Large Scale Bayesian Kernel Learning with the Subspace EM Algorithm

Joaquin Quiñonero-Candela
Ole Winther
Lars Kai Hansen
Institute of Mathematical Modelling
Technical University of Denmark
DK-2800, Kongens Lyngby, Denmark

Editor: unknown

Abstract

In this paper we consider Bayesian kernel learning and introduce a new way of using the expectation-maximization (EM) algorithm which amounts to taking the computationally most expensive E-step only in a subspace of the hidden representation. Using the subspace EM algorithm iteratively by adding examples while using a kernel that leads to a sparse representation (Tipping, 2001), i.e. effectively removing dimensions from the subspace, leads to an algorithm that is very computationally effective. In this way, a sparse Bayesian kernel algorithm is available that can handle the same training set sizes as for example support vector machines (SVMs). An active learning strategy based upon the same concept is also introduced. We demonstrate the usefulness of the algorithm on two large regression data sets, of order $10^5$ examples.

keywords: sparseness, kernel regression, Gaussian processes, expectation-maximization (EM) algorithm, active learning, linear models

1. Introduction

Kernel methods are among the current most popular approaches to learning. They are basically two views, the Bayesian based upon a Gaussian process (GP) prior over functions (Rasmussen, 1996, Williams and Rasmussen, 1996, Gibbs, 1997, MacKay, 1997) and support vector machines (SVMs) that aim at maximizing margins (Cortes and Vapnik, 1995, Vapnik, 1999, Schölkopf and Smola, 2002).

The Bayesian approach has the advantage over SVMs that predictions are probabilistic and the training is readily defined in probabilistic terms. The solution of the SVM optimization problem is typically sparse, i.e. only a small fraction of the training examples span the solution. Using this property, SVM optimization schemes have been developed that make application to training sets of size up to $N = 10^5$ possible (Platt, 1998, Flake and Lawrence, 2002, Schölkopf and Smola, 2002).

The Bayesian GP approach does not lead to sparse solutions and is thus not directly applicable to large data sets. Therefore a number of authors have considered schemes to overcome this problem (Csató and Opper, 2001, Smola and Bartlett, 2001, Williams and Seeger, 2001, Tipping, 2001). Of special interest here will be the Relevance Vector Machine
(RVM) introduced by Tipping (2001) which is a GP with a generalized type of kernel. Maximum likelihood II (ML-II) training with this type of kernel leads to sparse solutions.

The iterative expectation-maximization (EM) algorithm (Dempster et al., 1977) provides a framework for doing ML-II training that guarantees convergence to a local maximum of the likelihood of the model parameters. For Bayesian kernel methods, the E-step, where a trial distribution is estimated based upon the current value of the model parameters, typically scales as \( N^3 \), where \( N \) is the number of training examples. This can be understood from the fact that there is one random variable associated with each training point and training and making predictions requires inverting their covariance matrix. Unfortunately, this is also true for RVMs because initially all examples are included. Only eventually, (during the M-steps) the sparse representation kernels will effectively remove examples that are not necessary for the mapping.

The main contribution of this paper is to show that the EM algorithm can be used in an incremental fashion where the E-step is only performed in a subspace of the random variables and examples are added one at a time. The complexity of the E-step will only scale with \( M \), where \( M \) is the size of the subset. We call this approach the Subspace EM (SSEM). Used in conjunction with a finite expansion kernel (that removes examples and thus dimensions from the subspace), the subspace EM algorithm provides a very powerful approach that can handle very large data sets. Furthermore, the same kind of reasoning can be used to derive active learning strategies.

Even though we work in subspaces, it can still be proved that each SSEM step increases the likelihood, and that the algorithm converges to a local maximum of the likelihood. Though empirical results show that the same solution as with the classical (batch) EM algorithm is not obtained, the values of the likelihood and test error reached are very close for both methods, implying a large robustness of the SSEM method. It must be noted that SSEM can also be applied directly to the training of any model classes that rely on a linear superposition of general basis functions.

Reducing the computational burden of Bayesian kernel learning is a subject of current interest. This can be achieved by numerical approximations to matrix inversion (Williams and Seeger, 2001) and suboptimal projections onto finite subspaces of basis functions without having an explicit parametric form of such basis functions (Smola and Bartlett, 2001, Csató and Opper, 2001). Using mixtures of GPs (Tresp, 2000, Rasmussen and Ghahramani, 2002) to make the kernel function input dependent is also a promising technique because it can split the computation of the covariance matrix into subspaces to reduce the complexity. This approach also enhances the flexibility of modelling by relaxing the crude assumption of one set of kernel parameters (length scales and noise variances) being valid all over the input space.

