01-121

Van Gestel T., Suykens J.A.K., De Moor B., Vandewalle J., ``Bayesian inference for LS-SVMs on large data sets using the Nystrom method", in *Proc. of the World Congress on Computational Intelligence - International Joint Conference on Neural Networks (WCCI-IJCNN 2002)*, Honolulu, USA, May 2002, pp. 2779-2784., Lirias number: 182347.

Bayesian Inference for LS-SVMs on Large Data Sets using the Nyström Method

T. Van Gestel, J.A.K. Suykens, B. De Moor & J. Vandewalle

K.U.Leuven, Dept. of Electrical Engineering, ESAT-SISTA,
Kasteelpark Arenberg 10, B-3001 Leuven, Belgium
E-mail: {tony.vangestel,johan.suykens}@esat.kuleuven.ac.be.

Abstract

In Support Vector Machines (SVMs), Least Squares Support Vector Machines (LS-SVMs) and other kernel based techniques for regression and classification the solution follows from a convex optimization problem for a fixed choice of the hyperparameters. However, these methods involve the calculation, storage and typically also inversion of the kernel matrix with size equal to the number of data points. Therefore, large scale techniques like sequential minimal optimization (SMO) and conjugate gradient algorithms have been developed in order to solve the SVM and LS-SVM, respectively. In Bayesian inference for SVMs and LS-SVMs one also needs to compute the inverse and eigenvalue decomposition of the kernel matrix, which is again computationally intensive. In this paper, we discuss large scale approximations for Bayesian inference for LS-SVMs. A practical implementation using the Nyström method is implemented which allows to obtain approximate expressions at the different levels of inference within the evidence framework. The method is then evaluated on a number of benchmark problems.

1 Introduction

Recently, kernel based methods have become powerful models for regression and classification tasks [4, 8, 9, 16]. In Support Vector Machines (SVMs) [4, 15] and Least Squares Support Vector Machines (LS-SVMs) [9, 10] the solutions are obtained from a convex quadratic programming problem and a linear Karush-Kuhn-Tucker system, respectively. As the Least Squares Support Vector Machines involve the use of a least squares cost function, the sparseness property of SVMs is lost. On the other hand the LS-SVMs have been related to regularization networks, Gaussian Processes and kernel Fisher Discriminant Analysis [3, 5, 8, 13, 16]. Sparseness in the LS-SVM can be obtained by sequentially pruning the support value spectrum, while robustness is obtained by using a weighted least squares cost function [11]. Compared to multilayer perceptrons (MLPs) [1], the SVMs have the advantage of solving a convex optimization problem, while the cost function of MLPs typically has multiple local minima. On the other hand the SVM formulations involve the use of a square kernel matrix with size equal to the number of data points. This makes a straightforward implementation of kernel methods computationally less attractive for large data sets. Therefor, large scale implementations like sequential miminimal optimization and conjugate gradient implementations have been developed [4, 10] to solve the convex optimisation problem on the first level of inference.

A powerful tool to estimate the uncertainties on the prediction and classification of MLPs is the evidence framework [7]. Bayesian inference can also be used to select the regularization hyperparameters within the statistical framework and to perform model comparison. In Bayesian methods for SVMs and LS-SVMs the kernel matrix also appears in expressions for Bayesian hyperparameter inference [6, 12, 13] and in improved upper error bounds on the generalization behavior. The inverse kernel matrix is also needed when calculating error bars or class uncertainties on the output of the regressor or classifier, respectively. In this paper, we will discuss and compare large scale methods that allow to apply also the Bayesian evidence framework to large data sets.

This paper is organized as follows. The basic formulas in the dual space of LS-SVMs of the Bayesian framework are given in Section 2. The large scale formulations are derived in Section 3. An empirical evaluation is reported in Section 4.

2 Bayesian Framework for LS-SVMs

The LS-SVM regressor $y = w^T\varphi(x) + b$ and classifier $y = \text{sign}[w^T\varphi(x) + b]$ (with binary targets $y \in \{-1, +1\}$) are inferred from the data $D = \{(x_i, y_i)\}_{i=1}^{n_D}$ by minimizing the cost function [9]

$$\min_{w,b} \mathcal{J}_1(w, b) = \mu E_W + \zeta E_D = \frac{\mu}{2} w^T w + \frac{\zeta}{2} \sum_{i=1}^{n_D} e_i^2 \quad (1)$$

subject to the constraints

$$e_i = y_i - (w^T\varphi(x_i) + b), \qquad i = 1, \ldots, n_D. \quad (2)$$

The regularization and error term are defined as $E_W = \frac{1}{2} w^T w$ and $E_D = \frac{1}{2} \sum_{i=1}^{n_D} e_i^2$, respectively. The trade-off between regularization and training error is determined by the ratio $\gamma = \zeta/\mu$.

