A Working Set Algorithm for Accelerating the Relevance Vector Machine

David Ben-Shimon
Dept. of Information Systems Engineering
Ben-Gurion University
P.O.B 653, 84105 Beer-Sheva, Israel
dudibs@bgumail.bgu.ac.il

Armin Shmilovici
Dept. of Information Systems Engineering
Ben-Gurion University
P.O.B 653, 84105 Beer-Sheva, Israel
armin@bgumail.bgu.ac.il

Abstract

The runtime complexity of the Relevance Vector Machine (RVM) is $O(N^3)$ which makes it too expensive for moderately sized machine learning problems. We propose a working set algorithm which reduces the runtime complexity to $O(N^2)$. Experiments verify the viability of the method for classification benchmark problems

Keywords: Machine Learning, Data Mining, The Relevance Vector Machine, Working Set

1 Introduction

The Relevance Vector Machine (RVM) is a sparse method for training a generalized linear model (GLM) such as (1),

$$\mathbf{y(x,w)} = \sum_{i=1}^{N} w_i \cdot k(\mathbf{x}, x_i) + \varepsilon \qquad (1)$$

where $k(x, x_i)$ is a bi-variate kernel function centered on each one of the N training data points x_i, $w = [w_1, \ldots, w_N]^T$ is a vector of regression coefficients, and ε is the noise. This means that it will select a subset (often a small subset) of the provided basis functions to use in the final model.

Consider a dataset of input-target pairs, $\{x_i, t_i\}_{i=1}^{N}$ where targets are assumed, *jointly* Normal distributed - $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ are the unknowns to be determined by the algorithm. Each target t_i is also assumed Normally distributed with mean $y(x_i)$ and uniform variance σ^2 of the noise ε. The conditional probability of the targets given the parameters and the data can now be expressed as

$$p\left(\mathbf{t} \mid \mathbf{w}, \sigma^2\right) = \left(2\pi\sigma^2\right)^{\frac{N}{2}} \exp\left\{-\frac{1}{2\sigma^2}\left\|\mathbf{t} - \boldsymbol{\Phi}\mathbf{w}\right\|\right\} \qquad (2)$$

where the data is hidden in the NxN kernel function matrix $\boldsymbol{\Phi}$ representing all the pairs $\phi_{i,j} = k(x_i, x_j)$, $i, j \in [1, \ldots, N]$. ($\boldsymbol{\Phi}$ could be extended to include a possible bias term).

The goal of the RVM is to accurately predict the target function, while retaining as few basis functions as possible in (1). Sparseness is achieved via the framework of sparse Bayesian learning and the introduction of an additional vector of hyper parameters α_i that controls the width of a Normal prior distribution over the precision of each element of w_i.

$$p(w_i \mid \alpha_i) = \sqrt{\frac{\alpha_i}{2\pi}} \exp(-\frac{1}{2}\alpha_j {w_j}^2) \qquad (3)$$

The solution is derived via the following iterative type II maximization of the marginal likelihood $p(\mathbf{t} \mid \boldsymbol{\alpha}, \sigma^2)$ with respect to $(\boldsymbol{\alpha}, \sigma^2)$:

$$\alpha_i^{new} = \frac{1 - \alpha_i \Sigma_{ii}}{{\mu_i}^2} \qquad (4)$$

$$(\boldsymbol{\sigma}^2)^{new} = \frac{\left\|\mathbf{t} - \boldsymbol{\Phi}\boldsymbol{\mu}\right\|^2}{N - \sum_{i=1}^{N}\left(1 - \alpha_i \Sigma_{ii}\right)} \qquad (5)$$

The unknowns $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ are computed as

$$\boldsymbol{\Sigma} = (\boldsymbol{\Phi}^{\mathbf{T}}\mathbf{B}\boldsymbol{\Phi} + \mathbf{A})^{-1} \qquad (6)$$

$$\boldsymbol{\mu} = \boldsymbol{\Sigma}\boldsymbol{\Phi}^{\mathbf{T}}\mathbf{B}\mathbf{t} \qquad (7)$$

where $\mathbf{B} \equiv \sigma^{-2}\mathbf{I}_{NxN}$. The basic RVM algorithm cycles between (4),(5),(6),(7), reducing the dimensionality of the problem when any α_i larger than a preset threshold. The algorithm stops when the likelihood $p(\mathbf{t} \mid \boldsymbol{\alpha}, \sigma^2)$ ceases to increase. The non-zero elements of $\boldsymbol{w}$ are called Relevance Values, and their corresponding data-points are called Relevance Vectors (RVs) as an analogy to the Support Vector Machine [3].

