The Generalized Bayesian Committee Machine

[Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD-2000]

Volker Tresp
Siemens AG
Corporate Technology
Otto-Hahn-Ring 6
81730 München, Germany
Volker.Tresp@mchp.siemens.de

ABSTRACT

In this paper we introduce the Generalized Bayesian Committee Machine (GBCM) for applications with large data sets. In particular, the GBCM can be used in the context of kernel based systems such as smoothing splines, kriging, regularization networks and Gaussian process regression which —for computational reasons— are otherwise limited to rather small data sets. The GBCM provides a novel and principled way of combining estimators trained for regression, classification, the prediction of counts, the prediction of lifetimes and other applications which can be derived from the exponential family of distributions. We describe an online version of the GBCM which only requires one pass through the data set and only requires the storage of a matrix of the dimension of the number of query or test points. After training, the prediction at additional test points only requires resources dependent on the number of query points but is independent of the number of training data. We confirm the good scaling behavior using real and experimental data sets.

Categories and Subject Descriptors

I.2.6 [**Computing Methodologies**]: Artificial Intelligence—*Learning*; H.2.8 [**Database Management**]: Database Applications—*Data Mining*

General Terms

Committee Machines, Combining Estimators, Data Mining, Gaussian Processes, Kernel-Based Systems, Support Vector Machines

1. INTRODUCTION

Estimators for KDD applications should be able to exploit the complete information which is contained in a large data set. If the estimators have a fixed architecture, the prediction performance will eventually be determined by the bias of the estimator and performance will not significantly improve with larger data sets. It is therefore desirable that the complexity of the architecture of the estimators grows appropriately with data size. Naturally, the resources in kernel based estimators such as regression splines, kriging, regularization networks and Gaussian process regression grow with data size but these approaches have a big disadvantage: all four approaches require the inversion of matrices of the dimension of the number of training data[1] which limits their applicability to data sets of not much more than 1000 samples. The goal of this paper is to extend the applicability of those methods to large data sets. Since all four methods can give equivalent solutions we will focus on only one of them, namely Gaussian processes regression, an approach recently introduced into the machine learning community [14, 18, 29, 11]. In our approach, the data set is divided up into M sets of approximately the same size and M models are developed on the individual data sets. The predictions of the individual models are combined using a weighting scheme which is derived from a Bayesian perspective. The computational complexity of this generalized Bayesian committee machine (GBCM) scales linearly in the number of training data. As the name implies, the GBCM is a generalization of the Bayesian committee machine. Whereas the latter is only applicable to regression [20], the former one can be applied to any probability density of the exponential family, and therefore is suitable among others for classification problems, prediction of counts and prediction of life times. We show experimentally that the GBCM efficiently exploits the information in large data sets and present online learning solutions which require only one pass through the data set. After learning the generalization to new query points only requires resources dependent on the number of query points but not the number of training data. The GBCM is an addition to the theory of combining estimators. Whereas the BCM for regression can be seen as a generalization to the variance-based combination of estimators [22], the GBCM provides a principled theory for combining classifiers and estimators for counts, lifetimes and other applications which can be derived from the exponential family of distributions.

[1] If the input dimension is larger than two.

In Section 2 we derive the theoretical foundation of the GBCM based on a Bayesian approach and in Section 3 we derive the BCM for regression. In Section 4 we apply the BCM to Gaussian process regression. We follow here the derivation in [20]. In Section 5 we introduce the GBCM for probability densities of the exponential family and use it in the context of Gaussian processes. In Section 6 we discuss design and computational issues. In Section 7 we present experimental results demonstrating the excellent performance of the GBCM approximation. Section 8 contains the conclusion of the paper.

2. THEORETICAL FOUNDATIONS

Let x be a vector of input variables and let y be the target variable. We assume that, given a function $f(x)$, the targets are (conditionally) independent with conditional density $P(y|f(x))$. Let $X^m = \{x_1^m, \ldots, x_K^m\}$ denote the input vectors of the training data set of size K (superscript m for measurement) and let $Y^m = (y_1^m \ldots y_K^m)'$ be the vector of the corresponding target measurements. Let $D = \{X^m, Y^m\}$ denote both inputs and targets.

Furthermore, let $X^q = \{x_1^q, \ldots, x_{N_Q}^q\}$ denote a set of N_Q query or test points (superscript q for query) and let $f^q = (f_1^q, \ldots, f_{N_Q}^q)$ be the vector of the corresponding unknown response variables.

We assume now a setting where instead of training one estimator using all the data we split up the data into M data sets $D = \{D^1, \ldots, D^M\}$ (of typically approximately same size) and train M estimators separately on each training data set. Gaussian process regression which will be introduced in the next sections is one example where this procedure is useful. Correspondingly, we partition inputs $X^m = \{X_1^m, \ldots, X_M^m\}$ and targets $Y^m = \{Y_1^m, \ldots, Y_M^m\}$. Let $\mathcal{D}^i = \{D^1, \ldots, D^i\}$ denote the data sets with indices smaller or equal to i with $i = 1, \ldots M$. In a Bayesian setting each estimator will provide us with an estimate of the posterior predictive probability density $P(f^q|D^i)$. As an example, for an estimator parameterized by a weight vector w, the posterior predictive probability density is [1]

$$P(f^q|D^i) = \int P(f^q|w)P(w|D^i)dw.$$

Then we have in general[2]

$$P(f^q|\mathcal{D}^{i-1}, D^i) \propto P(f^q)P(\mathcal{D}^{i-1}|f^q)P(D^i|\mathcal{D}^{i-1}, f^q).$$

Now we would like to approximate

$$P(D^i|\mathcal{D}^{i-1}, f^q) \approx P(D^i|f^q). \tag{1}$$

This is not true in general unless $f^q \equiv f$: only conditioned on the complete function f are targets independent. The approximation might still be reasonable when, first, N_Q is large — since then f^q determines f everywhere — and when, second, the correlation between the targets in $\mathcal{D}^{i-1}$ and D^i is small, for example if inputs in those sets are spatially separated from each other.