The known bounds on the a priori generalization capability for kernel methods are based either on the VC-dimension or on margins (Vapnik, 1998, Schölkopf and Smola, 2002). However, recent results (Ben-David et al., 2002) show that insofar there exist bounds that can give practical a priori generalization guarantees, these cannot be based upon VC-dimension or margins. Sparseness could be an alternative (Ben-David, 2001). This indicates that the use of a sparse representation could be desirable also from a generalization point of view. More generally it is worth noting that GPs, SVMs and regularization networks share the concept of regularization via reproducing kernel Hilbert spaces (Girosi, 1998), which
means the smoothness properties of the class of functions spanned by the model can be directly specified.

The rest of the paper is organized as follows: In section 2 we present extended linear models in a Bayesian perspective and the standard EM approach to train them. In section 3, a variation of the EM algorithm, that we call the Subspace EM (SSEM) is presented as an alternative to the standard EM for training linear models. Section 4 gives results for the Mackey-Glass time-series and the “abalone” data set and we conclude in section 5.

2. Extended linear models

An extended linear model is build by transforming the input space by an arbitrary set of basis functions \( \phi_j : \mathbb{R}^D \to \mathbb{R} \) that performs a non-linear transformation of the \( D \)-dimensional input space. A linear model is applied to the transformed space whose dimension is equal to the number of basis functions:

\[
y(x_i) = \sum_{j=1}^{M} \omega_j \phi_j(x_i) = \Phi(x_i) \cdot \omega
\]

where \( \Phi(x_i) \equiv [\phi_1(x_i), \ldots, \phi_M(x_i)] \) denotes the \( i \)th row of the design matrix \( \Phi \) and \( \omega = (\omega_1, \ldots, \omega_N)^T \) is the weights vector.

There are no restrictions in the choice of the basis functions. While it is possible to optimize the parameters of the basis functions for the problem at hand (Tipping, 2001, Quiñonero-Candela and Hansen, 2002), we will in this paper assume that they are given. The only parameters of the model are thus the weights \( \omega \).

In this paper we will restrict ourselves to the simplest possible model: regression with additive Gaussian noise. In a forthcoming paper we will generalize to more complicated scenarios, i.e. classification and regression with non-Gaussian noise. Given a set of input-target training pairs \( \{x_i, t_i\}_{i=1}^{N} \) and assuming that the observations are independent and contaminated with Gaussian noise of variance \( \sigma^2 \), the likelihood of the training set can be written as

\[
p(t|\omega, \sigma^2) = (2\pi\sigma^2)^{-N/2} \exp\left(-\frac{1}{2\sigma^2} \|t - \Phi \omega\|^2\right)
\]

where \( t = (t_1, \ldots, t_N)^T \) is the target vector.

Attempting to maximize the likelihood of the training data (2) leads to over-fitting except for very particular choices of the basis functions, and in most cases it does not yield a sparse model. Some form of regularization is needed. Rather than making point predictions of the optimal values of the weights, in Bayesian learning, a prior distribution over the over weights is introduced. In general, the implied prior over functions is a very complicated distribution. However, choosing a Gaussian prior on the weights the prior over functions also becomes Gaussian, i.e. a Gaussian process. For the specific choice of a factorized distribution with variance \( \alpha_j^{-1} \):

\[
p(\omega_j | \alpha_j) = \sqrt{\frac{\alpha_j}{2\pi}} \exp\left(-\frac{1}{2} \alpha_j \omega_j^2\right)
\]
the prior over functions \( p(\mathbf{y}|\mathbf{\alpha}) = \mathcal{N}(0, \mathbf{\Phi} \mathbf{A}^{-1} \mathbf{\Phi}^T) \), i.e. a Gaussian process with covariance function given by

\[
\text{Cov}(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^{M} \frac{1}{\alpha_k} \phi_k(x_i) \phi_k(x_j) \tag{4}
\]

where \( \mathbf{\alpha} = (\alpha_0, \ldots, \alpha_N)^T \) and \( \mathbf{A} = \text{diag}(\alpha_0, \ldots, \alpha_N) \). We can now see how sparseness in terms of the basis vectors may arise: if \( \alpha_k^{-1} = 0 \) the \( k \)th basis vector \( \Phi_k \equiv [\phi_k(x_1), \ldots, \phi_k(x_N)]^T \), i.e. the \( k \)th column in the design matrix, will not contribute to the model. Associating a basis function with each input point may thus lead to a model with a sparse representations in the inputs, i.e. the solution is only spanned by a subset of all input points. This is exactly the idea behind the relevance vector machine. We will see in the following how this leads to a lower computational complexity.