In the binary classification problem each target t_i is Binary: $t_i \in \{0,1\}$. The model (1) is assumed to be noise-free. That is $\sigma^2 \equiv 0$. The sigmoid function $\rho(y) = 1/(1+e^{-y})$ is used to generalize the linear model. The main idea of the sigmoid function is to make an approximation of the regression case to the two-class classification problem. With the sigmoid link function we can adopt the Bernoulli distribution $p(t \mid x)$ and rewrite the likelihood as:

$$p(t \mid w) = \prod_{i=1}^{N} \rho\left(\phi_i^T w\right)^{t_i} \left(1 - \rho\left(\phi_i^T w\right)\right)^{1-t_i} \qquad (8)$$

In the classification RVM framework, we find two solutions of two different coupled problems, a discrete problem and a continuous one.

In the first problem we assume that $\boldsymbol{\alpha}$ is known and we minimize the following cost function for the unknown $\boldsymbol{w}$ (the MAP solution):

$$\begin{aligned} J(\mathbf{w} \mid \mathbf{t}, \boldsymbol{\alpha}) &= \tfrac{1}{2}\mathbf{w}\mathbf{A}\mathbf{w} \\ &- \sum_{i=1}^{N}\left[t_i \log\left(\rho\left(\phi_i^T w\right)\right) + (1-t_i)\log\left(1 - \rho\left(\phi_i^T w\right)\right)\right] \end{aligned} \qquad (9)$$

We are going to solve the following regression RVM for the continuous approximation of the targets, $\tilde{\mathbf{t}}$ (with a different $\boldsymbol{B}$).

$$\tilde{\mathbf{t}} = \boldsymbol{\Phi}^{\mathbf{T}}\mathbf{w} + \mathbf{B}^{-1}\left(\mathbf{t} - \rho\left(\phi_i^T \mathbf{w}\right)\right) \qquad (10)$$

$\mathbf{B} \equiv diag[\beta_1, \ldots, \beta_N]$, and β_i is defined as follows:

$$\beta_i = \rho\left(\phi_i^T \mathbf{w}\right)\left(1 - \rho\left(\phi_i^T \mathbf{w}\right)\right) \qquad (11)$$

In the second problem we solve for the unknowns (μ, Σ) as with the regression RVM. $\mathbf{t}$ is replaced with $\tilde{\mathbf{t}}$, as computed from (10). However, if $\boldsymbol{\mu} \neq \mathbf{w_{MAP}}$ approximately, the solution is not optimal. In this case, we insert the values of α_i in equation (9), resolve again for $\boldsymbol{w}$ and start all over again.

The RVM typically produces *very sparse solutions* compared to the SVM [4], *while retaining similar accuracy*. The RVM is an approximate Bayesian method, thus it can generate not only predicted values, but also the *predicted distribution* of the values [4].

The matrix inversion operation in (6), which requires $O(N^3)$ operations is the computationally intensive part of the algorithm. The matrices $\boldsymbol{\Phi}$ and $\boldsymbol{\Sigma}$ are full rank, thus require initially $O(N^2)$ space complexity. Furthermore, it is common that the inversion of a large matrix becomes ill-conditioned after several cycles even for positive definite matrices unless the parameters of the kernel function are optimized. These problems limit the practicality of the basic RVM algorithm for moderately sized problems.

The motivation of this paper is to enhance the computational bottleneck of the RVM classifier, thus make it more useful to the machine learning community. In [1] we proposed three different partition heuristics *only for the regression RVM* and demonstrated significant speedups. This paper *focuses on the RVM classification problem* – which is more difficult and more interesting in practice. Here, in section 2 we introduce the working set algorithm for the classification problem; Section 3 validates its behavior on a set of benchmark datasets; and Section 4 concludes with a discussion.

2 The Working Set Algorithm

The basic Idea of the working set is consider that somehow we can guess an initial solution $\mathbf{w}^0$. It is not too expensive to use this initial solution for predicting the error, or class membership in classification, for each one of the training examples and rank them according to their predicted errors. The samples which have the largest predicted error are potentially the most informative for improving the solution, thus are introduced to the model as part of the working set. A working set of size $P << N$ is used to compute an improved guess to the solution $\mathbf{w}^1$, and this iterative process is repeated until either a predetermined convergence criteria is being met, the data is exhausted, or the computing time is exhausted.