Using the approximation and applying Bayes' formula, we obtain

$$P(f^q|\mathcal{D}^{i-1}, D^i) \approx const \times \frac{P(f^q|\mathcal{D}^{i-1})P(f^q|D^i)}{P(f^q)} \tag{2}$$

such that we can achieve an approximate predictive density

$$\hat{P}(f^q|D) = const \times \frac{\prod_{i=1}^M P(f^q|D^i)}{P(f^q)^{M-1}} \tag{3}$$

where *const* is a normalizing constant. The posterior predictive probability densities are simply multiplied. Note that since we multiply posterior probability densities, we have to divide by the priors $M-1$ times. This general formula can be applied to the combination of any Bayesian estimator.

[2]In this paper we assume that inputs are given.

3. THE BCM

In case that the predictive densities $P(f^q|D^i)$ and the prior densities are Gaussian (or can be approximately reasonably well by Gaussians) Equation 2 takes on an especially simple form. Let's assume that the *a priori* predictive density at the N_Q query points is a Gaussian with zero mean and covariance Σ^{qq} and the posterior predictive density for each module is a Gaussian with mean $E(f^q|D^i)$ and covariance $cov(f^q|D^i)$. In this case we achieve [20] ($\hat{E}$ and $\widehat{cov}$ are caclulated w.r.t. the approximate density $\hat{P}$)

$$\hat{E}(f^q|D) = C^{-1} \sum_{i=1}^M cov(f^q|D^i)^{-1} E(f^q|D^i) \tag{4}$$

with

$$C = \widehat{cov}(f^q|D)^{-1} = -(M-1)(\Sigma^{qq})^{-1} + \sum_{i=1}^M cov(f^q|D^i)^{-1}. \tag{5}$$

We recognize that the prediction of each module i is weighted by the inverse covariance of its prediction. But note that we do not compute the covariance between the modules but the covariance between the N_Q query points! This means that predictive densities at all query points contribute to the prediction of the BCM at a given query point. This way of combining the predictions of the modules is named the Bayesian committee machine (BCM). Modules which are uncertain about their prediction are automatically weighted less than modules which are certain about their prediction.

4. GAUSSIAN PROCESS REGRESSION AND THE BCM

The most interesting application of the BCM is kernel based regression in the form of regularization networks, smoothing splines, kriging and Gaussian processes regression. The reason is that first, the degrees of freedom in such systems increase with the number of training data points such that it is not suitable to update posterior parameter probabilities and that second, the approaches require the inversion of matrices of the dimension of the number of data points which is clearly unsuitable for large data sets. Since it is well known that all four approaches can give equivalent solutions we will focus on only one of them, namely the Gaussian process regression framework. In the next section we briefly review Gaussian process regression and in the following section we discuss the BCM in conjunction with Gaussian process regression.

4.1 A Short Review of Gaussian Process Regression

In contrast to the usual parameterized approach to regression, in Gaussian process regression we specify the prior model directly in function space. In particular we assume that *a priori* f is Gaussian distributed (in fact it would be an infinite-dimensional Gaussian) with zero mean and a covariance $cov(f(x_1), f(x_2)) = \sigma_{x_1,x_2}$. We assume that we can only measure a noisy version of f

$$y(x) = f(x) + \epsilon(x)$$

where $\epsilon(x)$ is independent Gaussian distributed noise with zero mean and variance $\sigma_\psi^2(x)$.

Let $(\Sigma^{mm})_{ij} = \sigma_{x_i^m, x_j^m}$ be the covariance matrix of the measurements, $\Psi^{mm} = \sigma_\psi^2(x)I$ be the noise variance between the targets of the measurements, $(\Sigma^{qq})_{ij} = \sigma_{x_i^q, x_j^q}$ be the covariance matrix of the query points and $(\Sigma^{qm})_{ij} = \sigma_{x_i^q, x_j^m}$ be the covariance matrix between training data and query data. I is the K-dimensional unit matrix.

Under these assumptions the conditional density of the response variables at the query points is Gaussian distributed with mean

$$E(f^q|D) = \Sigma^{qm}(\Psi^{mm} + \Sigma^{mm})^{-1}y^m, \tag{6}$$

and covariance

$$cov(f^q|D) = \Sigma^{qq} - \Sigma^{qm}\left(\Psi^{mm} + \Sigma^{mm}\right)^{-1}(\Sigma^{qm})'. \tag{7}$$

Note, that for the $i-$th query point we obtain

$$E(f_i^q|D) = \sum_{j=1}^{K} \sigma_{x_i^q, x_j^m}\, v_j \tag{8}$$

where v_j is the $j-$th component of the vector

$$(\Psi^{mm} + \Sigma^{mm})^{-1}y^m.$$

Equation 8 describes the weighted superposition of kernel functions $b_j(x_i^q) = \sigma_{x_i^q, x_j^m}$ which are defined for each training data point and is equivalent to some solutions obtained for kriging, regularization networks and smoothing splines [25, 17, 29, 11, 8]. The experimenter has to specify the positive definite covariance matrix. A common choice is that

$$\sigma_{x_i,x_j} = A\exp(-1/(2\gamma^2)||x_i - x_j||^2) \tag{9}$$

with $A, \gamma > 0$ such that we obtain Gaussian basis functions although other positive definite covariance functions are also used.

Gaussian processes have been shown to be a very competitive approach for regression with small data sets [18].

4.2 The BCM for Gaussian Process Regression

Since the computational cost of the matrix inversion in Equation 6 scales as $\mathcal{O}(K^3)$ where K is the number of training data, the cost becomes prohibitive if K is large. The big advantage of the application of the BCM (Equation 4) to Gaussian process regression (Equations 6 and 7) is that now instead of having to invert a K-dimensional matrix we only have to invert approximately (K/M)-dimensional and N_Q-dimensional matrices. If we set $M = K/\alpha$ where α is a constant, the BCM scales linearly in K.

5. THE GBCM

In the following section we will discuss the general case that the distribution for the measurement process is from the exponential family of distributions. Readers who are interested only in classification —which is the most interesting special case from an application point of view— can directly go to sections 5.2 and 5.3.