The posterior distribution over the weights, obtained using Bayes rule, is a Gaussian distribution

\[
p(\mathbf{w}|\mathbf{t}, \mathbf{\alpha}, \sigma^2) = \frac{p(\mathbf{t}|\mathbf{w}, \sigma^2) p(\mathbf{w}|\mathbf{\alpha})}{p(\mathbf{t}|\mathbf{\alpha}, \sigma^2)} = \mathcal{N}(\mathbf{t}|\mathbf{\mu}, \mathbf{\Sigma}) \tag{5}
\]

where \( \mathcal{N}(\mathbf{t}|\mathbf{\mu}, \mathbf{\Sigma}) \) is a Gaussian distribution with mean \( \mathbf{\mu} \) and covariance \( \mathbf{\Sigma} \) evaluated at \( \mathbf{t} \). The mean and covariance are given by

\[
\mathbf{\mu} = \sigma^{-2} \mathbf{\Sigma} \mathbf{\Phi}^T \mathbf{t} \tag{6}
\]

\[
\mathbf{\Sigma} = (\sigma^{-2} \mathbf{\Phi}^T \mathbf{\Phi} + \mathbf{A})^{-1} \tag{7}
\]

The maximum a posteriori (MAP) estimate of the weights is given by

\[
\hat{\mathbf{w}} = (\mathbf{\Phi}^T \mathbf{\Phi} + \sigma^{-2} \mathbf{A})^{-1} \mathbf{\Phi}^T \mathbf{t}.
\]

This is a version of the normal equations that includes a regularization term determined by the hyperparameters \{\( \alpha_j \)\} and by the noise \( \sigma^2 \).

The likelihood distribution over the training targets \( \mathcal{L} \) can be “marginalized" with respect to the weights to obtain the marginal likelihood, which is also a Gaussian distribution

\[
p(t|\mathbf{\alpha}, \sigma^2) = \int p(t|\mathbf{w}, \sigma^2) p(\mathbf{w}|\mathbf{\alpha}) \, d\mathbf{w} = \mathcal{N}(\mathbf{t}|0, \mathbf{C}) \tag{8}
\]

with covariance

\[
\mathbf{C} = \sigma^2 \mathbf{I} + \mathbf{\Phi} \mathbf{A}^{-1} \mathbf{\Phi}^T \tag{9}
\]

The uncertainty about the optimal value of the weights captured by the posterior distribution \( \mathcal{P} \) can be used to build probabilistic predictions. Given a new input \( \mathbf{x}_s \), the model gives a Gaussian predictive distribution of the corresponding target \( t_s \)

\[
p(t_s|\mathbf{x}_s, \mathbf{\hat{\alpha}}, \sigma^2) = \int p(t_s|\mathbf{x}_s, \mathbf{w}, \sigma^2) p(\mathbf{w}|\mathbf{t}, \mathbf{\hat{\alpha}}, \sigma^2) \, d\mathbf{w} = \mathcal{N}(t_s|y_s, \sigma^2_s) \tag{10}
\]

where

\[
y_s = \mathbf{\Phi}^T \mathbf{\hat{\alpha}} \cdot \mathbf{\mu} \tag{11}
\]

\[
\sigma^2_s = \sigma^2 + \mathbf{\Phi}^T \mathbf{\hat{\alpha}} \cdot \mathbf{\Sigma} \cdot \mathbf{\Phi} \mathbf{\hat{\alpha}}^T \tag{12}
\]
Figure 1: Relevance Vectors chosen from the training set to build a generalized linear model for prediction.

For regression it is natural to use $y_s$ and $\sigma_s^2$ as the prediction and the error bar on the prediction respectively. The computational complexity of making predictions is thus $O(M^3 + M^2P)$, where $M$ is the number of selected basis functions (RVs) and $P$ is the number of predictions.

