Chemometrics and Intelligent Laboratory Systems 147 (2015) 75–85

Contents lists available at ScienceDirect

Chemometrics and Intelligent Laboratory Systems

journal homepage: www.elsevier.com/locate/chemolab

Co-training partial least squares model for semi-supervised soft sensor development

Liang Bao, Xiaofeng Yuan, Zhiqiang Ge *

State Key Laboratory of Industrial Control Technology, Institute of Industrial Process Control, College of Control Science and Engineering, Zhejiang University, Hangzhou, 310027, PR China

ARTICLE INFO

Article history:
Received 19 January 2015
Received in revised form 21 July 2015
Accepted 2 August 2015
Available online 7 August 2015

Keywords:
Soft sensor modeling
Semi-supervised learning
Co-training strategy
Partial least squares

ABSTRACT

Typically, the easy-to-measure variables are used to predict the hard-to-measure ones in soft sensor modeling. In practice, however, the easy-to-measure variables are redundant while the other ones are quite rare, which are often obtained from offline lab analyses. In this paper, the semi-supervised learning method is introduced for soft sensor modeling. Particularly, the co-training strategy is combined with the conventionally used partial least squares model (PLS). A co-training styled algorithm called co-training PLS is proposed for the development of a semi-supervised soft sensor. By splitting the whole process variables into two different parts, two diverse PLS regression models can be developed. Through an iterative learning procedure, the final new labeled data sets can be determined, based on which two new regressors are constructed for soft sensing. Two examples are provided for performance evaluation of the proposed method, with detailed comparative studies to the traditional PLS and co-training *k*NN model based soft sensors.

1. Introduction

Successful implementation of advanced monitoring and control techniques highly depends on process models as well as accuracy and reliability of measurements in industrial processes [1]. In particular, real-time analyses of key performance indicators have a huge impact on advanced monitoring and control. However, inadequacy of measurement techniques and low reliability of measuring devices bring a lot of constrains on the on-line measurement of quality variables. As a result, those key performance indicators are normally determined by offline sample laboratory analysis or on-line product quality analyzers, both of which are often expensive and require frequent and high-cost maintenance. In practice, such limitations may cause a severe influence on the quality of products, control of waste, and safety of operations. Recently, large amounts of data have been measured and stored in the industry process to build predictive models for quality estimation and control. Such predictive models devoted to producing real-time estimates of desired plant variables can help to overcome measuring device restriction, improve system reliability, and develop tight control policies [2].

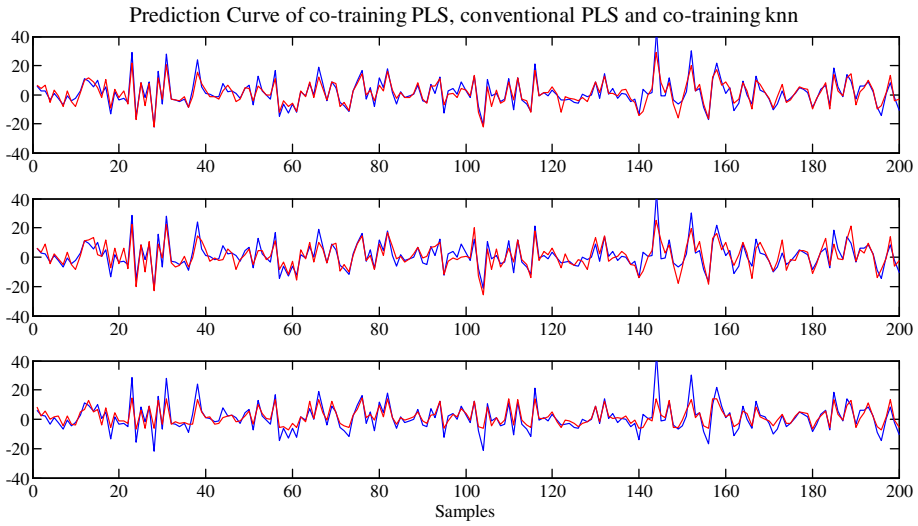
Particularly, soft sensors use easy-to-measure process variables to predict hard-to-measure ones, which have now been widely used in industrial processes [3–5]. Compared to the soft sensors based on the first-principle model, data-driven soft sensors provide more flexible design procedures and can be used in various industrial processes [6–8]. For development of a data-based soft sensor, a training data set containing both input and output variables should be incorporated. While those input variables corresponding to easy-to-measure process variables are routinely recorded, the output variables corresponding to hard-to-measure variables are expensive or difficult to obtain. In this case, there are imbalanced numbers of input and output data for soft sensor modeling. In this paper, the data that have both input and output variables are denoted as labeled data, while those data only contain input variables are called unlabeled data.