5.1 The General Case

In the previous section it was assumed that both the prior distribution of f and the noise distribution $P(y_i|f_i)$ are Gaussian where for a generic x_i, we define $f_i \equiv f(x_i)$. In the following sections we will still assume a prior Gaussian distribution for f_i but we will allow for more general distributions for the measurement process $P(y_i|f_i)$. In particular we will assume that $P(y_i|f_i)$ is from the exponential family for which at x_i

$$P(y_i|f_i) = \exp\left(\frac{y_i\theta_i - b(\theta_i)}{\phi} - c(y_i, \phi)\right).$$

Here, θ_i is the natural parameter, ϕ is a scale parameter, and $b()$ and $c()$ are specific functions corresponding to the type of exponential family. Furthermore, we have the relationship

$$\mu_i = h(f_i) = E(y_i|x_i) = \frac{\partial b(\theta_i)}{\partial \theta_i}$$

where $h(.)$ is the link function. The variance of y_i is $\sigma_i^2 = \phi\frac{\partial^2 b(\theta_i)}{\partial\theta_i^2}$. By fixing the functions $b(.)$ and $c(.)$ we obtain particular distributions out of the exponential family such as the normal (Gaussian), Bernoulli, Poisson, Gamma and inverse Gaussian distributions which allow, among others, the modeling of classification problems, prediction of counts and prediction of life times. This generalization of the Gaussian model corresponds the transition from linear models to generalized linear models as described in McCullagh and Nelder [12] and Fahrmeir and Tutz [4].

Due to the fact that the conditional densities are non-Gaussian, the posterior distribution of f will also be non-Gaussian. We consider a Laplace approximation of the posterior probability density at the measurements points f^m i.e. a Gaussian approximation to the posterior density. The center and the covariance matrix of this Gaussian is calculated as the mode and the inverse Hessian of the cost function

$$cost(f^m) = -\sum_{i=1}^{K} \log P(y_i^m|f_i^m) + \frac{1}{2}(f^m)'(\Sigma^{mm})^{-1}f^m.$$

Based on a Fisher scoring approach it can be shown (see e.g. Fahrmeir and Tutz [4] and Appendix A) that the mode can be found by iterating the reweighted least squares equations

$$\hat{f}^{m,(k+1)} = \Sigma^{mm}(\Psi^{(k)} + \Sigma^{mm})^{-1}\tilde{y}^{(k)}.$$

Here,

$$\tilde{y}_i^{(k)} = \frac{y_i^m - h(f_i^{m,(k)})}{D_i^{(k)}} + f_i^{m,(k)}$$

where

$$\Psi^{(k)} = diag\left(\frac{\left(\sigma_i^{(k)}\right)^2}{\left(D_i^{(k)}\right)^2}\right)$$

and

$$D_i^{(k)} = \frac{\partial h\left(f_i^{m,(k)}\right)}{\partial f_i^{m,(k)}}.$$

After convergence (k=k'), an estimate of the covariance matrix is obtained by

$$\widehat{cov}(f^m|D) = \Sigma^{mm}(\Psi^{(k')} + \Sigma^{mm})^{-1}\Psi^{(k')}.$$

The mode and the covariance matrix at the query points can now be calculated as

$$\hat{E}(f^q) = \Sigma^{qm}(\Sigma^{mm})^{-1}\hat{f}^m = \Sigma^{qm}(\Psi^{(k')} + \Sigma^{mm})^{-1}\tilde{y}^{(k')}$$

and

$$\widehat{cov}(f^q|D) = \Sigma^{qq} - \Sigma^{qm}\ (\Psi^{(k')} + \Sigma^{mm})^{-1}\ (\Sigma^{qm})'.$$

The GBCM is obtained if these estimates are used in the formulas 4 and 5 to combine the estimates of f^q trained on different data sets.

5.2 Two-class Classification

A special case of great practical interest is the case that $y_i \in \{0, 1\}$ and that

$$P(y_i|f_i) = \mu_i^{y_i}(1-\mu_i)^{1-y_i}$$

is Bernoulli distributed and corresponds to the posterior probability for class one. If the natural link function (logistic distribution function)

$$\mu_i = h(f_i) = \frac{\exp f_i}{1+\exp f_i}$$

is used then $b(\theta_i) = \log(1+\exp(\theta_i))$, $\phi_i = 1$, $\theta_i = f_i$, and

$$\sigma_i^2 = \mu_i(1-\mu_i).$$

Here,

$$\Psi^{(k)} = diag\left(\mu_i^{(k)}\left(1-\mu_i^{(k)}\right)\right)^{-1},$$

and

$$D_i^{(k)} = \mu_i^{(k)}\left(1-\mu_i^{(k)}\right)$$

such that

$$\tilde{y}_i^{(k)} = \frac{y_i^m - \mu_i^{(k)}}{\mu_i^{(k)}\left(1-\mu_i^{(k)}\right)} + f_i^{m,(k)}.$$

The application of Gaussian processes to classification has been discussed in a Bayesian setting in a paper by Williams and Barber [28]. They also provide detailed derivations of the Fisher scoring algorithm and discuss the multiple-class case.

5.3 Multiple-class Classification

There are several ways of approaching the multiple-class classification case [4]. A straightforward generalization to multiple classes assumes C binary variables where $y_{i,c} \in \{0,1\}$ and $y_{i,c} = 1$ indicates that at input x_i class c is the correct class. Furthermore, we use C Gaussian processes where $f_{i,c}$ is the activation of the c-th Gaussian process at x_i and $\forall i$

$$P(y_{i,c} = 1|\{f_{i,c}\}_{c=1}^C) = \mu_{i,c} = \frac{\exp f_{i,c}}{\sum_{c'=1}^C exp f_{i,c'}}$$

is the probability of class c at x_i. The cost function becomes

$$cost = -\sum_{i=1}^K\left[\sum_{c=1}^C y_{i,c}f_{i,c}^m - \log\sum_{c'=1}^C e^{f_{i,c'}^m}\right]$$

$$+\frac{1}{2}\sum_{c=1}^C (f_c^m)'\,(\Sigma_c^{mm})^{-1}\,f_c^m$$

where $f_c^m = (f_{1,c}^m, \ldots, f_{K,c}^m)'$ and Σ_c^{mm} is the covariance at the data points for the c-th Gaussian.