It has been shown by MacKay (1997) that the same predictive distribution as in (10) is obtained by conditioning the joint distribution of training examples and target associated to a new input on the training examples, which is the standard procedure in Gaussian processes. The error bars (12) are strongly conditioned on the form of the basis functions used. For example, the use of local basis functions results in small error bars if the new input happens to have no significant output from any of the basis functions, i.e. if the new point is far away from the training data.

2.1 Training with the EM algorithm

In this section we derive the EM algorithm to provide the background for the SSEM algorithm that we describe in section 3. Estimating the model parameters from the marginal likelihood (8) is known as empirical Bayes, or as the maximum-likelihood II approach, and does not constitute a fully Bayesian approach in the strict sense. A fully Bayesian approach would require a hierarchical definition of priors and marginalization over all parameters. However, empirical studies for Gaussian processes have shown that both approaches give
similar results. For RVMs in particular, this has been shown by Tipping (2001). The maximization of the marginal likelihood with respect to \( \{\alpha_j\} \) and \( \sigma^2 \) cannot be done in a closed form, and requires an iterative approach, like the EM algorithm. The logarithm of the marginal likelihood (8) is our objective function

\[
\mathcal{L}(\sigma^2, \alpha) = \log p(\mathbf{t} | \mathbf{\alpha}, \sigma^2) = \log \int p(\mathbf{t}, \mathbf{\omega} | \mathbf{\alpha}, \sigma^2) \, d\mathbf{\omega}
\]  

(13)

Using Jensen’s inequality with an arbitrary distribution \( q(\mathbf{\omega}) \) we obtain a lower bound for (13)

\[
\log \int q(\mathbf{\omega}) p(\mathbf{t}, \mathbf{\omega} | \mathbf{\alpha}, \sigma^2) \, d\mathbf{\omega} \geq \int q(\mathbf{\omega}) \log \frac{p(\mathbf{t}, \mathbf{\omega} | \mathbf{\alpha}, \sigma^2)}{q(\mathbf{\omega})} \, d\mathbf{\omega} \equiv \mathcal{F}(q, \sigma^2, \mathbf{\alpha})
\]  

(14)

The E-step corresponds to maximizing \( \mathcal{F}(q, \sigma^2, \mathbf{\alpha}) \) with respect to \( q(\mathbf{\omega}) \), which is equivalent to minimizing the Kullback-Liebler (KL) divergence between \( q(\mathbf{\omega}) \) and the posterior distribution of the weights (5). The minimization can be made exact by taking \( q(\mathbf{\omega}) = \) the posterior, therefore the E-step amounts to estimating (6) and (7) for \( \mathbf{\alpha} \) and \( \sigma^2 \) given. The M-step corresponds to maximizing \( \mathcal{F}(q, \sigma^2, \mathbf{\alpha}) \) with respect to \( \sigma^2 \) and \( \mathbf{\alpha} \) with a fixed \( q(\mathbf{\omega}) \), which gives the following update rules:

\[
\alpha_j^{\text{new}} = \frac{1}{(\omega_j^2)^r \sigma_j^2} \frac{1}{\frac{1}{\mu_j^2} + \Sigma_{jj}}
\]  

(15)

and

\[
(\sigma^2)^{\text{new}} = \frac{||\mathbf{t} - \Phi \mathbf{\mu}||^2 + (\sigma^2)^{\text{old}} \Sigma_{jj} \gamma_j}{N}
\]  

(16)

where the quantity \( \gamma_j \equiv 1 - \alpha_j \Sigma_{jj} \) is a measure of how “well-determined” each weight \( \omega_j \) is by the data (MacKay, 1992, Tipping, 2001). One can obtain a different update rule that gives faster convergence. Although it is suboptimal in the EM sense, we have never observed it decrease the lower bound on the marginal log-likelihood. The rule, derived by Tipping (2001), is obtained by differentiation of (8) and by an arbitrary choice of independent terms as is done by MacKay (1992). It makes use of the terms \( \{\gamma_j\} \):

\[
\alpha_j^{\text{new}} = \frac{\gamma_j}{\mu_j^2}
\]  

(17)

\[
(\sigma^2)^{\text{new}} = \frac{||\mathbf{t} - \Phi \mathbf{\mu}||^2}{N - \Sigma_{jj} \gamma_j}
\]  

(18)

As the optimization progresses, many of the hyperparameters \( \alpha_j \) happen to grow to infinity, which effectively deletes the corresponding weight and basis function, since the prior on that weight tends to become a delta function centered on zero. Recently, Pauł and Tipping (2002) have shown that fixing all but one of the hyperparameters, the remaining ones can either have finite value as a maximizer of the marginal likelihood, or infinity. Figure 1 illustrates the resulting sparseness in terms of the targets associated to the chosen relevance vectors for prediction of the Mackey-Glass time series. The details are explained in section 4.

