our memory requirement is shown to be on the order of square of the number of non-bound support vectors. We show that this methodology is competitive with Scheinberg's implementation, while eliminating the additional complexity for dealing with singularities, and is competitive with two state-of-the-art training algorithm implementations, LIBSVM and SVMLight.

2. Revised simplex method

The problem statement for the revised simplex method for quadratic programming, as discussed by Rusin, is of the following form

$$\begin{aligned} &\min\ p^T x - \frac{1}{2} x^T Q x \\ &\text{s.t. } Ax = b \\ &\quad x \geq 0 \end{aligned} \tag{1}$$

where p is a $n \times 1$ vector of linear costs, x is a $n \times 1$ vector representing the variables to be solved, Q is a $n \times n$ matrix of quadratic costs, A is a $m \times n$ matrix consisting of m constraints applied to n variables and b is a $m \times 1$ vector representing the constant offsets for the constraints. In this problem, Rusin assumes the problem is convex and Q is negative semi-definite. We can reformulate the SVM problem to match Rusin's form by introducing additional slack variables, as follows

$$\begin{aligned} &\min\ \begin{pmatrix} -1^T & 0^T \end{pmatrix} \begin{pmatrix} \alpha \\ s \end{pmatrix} - \frac{1}{2} \begin{pmatrix} \alpha^T & s^T \end{pmatrix} \begin{pmatrix} -Q & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ s \end{pmatrix} \\ &\text{s.t. } \alpha^T y = 0 \\ &\quad \alpha + s = 1C \\ &\quad \alpha \geq 0, s \geq 0 \end{aligned}$$

where we have added the vector, s, as a set of slack variables to handle the inequality constraint $\alpha \leq C$. Comparing this with (1) we see that

$$p^T = \begin{pmatrix} -1^T & 0^T \end{pmatrix}, Q = \begin{pmatrix} -Q & 0 \\ 0 & 0 \end{pmatrix}$$
$$A = \begin{pmatrix} y^T & 0 \\ I & I \end{pmatrix}, b = \begin{pmatrix} 0 \\ 1C \end{pmatrix}$$

At each iteration of the revised simplex method, once the basic (non-zero) versus non-basic (zero) variables are identified, the following problem is typically solved

$$\begin{pmatrix} 0 & B \\ B^T & Q_B \end{pmatrix} \begin{pmatrix} g \\ h \end{pmatrix} = \begin{pmatrix} a_i \\ q_i \end{pmatrix} \tag{2}$$

where B is comprised of the columns of A corresponding to the basic variables, g and h are the updates to π^T and x_B, respectively, Q_B is the portion of Q corresponding to the basic variables, and a_i is the column of A and q_i is the column of Q for a variable entering the basis.

For the SVM problem, this update equation takes the following form, when a non-support vector is entering the basis,

$$\begin{pmatrix} 0 & {y_s}^T \\ y_s & -Q_{ss} \end{pmatrix} \begin{pmatrix} g_\beta \\ h_s \end{pmatrix} = \begin{pmatrix} y_i \\ q_{si} \end{pmatrix} \quad h_i = 1 \tag{3}$$

and changes, when a bound support vector becomes unbounded or is entering the basis, to

$$\begin{pmatrix} 0 & {y_s}^T \\ y_s & -Q_{ss} \end{pmatrix} \begin{pmatrix} g_\beta \\ h_s \end{pmatrix} = \begin{pmatrix} -y_i \\ -q_{si} \end{pmatrix} \quad h_i = 1 \tag{4}$$

where h_s, Q_{ss}, and y_s are components of the original variables corresponding to the non-bound support vectors, g_β is the component of g corresponding to an update in β, which is π_0 in Rusin and is also the threshold b, q_{si} is the column of Q for the variable entering the basis, and y_i is the label. The slack variables are dealt with implicitly, and, therefore, do not appear in the updates.

Rusin shows that the matrix in (2) will be nonsingular at every iteration. It is easy to show this implies the matrix in (3) and (4) will be non-singular. Q_{ss} can have a single, zero eigenvalue, in which case, simple refactoring of the system of equations still results in a finite descent direction. For efficient implementation, a Cholesky factorization of Q_{ss} is maintained. At each iteration of the revised simplex method, a single row and column is added

to or removed from Q_{ss}. As a result, a rank-one update of the Cholesky factorization can be computed at each iteration following the method described in Golub Van Loan [5].