In machine learning area, a model trained based on labeled data is known as supervised learning, while unsupervised learning only incorporates unlabeled data. For a typical soft sensor predictive modeling, particularly, it is an obvious supervised learning which need both input and output data. However, industrial processes may only provide a limited number of labeled data samples due to cost, time, or limitations of other resources. In this situation, the performance of soft sensor cannot be well guaranteed under the circumstance that only a limited number of labeled samples are used for model training. Therefore, additional unlabeled data samples are incorporated in the model training of a supervised model to improve the predictive performance, which is known as semi-supervised learning in machine learning area. Distinguished from both supervised and unsupervised learning methods, semi-supervised learning is able to exploit both unlabeled and labeled data samples in the predictive model [9,10].

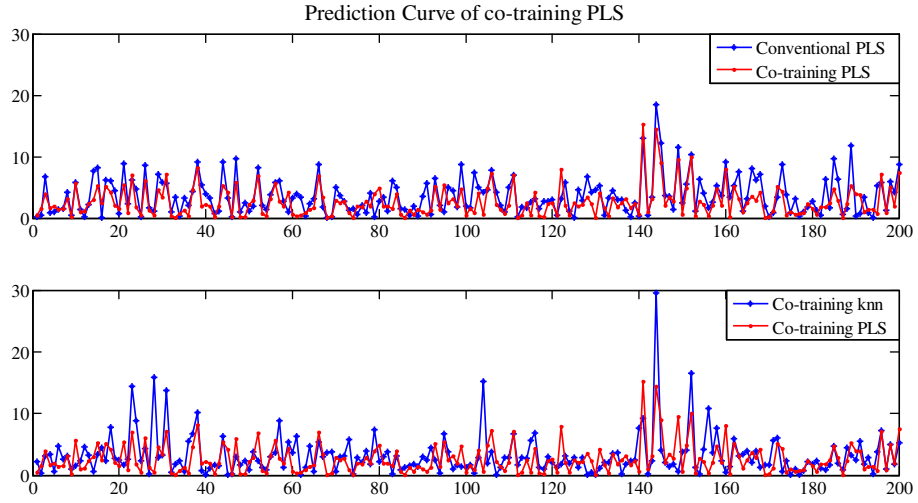
Semi-supervised learning methods have been intensively researched and applied in various areas in the past years [11–13]. Conventional semi-supervised learning methods include generative models, graph-based semi-supervised learning, semi-supervised support vector machines, self-training, co-training paradigm, etc. Generally, semi-supervised learning seeks to exploit unlabeled data to improve the

* Corresponding author. Tel.: +86 87951442.
E-mail address: gezhiqiang@zju.edu.cn (Z. Ge).

http://dx.doi.org/10.1016/j.chemolab.2015.08.002
0169-7439/



(a) Prediction curves of three methods when proportion of labeled data is 15%



(b) D-values of three methods when proportion of labeled data is 15%

Fig. 1. The prediction curve and *D*-values of co-training PLS, conventional PLS, and co-training *k*NN; the proportion of labeled data is 15%.

performance of the supervised data model. Among all those semi-supervised methods, co-training has its peculiar superiorities. Being a wrapper method, the co-training strategy has no limit on the structure of the data model, which means any model can be incorporated into this method. Different from the self-training strategy, it avoids the disadvantage that the error may be reinforced during the self-training process. The ideology of utilizing different views in co-training has aroused great attention in machine learning area, which also established the foundation of multi-view data learning. Due to its good adaptability, satisfactory performance and easy understanding, co-training has gained much attention and been applied in diverse fields such as natural language processing [14,15], content based image retrieval [16,17], etc.