A full Fisher scoring approach for learning would require the inversion of matrices of dimension CN. It is therefore appropriate to ignore terms in the Fisher information matrix which couple the different Gaussian processes. In this case we obtain update equations identical to the ones in Section 5.2 if we use the $\mu_{i,c}$ as defined in this section. Correspondingly, we would require one GBCM for each class.

6. DESIGN ISSUES

6.1 Effective Number of Parameters

In general, the GBCM will have good performance if Equation 1 is a good approximation. This is the case if data sets are independent conditioned on f^q. A trivial solution would be that f^q are the functional values at the training data since given those, all target measurements are independent. Of course, this is unsuitable since the BCM requires computations which scale to the third power with the number of query points but this discussion makes clear that the query data should be similar to the training data, i. e. should come from the same input data distribution.

From another point of view, to obtain a good approximation, we would require the query points to uniquely define the function since then target measurements are independent. In general, the degrees of freedom of a kernel based system are infinite which would mean that an infinite number of query points would be required to obtain independence. In [20] it was shown that even if in theory a Gaussian process regression system might have infinite degrees of freedom, in typical cases only a limited number of degrees of freedom are really used such that even with a small number of query points, the GBCM approximation is reasonable.

Let's define the effective degrees of freedom as

$$P_{eff}^{Data} = trace\,(\Sigma^{mm}(\Psi^{mm} + \Sigma^{mm})^{-1})$$

which measures how many degrees of freedom are used by the given data. This is analogous to the definition of the effective number of parameters by Wahba [24], Hastie and

Tibshirani [8], Moody [13] and MacKay [10].[3] Let $\{\lambda_0, \dots \lambda_K\}$ be the eigenvalues of Σ^{mm}. Then we obtain

$$P_{eff}^{Data} = \sum_{i=1}^{K} \frac{\lambda_i}{\lambda_i + \sigma_\psi^2}.$$

The effective degrees of freedom are therefore approximately equal to the number of eigenvalues which are greater than the noise variance. If σ_ψ^2 is small, the system uses all degrees of freedom to fit the data. For a large σ_ψ^2, $P_{eff}^{Data} \rightarrow 0$ and the data barely influence the solution. One can therefore conclude that only P_{eff}^{Data} query points are necessary to fix the relevant degrees of freedom of the system —assuming that training data and query data come from the same distribution— such that the GBCM performs well.

6.2 Online Implementations

In this section we consider an online version. We assume that data arrive continuously in time. Let D^k denote the data points collected between time $t(k)$ and $t(k-1)$ and let $\mathcal{D}^k = \{D^1, \dots, D^k\}$ denote the set of all data collected up to time $t(k)$.

We can derive an algorithm which directly adapts the posterior probabilities of f^Q. The iterations are based on the Kalman filter and yields for $k = 1, 2, \dots$

$$A_k = A_{k-1} + K_k(E(f^q|D^k) - A_{k-1})$$

$$K_k = S_{k-1}(S_{k-1} + cov(f^q|D^k))^{-1}$$

$$S_k = S_{k-1} - K_k(S_{k-1} + cov(f^q|D^k))K_k'$$

with $S_0 = \Sigma^{qq}$ and $A_0 = (0, \dots, 0)'$. At any time

$$\hat{E}(f^q|\mathcal{D}^k) = \frac{1}{k}\Sigma^{qq}(\frac{1}{k}\Sigma^{qq} - S_k)^{-1}A_k.$$

Note that only one matrix and one vector of the size of the number of query points need to be stored which makes this online version of the GBCM applicable for KDD applications.

Another formulation of the BCM and the corresponding online version can be found in Appendix B.

6.3 Parallelization

Due to the modularity of the approach, the GBCM is easily parallelizable. The Gaussian approximation of the posterior density of the individual committee members can be calculated in parallel by M processors.

6.4 New Test Points

After training, we might be interested in the responses at additional query points. The goal is then to infer from the estimated density at the query points f^q the density at other query points f^*. $\hat{P}(f^*|D)$ is approximately Gaussian distributed with expectation

$$\hat{E}(f^*|D) = \Sigma^{*q}(\Sigma^{qq})^{-1}\hat{E}(f^q|\mathcal{D}^k). \quad (10)$$

[3]Note, that $\Sigma^{mm}(\Psi^{mm} + \Sigma^{mm})^{-1}$ maps training targets into predictions at the same locations. This expression is therefore equivalent to the so called hat matrix in linear regression. There analogously, the trace of the hat matrix is defined as the effective number of parameters.

Note, that this equation describes the superposition of basis functions defined for every query point f^q and evaluated at f^* with weights $(\Sigma^{qq})^{-1}\hat{E}(f^q|\mathcal{D}^k)$. Note also, that the inverted matrix is of the dimension of the number of query points. Furthermore the approximate covariance becomes

$$\widehat{cov}(f^*|D) = \Sigma^{**}$$

$$+\Sigma^{*q}\left[(\Sigma^{qq})^{-1}\ \widehat{cov}(f^q|D)\ (\Sigma^{qq})^{-1} - (\Sigma^{qq})^{-1}\right](\Sigma^{*q})'.$$

The covariance of the prediction is small if the covariance $\widehat{cov}(f^q|D)$ has small elements and if f^* can be well determined from f^q.

6.5 A Fast Version for Two-class Classification

Whereas the prediction for Gaussian process regression can be computed in closed form, for the generalized model training is iterative and requires typically around 10 iteration steps. There is a fast alternative which in our experiments gave very comparable classification results. It is applicable when the experimenter is not interested in posterior class probabilities as the GBCM would supply but just in the classification decision. In this case one can change the targets to 1 if a training pattern belongs to class 1 and to -1 if a training pattern belongs to class 2. Then treat the problem as a regression problem and use Equation 4 and Equation 5 for the calculation of the estimate and the covariance matrix. Finally combine the predictions using the BCM. For a new test point use the sign of the output of the BCM as classification decision.

