Bagging for Gaussian Process Regression

Tao Chen [a,*], Jianghong Ren [b]

[a] *School of Chemical and Biomedical Engineering, Nanyang Technological University, Singapore 637459*

[b] *College of Automation, Chongqing University, Chongqing 400044, China*

Abstract

This paper proposes the application of bagging to obtain more robust and accurate predictions using Gaussian process regression models. The training data is re-sampled using the bootstrap method to form several training sets, from which multiple Gaussian process models are developed and combined through weighting to provide predictions. A number of weighting methods for model combination are discussed, including the simple averaging rule and the weighted averaging rules. We propose to weight the models by the inverse of their predictive variance, and thus the prediction uncertainty of the models is automatically accounted for. The bagging method for Gaussian process regression is successfully applied to the inferential estimation of quality variables in an industrial chemical plant.

Key words: Bagging, Bayesian method, Bootstrap, Gaussian process, Model robustness, Soft sensor

1 Introduction

Bagging, short for “Bootstrap AGGregatING”, is a method of obtaining more robust and accurate models using bootstrap re-samples of the training data [1,2]. The procedure for bagging consists of two stages. First bootstrap samples are obtained from the original data to form a set of training sets, from which multiple models are developed. Then these models are combined in some way to make predictions. Bagging can be applied to both regression and classification models, whilst the focus is on regression in this study. It was shown [1] that bagging is especially suitable for “unstable” models, i.e. the models that

* Corresponding author, Email: chentao@ntu.edu.sg; Tel.: +65 6513 8267; Fax: +65 6794 7553.

are sensitive (in terms of model parameters and predictive performance) to small changes in the training data. In practice, bagging has been applied to a large number of model structures, including regression trees [3], regression with variable selection [4] and neural networks [5,6].

The primary purpose of this paper is to apply the bagging procedure to improve the robustness and accuracy of Gaussian process regression models that have recently received significant interests in the community of machine learning and applied statistics [7–11]. Initially proposed by O'Hagan [12], Gaussian process regression was viewed as an alternative approach to artificial neural networks, primarily as a result of the seminal research of Neal [13]. Neal showed that a large class of Bayesian regression models, based on artificial neural networks, converged to a Gaussian process, in the limit of an infinite network [13]. Gaussian processes can also be derived from the perspective of non-parametric Bayesian regression [14], by directly placing Gaussian prior distribution over the space of regression functions. As a result of its good performance in practice and desirable analytical properties, Gaussian process models have seen wide applications, such as rehabilitation engineering [15], machining optimization [16] and calibration of analytical sensors [17].

The motivation of bagging Gaussian process models is that we have observed that a single Gaussian process does not always give satisfactory predictions in practice. In particular we are considering a specific application scenario in chemical plants where the product quality variables (e.g. melt flow rate of polypropylene) are inferred from process operational variables (e.g. reactor temperature and feed rate of raw material). The details are given in Section 4. Although instruments for measuring the quality variables are available, these instruments usually possess substantial delays, and thus are not suitable for the on-line quality assurance and control purpose. Therefore the key in inferential estimation is to identify the relationship between the difficult-to-measure (quality) variables and the easy-to-measure (operational) variables [6]. The inferential models serve as “virtual” sensors to provide the information about process quality variables. In this study, Gaussian process regression model, enhanced by the bagging procedure, is employed to achieve this purpose. To the best of our knowledge there has been little report of applying bagging to Gaussian process models in the literature. Hence the results in this paper could provide a guideline to other modeling practice where Gaussian process is utilized.

The rest of this paper is organized as follows. Section 2 gives a brief overview of Gaussian process regression models, followed by the introduction of bagging in Section 3. The conventional model combination methods are discussed, and a novel model weighting strategy is proposed. The effectiveness of the bagging method for Gaussian process models is demonstrated through its industrial application in Section 4. Finally Section 5 concludes this paper.

2 An overview of Gaussian process

The idea of Gaussian process can be dated back to the classic paper by O'Hagan [12]. However, the application of Gaussian process as a regression (and classification) technique in the community of pattern recognition was not common until the late 1990's, where the rapid advancement of computational power helped facilitate the implementation of Gaussian process for larger data sets. The Gaussian process regression model can be derived from the perspectives of neural networks and Bayesian non-parametric regression; see [13,14,10] for details. In this section a brief overview of Gaussian process regression model is given, including the formulation and implementation of the model.