Appealing though it might seem, this learning approach limits the maximum size of the training set, since covariance of the weights (7) needs to be computed for each iteration,
1. Set \( \alpha_j = L \) for all \( j \). (\( L \) is a very large number) Set \( n = 1 \).
2. Update the set of active indexes \( R_n \).
3. Perform an E-step in subspace \( \omega_j \) such that \( j \in R_n \).
4. Perform the M-step for all \( \alpha_j \) such that \( j \in R_n \).
5. If all basis functions have been visited, end, else go to 2.

Figure 2: Schematics of the SSEM algorithm.

which implies inverting an \( N \times N \) matrix, \( N \) being the number of weights which is equal to the number of training patterns for RVMs. The trick used by Tipping in (Tipping, 2000) consisting in irreversibly deleting the basis functions as their corresponding \( \alpha \) reaches infinity does not completely solve the problem, since the first iterations are still \( O(N^2) \) and the process only speeds up when enough basis functions have been thrown out so that \( M \ll N \), \( M \) being the number of relevance vectors. Another approach is needed to cope with large training sets, in this case larger than 2,000 to 3,000 training patterns.

In the next section we propose a modified version of the EM algorithm for training linear models as an alternative to the batch training algorithm, where the number of basis functions is grown from scratch, rather than decreased from the full set.

3. Subspace EM

We introduce a new approach to the training of the RVM, that can easily be generalized to the training of other model classes that rely on a linear superposition of completely general basis functions. Instead of starting with a full model, i.e. where all the basis functions are present with finite \( \alpha \) values, we start with a fully pruned model with all \( \alpha_j \) set to infinity.

Effectively, we start with no model. The model is grown by iteratively including some \( \alpha_j \) previously set to infinity to the active set of \( \alpha \)'s. The active set at iteration \( n \), \( R_n \), contains the indices of the basis vectors with \( \alpha \) less than infinity. This procedure is equivalent to iteratively presenting a new basis function to the model, and letting it adjust its parameters to decide whether it incorporates the new basis function and whether it prunes some older basis function in the light of the newly acquired basis function.

As we already mentioned, we call the resulting modification of the EM algorithm Subspace EM (SSEM). In SSEM, the \( n \)-th EM step is taken only in a subspace of the parameters space, given by those parameters whose index is contained in the active set \( R_n \)

\[
R_1 = 1 \\
R_n = \{ i \mid i \in R_{n-1} \land \alpha_i \leq L \} \cup \{ n \}
\]

(19)

where \( L \) is a finite very large number arbitrarily defined. Observe that \( R_n \) contains at most one more element (index) than \( R_{n-1} \). If some of the \( \alpha \)'s indexed by \( R_{n-1} \) happen to reach \( L \) at the \( n \)-th step, \( R_n \) can contain less elements than \( R_{n-1} \). In figure 2 we give a schematic description of the SSEM algorithm.

At step \( n \), we want to maximize the marginal likelihood (8) with respect to the noise \( \sigma^2 \) and to the hyperparameters \( \{ \alpha_j \} \) such that \( j \in R_n \)

\[
\mathcal{L}^n(\alpha, \sigma^2) = \log p(\mathbf{t} | \alpha, \sigma^2, R_n)
\]

(20)
the remaining $\alpha_j$ with index not included in $R_n$ are treated as constants with infinite value.
The E-step corresponds to minimizing the $KL$ divergence between the posterior (5) and
an arbitrary probability distribution $q(\omega)$. The $KL$ divergence can be made zero (absolute
minimum) by simply taking $q(\omega)$ as the posterior distribution only of those weights whose
indexes belong to the active set $R_n$: 

$$
\mu^n = \sigma^{-2}(\Sigma^n (\Phi^n)^T t \\
\Sigma^n = (\sigma^{-2}(\Phi^n)^T \Phi^n + A^n)^{-1}
$$

where $A^n = \text{diag}(\alpha_j)$ and $\Phi^n = [\phi_j]$ for all $j \in R_n$. The M-step is as given by (16) and
(17), replacing $\mu$ and $\Sigma$ by $\mu^n$ and $\Sigma^n$ respectively, and working only with the indexes $j$
contained in $R_n$. The computational complexity of the SSEM algorithm is bounded from
above by $O(M^3)$, where $M$ is the number of relevance vectors selected by the model.