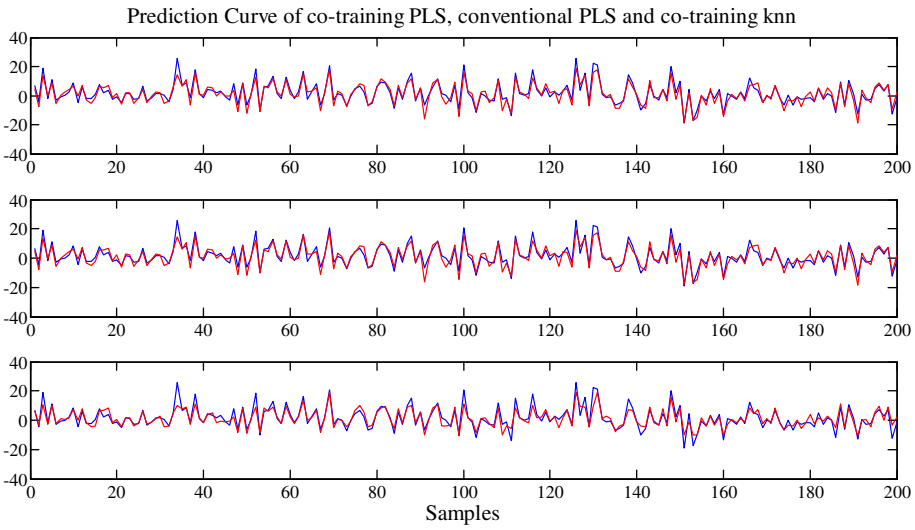
This paper intends to combine the co-training strategy with the typically used partial least squares (PLS) [18–21] algorithm for soft sensor modeling. First, the whole attribute set X is split into two different parts x^1 and x^2, based on which two PLS regression models are trained separately. In order to ensure the diversity, the two regressors are trained not only on different attributes, but also the two data sets differ gradually as the iteration cycle proceeds. To estimate the labeling confidence, the influences of the newly labeled data examples are evaluated. As a result, the selected unlabeled sample will be the one which makes the new regressor most consistent with the initial labeled data set. The final outputs are two labeled data sets that contain the full X attributes of the new labeled data sets. Two regressors are then built on those two data sets and predicting with their average values. Experiment results show that this algorithm can effectively exploit unlabeled data to improve the performance of PLS, especially when the number of labeled data is quite few.

The rest of this paper is organized as follows. Section 2 briefly introduces the basic PLS algorithm, followed by the detailed description of the co-training PLS soft sensing method in Section 3. Section 4 provides case studies on both numerical example and the TE benchmark process. Finally, conclusions are made.

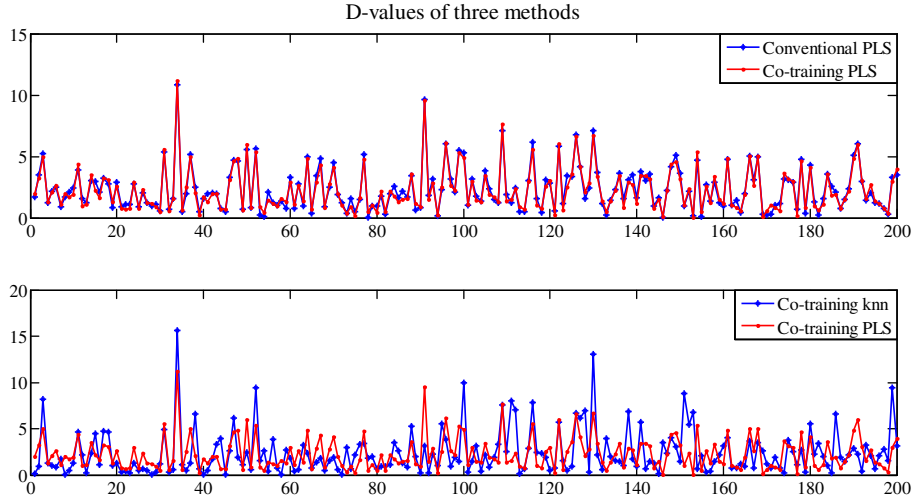
2. Partial least squares

Partial least squares is a multivariate regression method being used as a basic tool in chemometrics, which is ideally situated to studying the variations in large numbers of highly correlated input variables X and relating them to a set of output variables Y. PLS deals not only the internal but also external relationship of X and Y. X is decomposed as follows:

$$X = TP^T + E \tag{1}$$



(a) Prediction curves of three methods when proportion of labeled data is 40%



(b) D-values of three methods when proportion of labeled data is 40%

Fig. 2. The prediction curve and *D*-values of co-training PLS, conventional PLS, and co-training *k*NN; the proportion of labeled data is 40%.