From the perspective of a regression problem, a functional relationship is identified between the D dimensional input variables, $\mathbf{x}$, and the output variable, y. Here the formulation is restricted to univariate output. A discussion on Gaussian process model with multivariate output is given in [14,17]. A Gaussian process defined on multiple outputs has several difficulties, such as the complexity in the model and its implementation [17]. In practice a simplified solution could be adopted to model each output variable separately [14]. By adopting this approach, it can be argued that significant information contained in the correlation structure between the output variables is ignored. However in the literature of regression modeling, there is no consensus on whether multi-output modeling can achieve better predictive performance than separate modeling. The use of separate models is also justified by their good performance in various applications [6,17].

2.1 The model

Consider a training data set consisting of N data points, $\{\mathbf{x}_i, y_i; i = 1, \ldots, N\}$. A Gaussian process for regression is defined such that the regression function $y(\mathbf{x})$ has a Gaussian prior distribution with zero mean, or in discrete form:

$$\mathbf{y} = (y_1, \ldots, y_N)^{\mathrm{T}} \sim G(\mathbf{0}, \mathbf{C}) \tag{1}$$

where $\mathbf{C}$ is an $N \times N$ covariance matrix of which the ij-th element is defined by the covariance function: $\mathbf{C}_{ij} = C(\mathbf{x}_i, \mathbf{x}_j)$. An example of such a covariance function is:

$$C(\mathbf{x}_i, \mathbf{x}_j) = a_0 + a_1 \sum_{d=1}^{D} x_{id} x_{jd} + v_0 \exp\left(-\sum_{d=1}^{D} w_d (x_{id} - x_{jd})^2\right) + \delta_{ij}\sigma^2 \qquad (2)$$

where $\mathbf{x}_i = (x_{i1}, \ldots, x_{iD})$; $\delta_{ij} = 1$ if $i = j$, otherwise it is equal to zero. We denote $\boldsymbol{\theta} = (a_0, a_1, v_0, w_1, \cdots, w_D, \sigma^2)$ as "hyper-parameters" defining the covariance function. The hyper-parameters must be non-negative to ensure that the covariance matrix is non-negative definite. The term "hyper-parameter" is used to differentiate Gaussian processes from parametric regression, where the parameter is required to be estimated. For the covariance function depicted in Eq. (2), the first two terms represent a constant bias (offset) and a linear correlation term, respectively. The exponential term is similar to the form of a radial basis function, and it takes into account the potentially strong correlation between the outputs for nearby inputs. The term σ^2 captures the random error effect. By combining both linear and non-linear terms in the covariance function, Gaussian process is capable of handling both linear and non-linear data structures [17]. Other forms of covariance functions are discussed in [14].

For a new data point with input vector $\mathbf{x}^*$, the predictive distribution of the output y^* conditional on the training data is also Gaussian, of which the mean ($\hat{y}^*$) and variance ($\sigma^2_{\hat{y}^*}$) are calculated as follows:

$$\hat{y}^* = \mathbf{k}^{\mathrm{T}}(\mathbf{x}^*)\,\mathbf{C}^{-1}\mathbf{y} \qquad (3)$$

$$\sigma^2_{\hat{y}^*} = C(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}^{\mathrm{T}}(\mathbf{x}^*)\,\mathbf{C}^{-1}\,\mathbf{k}(\mathbf{x}^*) \qquad (4)$$

where $\mathbf{k}(\mathbf{x}^*) = [C(\mathbf{x}^*, \mathbf{x}_1), \ldots, C(\mathbf{x}^*, \mathbf{x}_N)]^{\mathrm{T}}$. The capability to providing the prediction uncertainty in terms of the variance is an important feature of Gaussian process. The predictive variance in Eq. (4) plays an important role in model combination that is presented in Section 3.