In the simplex method, a pricing step determines the variable to be added to the basis at each iteration. The pricing step, as listed in Rusin, is

$$\delta_j = p_j - \pi a_j - x_B^T q_j.$$

This can be reformulated for the SVM problem as follows,

$$\begin{aligned} \delta_j &= 1 + y_j\beta + Q_{js}\alpha_s + Q_{jc}\alpha_c \forall j \in I_o \\ \delta_j &= 1 + y_j\beta - Q_{js}\alpha_s - Q_{jc}\alpha_c \forall j \in I_c \\ \delta_j &= 0 \forall j \in I_s \end{aligned}$$

where I_o represents the set of non-support vectors, I_c is the set of bound support vectors, and I_s is the set of non-bound support vectors. Of course, optimality occurs when $\delta_j \geq 0 \forall j$. The pricing step is the most computationally intensive portion of each iteration. As a result, we can apply shrinking [6] and partial pricing [8] to reduce the number of computations during the pricing step and significantly reduce the overall computation time. An initial basic feasible solution must be found to start the simplex method and is found to be the following.

$$\begin{aligned} \alpha_1 = 0, s = C, \beta = -y_1 \\ I_s = 1, I_0 = 2..N, I_c = \emptyset \end{aligned}$$

3. Experimental Results

In this study, we compared the performance of our algorithm, SVM-RSQP, with SVMLight (version 6.01), which is a decomposition-based approach using an interior point solver for each subproblem [6], LIBSVM (version 3.85) [2], which is an SMO implementation based upon Fan *et. al* [4], and SVM-QP introduced in [12]. Several datasets, typically reported in the SVM literature, were tested using the RBF kernel and linear kernel. The accuracy tolerance or stopping criterion for all algorithms was $1e-6$. All algorithms employ kernel caching and shrinking, and both SVM-RSQP and SVM-QP employ partial pricing. Testing was performed with the **abalone**, **adult-1a**, **spam**, and **ocr-9** datasets. The **ocr-9** dataset is based upon the United States Postal Service (USPS) optical character recognition (OCR) database and modified to classify "9" from the remaining digits, while the remaining datasets are all obtained from the UCI repository [1].

The results show, for the datasets tested, that SVM-RSQP consistently outperforms SVMLight and the performance gap increases, exponentially, for larger values of the regularization parameter, C. Although SVM-RSQP does not consistently outperform LIBSVM, we see that LIBSVM exhibits the same exponentially increasing performance gap as C is increased for the **abalone** and **ocr-9** datasets. For low values of C, LIBSVM tends to exhibit comparable or better performance. In the case of the **adult-1a** dataset, for C between 0.01 and 10, LIBSVM training times are 0.3 seconds while SVM-RSQP training times are 0.72 seconds. While the LIBSVM algorithm is competitive for the RBF kernel, we found the SVM-RSQP algorithm outperforming LIBSVM for the linear kernel. Of note is that LIBSVM failed to converge for C greater than 0.1 and SVMLight failed to converge for C greater than 1 on the **spam** dataset. Scheinberg also reports this issue for SVMLight [12].

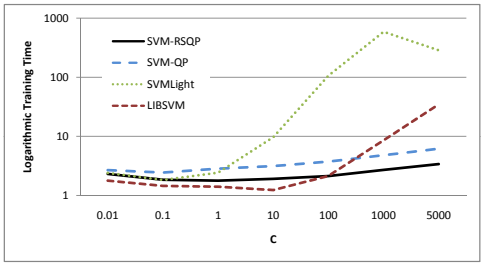
When comparing to SVM-QP, we note slightly better performance for the **abalone** and **spam** datasets. For the **spam** dataset, at $C = 10$ we begin to note a significant performance gap with a 6.2 second training time for SVM-QP and 3.2 second training time for SVM-RSQP. With the **abalone** dataset, we note a similar situation at $C = 1000$ with 4.8 seconds for SVM-RSQP and 2.7 for SVM-QP.

In summary, both LIBSVM and SVMLight exhibit exponentially increasing training times for the datasets tested, as C is increased, which is highly disadvantageous when the optimal value for C must be discovered through a grid search and cross validation. In addition, we note that SVM-RSQP tends to perform comparably with SVM-QP with a speed-up factor of 2 in two cases for high values of C.

