Training Support Vector Machines Using Greedy Stagewise Algorithm

Liefeng Bo, Ling Wang, and Licheng Jiao

Institute of Intelligent Information Processing,
Xidian University, Xi'an 710071, China
{blf0218, wliiip}@163.com

Abstract. Hard margin support vector machines (HM-SVMs) have a risk of getting overfitting in the presence of the noise. Soft margin SVMs deal with this problem by the introduction of the capacity control term and obtain the state of the art performance. However, this disposal leads to a relatively high computational cost. In this paper, an alternative method, greedy stagewise algorithm, named GS-SVMs is presented to deal with the overfitting of HM-SVMs without the introduction of capacity control term. The most attractive property of GS-SVMs is that its computational complexity scales quadratically with the size of training samples in the worst case. Extensive empirical comparisons confirm the feasibility and validity GS-SVMs.

1 Introduction

Hard margin support vector machines have a risk of getting overfitting in the presence of the noise [1]. To deal with this problem, soft margin SVMs [2] introduce the capacity control parameter that allows a little training error to obtain the large margin. This is a highly effective mechanism for avoiding overfitting, which leads to good generalization performance. Though very successful, we can identify some shortages of soft margin SVMs:

① The training procedure of soft margin SVMs amounts to solving a constrained quadratic programming. Although the training problem is, in principle, solvable, in practice it is intractable by the classical optimization techniques, e.g. interior point method because their computational complexity usually scales cubically with the size of training samples.

② Capacity control parameter depends on the task at hand; hence there is no foolproof method for determining it before training. Usually, we have to resort to a cross validation procedure, which is wasteful in computation [3].

In the past few years, a lot of fast iterative algorithms were presented for tackling the problem ①. Probably, the most famous method among them is sequential minimization optimization algorithm (SMO), which is proposed by Platt [4] and further improved by Keerthi [5]. Some other examples include SVM^{light} [6], SimpleSVM [7], *SVMTorch* [8], and so on. These algorithms proved to be effective and boosted the development of SVMs.

In this paper, an alternative method, greedy stagewise algorithm, named GS-SVMs is presented to deal with the overfitting of HM-SVMs. Instead of employing the capacity control term, GS-SVMs attempts to control the capacity of hypothesis space by algorithm itself. In summary, the proposed algorithms possess the following two attractive properties:

① The computational complexity of GS-SVMs is $O(nl)$, where l and n are the size of training samples and support vectors, respectively. Even in the worst situation that all the training samples are the support vectors, the computational complexity of GS-SVMs is only $O(l^2)$.

② No extra capacity control parameter is required.

2 Greedy Stagewise Algorithm for SVMs

The Wolfe dual of hard margin SVMs

$$\begin{aligned}&\min\left(\frac{1}{2}\sum_{i,j=1}^{l}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i,\mathbf{x}_j)-\sum_{i=1}^{l}\alpha_i\right)\\ &s.t.\quad 0\le\alpha_i \quad i=1,\cdots,l,\end{aligned}\tag{1}$$

can be regarded as a loss function induced by reproducing kernel Hilbert space norm. This allows us to approximate it using greedy algorithm. Due to the room limitation, the detailed interpretation is ignored and the interested reader can refer to [9]. Though HM-SVM is, in principle, solvable by the classical optimization technique, in practice it suffers from two serious problems: (1) their computational complexity usually scales cubically with the size of training samples; (2) there often is a risk of getting overfitting due to no capacity control term. Here, we will deal with the two problems by greedy stagewise algorithm, which attempts to approximate (1) quickly while avoids the overfitting. Greedy stagewise algorithm [10] can be described as the following.

For $m=1,2,\cdots l$,

$$\begin{aligned}(w_m,\beta_m)&=\arg\min_{w,\beta}\left(\sum_{i=1}^{l}L\left(y_i,f_{m-1}(\mathbf{x}_i)+wK(\mathbf{x}_i,\mathbf{x}_\beta)\right)\right)\\ &s.t.\quad \beta\ne\beta_j \quad j=1,2,\cdots m-1\end{aligned}\tag{2}$$

and then

$$f_m=f_{m-1}+w_mK\left(\mathbf{x},\mathbf{x}_{\beta_m}\right).\tag{3}$$

where $L(\cdot)$ denotes loss function, $f_0\equiv 0$ and the constraint terms guarantee that each basis function is used once at most.