This conforms to the Bayesian framework of the RVM algorithm: each new data contains further evidence regarding the (unknown) distribution of the full dataset. This new evidence is combined with the a-priori assumption about the distribution of the data (the RVM solution to the previous working set) to form an a-posteriori distribution assumption (via the RVM solution to the current working set). Thus, data needs to be considered in the working set at most once.

The key parameter in the efficiency of the working set algorithm is the size *P*. Based on experiments presented in [1], $P \propto O(\sqrt{N})$.

The outline of the working set algorithm is as follows:

1. Run the basic RVM on a randomly selected working set of size P^0 and estimate P from (14).
2. Run the basic RVM on the first working set of size P and store the resulting RVs.
3. Construct a model from the RVs, predict the error for each one of the remaining (unused) data, and construct a new partition of size P from the data associated with large prediction error. The distribution of classes in the working set should approximate the distribution of classes in the full training set.
4. Update the working set by merging the current RVs with the newly generated partition.
5. Run the basic RVM on the working set and store the resulting RVs.
6. Repeat steps 3, 4, and 5 until exhausting the data.
7. For the last working set, let the basic RVM run until convergence and obtain the final solution.

Complexity analysis: Considering that a sparse solution exists, the premature stopping of the basic RVM on each working set halves the number of RVs in each step. Due to the merging process of the current RV candidates with new chunks of data of size P, there is a gradual increase in the size of the working set to a limit of $2P$: $P, P+\frac{P}{2}, P+\frac{3P}{4}, P+\frac{7P}{8}, \cdots, \approx 2P$. There are approximately $\frac{N}{P}$ working sets to process. Each one of them is of size order $P \propto O(\sqrt{N})$, and its runtime complexity is $O(P^3)$. Thus, the overall runtime complexity of the working set algorithm $\frac{N}{P} \cdot P^3 = NP^2 = N(\sqrt{N})^2$ is $O(N^2)$.

3 Experiments

The purpose of this section is to validate experimentally the speedup resulting from the working set algorithm. The basic RVM and the working set RVM algorithms were implemented using the MATLAB scripting language, on a Pentium IV processor running the Windows XP operating system at 3.1 GHz equipped with 2GB physical memory. Each algorithm was simulated for 10 different splits into training and testing sets (same data for both algorithms). The means and standard deviations for each set of ten simulations are presented in the Tables. A Gaussian kernel was used in all the experiments. Its width parameter was optimized from cross-validations of the training sets (same width used for both algorithms).

For the first set of experiments we used Ripley's data - a two dimensional synthetic data set, generated from mixtures of two Gaussians [4] There is 8% class overlap for the optimal theoretical classifier. A testing set of size 500 was used to measure the accuracy for a Gaussian kernel with width parameter optimized to 0.5.

Table 1 presents the comparison. The classification test error is practically the same for both algorithms – about 9%. The working set algorithm is much faster – especially for the larger N. The number of RVs seems similar for both algorithms.

Table 1: Comparison for the Ripley dataset

	Time		# RVs	
N	**Basic**	**W.S.**	**Basic**	**W.S.**
200	1.43±0.18	1.6±0.3	4.5±0.6	4.5±0.6
400	9.4±0.83	5.5±0.7	5.3±0.5	5.0±0.8
600	25.8±0.9	11.4±3.4	6±1.2	6.5±0.6
800	60.5±0.9	18.8±3.2	5.3±0.5	5.5±0.6
1000	108.6±9.8	24.38±2	5.3±1.0	5.8±0.5
1200	190.5±6.87	33.46±6.8	6±0.5	5.8±0.5
1400	278.5±8.2	44.6±2.3	6.3±1.7	7.0±0.8
1600	415.7±65.3	59.2±5.2	6±0.8	6.8±1.3
1800	564.1±30.8	69.3±3.4	6.5±1	6.8±.96
2000	808±58.4	88.5±2.8	5.8±1.0	6.8±0.5

Figure 1 presents a graphical representation of the time as a function of N for both algorithms.

