Graph-based Semi-Supervised Regression and Its Extensions

Xinlu Guo
Graduate School of System Informatics
Kobe University
Kobe, Japan 657–8501

Kuniaki Uehara
Graduate School of System Informatics
Kobe University
Kobe, Japan 657–8501

***Abstract*—In this paper we present a graph-based semi-supervised method for solving regression problem. In our method, we first build an adjacent graph on all labeled and unlabeled data, and then incorporate the graph prior with the standard Gaussian process prior to infer the training model and prediction distribution for semi-supervised Gaussian process regression. Additionally, to further boost the learning performance, we employ a feedback algorithm to pick up the helpful prediction of unlabeled data for feeding back and re-training the model iteratively. Furthermore, we extend our semi-supervised method to a clustering regression framework to solve the computational problem of Gaussian process. Experimental results show that our work achieves encouraging results.**

***Keywords*—Semi-supervised learning; Graph-Laplacian; Regression; Gaussian Process; Feedback; Clustering**

I. Introduction

Regression is a fundamental task in data mining and statistical analysis. A regression task aims to analyze and model the relationship between variables so that the value of a given variable can be predicted from one or more other labeled variables. By using enough labeled training data, supervised regression algorithm can learn reasonably accurate model. However, in many machine learning domains, such as bioinformatics and text processing, labeled data is often difficult, expensive and time consuming to obtain. Meanwhile unlabeled data may be relatively easy to collect in practice. For this reason, in recent years, semi-supervised regression has received considerable attention in the machine learning literature due to its potential in utilizing unlabeled data to improve the predictive accuracy [6] [28].

An early semi-supervised regression method is iterative labeling [9], such as co-training algorithm [4][27], which employs supervised regressors as the base learners, then labels and selects unlabeled data in an iterative process. Similarly [5] performed another co-training style semi-supervised regression algorithm by employing multiple learners. Although these methods achieved considerable improvements, they didn't take full advantage of the inherent structure between labeled and unlabeled data. Indeed, they just kept the supervised learning algorithm and changed the form of the labels of data, i.e., they label and relabel the unlabeled data iteratively. Unfortunately, the iterative process causes computational problems for large datasets.

Besides co-training, regularization based method has also been widely employed in the semi-supervised regression [11][15][23][3]. This method combines a regularization term of all labeled and unlabeled data, with the predictive error of labeled data into a criterion. In such a criterion, the unlabeled data can help to get a better knowledge for what parts of the input space that the predictive function varies smoothly in. A variety of approaches using the regularization term have been proposed. Some well-known regularization terms are graph Laplacian regularizer [29], Hessian regularizer [10], parallel field regularizer [14], and so on. These methods have enjoyed a great success. However, they are transductive, which means they only work on the observed labeled and unlabeled training data and can't handle the unseen data.

In this paper, we propose an inductive semi-supervised regression model through incorporating graph prior information into the standard Gaussian process (GP) regression. Our method firstly builds an adjacent graph over all the labeled and unlabeled data. Then we consider the adjacent graph as a prior and incorporate it with the standard GP prior to generate a new GP prior condition on the graph and a graph-based covariance function. From the new conditional prior and the graph-based covariance function, the marginal likelihood and the prediction distribution of semi-supervised GP regression are derived. Since the prediction from the GP model takes the form of a full predictive distribution, the unseen data can also be predicted easily.

Additionally, to further boost the learning performance, we also extend our semi-supervised method to a feedback framework. The early semi-supervised learning methods, such as self-learning [19] and co-training [27], usually make use of a supervised learning algorithm to label and select unlabeled data in an iterative process. And these methods have been proved to be effective in improving the prediction accuracy. Thus, the predictions of the learning process must contain some valuable information, and under some metrics, they can help to construct more accurate model. In other words, when a learning process is performed repeatedly, we gain extra information from a new source: past unlabeled examples and their predictions, which can be viewed as a kind of experience. This kind of experience serves as a new source of knowledge related to the prediction model. The new knowledge provides the possibility of improving the performance of our semi-supervised GP regression. In this paper, to take advantage of such extra information, we also employ a feedback algorithm to pick up the helpful prediction of unlabeled data for feeding

back and re-training the model iteratively.

Furthermore, we empirically demonstrate a further extension of the semi-supervised GPr. GP has the computational problem due to an unfavorable cube scaling ($O(N3)$) during training, where, N is the number of training data. In recent years, many methods have been proposed to address this problem: sparse GP approximation [24][20][12], localized regression [7][17]. In our work, we describe a clustering regression framework in order to bring the scaling down. Specifically, a clustering algorithm is employed as the first step in the process for identifying regions that have similar characteristics. Then for each cluster, a local semi-supervised regression model is built to describe the relationship between inputs and outputs. By partitioning the dataset and learning models locally, the computational cost for each local model is cubic only in the numbers of data points in each cluster, rather than in the entire dataset. As a result, even for large dataset, it can lead to a more favorable training scaling.

This paper is organized as follows: In Section 2, we discuss some related work. In Section 3 we give some preliminaries and a brief overview of the Gaussian process regression. The problem statement and our main theorem, as well as the key models are detailed in Section 4. In Section 5, we lay out an extension algorithm that detects usefull predictions and feeds them into the training set. In Section 6 we experimentally compare our method with the state-of-art approaches and make a detailed discussion. According to the results, we find out a problem of our method, and also describe a clustering regression framework to fixing it. Finally, section 7 concludes our work.

II. Related Work

Our work is closely related to several semi-supervised learning methods. One is the semi-supervised classification method proposed by [21]. We both define a prior for the graph variables and attempt to incorporate it into the standard GP probability framework to derive a posterior distribution of latent variables condition on the graph. However, all their derivations are focused on semi-supervised classification problem but not the regression problem. Thus, we will not discuss in more detail.

Additionally, our work is also similar to Zhang's method of semi-supervised multi-task regression (SSMTR)[26]. It seems that we both construct an adjacent graph and incorporate the prior of this graph with the GP prior to generate a semi-supervised data dependent kernel function that defines over the entire data space. But there are several differences in deed. In Zhang's paper, they proposed a new GP likelihood $\prod_{i=1}^{m} p(y^i|X^i)p(\theta^i)$ for the supervised multi-tasks regression (named SMTR), and then changed the kernel function of the model to the semi-supervised kernel function, to extend their model to the semi-supervised setting. Here the semi-supervised kernel function has also been used in classification task before [22]. Actually, the prediction formulation of SMTR $p(y_*^i|x_*^i, X^i, y^i)$ is the same as standard GP but with different kernel functions. In our paper, we don't simply change the kernel function in a supervised GP, but take advantage of the prior of the adjacency graph to derive a new likelihood condition on the graph $p(y|X, \mathcal{G})$ and a conditional prediction distribution $p(y_*|x_*, X, y, \mathcal{G})$, which is the training model and prediction model for the semi-supervised regression. In other words, the major difference between our method and SSMTR is that the training and prediction models are totally different. Moreover, in Zhang's method, because of the large number of parameters, it is difficult to estimate the optimal values simultaneously. So the parameters are optimized through an alternating optimization algorithm. However, the parameters in our work are estimated by using the gradient descent method to minimize the negative log conditional likelihood $p(y|X, \mathcal{G})$, which means that the training processes of two methods are different.