For SVMs, w takes the form $\alpha y_\beta, \alpha \ge 0$. Using the loss function (1) we have

$$\begin{aligned}(\alpha_m,\beta_m) = \arg\min_{\alpha,\beta} & \left(\begin{array}{l}\frac{1}{2}\sum_{i=1}^{m-1}\sum_{j=1}^{m-1}\alpha_{\beta_i}\alpha_{\beta_j}y_{\beta_i}y_{\beta_j}K\left(\mathbf{x}_{\beta_i},\mathbf{x}_{\beta_j}\right)-\sum_{i=1}^{m-1}\alpha_{\beta_i}+\\ \alpha y_\beta\sum_{i=1}^{m-1}\alpha_{\beta_i}y_{\beta_i}K\left(\mathbf{x}_{\beta_j},\mathbf{x}_\beta\right)+\frac{1}{2}\alpha^2K\left(\mathbf{x}_\beta,\mathbf{x}_\beta\right)-\alpha\end{array}\right)\\ s.t. \quad & \alpha \ge 0 \\ & \beta \ne \beta_j \quad j=1,2,\cdots m-1\end{aligned} \quad . \tag{4}$$

Note that the first two terms of (4) can be ignored. Define the gradient vector

$$\mathbf{g}_\beta^m = \begin{cases}-1 & if\ m=0\\ y_\beta\sum_{j=1}^{m}\alpha_{\beta_j}y_{\beta_j}K\left(\mathbf{x}_{\beta_j},\mathbf{x}_\beta\right)-1 & if\ m\ge 1\end{cases}. \tag{5}$$

We can reformulate (4) as

$$\begin{aligned}(\alpha_m,\beta_m) = \arg\min_{\alpha,\beta} & \left(\frac{1}{2}\alpha^2K\left(\mathbf{x}_\beta,\mathbf{x}_\beta\right)+\alpha\mathbf{g}_\beta^{m-1}\right)\\ s.t. \quad & \alpha\ge 0\\ & \beta\ne\beta_j \quad j=1,2,\cdots m-1\end{aligned} \quad . \tag{6}$$

(6) can be solved in two steps. In the first step, we fix β and compute the minimal value h_β^{m-1} of (6) with respect to α. In the second step, we compute β_m by minimizing h_β^{m-1} with respect to β, and then compute α_m in terms of β_m. Fixing β, we have the subproblem

$$\begin{aligned}& \min_\alpha\left(\frac{1}{2}\alpha^2K\left(\mathbf{x}_\beta,\mathbf{x}_\beta\right)+\alpha\mathbf{g}_\beta^{m-1}\right)\\ & s.t. \quad \alpha\ge 0\end{aligned} \tag{7}$$

Since (7) is a single variable quadratic programming, we can give its analytical solution, i.e.

$$\alpha_\beta = \begin{cases}-\mathbf{g}_\beta^{m-1}/K(\mathbf{x}_\beta,\mathbf{x}_\beta), & if \quad -\mathbf{g}_\beta^{m-1}/K(\mathbf{x}_\beta,\mathbf{x}_\beta)>0\\ 0, & if \quad -\mathbf{g}_\beta^{m-1}/K(\mathbf{x}_\beta,\mathbf{x}_\beta)\le 0\end{cases}. \tag{8}$$

According to the positive definite property of kernel function, we have $K\left(\mathbf{x}_\beta,\mathbf{x}_\beta\right)>0$. Thus (8) can be further simplified as

$$\alpha_\beta = \begin{cases}-\mathbf{g}_\beta^{m-1}/K(\mathbf{x}_\beta,\mathbf{x}_\beta), & if \quad \mathbf{g}_\beta^{m-1}<0\\ 0, & if \quad \mathbf{g}_\beta^{m-1}\ge 0\end{cases}. \tag{9}$$

Combining (7) and (9), we get

$$h_{\beta}^{m-1}=\min_{\alpha\geq 0}\left(\frac{1}{2}\alpha^{2}K\left(\mathbf{x}_{\beta},\mathbf{x}_{\beta}\right)+\alpha\mathbf{g}_{\beta}^{m-1}\right)=\begin{cases}-\left(\mathbf{g}_{\beta}^{m-1}\right)^{2}\Big/2K\left(\mathbf{x}_{\beta},\mathbf{x}_{\beta}\right), if\ \mathbf{g}_{\beta}^{m-1}<0\\ 0,\quad if\ \mathbf{g}_{\beta}^{m-1}\geq 0\end{cases}. \tag{10}$$

In GS-SVMs, each basis function corresponds to a specified training sample and vice versa. Hence, if the basis function $K\left(\mathbf{x},\mathbf{x}_{\beta}\right)$ does not appear in f_m, we say its corresponding training sample $\mathbf{x}_{\beta}$ unused. From (10), we can derive that if the gradients of all the unused training samples are larger than zero, the loss function (1) will stop decreasing. Hence we will terminate the algorithm if the above condition is satisfied.