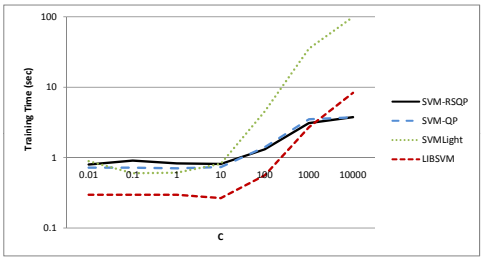
4. Conclusion

In conclusion, we show we can create an efficient active set method for SVM training using the revised simplex method proposed by Rusin. The advantage of this method is that the inner sub-problem has a guarantee of non-singularity, which simplifies implementation and may yield more efficient iterations. As the preliminary results show, SVM-RSQP performs similarly to SVM-QP and outperforms both SVMLight and LIBSVM at high values of C.

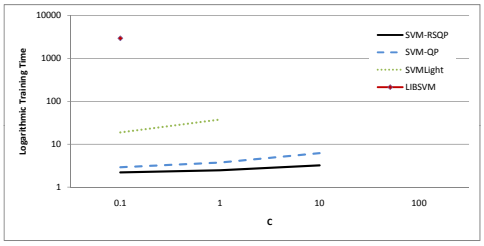
Decomposition methods maintain an advantage



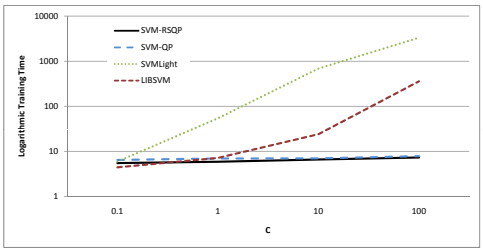
(a) Abalone dataset (RBF Kernel $\lambda = .25$)



(b) Adult-1a dataset (RBF Kernel $\lambda = .01$)



(c) Spam dataset (Linear Kernel)



(d) OCR-9 dataset (Linear Kernel)

Figure 1: SVM-RSQP performance as a function of the regularization parameter C

over even efficiently implemented active set methods due to their limited use of memory. However, guarantees of non-singularity within the revised simplex method may yield more choices for applying limited-memory, iterative techniques to solving the inner problem, a topic of future research.

References

[1] A. Asuncion and D. Newman. UCI machine learning repository, 2007.

[2] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[3] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press, 2000.

[4] R.-E. Fan, P.-H. Chin, and C.-J. Lin. Working set selection using second order information for training support vector machines. *Journal of Machine Learning Research*, 6:1889–1918, December 2005.

[5] G. H. Golub and C. F. V. Loan. *Matrix Computations*. The John Hopkins University Press, Baltimore and London, 3rd edition, 1996.

[6] T. Joachims. Making large-scale support vector machine learning practical. In A. S. B. Schölkopf, C. Burges, editor, *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA, 1998.

[7] C.-J. Lin. On the convergence of the decomposition method for support vector machines. *IEEE Transactions on Neural Networks*, 12(6):1288–1298, November 2001.

[8] I. Maros. *Computational Techniques of the Simplex Method*. Kluwer Academic Publishers, Boston, 2003.

[9] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *CVPR*, pages 130–136, San Juan, Puerto Rico, June 1997. 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'97).

[10] J. Platt. Making large-scale support vector machine learning practical. In A. S. B. Schölkopf, C. Burges, editor, *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA, 1998.

[11] M. H. Rusin. A revised simplex method for quadratic programming. *SIAM Journal on Applied Mathematics*, 20(2):143–160, March 1971.

[12] K. Scheinberg. An efficient implementation of an active set method for svms. *Journal of Machine Learning Research*, 7:2237–2257, December 2006.

[13] A. Shilton, M. Palaniswami, D. Ralph, and A. C. Tsoi. Incremental training of support vector machines. *IEEE Transactions on Neural Networks*, 16(1):114–131, January 2005.

[14] V. N. Vapnik. *Estimation of Dependences Based on Empirical Data*. Springer Series in Statistics. Springer-Verlag, New York, 1982.

[15] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Information Science and Statistics. Springer, 1st edition, 1995.