This cost function is obtained in [9, 10] by modifying Vapnik's SVM formulation [15] so as to obtain a linear system in the dual space. Constructing the Lagrangian by introducing the Lagrange multipliers α_i for the equality constraints (2), a linear system is obtained in the dual space

$$\left[\begin{array}{c|c} 0 & 1 \\ \hline 1^T & \Omega + \gamma^{-1} I_{n_D} \end{array}\right] \left[\begin{array}{c} b \\ \hline \alpha \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline y \end{array}\right] \quad (3)$$

with $\gamma = \zeta/\mu$, $y = [y_1; \ldots; y_{n_D}]$, $1 = [1; \ldots; 1]$, $\alpha = [\alpha_1; \ldots; \alpha_{n_D}]$, and where Mercer's condition is applied within the Ω matrix $\Omega_{ij} = \varphi(x_i)^T\varphi(x_j) = K(x_i, x_j)$. For large scale data sets the linear system (3) can be represented by two linear systems with positive definite system matrices. These linear systems can then be solved using the Hestenes-Stiefel conjugate gradient algorithm [10].

Possible kernel functions are, e.g., a linear kernel $K(x_1, x_2) = x_1^T x_2$ and an RBF-kernel $K(x_1, x_2) = \exp(-\|x_1 - x_2\|_2^2/\sigma^2)$, where Mercer's condition holds for all possible choices of the kernel parameter $\sigma \in \mathbb{R}$. The LS-SVM regressor is then constructed as follows

$$y(x) = w^T\varphi(x) + b = \sum_{i=1}^{n_D} \alpha_i K(x, x_i) + b, \quad (4)$$

while the classifier $y = \text{sign}(z)$ is obtained by taking the sign of the latent variable $z = \sum_{i=1}^{n_D} \alpha_i K(x, x_i) + b$.

2.1 Inference of model parameters (Level 1)

A Bayesian framework has been related to the LS-SVM regressor and classifier formulation (1) by taking the Gaussian prior

$$p(w, b|\log\mu, \mathcal{H}) \propto \left(\frac{\mu}{2\pi}\right)^{\frac{n_f}{2}} \exp(-\frac{\mu}{2} w^T w) \quad (5)$$

and the likelihood (assuming i.i.d. data)

$$\begin{aligned} p(D|w, b, \log\zeta, \mathcal{H}) &\propto \textstyle\prod_{i=1}^{n_D} p(y_i|x_i, w, b, \log\zeta, \mathcal{H}) \\ &= \textstyle\prod_{i=1}^{n_D} \left(\frac{\zeta}{2\pi}\right)^{\frac{1}{2}} \exp(-\frac{\zeta}{2}(y_i - (w^T\varphi(x_i) + b))^2) \\ &= \left(\frac{\zeta}{2\pi}\right)^{\frac{n_D}{2}} \exp(-\tfrac{\zeta}{2} \textstyle\sum_{i=1}^{n_D} e_i^2). \end{aligned}$$

Observe that the prior depends on the hyperparameter μ, while the likelihood depends on the hyperparameter ζ. The kernel function K and the possible kernel parameters are represented by means of $\mathcal{H}$. Applying now Bayes' formula, one obtains the posterior probability as

$$\begin{aligned} &p(w, b|D, \log\mu, \log\zeta, \mathcal{H}) \\ &= \frac{p(D|w, b, \log\mu, \log\zeta, mH) p(w, b|\log\mu, \log\zeta, \mathcal{H})}{p(D|\log\mu, \log\zeta, \mathcal{H})} \\ &\propto p(D|w, b, \log\mu, \log\zeta, mH) p(w, b|\log\mu, \log\zeta, \mathcal{H}), \end{aligned} \quad (6)$$

where the last step is obtained since the evidence $p(D|\log\mu, \log\zeta, \mathcal{H})$ is a normalizing constant that does not depend upon w and b. Taking the negative logarithm of the posterior probability $p(w, b|D, \log\mu, \log\zeta, \mathcal{H})$ and neglecting all constants one obtains the least squares cost function (1) with ridge regression.