III. An Overview of Gaussian Process Regression

GP has been proved to be a powerful tool for the purpose of regression. The important advantage of GP is the explicit probabilistic formulation. This not only provides probabilistic predictions but also gives the ability to infer model parameters. Here, we offers a brief summary of GP for supervised regression, see [18] for more details. We assume that the input training data is given as $X_D = \{X_L, X_U\} = \{x_1, \ldots, x_\ell, x_{\ell+1}, \ldots, x_N\}$, where $x_i \in R^d$, N is the total number of input data and ℓ is the number of labeled data. X_L and X_U denote the inputs of labeled and unlabeled dataset respectively. We use $y = \{y_1, \ldots, y_\ell\}$ to represent the corresponding outputs of labeled data X_L.

In supervised GP regression, the corresponding output label y is assumed relating to an latent function $f(x)$ through a Gaussian noise model: $y = f(x) + \mathcal{N}(0, \sigma^2)$, where $\mathcal{N}(m, c)$ is a Gaussian distribution with mean m and covariance c. The regression task is to learn a specific mapping function $f(x)$, which maps an input vector to a label value. Usually, a zero-mean multivariate Gaussian prior distribution is placed over f. That is:

$$\begin{aligned} p(f|X_L) &= \mathcal{N}(0, K_L) \\ &= (2\pi)^{-\frac{\ell}{2}} |K_L|^{-\frac{1}{2}} \exp\left(-\frac{1}{2} f^T K_L^{-1} f\right) \end{aligned} \quad (1)$$

where K_L is an $\ell \times \ell$ covariance matrix. In particular, the element of K_L is built by means of a covariance function (kernel) $k(x, x')$. A simple example is the standard Gaussian covariance defined as:

$$k(x, x') = c \cdot \exp\left(-\frac{1}{2} \sum_{j=1}^{d} b_j \left(x^j - x'^j\right)^2\right), \theta = \{c, b\} \quad (2)$$

where $b = \{b_j\}_{j=1}^d$ plays the role of characteristic length-scales. c is the kernel over scale. The parameters b and c are initially unknown and are added to a parameter set θ, which is defined as containing all such hyper-parameters.

For a GP model, the marginal likelihood is equal to the integral over the product of likelihood $p(y|f) = \mathcal{N}(f, \sigma^2 I)$ and the prior $p(f|X_L)$, given as:

$$p(y|X_L) = \int p(y|f)\ p(f|X_L)\ df = \mathcal{N}(0, K_L + \sigma^2 I) \quad (3)$$

which is typically thought as the training model of GP. Given some observations and a covariance function, we want to find out the most appropriate θ and σ, and make a prediction on

the test data. There are various methods for determining the parameters. A general one is the gradient ascent, which seeks the optimal parameters by maximizing the marginal likelihood.

Given the observations and optimal θ and σ, the prediction distribution of the target value f_* for a test input x_* can be expressed as [18]:

$$p\left(f_* \mid x_*, X_L, y\right) = \mathcal{N}(m_*, c_*) \tag{4}$$

where the predictive mean and variance are:

$$\begin{aligned} m_* &= k_*^T \left(K_L + \sigma^2 I\right)^{-1} y \\ c_* &= k_{**} - k_*^T \left(K_L + \sigma^2 I\right)^{-1} k_* \end{aligned} \tag{5}$$

where k_* is a matrix of covariances between the training data and test data. The matrix k_{**} consists of the covariance of the test data.

IV. Semi-supervised Gaussian Process Regression

As we can see in standard GPr, neither the prior of latent function f (Eq.(1)) nor the predictive distribution (Eq.(4)) contains any information of the unlabeled data. Evidently, to train a accurate GP model, we need to get sufficient training data (labeled data). However, the training data is often difficult and expensive to obtain, while the unlabeled data is relatively easy to collect. Therefore, it appears necessary to modify the standard GP model to make it capable of learning from unlabeled data, and thereby improve the performance of prediction. In this section we present how to effectively use the information of unlabeled data to extend the standard GP model into the semi-supervised framework.

According to semi-supervised smoothness assumption, if two points are close, then so should be the corresponding outputs. Based on this assumption, the unlabeled data should be helpful in regression problem. They can help explore the nearness or similarity between outputs. And the output should vary smoothly with this distance. So, to utilize the unlabeled data, we consider building an adjacent graph to define the nearness between labeled and unlabeled data. Then we attempt to incorporate the graph information into the standard GP probabilistic framework to generate a new probability model for semi-supervised GPr.

A. Prior Condition On Graph

In order to take advantage of the information of unlabeled data, we build an adjacent graph $\mathcal{G} = (V, E)$ on all observed data points $X_D = \{X_L, X_U\}$, to find the adjacent relationship between labeled and unlabeled data, where V is the set of nodes composed by all data points, E is the set of edges between nodes. The graph can be represented by a weight matrix $W = \{w_{ij}\}_{i,j=1}^N$, where $w_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\eta}\right)$ is the edge weight between nodes i and j, with $w_{ij} = 0$ if there is no edge.

From the previous section, we can see that regression by GP is a probabilistic approach. Probabilistic approaches to regression attempt to model $p(y|X_D)$. In this case, in order to make the unlabeled data affect our predictions, we must make some assumptions on the underlying distribution of input data. In our work, we attempt to combine the graph information with the GP. Thus, we focus on incorporating a prior of $p(\mathcal{G}|f)$ with the prior of $p(f|X_D)$ to infer a posterior distribution of f condition on the graph $\mathcal{G}$.

Here, we consider the graph $\mathcal{G}$ itself as a random variable. There are many ways to define an appropriate likelihood of the variable $\mathcal{G}$. [21] provides a simple likelihood of observing the graph:

$$p\left(\mathcal{G}|f\right) \propto \exp\left(-\frac{1}{2} f^T \Delta f\right) \tag{6}$$

where Δ is a graph regularization matrix, which is defined as the graph Laplacian here. We can derive Δ in the following way: let $\Delta = \lambda L^\upsilon$, where λ is a weighting factor, υ is an integer, and L denotes the combinatorial Laplacian of the graph $\mathcal{G}$. Let $D_{ii} = \sum_j w_{ij}$, the combinatorial Laplacian is defined as $L = D - W$.

Combining the Gaussian process prior $p(f|X_D)$ with the likelihood function Eq.(6), we can obtain the posterior distribution of f on the graph $\mathcal{G}$ as follows:

$$p(f|X_D, \mathcal{G}) = \frac{1}{p(\mathcal{G})} p(\mathcal{G}|f) p(f|X_D) \tag{7}$$

Observably, the posterior distribution Eq.(7) is proportional to $p(\mathcal{G}|f)p(f|X_D)$, which is a multivariate Gaussian as follows:

$$p(f|X_D, \mathcal{G}) = \mathcal{N}\left(0, (K_{DD}^{-1} + \Delta)^{-1}\right) \tag{8}$$

The posterior distribution Eq.(8) will be used as the prior distribution for the following derivation. To proceed further, we have to derive the posterior of f_X independent of graph $\mathcal{G}$. Here X denotes the more general dataset, which contains observed dataset X_D and a set of unseen test data X_T, *i.e.*, $X = \{X_D, X_T\}$. In standard GP, the joint Gaussian prior distribution of f_X can be expressed as follows:

$$p\left(f_X|X\right) = \mathcal{N}\left(\left[\begin{array}{c} 0 \\ 0 \end{array}\right], \left[\begin{array}{cc} K_{DD} & K_{DT} \\ K_{DT}^T & K_{TT} \end{array}\right]\right) \tag{9}$$