2.2 Implementation

The hyper-parameters $\boldsymbol{\theta}$ can be estimated by maximizing the following log-likelihood function:

$$L = -\frac{1}{2}\log\det\mathbf{C} - \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{y} - \frac{N}{2}\log 2\pi \qquad (5)$$

Most training algorithms also require the derivative of L with respect to each hyper-parameter θ:

$$\frac{\partial L}{\partial \theta} = -\frac{1}{2}\mathrm{tr}\left(\mathbf{C}^{-1}\frac{\partial \mathbf{C}}{\partial \theta}\right) + \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{C}^{-1}\frac{\partial \mathbf{C}}{\partial \theta}\mathbf{C}^{-1}\mathbf{y} \tag{6}$$

where $\partial\mathbf{C}/\partial\theta$ can be obtained from the covariance function.

In situations where the prior distribution is assigned to the hyper-parameters, it can be incorporated into the likelihood function to realize a maximum a posterior (MAP) estimation. Both maximum likelihood estimation and the MAP method are known to be sensitive to initializations and may converge to local optima [10]. A more elaborate approach is to use Markov chain Monte Carlo (MCMC) [10] method to generate samples for approximation of the posterior distribution of hyper-parameters. MCMC introduces significantly higher computation cost, and thus its application in engineering practice is restricted. In this study the maximum likelihood estimation is adopted for model training, where a conjugate gradient method is employed to search for the hyper-parameters that maximize the likelihood [10].

It should also be noted that the calculation of the likelihood and the derivatives involves a matrix inversion step and takes time of the order $O(N^3)$, which is feasible for a moderate size of training data sets (less than several thousand) on a conventional computer. For larger data sets, sparse training strategies may be required to reduce the overall computational burden [18,19]. A comprehensive investigation of sparse Gaussian processes is beyond the scope of this paper. The computational aspect will be discussed within the case study in Section 4.

3 Bagging for Gaussian process models

The basic idea of bagging is straightforward. Instead of making predictions from a single model that is fit to the observed data, a number of models are developed to characterize the same relationship between input and output variables. Each model is developed from a bootstrap [20] re-sampled set of the original training data. Then the predictions from the multiple models are combined to improve model accuracy and robustness. The procedure of bootstrap re-sampling is briefly described here.

Suppose the original training data is $Z = \{\mathbf{x}_i, y_i; i = 1, \ldots, N\}$. We randomly sample N data points with replacement from Z where the probability of each data point being selected is $1/N$. These N data points are regarded as a re-sampled training set that is denoted Z_1. Then we repeat the procedure for

K times and obtain K re-sampled data sets: $Z_1, \dots, Z_K$. Finally K separate models can be developed from these re-sampled training sets.

The method of bagging is especially applicable for unstable modeling procedures, i.e. the models that are sensitive to small changes in the data, such as neural networks and classification and regression trees [1]. Gaussian process is among the unstable modeling procedures where bagging is potentially useful. The model may be sensitive to data if the amount of training data is limited. The availability of data is an important factor when developing models for industrial applications, where the data collection process involves significant cost [6]. In addition, the choice of training algorithms also affects the accuracy and robustness of a Gaussian process model. The conventional method, e.g. conjugate gradient, for hyper-parameter estimation is sensitive to initialization and may find a local optima that does not provide good prediction performance over the entire data space. Although the MCMC method can be utilized to partially address these issue, it is not considered in this paper due to the computational constraints in practice, and the focus is on the implementation of bagging for Gaussian process models.

Suppose K Gaussian process models, $\mathscr{M} = \{M_1, \dots, M_K\}$, have been developed from the bootstrap re-sampled data sets. The rest of this section discusses the methods to combine the multiple models for prediction, including simple averaging and weighted averaging rules.

3.1 Combination using simple averaging rule

The averaging rule for multi-model combination is the original method in bagging [1]. Formally the prediction y^* is the average of the predictive distributions from the K models:

$$p(y^*|\mathscr{M}) = \frac{1}{K}\sum_{k=1}^{K} p(y^*|M_k) \tag{7}$$

where $p(y^*|M_k)$, $k = 1, \dots, K$ are Gaussian with mean and variance calculated from Eqs. (3) and (4), i.e. $p(y^*|M_k) = G(\hat{y}^*(k), \sigma^2_{\hat{y}^*}(k))$. Essentially Eq. (7) is not a Gaussian distribution any more but a mixture of K Gaussians, or the well known Gaussian mixture model [21]. The predictive mean and variance for the combined models can be calculated based on the property of the Gaussian mixture model:

$$E(y^*) = \frac{1}{K}\sum_{k=1}^{K} \hat{y}^*(k) \tag{8}$$

$$Var(y^*) = \frac{1}{K}\sum_{k=1}^{K}\sigma^2_{\hat{y}^*}(k) + \frac{1}{K}\sum_{k=1}^{K}\left(\hat{y}^*(k) - E(y^*)\right)^2 \tag{9}$$

It has been observed that this simple averaging rule can significantly improve the model accuracy and robustness in various applications [4,6,22,23].