It must be noted that since the initial value of $\alpha_j$ is infinity for all $j$, the E-step yields
always an equality between the log marginal likelihood and its lower bound. At any step
$n$, the posterior can be exactly projected onto the space spanned by the weights $\omega_j$ such
that $j \in R_n$, because the $\alpha_k = \infty$ for all $k$ not in $R_n$. Hence SSEM never decreases
the objective function:

$$
L^{n+1}(\alpha, \sigma^2) \geq L^n(\alpha, \sigma^2)
$$

Figure 3 illustrates the convergence process of the SSEM algorithm compared to that of
the EM algorithm in an example consisting in predicting the Mackey-Glass time series. For
better visualization, instead of $\alpha_j$ we have represented the value $\log(1 + \alpha_j^{-1})$.

Once all the examples have been visited, we switch to the batch EM algorithm on the active set
until some convergence criteria has been satisfied, for example until the relative increase in the likelihood is smaller than a certain threshold.

### 3.1 Active learning

The order in which the basis functions are presented to the SSEM algorithm has a strong
influence on the local maximum of the likelihood that is reached during the training process.
For the time series prediction problem that constitutes our first example in this paper,
presenting the basis functions in a random sequence yields much better results than following
the temporal order. Here we will present two active learning strategies, both aiming
at choosing as new basis function the one that maximizes the increase in marginal log-
likelihood. The first approximate approach is based on maximizing the log-likelihood with
respect to a given $\alpha$ while keeping the others fixed, while the second approach is based on
performing an tentative maximization of the log-likelihood with respect to an augmented
set of $\alpha$’s.

The first approximate approach goes as follows: at the $n$-th step of the SSEM algorithm,
the log-likelihood of the set of $\alpha$’s that belong to the active set $R_n$ is given by (20). Introducing
a new basis function $\phi_i$ with its associated $\alpha_i$ to the model changes the log-likelihood
to:

$$
L^{n(i)}(\alpha, \alpha_i, \sigma^2) = L^n(\alpha, \sigma^2) + \frac{1}{2} \left[ \log \frac{\alpha_i}{\alpha_i + S_i} + \frac{Q_i^2}{\alpha_i + S_i} \right]
$$
Figure 3: Training on 400 samples of the Mackey-Glass time series, testing on 2000 cases. Top: values taken by the hyperparameters as the optimization progresses for the SSEM (left) and the EM (right) algorithms. Bottom: log marginal likelihood as a function of the elapsed CPU time (left) and corresponding number of relevance vectors (right) for both SSEM and EM.

where the contribution of $\alpha_i$ has been isolated in the second term (Faul and Tipping, 2002). The gradient of the log-likelihood (24) with respect to the new $\alpha_i$ can be computed as:

$$\frac{\partial L^n[i]}{\partial \alpha_i} = \frac{\partial l^n(\alpha_i)}{\partial \alpha_i} = \frac{\alpha_i^{-1} S_i^2 - (Q_i^2 - S_i)}{2(\alpha_i + S_i)^2}$$

(25)

where we have used the notation introduced by Faul and Tipping (2002) for simplification:

$$Q_i = \Phi_i^T C^n^{-1} t, \quad S_i = \Phi_i^T C^n^{-1} \Phi_i.$$

(26)
From (25) maximizers of the log-likelihood as a function of \( \alpha_i \) points can occur both for \( \alpha_i = \infty \) and for:

\[
\alpha_i = \frac{S_i^2}{Q_i^2 - S_i}.
\]

(27)