where $X \in R^{m \times n}$ is input matrix, $T \in R^{n \times a}$ is score matrix, $P \in R^{m \times a}$ is loading matrix, and $E \in R^{m \times n}$ is noise matrix. If we replace TP^T with sum of the product of score vector t_j (the jth row of T) and loading vector p_j (the jth row of P), then we can get $X = \sum_{j=1}^{a} t_j p_j^T + E$. In the same way, Y can be decomposed as

$$Y = UQ^T + F \tag{2}$$

where $Y \in R^{n \times m}$ is output matrix, $U \in R^{n \times a}$ is score matrix, $Q \in R^{m \times a}$ is loading matrix, and $F \in R^{n \times m}$ is noise matrix. Similarly, we can replace UQ^T with sum of the product of u_j and q_j, which results in $Y = \sum_{j=1}^{a} u_j q_j^T + F$. Assume that $\tilde{u}_j = b_j t_j$, where b_j is regression coefficient, $U = TB$, and $B \in R^{a \times a}$ is regression matrix. The relationship between X and Y can be represented as $Y = TBQ^T + F$.

Table 1
RMSE of models sharing and not sharing unlabeled data in numerical study.

Proportion of labeled data	15%	20%	25%	30%	35%	40%	45%
Sharing	3.7585	3.4900	3.3149	3.1656	3.2104	3.1957	3.0827
Not sharing	3.6153	3.3719	3.2433	3.1429	3.1599	3.1503	3.0541

3. Co-training PLS for soft sensor modeling and prediction

The co-training method is proposed by Blum and Mitchell in 1998. It trains two learners separately on two sufficient and redundant views and use the predictions of one learner on unlabeled examples to augment the training set of the other [22]. Quite a lot of studies have been carried out on this method since it has been proposed. K. Nigan and R. Ghani investigated on circumstances which do not conform to the sufficient and redundant views requirements [23]. They showed that co-training can achieve satisfactory performance by dividing the attributes randomly into two different views. S. Goldman and Y. Zhou proposed an adapted co-training algorithm, which does not require sufficient and redundant views [24]. Z-H Zhou and M. Li put forward Tri-training, which requires neither the sufficient and redundant views nor different learners as the previous co-training methods [25]. Considering the difficulty of fulfilling the sufficient and redundant views, Wang and Zhou

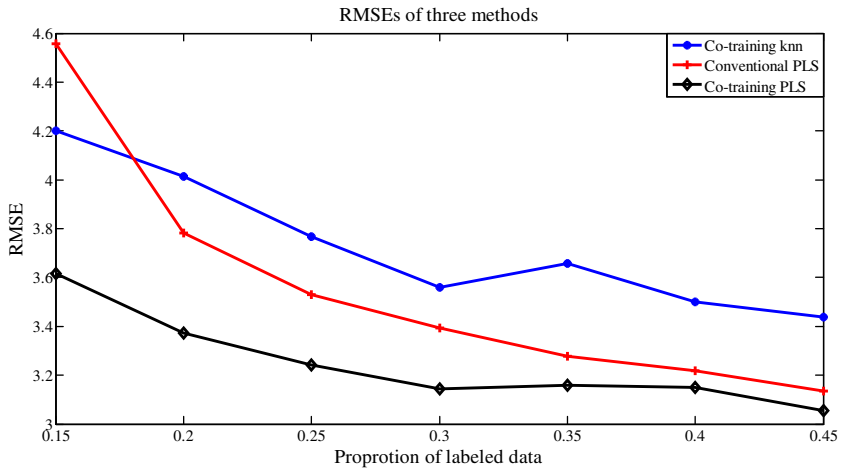


Fig. 3. RMSE values of different proportions of labeled data.

Table 2
Time cost of 100 simulation times under different circumstances.

Proportion of labeled data	15%	20%	25%	30%	35%	40%	45%
Co-training PLS	205.50s	193.54 s	176.39 s	164.07 s	150.04 s	132.24 s	121.71 s
Co-training *k*NN	466.69 s	490.33 s	498.88 s	496.44 s	489.39 s	460.52 s	425.64 s

proved that only if two learners are of great diversity the performance of them can be enhanced [26]. For regression purpose, Zhou and Li developed a co-training regression method, which has gained lots of attention in recent years [27–29]. Due to the implicit usage of manifold assumption, the co-training regression strengthens the ability of dealing nonlinear problems of the base learner.