6.6 Valid Covariance Matrices

In our work we always used a Gaussian dependency (i.e. a Gaussian kernel) for the covariance as in Equation 9. Here, the experimenter has great freedom in choosing an alternative kernel. The only condition is that it must generate a non-negative definite covariance matrix for any set of points. Mackay [11] has published a number of recipes of how one can design (problem specific) covariance functions. As an example, Williams and Rasmussen report good results by adding a constant (and sometimes additional terms) to the Gaussian kernel.

There is a very close relationship between Gaussian processes and regression with fixed basis functions. Consider that $f(x)$ is a superposition of N fixed basis functions, i.e.

$$f(x) = \sum_{i=1}^{N} w_i \phi_i(x)$$

where $w = (w_1, \dots, w_N)'$ is the $N-$dimensional weight vector with a Gaussian weight prior with mean 0 and covariance Σ_w. We can now calculate the prior distribution for $f(x)$ itself which is a zero mean Gaussian process with

$$\sigma_{x_i^q, x_j^m} = \phi(x_i^q)'\Sigma_w\phi(x_j^m) \quad (11)$$

where $\phi(x) = (\phi_1(x), \dots, \phi_N(x))'$ is the activation of the fixed basis functions at x. In this way we can derive covariance functions from systems with fixed basis functions. Neal [15] and Williams [27] have shown that multi-layer perceptron neural networks with an infinite number of hidden units also can be described by Gaussian processes.

One might also ask: given a Gaussian process can we find an equivalent system of fixed basis functions or in other words is there always a decomposition as described in Equation 11. The answer is generally yes but we will typically require an infinite number of fixed basis functions (see also the discussion on the effective number of parameters in Section 6.1). In this sense Gaussian process regression corresponds to the fitting of functions to an infinite number of fixed basis functions.

6.7 Support Vector Machines

Let's consider the cost function [26, 16]

$$cost(f^m) = const \sum_{i=1}^{K} |1 - (2y_i^m - 1)(f_i^m + b)|_+$$

$$+\frac{1}{2}(f^m)'(\Sigma^{mm})^{-1} f^m.$$

where $|.|_+$ is equal to its argument when the argument is larger than zero and zero otherwise and where $const > 0$.[4] After optimization a large number of training data will not contribute to the first term in the cost function which means that the corresponding components in the weight vector (see Section 4.1) v are zero. The "surviving" weights correspond to the support vectors [23] and the output to a new input x_i^q is

$$sign(\sum_{\text{support vectors}} \sigma_{x_i^q, x_j^m}\, v_j).$$

In principle the SVM-approach would also be suitable for the GBCM although due to the highly nonlinear cost function the second order Laplace approximation used for the GBCM might not be very good.

As a note, Platt [16] discusses a version of the SVM which allows the calculation of posterior class probabilities and achieved results comparable to the solution obtained using the approach described in Section 5.2.

6.8 Other Interesting Models

We have already seen in Section 5.3 how several Gaussian processes can interact in multiple-class classification. Interacting Gaussian processes can form many more interesting models [4]. In [21] it is shown how Gaussian Processes can be used to model the dependency structure in a Bayesian belief network and the ideas are applied to the mixtures of experts system [9]. These models can be used in connection with the GBCM where the number of GBCMs is equal to the number of Gaussian processes used in the model.

6.9 Preliminary Results on Boosting the GBCM

The GBCM can be considered to be a committee in which each committee member is trained independently from one another. In contrast in boosting the training of a committee member depends on the performance of the previously trained committee members. Typically one puts more weight on training patterns which were previously misclassified. In boosting not only the variance in the prediction is decreased as in other approaches to committee machines but also the bias in the prediction [7]. Boosting was introduced by Schapire and Freund [19, 5, 6] and a description of the state of research and current variants can be found in [3].

Let's look at a combination of the GBCM with boosting and consider a particular variant, i.e. the Gentle AdaBoost algorithm [7]. If applied to the GBCM we would change the weights on the i-th training exemplar for the current committee member to

$$w_i = e^{-y_i F_i}$$

where $y_i \in \{-1, 1\}$ indicates the class label and F_i is the output of the current committee. Note that more weight is placed on mis-classified patterns. The new committee member is formed by a weighted regression using Equation 6 where

$$\Psi^{mm} = diag(\frac{\sigma_\psi^2}{w_i})$$

incorporates the weights on the training patterns. The committee is formed either by the BCM algorithm or by simple addition of the ouputs of the committee members. The classification is performed based on the sign of the output of the committee.

We have done some preliminary experiments which did not show a significant improvement by the addition of boosting to the GBCM. The reason might be that Gausssian processes are rather strong learners and boosting was mostly successful with weak learners such as very simple decision trees. Our result so far is that the reweighting did not strongly influence the prediction of the Gaussian process classifier.

7. EXPERIMENTS

In the first experiment we investigated the computational scaling behavior of the BCM in conjunction with Gaussian process regression. Figure 1 shows the CPU-time as a function of the number of training samples. Clearly, the CPU-time for Gaussian process regression scales as K^3. The two versions of the BCM schemes scale linearly. Also note that for $K/M = 100 \approx K/N_Q$ (in the experiment, $N_Q = 128$) we obtain the least computational cost.

In the second set of experiments we tested the performance of the GBCM using a number of real and artificial data sets. In the experiments we set for two locations in input space x_i and x_j: $\sigma_{x_i,x_j} = A\exp(-1/(2\gamma^2)||x_i - x_j||^2)$ where γ is a positive constant optimized with cross validation and $A = 10$, implementing a weak prior. The input for the artificial data set space was $Dim-$dimensional and the inputs were randomly chosen in $x \in [-1, 1]^{Dim}$. We generated first continuous targets by adding independent Gaussian noise with variance σ_ψ^2 to a map defined by 5 normalized Gaussian basis functions (see Appendix C). We then assigned patterns with positive target values to class one and patterns with negative target values to class two. By varying the variance of the noise we obtain non-overlapping classes ($\sigma_\psi^2 = 0$) and overlapping classes ($\sigma_\psi^2 > 0$). Table 1 summarizes the results. Well demonstrated is the excellent performance of the GBCM. For the real world data (Table 1)

[4] In a previous version of the paper we defined $|.|_+$ to be equal to one if the argument is larger than zero and zero otherwise. Our new definition includes the case that classes are not linearly separable.