This probabilistic framework also allows to infer the uncertainty on the model parameters around their most probable value obtained from (3) in the dual space. Calculating the Hessian and using matrix algebra, one obtains expressions in the dual space that yield confidence bounds for prediction problems and posterior class probabilities for classification problems. In [12, 13] these expressions are evaluated using the eigenvalue decomposition of the centered kernel matrix $M_c\Omega M_c$, with M_c the idempotent centering matrix $M_c = I_{n_D} - \frac{1}{n_D} 1 1^T$. For large scale data sets this eigenvalue decomposition is computationally expensive. Therefore, there is a need for approximations for large scale data sets.

2.2 Inference of hyperparameters (Level 2)

The performance of the LS-SVM also depends upon the choice of the regularization parameters μ and ζ and the kernel parameter σ of the RBF kernel. Within the Bayesian framework the regularization parameter $\gamma = \zeta/\mu$ is obtained by minimizing [12, 13]

$$\begin{aligned} &\min_\gamma \mathcal{J}_2(\gamma) = \textstyle\sum_{i=1}^{n_D - 1} \log(\lambda_{\Omega,i} + \gamma^{-1}) \\ &\quad + (n_D - 1) \log[E_W(w_{MP}) + \gamma E_D(w_{MP}, b_{MP})], \end{aligned} \quad (7)$$

with gradient

$$\frac{\partial \mathcal{J}_2(\gamma)}{\partial \gamma} = -\frac{1}{\gamma}\sum_{i=1}^{n_D-1}(1+\gamma\lambda_{\Omega,i})^{-1} + (n_D-1)\frac{E_D(w_{MP},b_{MP})}{E_W(w_{MP})+\gamma E_D(w_{MP},b_{MP})}, \quad (8)$$

where $\lambda_{\Omega,1}, \ldots, \lambda_{\Omega,n_D-1}$ are the largest n_D-1 eigenvalues of the centered kernel matrix $M_c\Omega M_c$. The level 1 cost function $E_W(w_{MP})+\gamma E_D(w_{MP},b_{MP})$ can be expressed as

$$E_W(w_{MP})+\gamma E_D(w_{MP},b_{MP}) = \frac{1}{2}y^T M_c(M_c\Omega M_c+\gamma^{-1}I_{n_D})^{-1}M_c y.$$

This expressions avoids the explicit solution of the linear system (3) for all possible γ values. As these expressions need to be evaluated for different values of γ one can compute first an eigenvalue decomposition of $M_c\Omega M_c$. This allows then to compute E_W, E_D and $E_W+\gamma E_D$ from matrix vector multiplications only, avoiding the (computationally intensive) calculation of the inverse $(M_c\Omega M_c+\gamma^{-1}I_{n_D})^{-1}$ [13, 14].

Given the optimal γ from (7) one finds the effective number of parameters γ_{eff} from $\gamma_{eff} = (n_D+\gamma E_D/E_W)/(1+\gamma E_D/E_W)$. The optimal μ and ζ are obtained from $\mu = (\gamma_{eff}-1)/(2E_W(w_{MP})$ and $\zeta = (n_D-\gamma_{eff})/(2E_D(w_{MP},b_{MP}))$.

2.3 Tuning of the parameters of the RBF-kernel (Level 3)

At the third level of inference one also takes the uncertainty on the inferred hyperparameters μ and ζ into account in order to infer $p(\mathcal{H}|D)$. These error bars are approximately equal to $\sigma^2_{\log\mu} = 2/(\gamma_{eff}-1)$ and $\sigma^2_{\log\zeta} = 2/(n_D-\gamma_{eff})$, respectively. Using this additional uncertainty into the Occam factor on level 3, one obtains $p(\mathcal{H}|D)$ [12, 13].

A practical approach is to define a grid $\boldsymbol{\sigma} = [\sigma_1;\ldots;\sigma_n]$ of kernel parameters and to evaluate the level 3 probability $p(D|\mathcal{H})$ for each σ_i value. For this evaluation one needs to solve (7) with respect to γ and then estimate μ and ζ. For this evaluation one needs to compute the eigenvalue decomposition of $M_c\Omega M_c \in \mathbb{R}^{n_D\times n_D}$. As the memory and computational requirements are $\mathcal{O}({n_D}^2)$ and $\mathcal{O}({n_D}^3)$, respectively, this becomes infeasible for large data sets. In the next Section approximations for the eigenvalues and eigenvectors are made.