3.2 Combination using weighted averaging rules

An alternative combination approach is to assign different weights to the models, the rationale being that the models should be weighed according to the corresponding prediction capability. A linear regression method was proposed to learn these weights so that the combined predictors minimize the training error [6]. More specifically for the entire training data set, the output variable y is regressed on the predicted outputs from each model $\hat{y}(k)$:

$$y_i = w_1\,\hat{y}_i(1) + w_2\,\hat{y}_i(2) + \cdots + w_K\,\hat{y}_i(K), \quad i = 1, \ldots, N \tag{10}$$

where $\{w_k, k = 1, \ldots, K\}$ are the weights. The linear regression in Eq. (10) is an ill-conditioned problem because predictions from the multiple models are highly correlated. This collinearity issue was addressed by using principal component analysis [6], whereas other approaches are also applicable, including ridge regression and partial least squares [24]. This regression based weighting approach tends to assign larger weights to the models that give better "prediction" to the training data. In fact the models are weighted according to how well they fit, as opposed to predict, the training data, since the data has already been used for model development. As a result, if some of the models severely over-fit the training data, the effect of over-fitting could be "amplified" by these models being assigned large weights. Therefore the generalization capability of the models may be compromised.

In this paper we propose a weighting strategy that relies on the prediction uncertainty. In effect the model that is uncertain about the prediction should be discounted with a smaller weight. Therefore the weights are not pre-calculated through fitting the training data and fixed for prediction; rather they are automatically adjusted based on the prediction uncertainty. The proposed strategy is originally motivated by the approach of Bayesian committee machine (BCM) [19]. BCM was derived based on the assumption that the correlation between the K Gaussian process models is negligible, an assumption that is unrealistic in the application of bagging. Hence we do not follow the derivation in [19]; rather we propose to combine the predictors based on the following product rule:

$$p(y^*|\mathscr{M}) \propto \prod_{k=1}^{K} p(y^*|M_k) \qquad (11)$$

For Gaussian process models, each predictor $p(y^*|M_k)$ is a Gaussian distribution. Therefore the product in Eq. (11) is still Gaussian with mean and variance defined by:

$$E(y^*) = Var(y^*) \sum_{k=1}^{K} \hat{y}^*(k)\, \sigma_{\hat{y}^*}^{-2}(k) \qquad (12)$$

$$Var(y^*) = \left(\sum_{k=1}^{K} \sigma_{\hat{y}^*}^{-2}(k) \right)^{-1} \qquad (13)$$

In Eqs. (12) and (13) the weights are $w_k = \sigma_{\hat{y}^*}^{-2}(k) / \sum_{k=1}^{K} \sigma_{\hat{y}^*}^{-2}(k)$ and the models are weighted by the inverse of the corresponding prediction variance. Models that are uncertain about their predictions, i.e. with large variances, are automatically given smaller weights than models that are certain about their predictions. Note that the weighting strategy of the BCM [19] is similar to Eqs. (12) and (13) with the difference being the variance term. Specifically the variance term of the BCM is:

$$Var(y^*) = -(K-1)C(\mathbf{x}^*, \mathbf{x}^*) + \left(\sum_{k=1}^{K} \sigma_{\hat{y}^*}^{-2}(k) \right)^{-1} \qquad (14)$$

Conceptually the BCM is less appealing since the sum of weights is not equal to unity. We have not found that the BCM attains better prediction accuracy than the proposed product rule (detailed results not reported) when applied to the industrial example in the next section. Therefore the BCM method is not considered further in this paper for bagging Gaussian process models.