3.1. Co-training regression

Lots of studies have conducted on semi-supervised learning in the past few years. However, despite the importance of regression in data analytics, investigations mainly focus on classification. It is not until 2005 that co-training regression began to be studied. One of the main

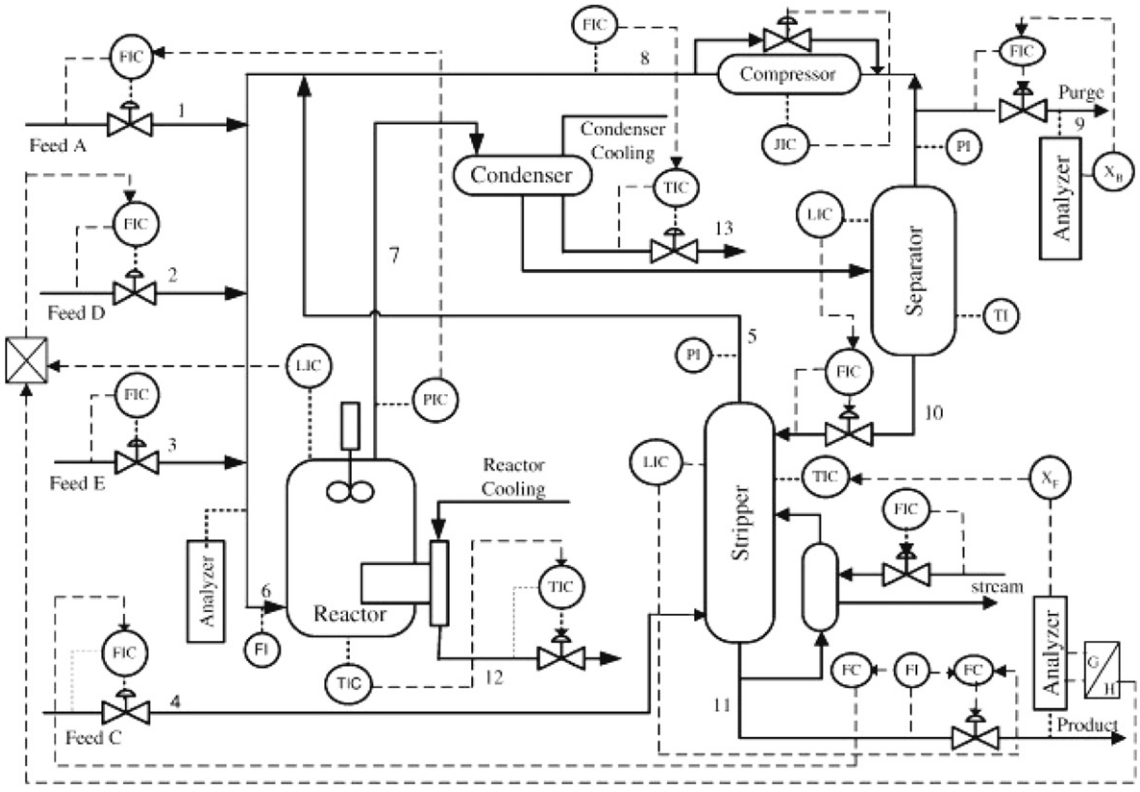


Fig. 4. Flowchart of TE process.

Table 3
Input variables for soft sensor models in TE process.

No.	Input variable	No.	Input variable
1	A feed	9	Separator temperature
2	D feed	10	Separator pressure
3	E feed	11	Separator underflow
4	A and C feed	12	Stripper pressure
5	Recycle flow	13	Stripper temperature
6	Reactor feed rate	14	Stripper steam flow
7	Reactor temperature	15	Reactor cooling water outlet temperature
8	Purge rate	16	Separator cooling water outlet temperature

difficulties to implement co-training regression is how to find the most confident unlabeled data. In the co-training regression algorithm (Coreg) proposed by Zhou and Li [29], this problem is settled. For unsupervised data labeling, Coreg chooses the most confident unlabeled data by analyzing their influences on the labeled data as follows. An original regressor is trained on the labeled data. Thus, the root mean squared error (RMSE) of this regressor on the labeled example set can be evaluated first. A new regressor is built by adding $(x_u, \hat{y}_u)$ into the train set, where x_u is an unlabeled instance while $\hat{y}_u$ is the real-valued label generated by the original regressor. RMSE can be evaluated as the same way. To estimate the influence of the new labeled data, Coreg take the one that maximize the following value as the best one

$$\Delta_{x_u} = \sum_{x_i \in \Omega} \left((y_i - h(x_i))^2 - (y_i - h'(x_i))^2 \right) \tag{3}$$

where x_u is unlabeled data, Ω is the set of its k nearest labeled data, y_i is the real label, $h(x_i)$ is the result labeled by the original regressor, while $h'(x_i)$ corresponds to the regressor refined by the new labeled data x_u. In summary, it tells that the new labeled data, which make the updated regressor most consistent with the given labeled data is with the highest confidence.