2.4 Automatic Relevance Determination

Automatic Relevance Determination (ARD) allows to infer the relevance from the different inputs from the given data. For LS-SVMs ARD is performed by using a diagonal weighting matrix [8] $U = \mathrm{diag}(\boldsymbol{u}) = \mathrm{diag}([u_1;\ldots;u_n])$. Each weight $u_l \in \mathbb{R}^+$ weights the corresponding input x_l, $l=1,\ldots,n$ in the kernel function K. Inputs corresponding to low (almost zero) weights have a small relevance, while input with relatively high weights have a large relevance. For an RBF-kernel, the kernel function becomes

$$K(x_1,x_2) = \exp(-(x_1-x_2)^T U(x_1-x_2)/\sigma^2) = \exp(-(x_1-x_2)^T\bar{U}(x_1-x_2)),$$

where the positive scale parameter σ is taken into account by defining $\bar{U} = U/\sigma = \mathrm{diag}(\bar{u})$. The weights $\bar{u}$ are inferred by maximizing the model evidence [14].

3 Large scale approximation

For large data sets the kernel matrix Ω becomes too large to compute the eigenvalues $\lambda_{\Omega,i}$. Therefor, one has to use an approximation to calculate the eigenvalues [2, 17]. Given the kernel function $K(x,v)$ with eigenfunction ϕ_i such that

$$\int K(x,v)\phi_i(x)p(x)dx = \lambda_i\phi_i(v), \quad (9)$$

one approximates the eigenfunction by sampling from the input probability $p(x)$ using

$$\frac{1}{n_D}\sum_{l=1}^{n_D}K(v,x_l)\phi_i(x_l) \simeq \lambda_i\phi_i(v). \quad (10)$$

Using the eigenvalue decomposition $\Omega = U\Lambda U^T$ the Nyström approximation [17] to the eigenvalues $\lambda_i \simeq \Lambda_{ii}$ and the eigenfunction $\phi_i(u)$ is obtained as

$$\phi_i(u) \simeq \frac{\sqrt{n_D}}{\lambda_{\Omega,i}}\sum_{l=1}^{n_D}K(v,x_l)U_{li}. \quad (11)$$

For instances from the training set, the above approximation simplifies to $\phi_i(x_j) \simeq \sqrt{n_D}U_{ji}$.

We now use two sets of data points to approximate ϕ_i and λ_i. The first set is the full training data set $D = \{x_i,y_i\}_{i=1}^{n_D}$, while the second data set $\tilde{D}_S = \{\tilde{x}_j,\tilde{y}_j\}_{j=1}^{\tilde{n}_{\tilde{D}}}$ is obtained by taking a subsample of D. Typically we use $\tilde{n}_{\tilde{D}} \ll n_D$ in order to reduce the computational requirements significantly. Both data sets can now be used to approximate the eigenvalues λ_i and the corresponding eigenvalues ϕ_i. Putting both approximations equal to each other, we find that

$$\lambda_{\Omega,i} \simeq \frac{n_D}{\tilde{n}_{\tilde{D}}}\tilde{\lambda}_{\tilde{\Omega}_i}$$
$$\Omega \simeq K\tilde{\Omega}^{-1}K^T, \quad (12)$$

where $\tilde{\Omega} \in \mathbb{R}^{\tilde{n}_{\tilde{D}}\times\tilde{n}_{\tilde{D}}}$ and $K \in \mathbb{R}^{n_D\times\tilde{n}_{\tilde{D}}}$ have elements $K_{ij} = K(x_i,\tilde{x}_j)$ and $\Omega_{ij} = K(\tilde{x}_i,\tilde{x}_j)$,

respectively. Defining the feature space matrices $\Phi = [\varphi(x_1), \ldots, \varphi(x_{n_D})] \in \mathbb{R}^{n_f \times n_D}$ and $\tilde{\Phi} = [\varphi(\tilde{x}_1), \ldots, \varphi(\tilde{x}_{\tilde{n}_{\tilde{D}}})] \in \mathbb{R}^{n_f \times \tilde{n}_{\tilde{D}}}$, it is seen that the approximation (12) corresponds to the approximation $\Omega \simeq \Phi^T[\tilde{\Phi}(\tilde{\Phi}^T\tilde{\Phi})^{-1}\tilde{\Phi}^T]\Phi$. This approximation is exact when $\text{rank}([\Phi\ \tilde{\Phi}]) = \text{rank}(\tilde{\Phi})$. When $n_f \leq \tilde{n}_{\tilde{D}}$ (e.g., in the case of a linear kernel) this assumption typically holds, while it is assumed that still a good approximation is obtained in other cases.