4 Industrial case study

In this section we apply the bagging method for Gaussian process models to the inferential estimation of the quality variables, i.e. the melt flow rate (MFR), in a polypropylene polymerization process. A major difficulty in the monitoring and control of product quality in industrial polymerization reactors is the lack of suitable on-line polymer quality measurements [25,6]. In this case study, MFR is measured through laboratory analysis every two hours, which is typical in the polymerization industry. For the purpose of monitoring and control, on-line prediction of MFR is required.

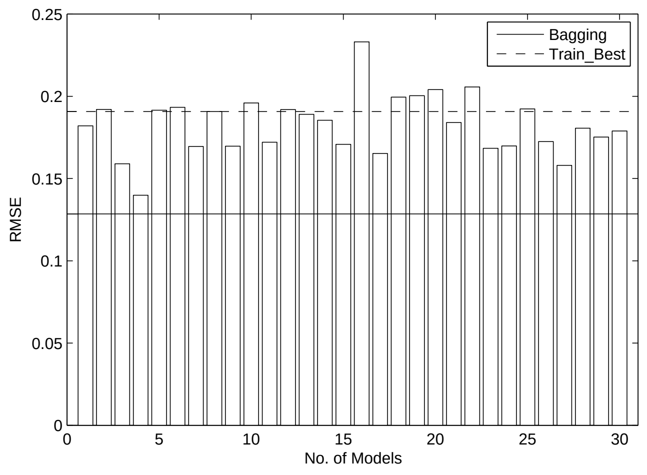
The quality variables could be inferred through the mathematical modeling of the process based on physical and chemical mechanism [25,26]. However the development of mechanistic models require a deep understanding of the process and is typically time-consuming. To meet the rapid changing of the market, a fast modeling approach is desirable. As a result the technique of data-based empirical modeling has emerged as a solution to infer the quality variables from process measurements that are routinely recorded [6,27,28]. The inferential model is also termed soft sensor in process engineering to differentiate it from hardware sensors.

A total of 360 data points were collected from a propylene polymerization plant operated in a continuous mode. Eight process variables are selected as the input to the Gaussian process model, including hydrogen concentration, feed rates of two catalysts, feed rates of propylene and hydrogen, reactor temperature, pressure and level measurements. The data is randomly partitioned into training and test sets. Different sizes of training data are used to investigate the impact of the amount of data on the prediction performance. The random partition is repeated 50 times under each situation so that the results are not susceptible to a specific split of the data. The covariance function in Eq. (2) is adopted due to its reported good performance in the literature [17,15], and the conjugate gradient method is used to find the maximum likelihood estimation of the hyper-parameters. Following the common practice, the data is pre-processed to have zero mean and unit standard deviation at each variable before it is used for training a Gaussian process.

Figure 1 depicts the root mean square error (RMSE) of one of the random partitions with training set having 180 data points. Figure 1(a) shows that the 30 models, developed from bootstrap re-sampled training sets, achieve quite different prediction performance. The highest RMSE is 0.233 and the lowest 0.140. In practice the testing outputs, to be predicted, is not available to calculate the RMSE for the selection of the best model. We could select the model with the lowest training error, termed "Train_Best" in the figure, and it gives an RMSE of 0.191. Figure 1(b) shows that bagging, through the product rule as in Eqs. (12) and (13), makes the combined model significantly more accurate and robust. It appears that the combination of five or more models result in very stable predictors. The RMSE for bagging 30 models is 0.128 that is lower than the best single Gaussian process.

Figure 2 compares the prediction and measured (reference) MFR values for the "Train_Best" and "Bagging" models as labeled in Figure 1(a). The figure confirms the superior prediction accuracy of the bagging method to a single Gaussian process model.

(a)



(b)