Then the same as above, the posterior distribution of f_X conditioned on $\mathcal{G}$ is proportional to $p(\mathcal{G}|f_X)p(f_X|X)$, and it is explicitly given by a modified covariance function defined in the following:

$$p\left(f_X|X, \mathcal{G}\right) = \mathcal{N}\left(0, \tilde{K}_{XX}\right) \tag{10}$$

where

$$\tilde{K}_{XX}^{-1} = \left[\begin{array}{cc} K_{DD} & K_{DT} \\ K_{DT}^T & K_{TT} \end{array}\right]^{-1} + \left[\begin{array}{cc} \Delta & 0 \\ 0 & 0 \end{array}\right] \tag{11}$$

Eq.(10) gives a general description that for any finite collection of data X, the latent random variable f_X conditioned on graph $\mathcal{G}$ has a multivariate normal distribution $\mathcal{N}(0, \tilde{K}_X)$, where $\tilde{K}_X$ is the covariance matrix, whose elements are given by evaluating the following kernel function:

$$\tilde{k}\left(x, z\right) = k\left(x, z\right) - k_x^T \left(I + \Delta K\right)^{-1} \Delta k_z \tag{12}$$

in this equation, K is a $N \times N$ matrix of $k\left(\cdot, \cdot\right)$, and k_x and k_z denote the column vector $\left(k\left(x_1, x\right), \ldots, k\left(x_{l+u}, x\right)\right)^T$.

We notice that by incorporating the graph information Δ with the standard GP prior $p(f|X)$, we infer a new prior condition on the graph $\mathcal{G}$ and a graph-based covariance function $\tilde{k}$. In fact this semi-supervised kernel (covariance function) was first proposed by [22] from the Reproducing Kernel Hilbert Space view, and is used for the semi-supervised classification task. In our work, we mainly focus on how to utilize the new prior and the graph-based covariance function to derive the training and predicting distributions for semi-supervised GPr.

B. Objective Functions

Our objective training function for semi-supervised GPr is the marginal likelihood $p(y|X_D, \mathcal{G})$, which is the integral of the likelihood times the prior:

$$p\left(y|X_D, \mathcal{G}\right) = \int p(y|f)\ p(f|X_D, \mathcal{G})\ df \tag{13}$$

Similar with standard GP, the term marginal likelihood refers to the marginalization over the latent function value f. But the difference is that the prior of semi-supervised GP is the posterior obtained by conditioning the original GP with respect to graph $\mathcal{G}$.

According to Eq. 8 and the likelihood $p(y|f) = \mathcal{N}(f, \sigma^2 I)$, the marginal likelihood of the observed target values y is:

$$p\left(y|X_D, \mathcal{G}\right) = \mathcal{N}(0, \Sigma) \tag{14}$$

where $\Sigma = \left(K_{DD}^{-1} + \Delta\right)^{-1} + \sigma^2 I$. This formula can be seen as the training model of our proposed method. We can select the appropriate values of hyper-parameters $\Theta = \{\theta, \sigma\}$ by maximizing the log marginal likelihood $\log p(y|X_D, \mathcal{G})$. The goal is to solve $\hat{\Theta} = \arg\max \log p(y|X_D, \mathcal{G})$. In learning process we seek the partial derivatives of the marginal likelihood, and use them for the gradient ascent to maximize the marginal likelihood with respect to all hyper-parameters.

After learning the model parameters, we are now confronted with the prediction problem. In the prediction process, given a test data x_*, we are going to infer f_* based on the observed vector y. According to the prior Eq.(10) and Eq.(14), the joint distribution of the training output y and the test output f_* is

$$\begin{bmatrix} y \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \Sigma & \tilde{k}_* \\ \tilde{k}_*^T & \tilde{k}_{**} \end{bmatrix}\right) \tag{15}$$

Then we can use this joint probability and Eq.(14) to compute the Gaussian conditional distribution over f_*:

$$p\left(f_*|x_*, X_D, y, \mathcal{G}\right) \propto \exp\left(-\frac{1}{2}\left[y, f_*\right]\begin{bmatrix} \Sigma & \tilde{k}_* \\ \tilde{k}_*^T & \tilde{k}_{**} \end{bmatrix}^{-1}\begin{bmatrix} y \\ f_* \end{bmatrix}\right) \tag{16}$$

By using the partitioned inverse equations, we can derive the Gaussian conditional distribution of f_* at x_*:

$$p\left(f_*|x_*, X_D, y, \mathcal{G}\right) = \mathcal{N}\left(\hat{\mu}, C\right) \tag{17}$$

where

$$\begin{aligned} \hat{\mu} &= \tilde{k}_*^T \Sigma^{-1} y \\ C &= \tilde{k}_{**} - \tilde{k}_*^T \Sigma^{-1} \tilde{k}_* \end{aligned} \tag{18}$$

This is the key predictive distribution for our proposed semi-supervised GPr method. $\hat{\mu}$ is the mean prediction at the new point and C is the standard deviation of the prediction. For fixed data and fixed hyper-parameters of the covariance function, we can predict the test data from the labeled data and a large amount of unlabeled data.

Note that the graph $\mathcal{G}$ contains the adjacent information of labeled and unlabeled data, and it is helpful for regression according to the smoothness assumption of supervised learning. Then, the knowledge on $p(\mathcal{G}|f)$ that we gain through the unlabeled data carries information that is useful in the inference of $p(y|X_D, \mathcal{G})$ and $p(f_*|x_*, X_D, y, \mathcal{G})$, which is the training probability and predictive distribution for semi-supervised GP regression. Thus, our semi-supervised GPr method can be expected to yield an improvement over supervised one.

V. Regression with Feedback

In the semi-supervised regression, we learn a predictive model from labeled and unlabeled data. Then the output of the unlabeled data can be predicted through the model. In this process, predictive output can be viewed as a kind of experience. Such experience provides the possibility of improving the performance of semi-supervised GPr. Therefore, in this paper, we describe a feedback algorithm, which can pick up the useful prediction of unlabeled data for feeding back into the labeled dataset and re-train the model iteratively.

In a predictive system, we can not affirm that all the predictions of unlabeled data could be correctly predicted. For this reason, not all the predictions are helpful for re-training and we need to pick up the useful ones from them. Here we call one useful prediction a confident prediction. Now we have a problem that what the confident prediction is. Intuitively, if a labeled example can help to decrease the error of the regressor on the labeled data set, it should be the confident labeled data. Therefore, in each learning iteration of feedback, the confidence of unlabeled data point x_u can be evaluated using a criterion of:

$$E_{x_u} = \sum_{x_i \in X_L}\left(\left(y_i - M(x_i)\right)^2 - \left(y_i - M'(x_i)\right)^2\right) \tag{19}$$

here, M is the original semi-supervised regressor trained by the labeled dataset (X_L, y_L) and unlabeled dataset X_U, while M' is the one re-trained by the new labeled dataset $\{(X_L, y_L) \cup (x_u, \hat{y}_u\}$ and unlabeled dataset $\{X_U - x_{u'}\}$. Here x_u is an unlabeled data point while $\hat{y}_u$ is the real-valued output predicted by the original regressor M, i.e. $\hat{y}_u = M(x_u)$. The first term of Eq.(19) denotes the mean squared error (MSE) of the original semi-supervised regressor on labeled dataset, and the second term is expressed the MSE of the regressor utilizing the information provided by $(x_u, \hat{y}_u)$ on the labeled dataset. Thus, $(x_u, \hat{y}_u)$ associated with the biggest positive E_{x_u} can be regarded as the most confident labeled data. In other words, If the value of E_{x_u} is positive, it means utilizing $(x_u, \hat{y}_u)$ is beneficial. So we can use this unlabeled data paired with its prediction as labeled data in the next round of model training. Otherwise, $(x_u, \hat{y}_u)$ is not helpful to train models, and will be omitted. Then the x_u should remain in the unlabeled dataset X_U .