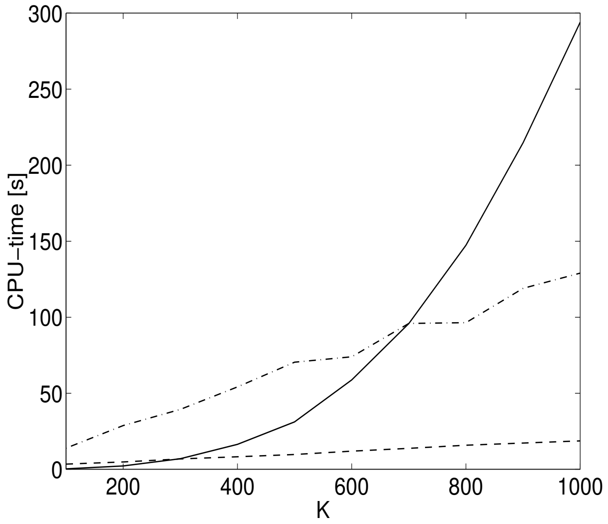

Figure 1: CPU-time as a function of the training set size K for $M = 1$ (Gaussian process regression, continuous) and the BCM with module size $K/M = 10$ (dash-dotted) and module size $K/M = 100$ (dashed). In these experiments the number of query points was $N_Q = 128$.

we obtain excellent performance for the BUPA and the DIABETES data sets.

In our third experiment (Figure 2) we tested the online GBCM described in Section 6.2 with a $Dim = 5-$dimensional input. Note, that the GBCM with $K = 60000$ achieves results unobtainable with Gaussian process regression which is limited to a training data set of approximately $K = 1000$. For $K = 60000$, Gaussian process regression would require one year of CPU-time.

8. CONCLUSIONS

The GBCM provides a new principled approach for the combination of estimators for regression, classification, the prediction of counts, the prediction of life times and other applications which can be derived from the exponential family of distributions. We found experimentally that the GBCM provides excellent predictions on several data sets. Since the CPU-time of the GBCM scales linearly in the number of training data, it can be used in applications with large data sets as in data mining and in applications requiring online learning.

As a final note, after completion of the manuscript the author became aware of independent work on scaling up Gaussian processes to large data sets based on mean field approaches from statistical physics [2].

9. REFERENCES

[1] J. M. Bernardo and A. F. M. Smith. *Bayesian Theory.* John Wiley and Sons, New York, 1994.

[2] L. Csató, E. Fokoué, M. Opper, B. Schottky, and O. Winther. Efficient approaches to gaussian process classification. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in neural information processing systems 12.* MIT Press, 2000.

[3] H. Drucker. Boosting neural networks. In A. J. C. Sharkey, editor, *Combining artificial neural nets.* Springer, 1999.

[4] L. Fahrmeir and G. Tutz. *Multivariate Statistical Modeling Based on Generalized Linear Models.* Springer, New York, 1994.

[5] Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995.

[6] Y. Freund and R. E. Schapire. A decision theoretic generalization of online learning and an application to boosting. *Journal of Computer and System Sciences*, 55, 1997.

[7] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 2000. To appear.

[8] T. J. Hastie and R. J. Tibshirani. *Generalized additive models.* Chapman & Hall, 1990.

[9] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and J. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3, 1991.

[10] D. J. C. MacKay. Bayesian model comparison and backprop nets. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in neural information processing systems 4*, pages 839–846. Morgan Kaufmann, 1992.

[11] D. J. C. MacKay. Introduction to gaussian processes. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, volume 168. NATO Asi Series F, Computer and Systems Sciences, Morgan Kaufmann, 1998.

[12] P. McCullagh and J. Nelder. *Generalized linear models.* Chapman & Hall, 1983.

[13] J. E. Moody. The effective number of parameters: an analysis of generalization and regularization in nonlinear learning systems. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in neural information processing systems 4*, pages 847–854. Morgan Kaufmann, 1992.

[14] R. M. Neal. *Bayesian learning for neural networks.* Springer, 1996.

[15] R. M. Neal. *Monte Carlo implementation of Gaussian process models for Bayesian regression and classification.* Chapman & Hall, 1997.

[16] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A. J. Smola, editor, *Advances in large Margin Classifiers.* MIT press, 2000.

[17] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78:1481–1497, 1990.