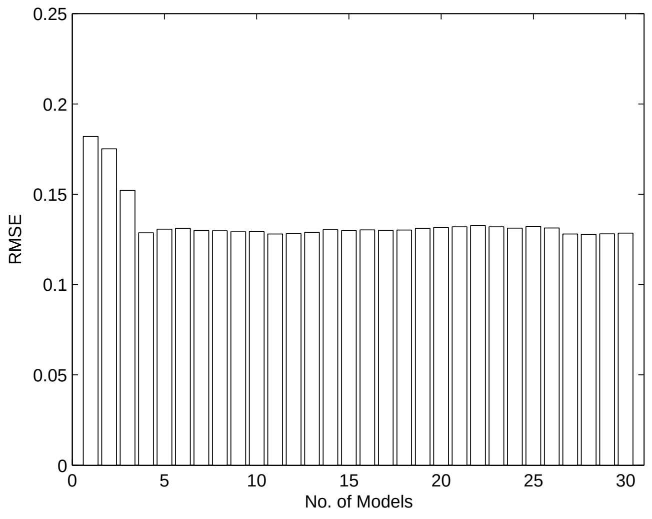


Fig. 1. Prediction errors of (a) single and (b) combined Gaussian process models. In (a) also depicted is the RMSE of bagging 30 models based on the product rule and of the "best" model selected from 30 models to minimize the training errors.

Table 1 summarizes the prediction performance for different training data size, where the average RMSE is calculated based on 50 random partitions of the training and test data. Bagging is performed using 10 bootstrap re-sampled training sets since Figure 1 suggests that the combination of 10 models is sufficient to produce a robust inferential estimation model. The following modeling methods are compared:

- Single: the single model that is developed from the original training data.
- Train_Best: the single model selected from 10 models to minimize the training errors.

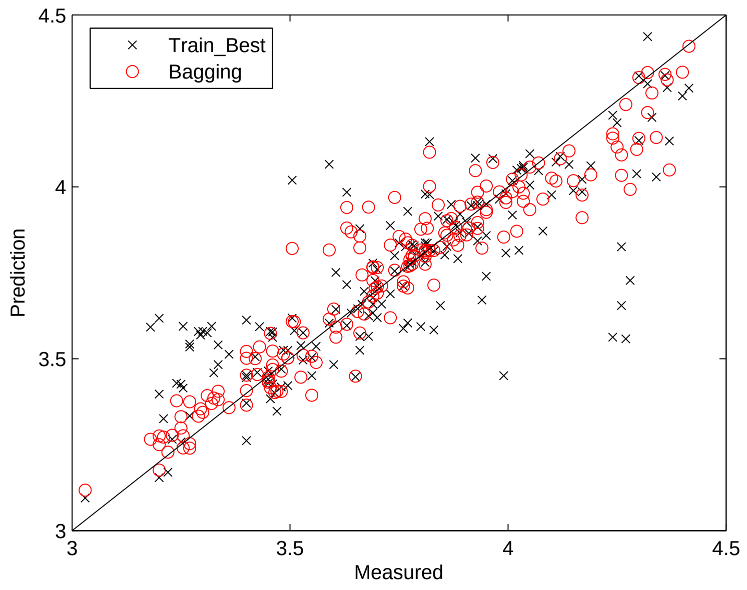


Fig. 2. Prediction vs. measured MFR values.

Table 1
Average RMSE for different training data size.

Training data size	60	120	180	240
Single	0.249	0.216	0.145	0.124
Train_Best	0.250	0.196	0.165	0.141
Bagging_Aveg	0.201	0.161	0.129	0.109
Bagging_Regn	0.202	0.163	0.133	0.117
Bagging_Prod	0.183	0.145	0.115	0.087

- Bagging_Aveg: Bagging 10 models through averaging as in Eqs. (8) and (9).
- Bagging_Regn: Bagging 10 models where the weights are determined using linear regression as in Eq. (10). Principal component analysis is utilized to address the collinearity issue.
- Bagging_Prod: Bagging 10 models where the weights are determined using the product rule as in Eqs. (12) and (13).

The general trend in Table 1 is that the more training data, the better the prediction is. In industrial practice the training data must be sufficient so that the prediction accuracy satisfies the requirement of process monitoring and control. However the cost involved in obtaining the data must be taken into account when deciding the amount of data to be collected. Typically data collection is a recursive procedure where an initial data set is obtained for modeling, and the prediction accuracy is assessed to decide whether more