Considering (9) and (10), we can obtain the parameter pairs $\left(\alpha_m,\beta_m\right)$ by the following equations

$$\beta_m=\arg\min_{\beta\in Q}\left(h_{\beta}^{m-1}\right). \tag{11}$$

$$\alpha_m=-\mathbf{g}_{\beta_m}^{m-1}\Big/K\left(\mathbf{x}_{\beta_m},\mathbf{x}_{\beta_m}\right). \tag{12}$$

Thus the greedy stagewise algorithm for SVMs (GS-SVMs) can be described as

```
Algorithm 1: GS-SVMs
 1. Set $f_0(\mathbf{x})=0$, $\alpha=\mathbf{0}$, $\mathbf{g}^0=-\mathbf{1}$, $\mathbf{h}^0=-\mathbf{1}$, $Q=\{1,2,\cdots l\}$, $P=\varnothing$;
 2. For $m=1$ to $l$, do:
 3.   If $\min_{\beta\in Q}\left(g_{\beta}^{m-1}\right)\geq 0$, stop;
 4.   $\beta_m=\arg\min_{\beta\in Q}\left(h_{\beta}^{m-1}\right)$, $\alpha_m=-\mathbf{g}_{\beta_m}^{m-1}\Big/K\left(\mathbf{x}_{\beta_m},\mathbf{x}_{\beta_m}\right)$;
 5.   $P=P\cup\{\beta_m\}$, $Q=Q-\{\beta_m\}$;
 6.   $\mathbf{g}_{\beta}^{m}=\mathbf{g}_{\beta}^{m-1}+\beta_m y_{\beta_m} y_{\beta} K\left(\mathbf{x}_{\beta},\mathbf{x}_{\beta_m}\right)$,  $\beta\in Q$;
 7. Update $h_{\beta}^{m-1}$,  $\beta\in Q$ according to (4.9);
 8.     $f_m(\mathbf{x})=f_{m-1}(\mathbf{x})+\alpha_m y_{\beta_m} K\left(\mathbf{x},\mathbf{x}_{\beta_m}\right)$;
 9. End For
10. End Algorithm
```

Fig. 1. Pseudo code of GS-SVMs

Updating $\mathbf{g}_{\beta}^{k}$, $\beta\in Q$ is an operation of cost $O(l)$ and successive n update incurs a computational cost of $O(nl)$, where n is the size of support vector. Besides that, the memory requirement of GS-SVMs is only $O(l)$.

3 Empirical Comparison

In all the experiments, the kernel matrix is constructed by Gaussian kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\theta\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2\right)$. Following [4], we compare the number of kernel evaluations of GS-SVMs and SMO, which is an effective measure of the algorithm's speed. For the sake of fair comparison, we use the same data sets and kernel parameter as in [4]. Note that the number of kernel evaluations of SMO in Table 1. denotes the average number under the different capacity control parameters.

Table 1. Number of kernel evaluations of GS-SVMs and SMO. Each unit corresponds to 10^6 kernel evaluations. SMO-1 and SMO-2 correspond to SMO-Modification 1 and SMO-Modification 2 in [5]

Problems	Size	θ	Dim	SMO-1	SMO-2	GS-SVMs
Adult-1	1605	0.05	123	29.518	17.375	0.845
Adult-4	4781	0.05	123	344.977	231.349	6.791
Adult-7	16100	0.05	123	856.212	698.864	73.014
Web-1	2477	0.05	300	11.543	11.187	0.439
Web-4	7366	0.05	300	79.415	79.008	3.224
Web-7	24692	0.05	300	691.419	703.495	31.981

From Table 1, we can see that GS-SVMs obtain the speedup range from 10 to 30 on the different data sets. In order to validate the performance of GS-SVMs, we compare it with hard magin and Soft margin SVMs on the fifteen benchmark data sets that are from UCI machine learning repository [11]. One-against-one method is used to extend binary classifiers to multi-class classifiers.