TABLE I. ALGORITHM OF FEEDBACK

Input: Labeled dataset(X_L, y_L), Unlabeled dataset X_U,
Learning iterations T, Initial parameters set $InitPara$
Output: Prediction model M
Step1: Training model
$M \leftarrow Semitrain(X_L, y_L, X_U, InitPara)$
Step2: Choosing and feedback
for $t = 1 : T$ **do**
Create pool $X_{U'}$ by randomly selecting data points from X_U
for each $x_u \in X_{U'}$ **do**
$\hat{y_u} \leftarrow M(x_u)$
$M' \leftarrow Semitrain\left((X_L, y_L) \cup (x_u, \hat{y_u}), \{X_U - x_u\}, InitPara\right)$
$E_{x_u} \leftarrow \sum_{x_i \in X_L} ((y_i - M(x_i))^2 - (y_i - M'(x_i))^2)$
end for
for each $E_{x_u} > 0$ **do**
$(X_L, y_L) \leftarrow (X_L, y_L) \cup (x_u, \hat{y_u})$
$X_U \leftarrow \{X_U - x_u\}$
$M \leftarrow M'$
end for
$M \leftarrow Semitrain(X_L, y_L, X_U, InitPara)$
end

The pseudo code of our feedback framework is shown in Table I, where the function $Semitrain$ returns a semi-supervised GP regressor. The learning process stops when the maximum number of learning iterations, T, is reached, or there is no unlabeled data.

VI. EXPERIMENTS

In this section, we firstly evaluate the performance of the proposed semi-supervised GPr (SemiGPr) on some regression datasets, and make a direct comparison to its standard version (GPr). Then we show the experimental results of SemiGPr with the feedback algorithm (named FdGPr). Finally, we introduce the clustering framework, and empirically demonstrate the exclusion time and accuracy of the local SemiGPr extension by this framework.

There are $d + 4$ hyper-parameters in SemiGPr: kernel length-scales $b = \{b_i\}_{i=1}^{d}$, where d is the dimension of input x, kernel over scale c, noise σ and edge weight length-scale η. In our experiment, we select the appropriate values of $\{b, c, \sigma\}$ by maximizing the marginal likelihood. To reduce the computing complexity, we fix $\eta = 10$ for all datasets. 4-fold cross validation is performed on each dataset and all the results are averaged over 40 runs of the algorithm.

The datasets used to evaluate the performance of our method are summarized in Table II. The examples contained in the artificial dataset Friedman is generated from the function: $y = \tan^{-1}\left(x_2 x_3 - 1/x_2 x_4\right)/x_1$. The constraint on the attribute is: $x_1 \sim U[0, 100]$, $x_2 \sim U[40\pi, 560\pi]$, $x_3 \sim U[0, 1]$, $x_4 \sim U[1, 11]$. Gaussian noise term is added to the function. The real-world datasets are from the UCI machine learning repository and StatLib.

In our experiment, for each dataset, we randomly choose 25% of the examples as test data, while the remaining are training data. We take 10% of the training data as labeled examples, and the remaining is used as the set of unlabeled examples. Note that all the datasets are normalized to the range $[0, 1]$.

A. Algorithmic Convergency

In this paper, we estimate hyper-parameters by using the gradient descent method to minimize the following log marginal likelihood.

$$-\log p\left(y|X, \mathcal{G}\right) = \frac{1}{2} y^T \Sigma^{-1} y + \frac{1}{2}\log|\Sigma| + \frac{N}{2}\log 2\pi \quad (20)$$

TABLE II. DATASETS USED FOR SEMIGPR. D IS THE FEATURE; N DENOTES THE SIZE OF THE DATA.

Dataset	Friedman	wine	chscase	no2
D	4	11	6	7
N	3000	1599	400	500
Source	Artificial	UCI	Statlib	Statlib
Dataset	kin8nm	triazines	pyrim	bodyfat
D	8	60	27	14
N	2000	186	74	252
Source	UCI	UCI	UCI	Statlib

Firstly, we discuss the convergence of the above training objective function. In Figure 1, we show how the objective function value decreases as a function of the iterations on triazines (left) dataset and no2 (right) dataset. The result of triazines shows a typical convergence process. As the number of iterations is increasing, the objective function value is decreasing smoothly. Meanwhile, the objective function value of no2 is converged in two stages. From the results, we can see that the objective function value decreases with the increase of the number of iterations and the iterative procedure guarantees a local optimum solution for the objective function in Eq.(20). According to our offline experiments, generally, the objective function converges after about 30-40 iterations for the datasets in Table II.

B. Efficiency of Unlabeled Data

To verify the SemiGPr model can take advantage of unlabeled data, for a fixed number of labeled data, we vary the number of unlabeled examples, and plot the mean squared error (MSE) for dataset triazines and no2. The corresponding curves are shown in Figure 2, where the dotted line and solid line indicate the predictive errors on unlabeled dataset and test dataset respectively. Note that when the proportion of unlabeled data is 0%, the result denotes the MSE of standard GPr. The figure shows that the proposed semiGPr algorithm has lower MSE compared to the standard GPr both on unlabeled and test dataset. Moreover, as the proportion of unlabeled examples increases, the advantage of semiGPr increases further. From this result, we can conclude that SemiGPr may bring extra advantage by utilizing the unlabeled data for model training. In other words, the unlabeled data provides some useful information, and our semi-supervised algorithm can make use of this information to improve the predictive accuracy.

While we observe a significant performance improvement of the proposed algorithm by using unlabeled examples, the unlabeled examples are not always helpful. For example, for data no2 (right figure of Figure 2), when the proportion of unlabeled data goes from 30% to 50%, the error rates are increased instead of reduced. The same happens to triazines when the size of unlabeled dataset goes from 90% to 100%. It is of interest to find out the cause of the negative effect of the unlabeled data experimentally in the future.