Given the approximation (12) we can now apply it in order to calculate

$$\begin{aligned}(\gamma^{-1}I_{n_D} + \Omega)^{-1} &\simeq (\gamma^{-1}I_{n_D} + K\tilde{\Omega}^{-1}K^T)^{-1} \\ &= \gamma I_{n_D} - \gamma^2 K(\tilde{\Omega} + \gamma K^T K)^{-1}K^T. \end{aligned} \quad (13)$$

As this function needs to be evaluated many times for different value of the parameter γ it is useful to consider a simulaneous diagonalization of both $\tilde{\Omega}$ and K^TK. First one computes the symmetric eigenvalue decomposition $\tilde{\Omega} = U_{\tilde{\Omega}}\Lambda_{\tilde{\Omega}}U_{\tilde{\Omega}}^T$ and then computes the eigenvalue decomposition $\Lambda_{\tilde{\Omega}}^{-1/2}U_{\tilde{\Omega}}^T K^T K U_{\tilde{\Omega}}\Lambda_{\tilde{\Omega}}^{-1/2} = U_2\Lambda_2 U_2^T$. Then one can write

$$\begin{aligned}&(\tilde{\Omega} + \gamma K^T K)^{-1} \\ &= (U_2^T\Lambda_{\tilde{\Omega}}^{1/2}U_{\tilde{\Omega}}^T)^{-1}(I + \gamma^{-1}\Lambda_2)^{-1}(U_{\tilde{\Omega}}\Lambda_{\tilde{\Omega}}^{1/2}U_2)^{-1}.\end{aligned}$$

When (13) needs to be evaluated for many values of γ, it becomes computationally more interesting to compute the two eigenvalue decompositions. The inverse can then be easily calculated using only matrix-vector multiplications.

One can also approximate the first term in (7) as follows

$$\begin{aligned}&\textstyle\sum_{i=1}^{n_D-1}\log(\lambda_{\Omega,i} + \gamma^{-1}) \\ &\simeq \textstyle\sum_{i=1}^{\tilde{n}_{\tilde{D}}-1}\log(\frac{n_D}{\tilde{n}_{\tilde{D}}}\lambda_{\tilde{\Omega},i} + \gamma^{-1}) - (n_D - \tilde{n}_{\tilde{D}})\log(\gamma).\end{aligned} \quad (14)$$

As at most $\tilde{n}_{\tilde{D}} - 1$ eigenvalues are non-zero, the remaining $n_D - \tilde{n}_{\tilde{D}}$ eigenvalues are put equal to zero. The computational advantage of the Nyström method [2, 17] becomes now clear as one only needs to compute eigenvalue decompositions and inverses of matrices of size $\tilde{n}_{\tilde{D}}$ instead of size n_D. Compared to other approaches from numerical linear algebra that only compute the largest eigenvalues and corresponding eigenvectors, the Nyström method has still a considerable advantage.

The above expressions are typically used for Gaussian Processes [17] where one also uses these approximations to estimate the parameters of the point prediction (4). As the SVM and LS-SVM involve a centering in the feature space, we will use these approximations on the centered kernel matrices $M_c\Omega M_c$ and $\tilde{M}_c\tilde{\Omega}_c\tilde{M}_c$, with $\tilde{M}_c = I_{\tilde{n}_{\tilde{D}}} - 1/\tilde{n}_{\tilde{D}}\tilde{1}\tilde{1}^T \in \mathbb{R}^{\tilde{n}_{\tilde{D}} \times \tilde{n}_{\tilde{D}}}$ and $\tilde{1} = [1; \ldots; 1] \in \mathbb{R}^{\tilde{n}_{\tilde{D}}}$. An important difference with [17] is that we use the approximations only at level 2 and 3, while we use the Hestenes-Stiefel conjugate gradient algorithm here in order to solve the level 1 problem (3) for large scale systems [10].

4 Simulation results

We will validate the results first on relatively small data sets which allows us to assess the performance of the approximations. Then the method is further analyzed on the adult data set. In the first example the LS-SVM is used to approximate the sinc function. The Nyström method is evaluated on benchmark data sets in the second example.