On each data set, ten-fold cross validation is run. The average accuracy of ten-fold cross validation is reported in Table 2. For each training-test pair, ten-fold cross validation is performed on training set for tuning free parameters. The detailed experiment setup is the following:

(a) For soft margin SVMs, Kernel width and capacity control parameter are chosen from intervals $\log 2(\theta) = [-8, -7, \cdots, 7, 8]$ and $\log 2(C) = [-1, 0, 1, \cdots 8, 9, 10]$. This range is enough for our problems. The number of trainings on each training-test pair needed by this method is $10 \times 17 \times 12 = 2040$.

(b) For GS-SVMs and HM-SVMs, Kernel width is chosen from interval $\log 2(\theta) = [-8, -7, \cdots, 7, 8]$. The number of trainings on each training-test pair needed by this method is $10 \times 17 = 170$.

The two-tailed *t*-tests also indicate that GS-SVMs are significantly better than SVMs on Glass and worse than SVMs on Liver. As for the remaining data sets, GS-SVMs and SVMs obtain the similar performance. Hence we have the conclusion that GS-SVMs are significantly better in speed than SMO and comparable in performance with SMO.

Tabel 2. Accuracy of GS-SVMs, HM-SVMs and SVMs

Problems	Size	Dim	Class	GS-SVMs	HM-SVMs	SVMs
Australian	690	15	2	**84.93**	78.55	84.49
German	1000	20	2	74.20	69.30	**75.40**
Glass	214	9	6	71.54	68.66	66.81
Heart	270	13	2	**83.70**	76.67	83.23
Ionosphere	351	34	2	94.00	94.00	**94.02**
Iris	150	4	3	95.33	92.00	**96.00**
Liver	345	6	2	66.03	61.69	**71.29**
Page	5473	10	4	96.45	96.50	**96.93**
Diabetes	768	8	2	**77.21**	70.55	77.08
Segment	2310	18	7	**97.32**	96.84	97.01
Splice	3175	60	3	**96.72**	96.31	96.25
Vowel	528	10	11	98.29	**99.05**	**99.05**
WDBC	569	30	2	**97.72**	96.49	97.54
Wine	178	13	3	**98.89**	96.64	**98.89**
Zoo	101	10	7	**97.09**	96.09	96.09
Mean	/	/	/	88.63	85.96	**88.67**

4 Conclusion

This paper proposes a greedy stagewise algorithm, named GS-SVMs to deal with the overfitting of HM-SVMs. Empirical comparisons confirm the feasibility and validity of GS-SVMs.

References

1. Boser, B., Guyon, I. and Vapnik, V.: A training algorithm for optimal margin classifiers, In D. Haussler, Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory, ACM Press, (1992) 144-152
2. Cotes, C. and Vapnik, V.: Support vector networks, Machine Learning 20 (1995) 273-279
3. Tipping, M.: parse Bayesian learning and the relevance vector machine, Journal of Machine Learning Research 1 (2001) 211-244
4. Platt, J.: Fast training of support vector machines using sequential minimal optimization In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, Advances in Kernel Methods --- Support Vector Learning, Cambridge, MA: MIT Press, (1999) 185-208
5. Keerthi, S.S., Shevade, S.K., Bhattacharyya, C. and Murthy, K.R.K.: Improvements to Platt's SMO algorithm for SVM classifier design. Neural Computation13 (2001) 637-649
6. Joachims, T.: Making large-scale SVM learning practical, Advances in Kernel Methods-Support Vector learning, Cambridge, MA: MIT Press, (1999) 169-184
7. Vishwanathan, S.V.N., Smola, A.J. and Murty. M.N.: SimpleSVM. In Proceedings of the Twentieth International Conference on Machine Learning, 2003
8. Collobert, R. and Bengio,S.: SVMTorch: Support Vector Machines for Large-Scale Regression Problems Journal of Machine Learning Research,1 (2001) 143-160

9. Girosi, F.: An equivalence between sparse approximation and support vector machines, Neural Computation. 10 (1998) 1455-1480
10. Friedman, J.H.: Greedy Function Approximation: A gradient boosting machine. Annals of Statistics, 29 (2001) 1189-1232
11. Blake, C.L. and Merz, C.J.: UCI repository of machine learning databases (1998).