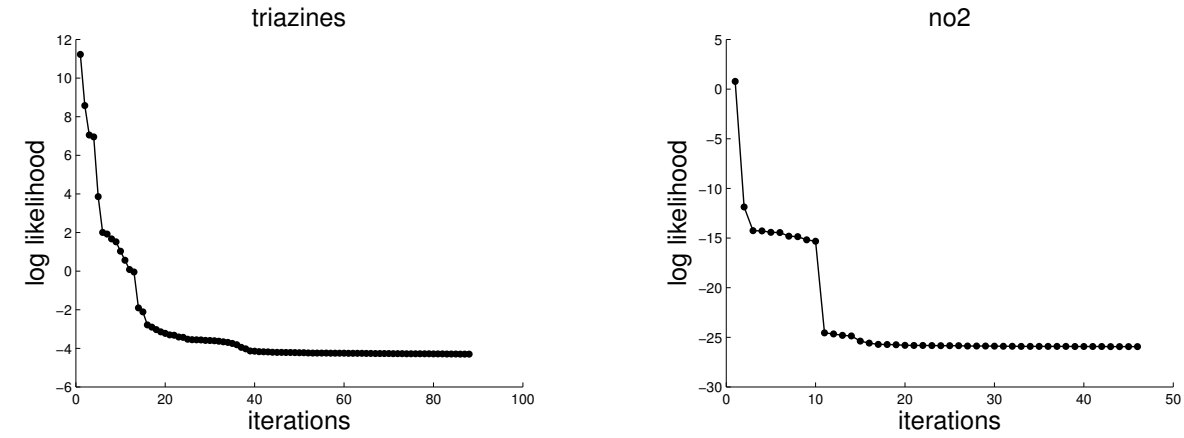

Fig. 1. log likelihood decreases along with the increase of the iteration No. for the triazines (left) and the no2 (right).

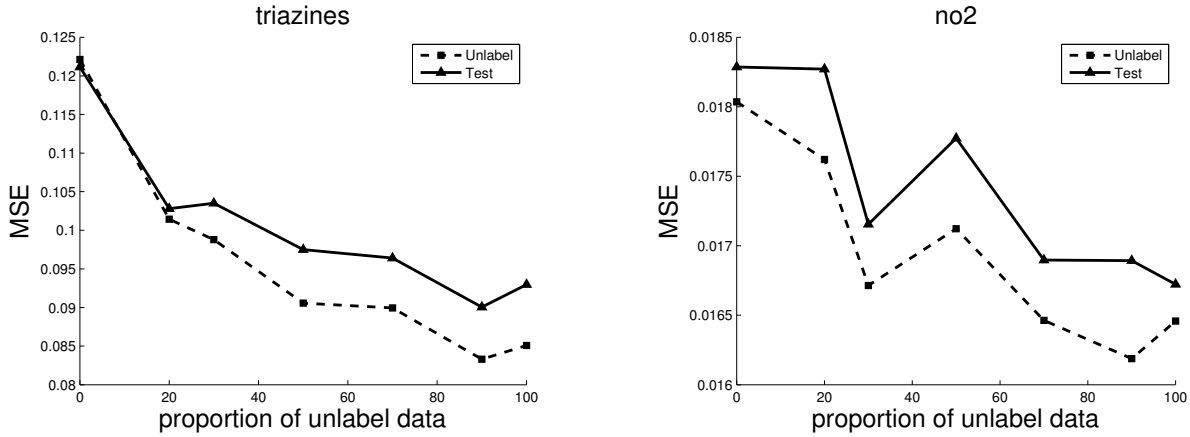

Fig. 2. Performance of SemiGPr as a function of number of unlabeled examples.

C. Evaluation of Regression Accuracy

To further clarify the effect of the proposed method, we compare the MSE between SemiGPr and GPr. The comparative results are summarized in Table III. The above value is the performance on unlabeled dataset, and the following value is on test dataset. In this experiment, we consider GPr as the baseline and compare the performance of SemiGPr with it. The improvements are also listed in the table. In addition to the average MSE, we test the significance of the performance difference between SemiGPr and GPr using a paired t-test on the MSE values. The differences are significant with a paired t-test at the 0.05 level, and the results with significant improvement in the table are bold-faced.

The result in Table III shows that our method SemiGPr performs as well as or better than the standard GPr in terms of the regression accuracy. We can observe that SemiGPr leads to improvements in most of the datasets, and the differences are significant in about half of the datasets. From the comparison, we can conclude that using the unlabeled training data with our semi-supervised regression framework, the GP regression accuracy can be improved. On some of the datasets like chscase, the precision of SemiGPr did not have a significant improvement over the standard one. There are two possible reasons for this result. One is the poor hyper-parameter choices made in optimization process. The other one is the negative effect of the unlabeled data as shown in the previous experiment.

D. Comparison with other methods

To further evaluate the performance of SemiGPr, we compare our results with other semi-supervised regression methods. In the first experiment, the co-training method (COREG) presented in [**?**] is compared. The code and documentation of COREG are available at $http : //lamda.nju.edu.cn/code/\_COREG.ashx$. All the experimental setting of COREG is the same as that of SemiGPr, i.e., the same splitting of the training and testing sets and preprocessing methods, and 40 randomly runs of the algorithm for each dataset. The obtained results are summarized in Table IV. We perform a paired t-test at the 5% significance level, and the results with obvious improvement in the table are bold-faced. In general, we observe that our method achieves a smaller error on all of the datasets compared to COREG. In particular, on Wine and kin8nm datasets, we observe a significant performance improvement of the MSE over COREG. It confirms the conclusion that our semi-supervised method can take advantage of the unlabeled data and it is effective even when only a limited amount of labeled data is available.

In the second experiment, in order to illustrate the difference between our method and the one proposed in [26],

TABLE III. COMPARISON OF SEMIGPR WITH THE STANDARD GPR ON DIFFERENT DATASETS.

Dataset	Friedman	Wine	chscase	no2	kin8nm	bodyfat	pyrim	triazines
GPr	0.0113	0.0196	0.0273	0.0180	0.0136	0.0026	0.0524	0.1215
	0.0114	0.0205	0.0268	0.0183	0.0134	0.0061	0.0544	0.1205
SemiGPr	0.0101	0.0190	0.0264	0.0161	0.0131	0.0026	0.0359	0.0843
	0.0102	0.0199	0.0265	0.0164	0.0132	0.0027	0.0495	0.0925
Improv.	10.62%	**3.06%**	3.30%	**10.56%**	**3.68%**	0%	**31.49%**	**30.62%**
	10.53%	**2.93%**	1.12%	**10.38%**	1.49%	55.74%	9.01%	**23.24%**

TABLE IV. COMPARISON OF SEMIGPR WITH CO-TRAINING METHODS.

Dataset	Friedman	Wine	chscase	no2	kin8nm
SemiGPr	0.0102	**0.0199**	0.0265	0.0164	**0.0132**
COREG	0.0115	0.0214	0.0282	0.0166	0.0190

we run experiments on exactly the same datasets of [26], following precisely their preprocessing and testing methods, where in Robot arm dataset, 2000 data points are selected independently for each task, with 1% as labeled data, 10% as unlabeled data and the remaining as test data, and in School dataset, for each task, 2% of the data is selected as labeled data, 20% as the unlabeled data and the rest as test data. Here, we use the systematic sampling method as the selection method. The normalized mean squared error, which is defined as the mean squared error divided by the variance of the test output, is calculated as the performance measure. The results are averaged over 10 runs of the algorithm. In Table V we report the test normalized mean squared error for two multi-task regression datasets.

It turns out that Zhang's method of SSMTR uses a very similar idea: constructing an adjacency graph and incorporating the prior of the graph with the GP prior to generate a semi-supervised data-dependent kernel function. Actually we derived the marginal likelihood and predictive distribution of the GP from different routes. As we discussed earlier, the major difference between two methods is that we have totally different training and prediction models.