4.1 Approximation of the sinc function

We illustrate the ARD algorithm for an RBF-kernel using the Nyström method on a synthetic data set constructed in the same way as in [14]. The data set consists of a training set and test set of $n_D = 200$ and $n_{test} = 1000$ data points, respectively. The 3 inputs were generated as follows. The first input $x(1)$ is generated by sampling uniformly in the interval $[-0.5, 0.5]$. The second input $x(2)$ is constructed from the first input by adding zero mean Gaussian distributed noise with variance 0.05^2. The third input $x(3)$ is zero mean random noise with variance 0.5^2. The output data y are generated from the first input by adding Gaussian noise as follows:

$$y_i = \text{sinc}(2\pi x_i(1)) + e_i, \quad (15)$$

with $\text{sinc}(x) = \sin(\pi x)/(\pi x)$ and e_i zero mean Gaussian noise with variance $\text{var}(e_i) = 0.1^2$. Hence, the first input is the most relevant, the second input is somewhat relevant and the third input is not relevant. The test data points are visualized in Figure 1.

sinc	$100 \times \text{MSE}_{test}$	$\log\sigma$	$\log\gamma$
100%	1.007	-0.223	4.084
75%	1.007 (0.00)	-0.273 (0.00)	4.087 (0.02)
50%	1.007 (0.00)	-0.270 (0.15)	3.931 (0.54)
25%	1.008 (0.00)	-0.317 (0.72)	3.784 (0.72)
10%	1.012 (0.00)	-0.552 (0.23)	3.225 (0.69)

Table 1: Average test set MSE and value of the hyperparameters γ and σ for a different number of sampling data points $\tilde{n}_{\tilde{D}} = 10\% \ldots 100\% n_D$.

All inputs were normalized to zero mean and unit vari-

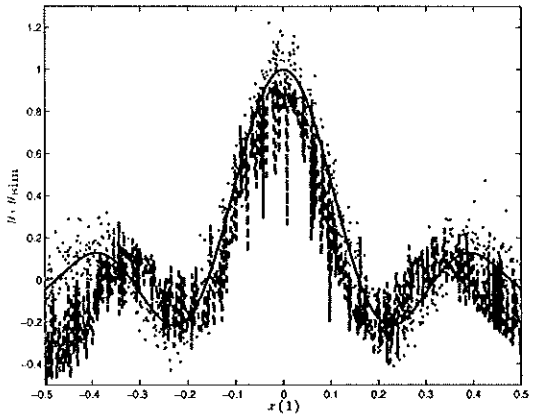

Figure 1: Test set data points $\{(x_i(1), y_i)\}_{i=1}^{1000}$ and predicted outputs $y_i = f(x_i([1:3]))$(dashed line) for a RBF-kernel function with constant σ and 3 inputs. The simulated output is a non-smooth function of the first (relevant) input because there are two irrelevant noise inputs with equal importance in the RBF-kernel. The true sinc function is depicted by the full line.

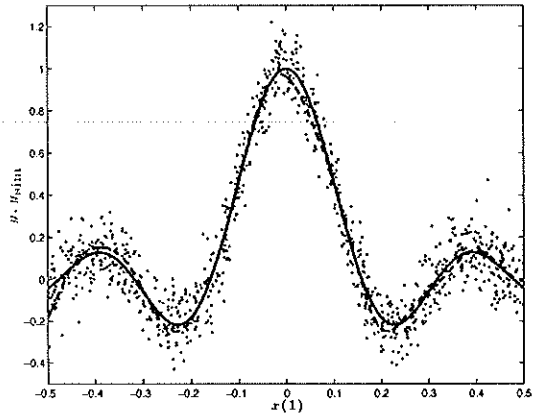

Figure 2: Test set data points $\{(x_i(1), y_i)\}_{i=1}^{1000}$ and predicted outputs $y_i = f(x_i(1))$ (dashed line) for a RBF-kernel function with one input $x(1)$. The true sinc function is depicted by the full line.