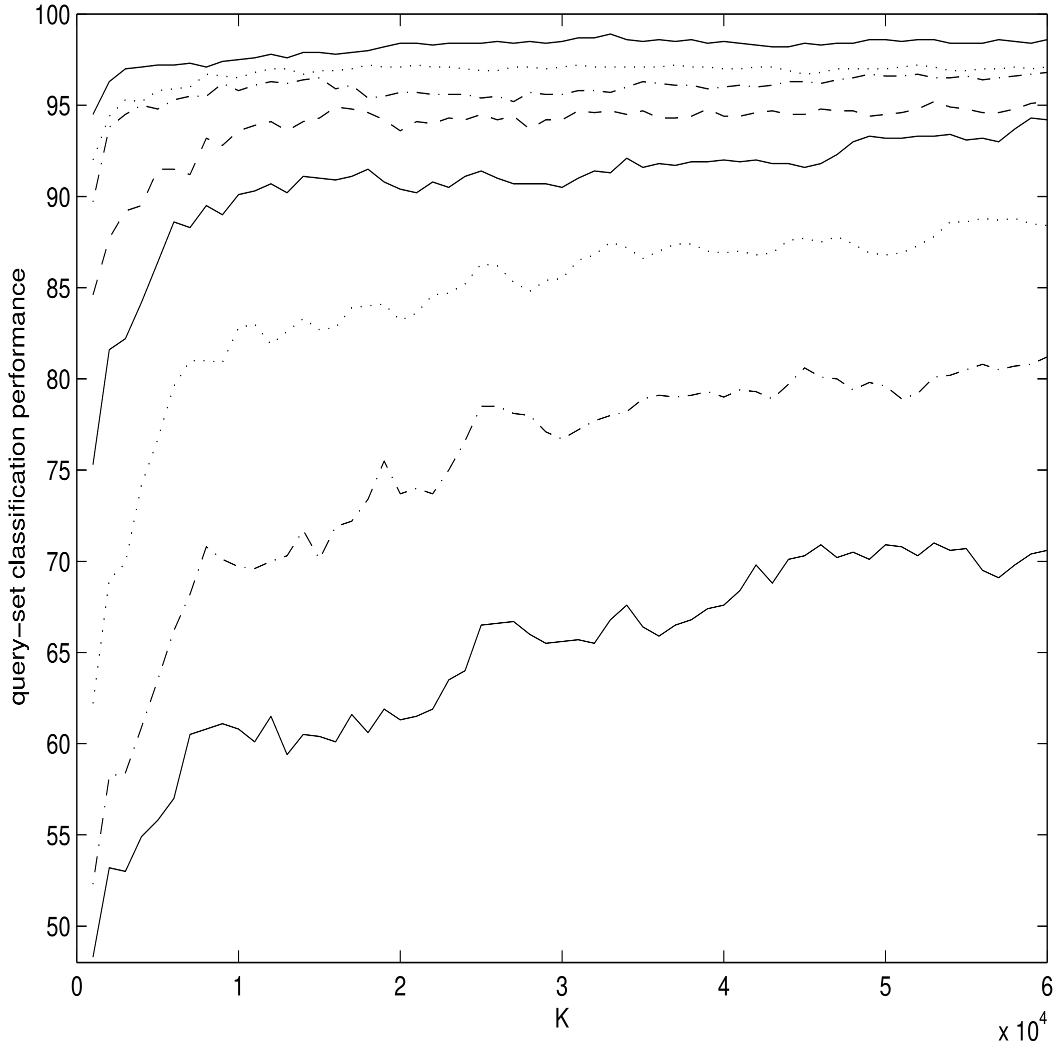

Figure 2: **The test of the online learning algorithms of Section 6.2 using the artificial data set. Shown is the percentage of correct classification on the query-set for $N_Q = 1000$ and a module size of $K/M = 1000$ as a function of the number of training data K. In the experiment, σ_ψ is varied. From top curve to bottom curve the degree of overlap between classes increases. From top to bottom: $\sigma_\psi = 0$ (continuous), $\sigma_\psi = 0.1$ (dotted), $\sigma_\psi = 0.2$ (dash-dotted), $\sigma_\psi = 0.5$ (dashed), $\sigma_\psi = 1$ (continuous), $\sigma_\psi = 2$ (dotted), $\sigma_\psi = 4$ (dash-dotted), $\sigma_\psi = 8$ (continuous). Note, that (generalized) Gaussian process regression is limited to a data set of approximately 1000 which corresponds to the furthest left points of the curves. The improvement in performance possible with the GBCM is impressive: For $\sigma_\psi = 8$ the percentage of correct classification increases from less than 50 % (for $K = 1000$) to more than 70% (for $K = 60000$) and for $\sigma_\psi = 0$ the percentage of correct classification increases from less than 95 % (for $K = 1000$) to more than 98.5% (for $K = 60000$).**

Table 1: The table shows results from real world data and artificial data. The real world data sets can be retrieved from the UCI data base at http://www.ics.uci.edu/ mlearn/MLRepository.html. The inputs to all data sets were normalized to a mean of zero and a standard deviation of one. Shown is the input dimension Dim, the number of training data used K, and the percentage of correct classification of various algorithms. Shown are averages over 20 experiments from which error bars could be derived. The width parameter γ was optimized using a validation set of size 100. The first result (GPR) shows the performance of (generalized) Gaussian process regression (M=1). GBCM(i, j) shows the classification performance with module size $K/M = i$ and query-set size $N_Q = j$. The artificial data are generated as described in the beginning of Section 7. In columns from left to right, the experiments using the artificial data are characterized as: no overlap between classes $\sigma_\psi = 0$, large overlap between classes $\sigma_\psi = 0.5$, high input dimension $Dim = 50$ and low input dimension $Dim = 2$. Note the excellent performance of the GBCM approximation.

	BUPA	DIAB.	ART	ART	ART	ART
Dim	6	8	5	5	50	2
K (train. size)	200	600	600	600	600	600
σ_ψ	-	-	0	.5	.1	.1
GPR	69.15 ± 0.5	77.5 ± 0.0	97.6 ± 0.4	93.2 ± 0.4	90.1 ± 0.4	98.4 ± 0.4
GBCM(10, 50)	70.0 ± 0.8	77.8 ± 1.3	98.0 ± 0.4	93.2 ± 0.9	89.9 ± 0.9	99.0 ± 0.4
GBCM(100, 50)	70.6 ± 1.7	78.2 ± 1.0	97.3 ± 0.5	93.5 ± 1.0	89.1 ± 1.0	98.6 ± 0.5
GBCM(10, 100)	69.9 ± 0.9	76.6 ± 0.7	97.4 ± 0.3	92.9 ± 0.6	90.6 ± 0.6	98.4 ± 0.3
GBCM(100,100)	69.2 ± 0.7	76.7 ± 0.8	97.6 ± 0.4	93.2 ± 0.5	91.7 ± 0.7	98.0 ± 0.4

[18] C. E. Rasmussen. Evaluation of gaussian processes and other methods for non-linear regression. Technical report, Department of Computer Science, University of Toronto, 1996.

[19] R. E. Schapire. The strength of weak learnability. *Machine learning*, 5(2):197–227, 1990.

[20] V. Tresp. A bayesian committee machine. *Neural Computation*, 2000. In print.

[21] V. Tresp. Interacting gaussian processes for graphical models. submitted, 2000.

[22] V. Tresp and M. Taniguchi. Combining estimators using non-constant weighting functions. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in neural information processing systems 7*, pages 419–426. MIT Press, 1995.