From the results of this experiment in Table V we can obtain the merits and demerits of these two different models. The result shows that SSMTR achieves a smaller error on robot arm dataset compared to our method because Zhang's method considers the relevance among tasks. Their model consists of GP and a common prior on the parameters for all tasks, and the common prior can model the relevance well. The robot arm dataset contains 7 tasks which are 7 joint toques of the robot arm. The 7 joint toques have strong association with each other, which means the 7 tasks have high relevance. Therefore, for such a multi-task regression, it's better to learn a multi-task model rather than building a single-task model for each task independently. However, for the second dataset (School score), although the tasks have some relevance with each other, our method still performs as well as the multi-task method SSMTR. In this dataset, the examination scores of students between different schools are related with the difficulty of the examination. In Zhang's paper, to model this latent relevance, they impose a common Gaussian prior $\theta^i \sim \mathcal{N}(m_\theta, \Sigma_\theta)$ on the kernel parameters for all tasks. On the other hand, our

TABLE V. COMPARISON OF SEMIGPR WITH SSMTR (SSRT:SUPERVISED SINGLE-TASK REGRESSION WHICH USES ONE GP FOR EACH TASK).

Method	SSTR	SSMTR	SemiGPr
robot arm	1.0228 ± 0.1318	0.3810 ± 0.1080	0.8389
	1.0270 ± 0.1450	0.3905 ± 0.1123	0.8710
school	1.2914 ± 0.3146	1.0506 ± 0.2804	1.3266
	1.3240 ± 0.3274	1.0612 ± 0.2813	1.0723

method is proposed for single task and can not model this relevance well. However, the common prior has no effect on a single task. Because when there is only one task in a dataset, the common prior becomes a fixed value. Therefore, it may be interesting in the future to compare which performs better for single-task.

Another major difference between two methods lies in hyper-parameter optimization. In Zhang's method, the number of parameters to estimate is large, since all the tasks are modeled in one formulation, and the number of parameters increases with the tasks. Because of the large number of parameters, it is difficult to estimate the optimal values simultaneously. So the parameters are optimized through an alternating optimization algorithm. And this could cause a computational problem for large multi-tasks datasets. However, in our work, the parameters of each task are estimated separately by maximizing the log-likelihood. Therefore, our work can be parallelized easily for a multi-task. It will also be interesting in the future to compare which performs better for hyper-parameter optimization and which saves training time.

E. Results of Feedback Algorithm

In this part, two of the datasets used in SemiGPr are presented to demonstrate the effectiveness of the SemiGPr extended by the feedback algorithm, which is denoted by FdGPr. Experimental setting is the same as the previous subsection.

To clarify unlabeled examples and their predictions really contain some valuable information and our feedback algorithm can utilize such information to improve the predictive accuracy, we plot the MSE of FdGPr for different iteration numbers. The results are shown in Figure 3. The dot line denotes the MSE on the unlabeled dataset, and the solid line is the result of the test dataset. The left figure is the result of dataset no2 and the right one is that for chscase. Note that when the feedback iteration is 0, the result denotes the MSE of SemiGPr.

From the figures we can see that when the iteration number is increased, the feedback algorithm cuts the error

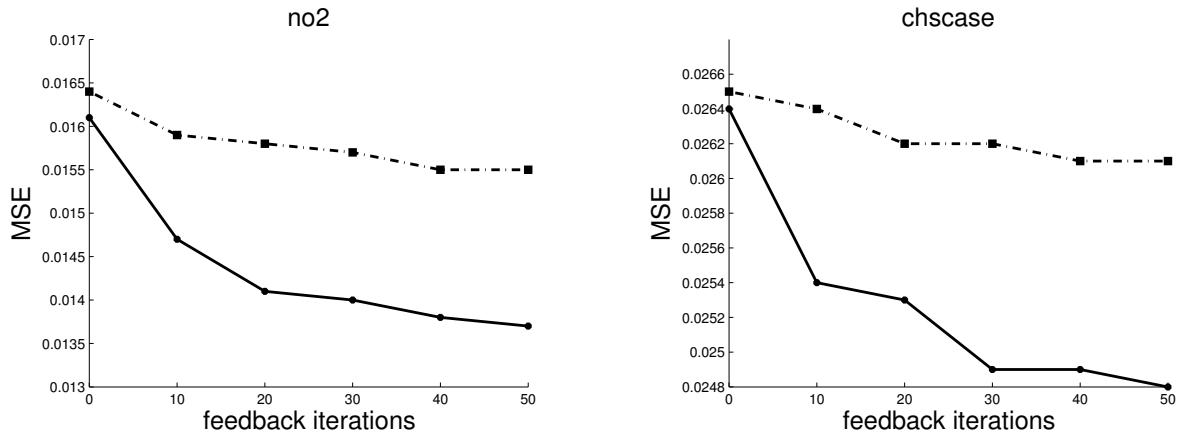

Fig. 3. The effect of different feedback iterations on unlabeled and test dataset.

rate drastically over SemiGPr. The results show clearly that the unlabeled examples and their predictions have a beneficial effect on model learning. From the experiment, larger iteration number almost always produces better results, while considering the computational cost, the iteration T should be set to 20. Although FdGPr achieves a comparable performance to a non-feedback baseline on the unlabeled dataset, it does not have a significant improvement over the other ones on the test dataset. Therefore we should point out that the feedback algorithm makes our work transductive, and we should find a new metric to select predictions of unlabeled examples to improve the performance on test dataset in future work. From this result, we can make a conclusion that by utilizing feedback information, FdGPr makes performance improvements over the other methods, especially on unlabeled data.

F. Extension by Clustering Regression Framework

A significant problem with GP is that it's computationally expensive to carry out the necessary matrix computation ($O(N3)$). To address this problem, we give a further extension of the SemiGPr by a clustering regression framework and discuss the possibility of improving the efficiency while keeping the accuracy.

Indeed, some regression methods with similar idea have already been proposed. For example, [25] proposed a regression clustering algorithm to solve the complex distribution regression problem. The proposed algorithm updates the data in each cluster by using a regression error. Then the clusters and the corresponding regression functions can be obtained simultaneously. This method is effective for the dataset that has multiple tasks within it. In addition, in [13] and [8], excellent results have been obtained on some specific datasets by clustering the input data into several parts, and learning a regression model inside each cluster. In these studies, the accuracy of combining the clustering and regression has been discussed for the specific dataset, such as mixture distribution dataset and multiple spatial dataset, while we focus on empirically demonstrating the execution time and accuracy on general regression dataset. Besides, in above studies supervised approaches have been used for regression, but in our work we take advantage of the proposed semi-supervised regression.

Our clustering regression framework consists of three stages: 1) data partition, 2) training models and 3) output prediction. The pseudo code for the different phases of this framework is shown in Table VII.. The first step is to partition the input data into several clusters. General clustering methods, for example k-means, divide the data into distinct clusters, where each data point belongs to exactly one cluster. However, this constraint is prone to cause unsuitable clustering results in the boundary areas among different clusters. Consequently, in this paper data partition is performed using a soft clustering model named fuzzy c-means clustering (FCM) [2].