ance on the training set. The Nyström method was applied with $\tilde{n}_{\tilde{D}} = 50$, which corresponds to 25% of the original training set. We first estimated σ on level 3 assuming a constant σ for the three inputs. This yielded $\sigma = 1.12$, $\mu_{MP} = 2.78$ and $\zeta_{MP} = 54.93$ (with $u = [1;1;1]$). The test set MSE was 0.0479. Without the Nyström approximation we obtained $\sigma = 0.88$, $\mu_{MP} = 5.24$ and $\zeta_{MP} = 106.87$ and a test set MSE of 0.0169 [14]. We then optimized $\bar{u}$ on level 3 for all inputs $x(1:3)$. This yielded $\bar{u} = [2.9692; 0.1332; 0.4012]$, $\mu_{MP} = 5.23$, $\zeta_{MP} = 106.87$, with a test set MSE of 0.0121. (Without the Nyström method we obtained $\bar{u} = [2.3908; 0.0020; 0.0007]$, $\mu_{MP} = 4.26$, $\zeta_{MP} = 113.77$, and a test set MSE of 0.0102). The test set performance is much smaller than the previous model by reducing the relevance of the noise inputs and is now almost equal to the variance of the additive noise. Observe also that due to the approximation we now first remove the second input. Stepwise removing all irrelevant inputs we retain the first input $x(1)$ as relevant and obtain $\bar{u} = 2.9893$, $\zeta_{MP} = 109.7$, $\mu_{MP} = 6.2$ and a test set MSE of 0.0103. (Without approximation we obtained $\bar{u} = 1.9855$, $\mu_{MP} = 2.99$, $\zeta_{MP} = 109.05$ and a test MSE of 0.0102). The approximations on the test set is depicted in Figure 2.

We then further evaluated the performance of the Nyström method for different sample sizes $\tilde{n}_{\tilde{D}}$. Averaging over ten random samples, we obtained the average test set MSE and hyperparameter values reported in Table 1. The standard deviations on these values are denoted between parantheses. From this Table it is observed that the Nyström method is a useful tool for hyperparameter selection.

4.2 Performance on benchmark classification data sets

We evaluated the performance of Bayesian hyperparameter selection with the Nyström approximation on benchmark classification data sets. A description of these data sets can be found in [13]. For each randomization we first inferred γ and σ on level 2 and 3, then we used solved the linear system (3) for the given set of hyperparameters. The average test set classification results are reported in Table 2. From this Table it is seen that the Nyström approximation typically yields good performances on all obtained data sets, even when a small sampling size is chosen.

We also applied the Nyström method on the adult data set which has 45222 instances data points. We used 33000 training data points and 12222 for testing. Applying the Nyström method with $\tilde{n}_{\tilde{D}} = 100$ we obtained the kernel parameter σ of the RBF-kernel from a grid $\sigma = [2; 4; 6; 8; 10; 12]$ of candidate kernel parameters. This yielded $\sigma = 6$ and $\gamma = 2.440$. We then estimate the LS-SVM using the large scale method [10] in order to solve the linear system (3). Basically this methods involves the solution of two linear systems of the size $n_D \times n_D$ with positive-definite system matrix using the Hestenes-Stiefel conjugate gradient algorithm. Evaluating the performance of the resulting LS-SVM on the test set this yields a test set performance of 84.5%. This performance is in line with a 10-fold cross-validation hyperparameter selection procedure which

cra	Acc.(TS)	$\log\sigma$	$\log\gamma$
100%	95.5	1.25	5.79
75%	95.6 (0.47)	1.15 (0.10)	5.65 (0.28)
50%	95.6 (0.63)	1.26 (0.09)	6.05 (0.22)
25%	95.5 (0.00)	1.60 (0.03)	4.99 (0.50)
10%	95.5 (0.00)	1.57 (0.11)	4.77 (0.32)
rsy	Acc.(TS)	$\log\sigma$	$\log\gamma$
100%	90.6	0.26	0.47
75%	90.6 (0.04)	0.26 (0.00)	0.50 (0.03)
50%	90.4 (0.04)	0.69 (0.00)	0.49 (0.03)
25%	90.5 (0.06)	0.19 (0.05)	0.55 (0.03)
10%	90.6 (0.08)	0.20 (0.07)	0.89 (0.17)
hea	Acc.(TS)	$\log\sigma$	$\log\gamma$
100%	85.6	0.61	2.28
75%	85.6 (0.00)	2.23 (0.03)	0.59 (0.02)
50%	85.6 (0.00)	2.26 (0.07)	0.60 (0.02)
25%	85.6 (0.00)	2.25 (0.07)	0.62 (0.01)
10%	85.6 (0.00)	2.22 (0.06)	0.64 (0.02)

Table 2: Empirical results on 5 classification data sets.

yielded $\gamma = 10$ and $\sigma = 10$ and a test set performance of 84.4%.