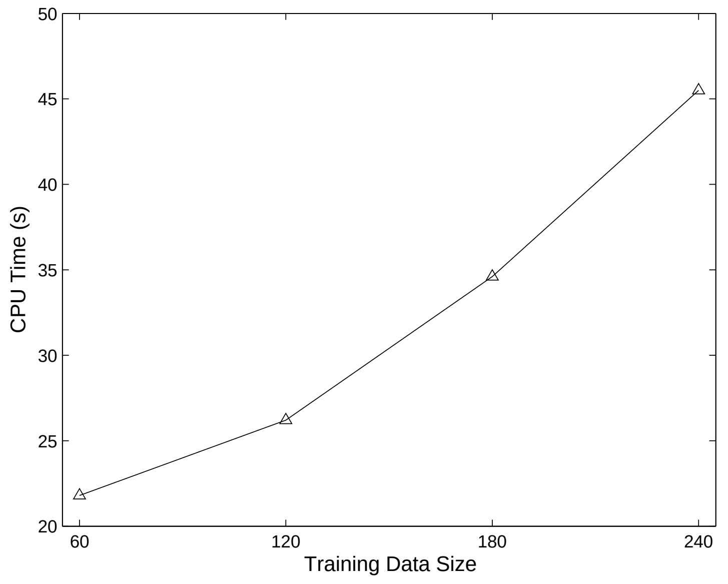


Fig. 3. CPU time for training of 10 models.

data is required.

Table 1 indicates that selecting the "best" model (in terms of minimal training errors) from 10 models (i.e. Train_Best) is not particularly superior to developing a single model from the original training data. In other words the training errors may not be a reliable criterion for model selection. In addition, the results in Table 1 clearly justify the application of bagging: model combination consistently gives lower RMSE in this example, regardless of the size of training data. A comparison between Bagging_Aveg and Bagging_Regn reveals that the weighting strategy based on regression does not outperform the simple averaging method for this data set. Finally the Bagging_Prod method, where the weights are assigned according to the prediction uncertainty, appears to be a promising approach to model combination that attains the lowest RMSE.

Figure 3 illustrates the CPU time (s) for training of 10 Gaussian process models from bootstrap re-sampled data, where the model was implemented within Matlab environment under Windows XP system equipped with a Pentium 2.8 GHz CPU. The values quoted are averaged over 50 partitions of the training and test data. The computational cost increases considerably with the size of training data, yet it is still manageable from a practical perspective. If a large data set is available, the so-called sparse training techniques, mentioned in Section 2.2, should be employed to reduce the training time. Approximately the CPU time for the training of a single model from the original data set is 1/10 (or more generally $1/K$) of that required for the bagging technique. Therefore the improvement in robustness and accuracy of the prediction is at the cost of additional computation.

5 Conclusions and discussions

This paper applies the bagging method to improve the robustness and prediction accuracy of Gaussian process regression models. A novel model combination rule is proposed to assign the weights to models based on their predictive uncertainty. Application study on an industrial example suggested that (1) through bagging the predictions are more reliable and accurate; (2) the proposed model combination strategy significantly outperforms the conventional methods. Therefore we recommend to utilize the bagging procedure to enhance the Gaussian process regression models if the additional computational cost is permitted in practice.

The bagging procedure essentially assumes that the training data are independent and identically distributed. However Gaussian process explicitly models the correlation between data points, which appears to contradict the underlying assumption. As a result, we hypothesize that the success of bagging, as demonstrated by the case study, is due to the fact that the instability of Gaussian process plays a more important role than the violation of the independence assumption in this example. Further theoretical study is needed to investigate this issue, which is an interesting on-going research topic.

References

[1] L. Breiman, Bagging predictors, Machine Learning 26 (1996) 123–140.

[2] D. H. Wolpert, Stacked generalization, Neural Networks 5 (1992) 241–259.

[3] P. Büchlmann, B. Yu, Analyzing bagging, Annals of Statistics 30 (2002) 927–961.

[4] S. Triadaphillou, E. Martin, G. Montague, A. Norden, P. Jeffkins, S. Stimpson, Fermentation process tracking through enhanced spectral calibration modeling, Biotechnology and Bioengineering 97 (2007) 554–567.

[5] R. Gencay, M. Qi, Pricing and hedging derivative securities with neural networks: Bayesian regularization, early stopping, and bagging, IEEE Transactions on Neural Networks 12 (2001) 726–734.

[6] J. Zhang, Inferential estimation of polymer quality using bootstrap aggregated neural networks, Neural Networks 12 (1999) 927–938.

[7] W. Chu, Z. Ghahramani, Gaussian processes for ordinal regression, Journal of Machine Learning Research 6 (2005) 1019–1041.