Faul and Tipping (2002) have shown that the stationary point is only finite, given by (27) if \( Q_i^2 - S_i > 0 \). If \( Q_i^2 - S_i \leq 0 \), introducing the new basis function with the remaining \( \alpha \)'s fixed will not increase the log-likelihood. For the finite case, the maximizer given by (27) can be introduced in the expression (24) to obtain the resulting improvement in the log-likelihood:

\[
\mathcal{L}^n(\theta, \alpha_i, \sigma^2) = \mathcal{L}^n(\alpha, \sigma^2) + \frac{1}{2} \left[ \log \frac{S_i}{Q_i^2} + \frac{Q_i^2 - S_i}{S_i} \right]
\]

(28)

The first active learning criterion we consider selects from a set of \( P \) candidates the basis function that gives the largest increase in the log-likelihood according to equation (28). The computational cost of this method is \( \mathcal{O}(PM^2) \). Using this method for training sets of diverse sizes, we systematically converge to a lower value of the log-likelihood than when we present the basis functions at random. We have also found out that the method fails to predict which candidate basis function will give a higher log-likelihood after including it and performing one SSEM step. This is because we assume that the \( \alpha \)'s in the active set remain fixed, while in the SSEM the \( \alpha \)'s in the active set are jointly modified together with the \( \alpha_i \) of the new basis function to increase the log-likelihood. This active learning criterion can still be interesting for a growth method based on iteratively maximizing the log-likelihood as a function of a single \( \alpha \), as suggested in (Faul and Tipping, 2002). It is unfortunately not useful for a growth method based on the SSEM algorithm.

The second method we propose consists simply in performing a tentative SSEM step for all the basis functions in the set of candidates, computing the resulting value of the
log-likelihood and selecting the basis function for which it is the highest. At step $n+1$ of the SSEM, for the candidate basis function $\Phi_i = [\phi_i(x_1), \ldots, \phi_i(x_N)]^T$ the E-step requires computing the new covariance of the weights:

$$
\Sigma^{n(i)} = \left( (\Sigma^n)^{-1} \frac{(\Phi^n)^T \Phi_i}{\Phi_i^T \Phi_i} \sigma^{-2} \Phi_i^T \Phi_i + \alpha_i \right)^{-1}
$$

(29)

Making use of inversion by partitioning using the fact that $(\Sigma^n)^{-1}$ has already been computed, the cost of computing (29) can be reduced to $O(MN + M^2)$. Inversion by partitioning gives also a structure of $\Sigma^{n[i]}$ that allows computing $\mu^{n[i]}$ from the previously computed $\mu^n$ with a very cheap additional cost of $O(N)$. The tentative M-step is taken using equations (15) and (16). It returns a new set of values $\alpha_i^{new}$ for the $\alpha_i$'s in the active set $R_n$, a new value $\alpha_i^{new}$ for $\alpha_i$ and a new estimate $(\sigma^2_{n(i)})^{new}$ of the noise $\sigma^2$. With these new values we can compute the marginal log-likelihood obtained after the tentative inclusion of the $i$-th candidate basis function:

$$
\mathcal{L}^{n(i)}(\sigma^2, \alpha) = -\frac{1}{2} \left[ N \log(2\pi) + \log |C^{n(i)}| + \mathbf{t}^T (C^{n(i)})^{-1} \mathbf{t} \right]
$$

(30)

where $C^{n(i)}$ can be written as:

$$
C^{n(i)} = (\sigma^2_{n(i)})^{new} \cdot \mathbf{I} - (\alpha_i^{n(i)})^{new} \cdot \Phi_i^{n(i)} \cdot (\Sigma^{n(i)})^{new} \cdot \Phi_i^{n(i)}
$$

(31)

with the augmented tentative design matrix given by $\Phi_i^{n(i)} = [\Phi_i; \Phi_i]$ and the covariance matrix of the weights modified after the M-step given by:

$$(\Sigma^{n(i)})^{new} = (\sigma^{-2}_{n(i)})^{new} (\Phi_i^{n(i)})^T (\Phi_i^{n(i)}) + \text{diag}(\alpha_i^{new}, \alpha_i^{new})].$$

From (31) the complexity of computing the marginal log-likelihood (30) after the $i$-th tentative step can be reduced to $O(NM + M^3)$.

Performing a tentative EM step P candidate basis functions has a cost of $O(P(NM + M^3))$. Active learning in a subset of the training examples has been used for example by Smola and Bartlett (2001).