TABLE VI. ALGORITHM OF FUZZY C-MEANS CLUSTERING

Input: $X = \{X_L, X_U\}$, P_{init}, $Params(m, \epsilon, A)$
Output: P, C
Repeat
Step1: Compute the centroid of cluster
for each cluster j, $1 \leq j \leq K$ **do**

$$c_j^{(t)} \leftarrow \frac{\sum_{i=1}^{N}(p_{ij}^{(t-1)})^m x_i}{\sum_{i=1}^{N}(p_{ij}^{(t-1)})^m} \quad \text{(a)}$$

Step2: Calculate the distances from data to center
for each data point x_i, $1 \leq i \leq N$ **do**

$$D_{ij}^2 \leftarrow (x_i - c_j)^T A (x_i - c_j),\ 1 \leq j \leq K \quad \text{(b)}$$

Step3: Update the partition matrix

$$p_{ij}^{(t)} \leftarrow \frac{1}{\sum_{k=1}^{K}(D_{ij}/D_{ik})^{2/(m-1)}} \quad \text{(c)}$$

Until $\| P^{(t)} - P^{(t-1)} \| < \epsilon$

The FCM algorithm attempts to partition a finite collection of N elements $X = \{X_L, X_U\} = \{x_1, \ldots, x_N\}$ into a collection of K fuzzy clusters with respect to some given criterion. Here, the X_L and X_U denote the labeled input set and the unlabeled input set separately. Given a finite set of data X, the algorithm returns a list of K cluster centers $C = \{c_1, \ldots, c_K\}$ and a $N \times K$ partition matrix $P = p_{ij} \in [0, 1]$, $i = 1, \ldots, N$, $j = 1, \ldots, K$, where each element p_{ij} can be interpreted as the probability that the element x_i belongs to cluster j.

Table VI provides the learning process of FCM. Given X and the initial partition matrix P, the FCM algorithm first computes the cluster centroid c_i for each cluster, using the formula Eq.(a) in Table VI. The centroid of a cluster is the mean of all points, weighted by their degree of belonging to the

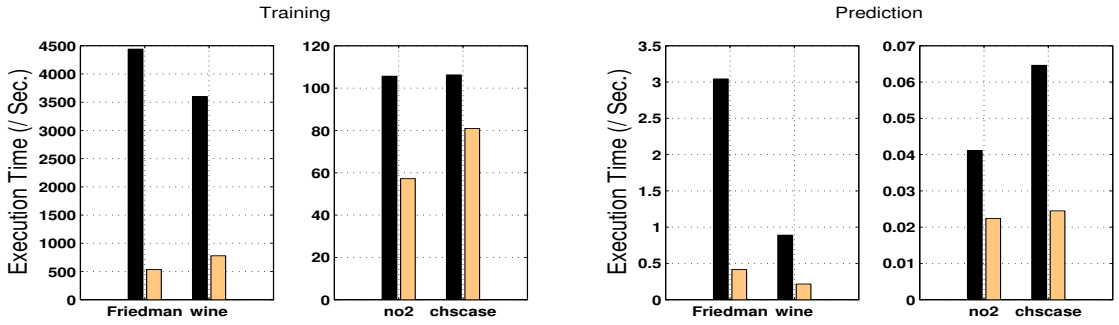

Fig. 4. Comparison of execution time

TABLE VII. PSEUDO CODE OF CLUSTERING FRAMEWORK

Input: X, y_L, x_t, P_{init}, $Params$, $hypara$
Output: P, C, M
Step1: Data Partition
$[P, C] = FuzzyCmeans(X, P_{init}, Params)$
Step2: Training Model
for each cluster $j(j = 1 : K)$ **do**
$M_j \leftarrow Semitrain(X^j, y_L^j, hypara)$
end for
Step3: Output Prediction
Compute the partition coefficient p_t of x_t
$w_{tj} \leftarrow p_{tj} / \sum_{p_t > \delta} p_t$
$\hat{y}_t \leftarrow \sum_{x_t \in j} w_{tj} * M_j(x_t)$

cluster. Then it calculates the distances D_{ij} from data point x_i to the cluster center c_j. Finally, for each data point, it updates the coefficients of being in the clusters. It is repeated until the convergence condition is satisfied.

Through the algorithm presented in Table VI, we can get the partition matrix P, where the rows indicate the probabilities that the data point x belongs to each cluster, and the columns denote the probabilities that all the data points X are partitioned into cluster j. But the final goal of clustering is to calculate an indicator matrix $Z = z_{ij} \in \{0, 1\}$, $i = 1, \ldots, N$, $j = 1, \ldots, K$. Here, the z_{ij} is one if x_i is assigned to the corresponding cluster j, or zero otherwise. The general idea of obtaining Z from P is that setting the maximum probability of each row to be 1, and the rest are 0. But it becomes the same as hard clustering to some extent. To gain fuzzy clusters, in this paper we exploit a threshold δ to filter the partition matrix. For example, set δ to be 0.4, then the corresponding z_{ij} is one if $p_{ij} > 0.4$, and zero otherwise. Thus, for a data point, it may not be assigned to only one cluster but to the clusters that have probability bigger than δ. Through adjusting the threshold δ, we can control how much clusters may overlap.

Several clusters with overlaps are obtained by the data partition step. Then in training step, local semi-supervised GPr model M (Eq.14) is trained for each cluster. Finally in predicting step, the prediction of a given data point x_t is a weighted combination of the predictions of the individual local models given by $\hat{y}(x_t) = \sum_{j=1}^{K} P_{x_t}(j) * M_j(x_t)$, where $P_{x_t}(j)$ denotes the weight of the model j. It effectively equals to the partition matrix that tells the probability of element x_t belonging to cluster j. And $M_j(x_t)$ represents the prediction of input x_t by using the model M_j, the formula of which is Eq. 18.

TABLE VIII. RESULTS OF CLUSTERING REGRESSION FRAMEWORK.

Dataset	Friedman	Wine	chscase	no2
SemiGPr	0.0101	0.0190	0.0264	0.0161
	0.0102	0.0199	0.0265	0.0164
CSGPr	0.0106	0.0220	0.0256	0.0154
	0.0107	0.0232	0.0261	0.0160

To evaluate the performance of the clustering regression framework (named FCMGPr), we experiment on the datasets described in Table II. The parameters chosen for the FCM algorithm remain unchanged for each dataset, $m = 2$, $\epsilon = 10^{-6}$, and A is a diagonal matrix with size $d \times d$, where d is the dimension of data X. Here, we set the threshold δ to be 0.4. And if the data points in a dataset are more than one thousand, then the number of clusters K equals to 4, if not, $K = 2$. The experimental setting of regression part is the same as the evaluation of SemiGPr.

Firstly, we compared the accuracy between SemiGPr and FCMGPr. The MSE results on the unlabeled and test data are shown in Table VIII. From the results we can see that the FCMGPr performs better than the semi-supervised one on no2 and chscase datasets. One of the reasons is that the local models are more flexible than a global one. In other words, constructing model locally can capture the details of data better than applying a global model across entire dataset. In addition, making predictions by weighted combination can help to avoid the inaccurate results due to incorrect clustering. However, there is a problem with clustering regression framework: if the amount of labeled data is too small in a cluster, particularly high-dimensional data, it will be easy to make poor hyper-parameter choices or occur under-fitting. Then it results in bad accuracy for FCMGPr. This is an explanation of the results on wine and Friedman datasets, where the local model did not have an improvement over the semi-supervised one.