[8] A. Girard, C. E. Rasmussen, J. Quiñonero-Candela, R. Murray-Smith, Multiple-step ahead prediction for non-linear dynamic systems - a Gaussian process treatment with propagation of the uncertainty, in: S. Becker, S. Thrun, K. Obermayer (Eds.), Advances in Neural Information Processing Systems 15, MIT Press, Cambridge, Massachusetts, 2003, pp. 529–536.

[9] M. Kennedy, A. O'Hagan, Bayesian calibration of computer models (with discussions), Journal of the Royal Statistical Society, B 63 (2001) 425–464.

[10] C. E. Rasmussen, C. K. I. Williams, Gaussian Processes for Machine Learning, MIT Press, 2006.

[11] C. K. I. Williams, C. E. Rasmussen, Gaussian processes for regression, in: D. S. Touretzky, M. C. Mozer, M. E. Hasselmo (Eds.), Advances in Neural Information Processing Systems 8, MIT Press, 1996.

[12] A. O'Hagan, Curve fitting and optimal design for prediction (with discussion), Journal of the Royal Statistical Society B 40 (1978) 1–42.

[13] R. M. Neal, Bayesian learning for neural networks, Springer-Verlag, New York, 1996.

[14] D. J. C. MacKay, Introduction to Gaussian processes, in: C. M. Bishop (Ed.), Neural Networks and Machine Learning, volume 168 of F: Computer and Systems Sciences, NATO Advanced Study Institute, Springer, Berlin, Heidelberg, 1998, pp. 133–165.

[15] R. Kamnik, J. Shi, R. Murray-Smith, T. Bajd, Nonlinear modeling of FES-supported standing-up in paraplegia for selection of feedback sensors, IEEE Transactions on Neural Systems and Rehabilitation Engineering 13 (2005) 40–52.

[16] J. Yuan, K. Wang, T. Yu, M. Fang, Reliable multi-objective optimization of high-speed WEDM process based on Gaussian process regression, International Journal of Machine Tools and Manufacture 48 (2008) 47–60.

[17] T. Chen, J. Morris, E. Martin, Gaussian process regression for multivariate spectroscopic calibration, Chemometrics and Intelligent Laboratory Systems 87 (2007) 59–67.

[18] L. Csató, M. Opper, Sparse on-line Gaussian processes, Neural Computation 14 (2002) 641–668.

[19] V. Tresp, A Bayesian committee machine, Neural Computation 12 (2000) 2719–2741.

[20] B. Efron, Nonparametric estimates of standard error: the jackknife, the bootstrap and other methods, Biometrika 68 (1981) 589–599.

[21] G. McLachlan, D. Peel, Finite Mixture models, John Wiley & Sons, 2004.

[22] S. Borra, A. Di Ciaccio, Improving nonparametric regression methods by bagging and boosting, Computational Statistics and Data Analysis 38 (2002) 407–420.

[23] C. Lu, A. Devos, J. Suykens, C. Arus, S. van Huffel, Bagging linear sparse Bayesian learning models for variable selection in cancer diagnosis, IEEE Transactions on Information Technology in Biomedicine 11 (2007) 338–347.

[24] E. Vigneau, M. F. Devaux, E. M. Qannari, P. Robert, Principal component regression, ridge regression and ridge principal component regression in spectroscopy calibration, Journal of Chemometrics 11 (1997) 239–249.

[25] C. Kiparissides, P. Seferlis, G. Mourikas, A. J. Morris, Online optimizing control of molecular weight properties in batch free-radical polymerization reactors, Industrial and Engineering Chemistry Research 41 (2002) 6120–6131.

[26] T. Chen, J. Morris, E. Martin, Particle filters for state and parameter estimation in batch processes, Journal of Process Control 15 (2005) 665–673.

[27] R. H. K. Entink, J.-P. Fox, B. H. L. Betlem, B. Roffel, Hierarchical process modeling: describing within-run and between-run variations, Journal of Process Control 17 (2007) 349–361.

[28] M. Kano, Y. Nakagawa, Data-based process monitoring, process control, and quality improvement: recent developments and applications in steel industry, Computers and Chemical Engineering 32 (2008) 12–24.