[23] V. N. Vapnik. *The nature of statistical learning theory.* Springer, New York, 1995.

[24] G. Wahba. *Spline models for observational data.* Society for Industrial and Applied Mathematics, 1983.

[25] G. Wahba. Bayesian "confidence intervals" for the cross-validated smoothing spline. *J. Roy. Stat. Soc. Ser. B*, 10:133–150, 1990.

[26] G. Wahba. Support vector machines, reproducing kernel hilbert spaces and randomized GACV. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods*, pages 69–88. MIT Press, 1999.

[27] C. K. I. Williams. Computing with infinite neural networks. *Neural Computation*, 10:1203–1216, 1998.

[28] C. K. I. Williams and D. Barber. Bayesian classification with gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:1342–1351, 1998.

[29] C. K. I. Williams and C. E. Rasmussen. Gaussian processes for regression. In M. C. Mozer and M. E. Hasselmo, editors, *Advances in neural information processing systems 8*, pages 514–520. MIT Press, 1996.

APPENDIX

A. FISHER SCORING ALGORITHM

We start with the cost function

$$cost(f^m) = -\sum_{i=1}^{K} \log P(y_i^m | f_i^m) + \frac{1}{2}(f^m)'(\Sigma^{mm})^{-1} f^m.$$

The mode can be found by a Fisher scoring algorithm of the form

$$f^{m,(k+1)} = f^{m,(k)} + F^{-1}(f^{m,(k)}) J(f^{m,(k)})$$

for $k = 1, 2, \ldots$ and where

$$J(f^{m,(k)}) = -\frac{\partial cost(f^{m,(k)})}{\partial f^{m,(k)}} = s(f^{m,(k)}) - (\Sigma^{mm})^{-1} f^{m,(k)}$$

is the vector of negative first derivatives and

$$F(f^{m,(k)}) = (\Psi^{(k)})^{-1} + (\Sigma^{mm})^{-1}$$

is the expected (penalized) Fisher information matrix. Furthermore

$$s(f^{m,(k)}) = vec\left(D_i^{(k)} (\sigma_i^{(k)})^{-2} (y_i^m - h(f_i^{m,(k)}))\right).$$

In all cases of interest here the expected (penalized) Fisher information is identical to the Hessian of the cost function and therefore the Fisher scoring update is identical to a Newton-Raphson update.

Then

$$f^{m,(k+1)} = \left((\Psi^{(k)})^{-1} + (\Sigma^{mm})^{-1}\right)^{-1}$$

$$\times \left(s(f^{m,(k)}) + (\Psi^{(k)})^{-1} f^{m,(k)}\right)$$

$$= \Sigma^{mm} \left(\Psi^{(k)} + \Sigma^{mm}\right) \Psi^{(k)} \left(s(f^{m,(k)}) + (\Psi^{(k)})^{-1} f^{m,(k)}\right)$$

$$= \Sigma^{mm} (\Psi^{(k)} + \Sigma^{mm})^{-1} \tilde{y}^{(k)}$$

where we have used that

$$\left((\Psi^{(k)})^{-1} + (\Sigma^{mm})^{-1}\right)^{-1} = \Sigma^{mm} \left(\Psi^{(k)} + \Sigma^{mm}\right) \Psi^{(k)}.$$

$\tilde{y}^{(k)}$ is defined in Section 5.1.

B. ANOTHER FORMULATION OF THE BCM

Based on our Gaussian assumptions we can also calculate the probability density of measurements given the query points $P(y^m|f^q)$. This density is also Gaussian with mean

$$E(y^m|f^q) = (\Sigma^{qm})'(\Sigma^{qq})^{-1} f^q, \tag{12}$$

and covariance

$$cov(y^m|f^q) = \Psi^{mm} + \Sigma^{mm} - (\Sigma^{qm})'(\Sigma^{qq})^{-1}\Sigma^{qm}. \tag{13}$$

Now, the BCM approximation can also be written as

$$\hat{P}(f^q|D) \propto P(f^q) \prod_{i=1}^{M} P(y_i^m|f^q).$$

Then we obtain with $A_i = (\Sigma^{qm,i})'(\Sigma^{qq})^{-1}$

$$\hat{E}(f^q|D) = C^{-1} \sum_{i=1}^{M} A_i' cov(y_i^m|f^q)^{-1} y_i^m \tag{14}$$

with

$$C = \widehat{cov}(f^q|D)^{-1} = (\Sigma^{qq})^{-1} + \sum_{i=1}^{M} A_i' cov(y_i^m|f^q)^{-1} A_i. \tag{15}$$

$\Sigma^{qm,i}$ is the the covariance matrix between training data for the i-th module and the query points. An online version can be obtained using the Kalman filter which yields the iterative set of equations, $k = 1, 2, \ldots$

$$\hat{E}(f^q|\mathcal{D}^k) = \hat{E}(f^q|\mathcal{D}^{k-1}) + K_k(y_k^m - A_k f^q)$$

$$K_k = (S_{k-1}A_k')(A_k S_{k-1} A_k' + cov(y_k^m|f^q))^{-1}$$

$$S_k = S_{k-1} - K_k(A_k S_{k-1} A_k' + cov(y_k^m|f^q))K_k'$$

with $S_0 = \Sigma^{qq}$. In the iterations, no matrix of size N_Q needs to be inverted.

C. ARTIFICIAL DATA

Explicitly, the target is calculated as

$$y_i =$$

$$\frac{1}{2}\left(1 + sign\left(\frac{\sum_{j=1}^{5} a_j \exp\left(-\frac{||x_i - center_j||^2}{2\sigma_a^2}\right)}{\sum_{j=1}^{5} \exp\left(-\frac{||x_i - center_j||^2}{2\sigma_a^2}\right)} + \epsilon_i\right)\right).$$

In the experiments

$$\sigma_a = 0.36 \quad a = (1.16, -0.63, -0.08, 0.35, -0.70)'$$

and $center_i$ is generated randomly according to a uniform density in the Dim-dimensional unit hypercube.