In figure 4 we show the results of apply partial active learning with a candidate set of 50 basis functions to the problem of predicting the Mackey-Glass time series. We use training set of 500 examples. It can be seen that the performance of the simple SSEM without active learning is almost as good as that of the SSEM with partial active learning and that of the batch EM algorithm. The difference is mainly that SSEM with random order presentation is much faster. For large training sets, partial active learning did not improve the performance with respect to random order presentation. For very small training sets it gave the same performance as the batch EM, at a comparable cost. We have in the experiments used the random order presentation, since it gives the best relation between cost and performance. We believe nevertheless that the contribution of active learning is problem-dependent, and depends as well on the type of basis functions used. Problems might exist for which active learning makes a significant difference with respect to the random order presentation.
4. Simulations

We select two very different data sets to compare the performance of the batch and growth approaches. The first set is a hard prediction problem, the Mackey-Glass chaotic time series, which is well-known for its strong non-linearity. Optimized non-linear models can have a prediction error which is three orders of magnitude lower than an optimized linear model (Svärter et al., 1993). In (Quiñonero-Candela and Hansen, 2002) we showed that the RVM has an order of magnitude superior performance than carefully tuned neural networks for time series prediction on the Mackey-Glass series.

The Mackey-Glass attractor is a non-linear chaotic system described by the following equation:

\[
\frac{dz(t)}{dt} = -bz(t) + a \frac{z(t-\tau)}{1 + z(t-\tau)^10}
\]  

(32)

where the constants are set to \(a = 0.2, b = 0.1\) and \(\tau = 17\). The series is resampled with period 1 according to standard practice. The inputs are formed by \(L = 16\) samples spaced 6 periods from each other \(x_k = [z(k-6), z(k-12), \ldots, z(k-6L)]\) and the targets are chosen to be \(t_k = z(k)\) to perform six steps ahead prediction (Svärter et al., 1993). We use Gaussian basis functions of fixed variance \(\nu^2 = 10\). The test set comprises 5804 examples. Figure 1 shows a piece of the chaotic time series and we have furthermore marked the training targets associated to the RV’s extracted from a training set composed by 500 samples of the Mackey-Glass chaotic time series.

The second data set we use is known as the ‘Abalone’ data set, available from the UCI Repository. The aim is to predict the age of certain marine animals from a series of measurements such as sex, length, diameter and diverse weight measures. The sex in the data set is encoded as \{Male, Female, Infant\}. We map it into \{(1,0,0),(0,1,0),(0,0,1)\} and we get 10-dimensional input vectors. From the fact that the physical measurements are done manually (see (Warwick J. Nash et al., 1994) for details) the data is very noisy, opposite to what is the case for the Mackey-Glass data. The test set comprises 1177 examples.

For both data sets we perform prediction experiments for different sizes of the training set. We perform in each case 10 repetitions with different partitions of the data sets into training and test. We compare the test error, the number of RVs selected and the computer time needed for the batch and the SSEM method. We present the results obtained with the growth method relative to the results obtained with the batch method in figure 5. As expected, the relative computer time of the growth method compared with the batch method decreases with size of the training set. For a few thousand examples the SSEM method is an order of magnitude faster than the batch method. The batch method proved only to be faster for 100 training examples in both experiments, and could not be used with data sets of thousands of examples on the machine on which we run the experiments because of its high memory requirements. This is the reason why we only ran the comparison for up to 2400 training example for the Mackey-Glass data set. For the Abalone data the test errors obtained with both methods are practically equivalent, while for Mackey-Glass slightly better values are obtained with the SSEM method.
Figure 5: Mean values over 10 repetitions of relative test error, number of RVs and computer time. Left, for the Mackey-Glass data, and right for the Abalone data.

5. Conclusion

We have presented a new approach to Bayesian training of linear models, based on a subspace extension of the EM algorithm that we call Subspace EM (SSEM). The new method iteratively builds models from a potentially big library of basis functions. We illustrate the method by applying it to training a relevance vector machine model for much larger training sets than what can be achieved by using the batch EM algorithm. We obtain similar performance with a computational complexity $O(M^3)$ instead of $O(N^3)$, where $N$ is the number of training examples and $M$ is the number of chosen basis functions (or relevance vectors (RV)). Presenting the candidate basis functions at random proves to be a valid approach. We have studied an active learning criterion based upon maximizing the increase in marginal log-likelihood, which did not give a significant improvement in performance with respect to random order presentation of the basis functions. We believe that devising more computationally efficient active learning approaches for the SSEM algorithm is an important open issue.

References