5 Conclusion

Bayesian inference is a powerful design method for MLPs and kernel based techniques like SVMs and LS-SVMs. While the SVM and LS-SVM involve convex optimization problems, the memory and computational complexity grow with the number of training data points. This is basically because one needs to solve linear systems and the eigenvalue decomposition of the Hessian. In this paper, we used the sampling idea of the Nyström method in order to compute approximate expressions on the second and third level of inference. Empirical results on the input selection problem and on benchmark data sets indicate that the Nyström method is a useful tool for expanding the Bayesian LS-SVM framework to large data sets.

Acknowledgments

This work was supported by grants and projects from the Flemish Gov.: (Research council KULeuven: Grants, GOA-Mefisto 666; FWO-Vlaanderen: Grants, res. proj. G.0240.99, G.0256.97, G.0407.02 and comm. (ICCoS and ANMMM); AWI: Bil. Int. Coll.; IWT: STWW Eureka SINOPSYS, IMPACT); from the Belgian Fed. Gov. (Interuniv. Attr. Poles: IUAP-IV/02, IV/24; Program Dur. Dev.); from the Eur. Comm.: (TMR Netw. (Alapedes, Niconet); Science: ERNSI). J.A.K. Suykens is a Postdoctoral Researcher with the Fund for Scientific Research-Flanders (FWO-Vlaanderen). The scientific responsibility is assumed by its authors.

References

[1] Bishop, C.M. *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.

[2] Baker, C.T.H. *The numerical treatment of integral equations*. Oxford: Clarendon Press, 1977.

[3] Baudat, G., & Anouar, F. "Generalized Discriminant Analysis Using a Kernel Approach," *Neural Computation*, 12, 2385-2404, 2000.

[4] Cristianini, N., & Shawe-Taylor, J. *An Introduction to Support Vector Machines*. Cambridge University Press, 2000.

[5] Evgeniou, T., Pontil, M., & Poggio, T. "Regularization Networks and Support Vector Machines," *Advances in Computational Mathematics*, vol. 13, 1-50, 2001.

[6] Kwok, J.T. "Moderating the outputs of support vector machine classifiers," *IEEE Trans. on Neural Networks, 10*, 1018-1031, 1999.

[7] MacKay, D.J.C. "Probable Networks and Plausible Predictions - A Review of Practical Bayesian Methods for Supervised Neural Networks," *Network: Computation in Neural Systems, 6*, 469-505, 1995.

[8] Poggio, T. & Girosi, F. "Networks for Approximation and Learning," *Proceedings of the IEEE*, 78(9), 1481-1497, 1990.

[9] Suykens, J.A.K. & Vandewalle, J. "Least squares support vector machine classifiers," *Neural Processing Letters, 9*, 293-300, 1999.

[10] Suykens, J.A.K. "Least Squares Support Vector Machines for Classification and Nonlinear Modelling," *Neural Network World, 10*, 29-48, 2000.

[11] Suykens, J.A.K., De Brabanter, J., Lukas, L., Vandewalle, J., "Weighted Least Squares Support Vector Machines: Robustness and Sparse Approximation," *Neurocomputing*, in press.

[12] Van Gestel T., Suykens J.A.K., Baestaens D.-E., Lambrechts A., Lanckriet G., Vandaele B., De Moor B., Vandewalle J. "Predicting Financial Time Series using Least Squares Support Vector Machines within the Evidence Framework," *IEEE Transactions on Neural Networks*, vol. 12(4), pp. 809-821, 2001.

[13] Van Gestel, T., Suykens J.A.K., Lanckriet, G., Lambrechts, A., De Moor, B. & Vandewalle, J. "Bayesian Framework for Least Squares Support Vector Machine Classifiers, Gaussian Processes and Kernel Fisher Discriminant Analysis," *Neural Computation*, in press.

[14] Van Gestel, T., Suykens, J.A.K., De Moor, B. Vandewalle, J., "Automatic Relevance Determination for Least Squares Support Vector Machine regression", In: *Proc. International Joint INNS-IEEE Conference on Neural Networks (IJCNN 2001)*, Washington DC, July 14-19, 2001.

[15] Vapnik, V. *Statistical learning theory*, John Wiley, New-York, 1998.

[16] Williams, C.K.I. "Prediction with Gaussian Processes: from Linear Regression to Linear Prediction and Beyond," In M.I. Jordan (Ed.), *Learning and Inference in Graphical Models*. Kluwer Academic Press, 1998.

[17] Williams, C.K.I., & Seeger, M. "Using the Nyström Method to Speed Up Kernel Machines," In T. Leen, T. Dietterich, V. Tresp (Eds.), *Proc. NIPS'2000*, vol. 13, MIT press.