Secondly, we tested the exclusion time of SemiGPr and FCMGPr. The results are shown in Figure 4, where the left two figures are the comparison of training time and the right two figures show the results of predicting time. As we can see in Figure 4, for all four datasets, both the training time and the predicting time of FCMGPr are reduced over SemiGPr. From these results we can conclude that by using the clustering regression framework, the computational efficiency of the semi-supervised GPr can be greatly improved and it also has the possibility for improving the prediction accuracy.

VII. Conclusion

In this paper we presented and evaluated a semi-supervised GPr by incorporating an adjacent graph within the standard GP probabilistic framework. Through exploring the standard GP to semi-supervised setting, we can learn a regression model from only small number of expensive labeled data and a large amount easily obtained unlabeled data. Moreover, we presented a feedback algorithm, which can choose the confident prediction for feedback to further improve the performance. Furthermore, to solve the computational problem of GP, we also gave a further extension of the semi-supervised GPr by a clustering regression framework. The experimental results indicate that our semi-supervised regression approach can improve the prediction accuracy. Besides, by choosing the confident prediction for feedback, it brings a significant improvement in the prediction accuracy over a non-feedback baseline. The extension by the clustering regression framework is successful in reducing the exclusion time.

In the experiments, we compared SemiGPr with some state-of-art methods. There also exist some other semi-supervised regression methods, such as regularization regression method [14], propagable graph method [16]. However, because of the different experimental settings, we could not compare the proposed method with them. Future work should include implementing these methods and empirical comparisons with them. We will also apply our scheme to harder regression tasks. Although the results of feedback extension were encouraged, it is noted that the algorithm has high time complexity due to the re-training of SemiGPr. Therefore, in the future work a new feedback criterion would need to be explored in order to obtain more accurate predictions but spending less computational cost.

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[2] Balasko, B., Abonyi, J., and Feil, B., *Fuzzy clustering and data analysis toolbox*, Department of Process Engineering, University of Veszprem, Veszprem, Computer software manual, 2005.

[3] Belkin, M., Niyogi, P., and Sindhwani, V., *Manifold regularization: A geometric framework for learning from labeled and unlabeled examples*, Journal of Machine Learning Research, 7, 2399–2434, 2006.

[4] Blum, A., and Mitchell, T., *Combining labeled and unlabeled data with co-training*, In Proceedings of the 11th Annual Conference on Computational Learning Theory, pp. 92–100, 1998.

[5] Brefeld, U., Gärtner, T., Scheffer, T., and Wrobel, S., *Efficient co-regularised least squares regression*, In Proceedings of the 23rd International Conference on Machine Learning, pp. 137–144, 2006.

[6] Chapelle, O., Schölkopf, B., and Zien, A., *Semi-supervised learning*, MIT Press, 2006.

[7] Chen, T., and Ren, J., *Bagging for gaussian process regression*, Neurocomputing, 72(7-9), pp. 1605–1610, 2009.

[8] Das, K., and Srivastava, A. N., *Block-gp: Scalable gaussian process regression for multimodal data*, In Proceedings of the 10th IEEE International Conference on Data Mining, pp. 791–796, 2010.

[9] Hanneke, S., and Roth, D., *Iterative labeling for semi-supervised learning*, Tech. Rep. No. UIUCDCS-R-2004-2442, Computer Science Department, University of Illinois at Urbana-Champaign, 2004.

[10] Kim, K. I., Steinke, F., and Hein, M., *Semi-supervised regression using hessian energy with an application to semi-supervised dimensionality reduction*, In Proceedings of the 23th Annual Conference on Neural Information Processing Systems, pp. 979–987, 2009.

[11] Lafferty, J., and Wasserman, L., *Statistical analysis of semi-supervised regression*, In Proceedings of the 21th Annual Conference on Neural Information Processing Systems, pp. 801–808, 2007.

[12] Lawrence, N. D., Seeger, M., and Herbrich, R., *Fast sparse gaussian process methods: The informative vector machine*, In Proceedings of the 16th Annual Conference on Neural Information Processing Systems, pp. 609–616, 2002.

[13] Lazarevic, A., Pokrajac, D., and Obradovic, Z., *Distributed clustering and local regression for knowledge discovery in multiple spatial databases*, In Proceedings of the 8th European Symposium on Artificial Neural Networks, pp. 129–134, 2000.

[14] Lin, B., Zhang, C., and Xiaofei, H., *Semi-supervised regression via parallel field regularization*, In Proceedings of the 25th Annual Conference on Neural Information Processing Systems, pp. 433–441, 2011.

[15] Luo, J., Chen, H., and Tang, Y., *Analysis of graph-based semi-supervised regression*, In Proceedings of the 5th International Conference on Fuzzy Systems and Knowledge Discovery, pp. 111–115, 2008.

[16] Ni, B., Yan, S., and Kassim, A. A., *Learning a propagable graph for semisupervised learning: Classification and regression*, IEEE Transactions on Knowledge and Data Engineering, 24(1), pp. 114–126, 2012.

[17] Rasmussen, C. E., and Ghahramani, Z., *Infinite mixtures of gaussian process experts*, In Proceedings of the 16th Annual Conference on Neural Information Processing Systems, pp. 881–888, 2002.

[18] Rasmussen, C. E., and Williams, C. K. I., *Gaussian processes for machine learning*, MIT Press, 2006.

[19] Rosenberg, C., Hebert, M., and Schneiderman, H., *Semi-supervised self-training of object detection models*, In Proceedings of the 7th IEEE Workshops on Application of Computer Vision, pp. 29–36, 2005.

[20] Seeger, M., Williams, C. K., and Lawrence, N. D., *Fast forward selection to speed up sparse gaussian process regression*, In Proceedings of the 9th International Workshop on Artificial Intelligence and Statistics, 2003.

[21] Sindhwani, V., Chu, W., and Keerthi, S. S., *Semi-supervised gaussian process classifiers*, In Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 1059–1064, 2007.

[22] Sindhwani, V., Niyogi, P., and Belkin, M., *Beyond the point cloud: from transductive to semi-supervised learning*, In Proceedings of the 22nd International Conference on Machine Learning, pp. 824–831, 2005a.

[23] Sindhwani, V., Niyogi, P., and Belkin, M., *A co-regularization approach to semi-supervised learning with multiple views*, In Proceedings of the ICML Workshop on Learning with Multiple Views, pp. 74–79, 2005b.

[24] Snelson, E., and Ghahramanin, Z., *Sparse gaussian processes using pseudo-inputs*, In Proceedings of the 19th Annual Conference on Neural Information Processing Systems, pp. 1257–1264, 2005.

[25] Zhang, B., *Regression clustering*, In Proceedings of the 3rd IEEE International Conference on Data Mining, pp. 451–458, 2003.

[26] Zhang, Y., and Yeungn, D. Y., *Semi-supervised multi-task regression*, In Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 617–631, 2009.

[27] Zhou, Z. H., and Li, M., *Semisupervised regression with cotraining-style algorithms*, IEEE Transactions on Knowledge and Data Engineering, 19(11), pp. 1479–1493, 2007.

[28] Zhu, X., *Semi-supervised learning literature survey*, Tech. Rep. No. 1530, Computer Sciences, University of Wisconsin-Madison, 2005.

[29] Zhu, X., Ghahramani, Z., and Lafferty, J. D., *Semi-supervised learning using gaussian fields and harmonic functions*, In Proceedings of the 20th International Conference on Machine Learning, pp. 912–919, 2003.