A nonlinear regression of the form $y = aN^b$ generated the following solutions (y denotes the time): for the basic RVM $y = 7 \cdot 10^{-7} N^{2.74}$ with $R^2 = 0.9996$ and for the working set algorithm $y = 2 \cdot 10^{-4} N^{1.71}$ with $R^2 = 0.9986$

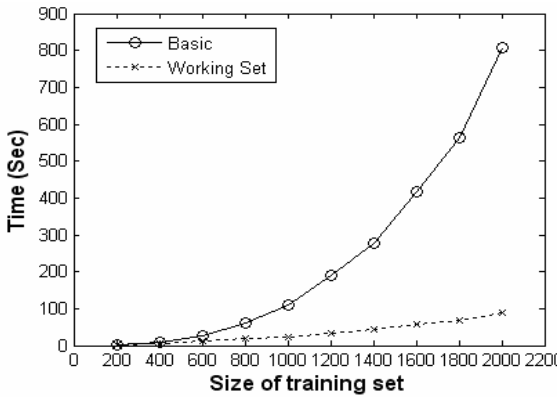

Figure 1: Comparing the runtimes of both algorithms for Ripley's classification problem.

The second set of experiments test the working set RVM classifier on natural two-class benchmark datasets from the literature [4]. Table 2 presents the attributes of these datasets, and the split between training and testing sets.

Table 3 presents a comparison of the simulation times and the #RVs for both algorithms. The two right most columns on Table 2 present the classification test error. For both algorithms the classification error and the #RVs are practically the same while the working set is much faster.

Table 2: Details of the benchmark datasets used for the classification experiments

Data set name	Size	d	Train /test set	Class. Error (%)	
				Basic	W.S.
Breast cancer	277	9	250/27	29.7±6.5	30± 7.5
Pima Indian	768	8	692/76	23± 4.1	22± 3.9
German	1000	20	900/100	26.5±4.6	25.9±4.2
Titanic	2201	3	1981/220	21.6±.02	21.6±.02
Banana	5300	2	1800/3500	9.45±.63	9.52±.56

The Titanic dataset presents a case where the #RV is much larger than the estimate for the size of the working set $P=78$. Nevertheless, the working set algorithm was able to find a solution comparable to that of the simple RVM, and still be faster, though less than expected.

Table 3: Results of the classification experiments

Data set	Time(Sec.)		# RVs	
	Basic	W.S.	Basic	W.S.
Breast Cancer	1.8±0.12	1.6±0.2	3.4±1	3.8±0.64
Pima Indian	28.4±2.4	9.2±0.9	9.8±1.9	10±1.5
German	73.9±3.3	24.7±1.5	25.1±5.4	24.7±3.5
Titanic	1344±287	597±145	503.9±5.4	458.9±85.2
Banana	4066±194	389±30	18.4±1.3.	18.1±1.6

4 Discussion

Bayesian methods are notorious for their runtime complexity. The basic RVM algorithm is an approximate Bayesian method with some very attractive properties; however, in its original form its runtime complexity of $O(N^3)$ is excessive for realistic medium or large datasets.

This paper introduces a working set algorithm for accelerating the basic RVM via splitting the dataset into manageable – carefully constructed chunks. Analysis of the complexity of the new algorithm indicates that its runtime complexity is reduced to $O(N^2)$. Extensive simulation experiments on benchmark datasets verify the runtime complexity reduction while the solution retains the accuracy and the sparseness of the basic RVM. A further advantage of the current algorithm is that it is expected to produce fairly decent solutions when aborted before normal completion. In this case, the RVs of the last completed working set represent the solution.

The idea of accelerating the RVM via partitioning was also suggested by [2]. However, they compute the error on each partition separately – effectively sub-sampling the original data and possibly reducing the accuracy of the solution. A different approach was taken by [5] a greedy algorithm which starts with a working set of size 1, and adds sequentially only candidate RVs which reduce the negative log-likelihood. Unfortunately, the log-likelihood is rather flat, so the algorithm may miss important RVs due to its greedy nature.

This paper provides another essential step for popularizing the RVM algorithm, so that eventually it will turn into another "standard tool" for the practitioner of machine learning.

References

[1] D. Ben-Shimon, A. Shmilovici (2006). Accelerating the Relevance Vector Machine via Data Partitioning, *Journal of Computing and Decision Sciences*, forthcoming.

[2] T.A. Down, T.J.P Hubbard (2003). Relevance Vector Machines for Classifying Points and Regions in Biological Sequences, Wellcome Trust, Sanger Institute.

[3] A. Shmilovici (2005). Support Vector Machines. In O. Maimon and L. Rokach (editors), *Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers*, Springer.

[4] M.E. Tipping (2001). Sparse Bayesian Learning and the Relevance Vector Machine. *Journal of Machine Learning Research* 1, 211-244.

[5] M.E. Tipping, A. Faul (2003). Fast Marginal Likelihood Maximization for Sparse Bayesian Models. *Proceedings of the 9th International workshop on Artificial Intelligence and Statistics*, January 3-6, Key West, Florida.