Coreg utilizes the co-training strategy by training two separated base learners. Rather than requiring sufficient and redundant views as the conventional co-training does, it employs two k nearest neighbor regressors with different distance matrices. As a key of kNN learner is how to determine the distance between different instances. In their Coreg algorithm, they use Minkowsky distance depicted as follows [29]:

$$\text{Minkowsky}_p(x_r, x_s) = \left(\sum_{i=1}^{d} |x_{r,i} - x_{s,i}|^p \right)^{1/p} \tag{4}$$

where d is the number of variables in x. By using different distance order p, two kNN regressors are trained on the same labeled data set. After the building of two kNN learners on the same labeled data set, regressor 1 and regressor 2 try to find the most confident unlabeled data for each other iteratively. When the break condition is met, the final regressor is decided by averaging the two regressors

$$h^*(x) \leftarrow \frac{1}{2}(h_1(x) + h_2(x)) \tag{5}$$

3.2. Co-training PLS model for soft sensor development

The basic idea of the co-training PLS method is to combine the co-training paradigm with PLS algorithm for soft sensor modeling. PLS is selected due to the following reasons. First of all, the building of base learners should not be time consuming because the co-training strategy is an iterative method. Commonly used methods such as neural network, locally weighted partial least squares, and kernel methods all need a lot of time for model construction. Second, industrial process data are collinear with each other, PLS is an effective method to handle this issue. Besides, the co-training regression method intends to label the unlabeled samples and uses the most confident one to update the labeled data set. Original model is built with initial labeled data set and the RMSE of this model is calculated on the original labeled ones. Then the newly labeled data are added into the labeled data set one by one to build a new model for the calculating of new RMSE. While the kNN method may not well use the newly added labeled samples, PLS builds a global model with all samples will guarantee certain difference of the models and RMSE values in different iterations. It can definitely help us to find a satisfying sample if it does exist.

The procedure of co-training PLS algorithm is depicted as follows. Let $L = \{X, Y\} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_{|L|}, y_{|L|})\}$ denote the labeled data set, where x_i is the ith instance described by d attributes, y_i is the real-valued data, and $|L|$ is the number of labeled data; let U denote the unlabeled data, which contain only the d attributes while their real labels are unknown. Different from the self-training algorithm which only incorporates a single learner, it is worth to note that there are two regressors incorporated in the co-training algorithm which should be diverse from each other.

First, X is split into two parts, which represent two different views. For each data sample x_i, the first half denotes as x_i^1 and the other half is x_i^2. When splitting the X portion of each data, we can either divide them according to their element property or split them equally. Given

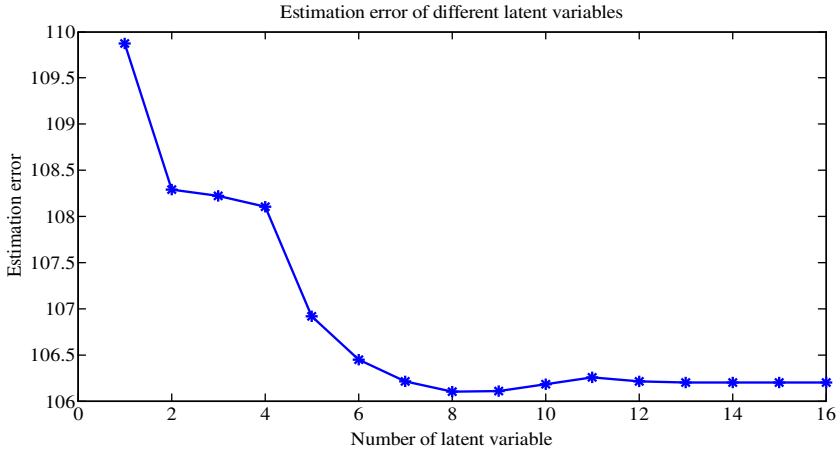


Fig. 5. Estimation error of different latent variables: with all variables.

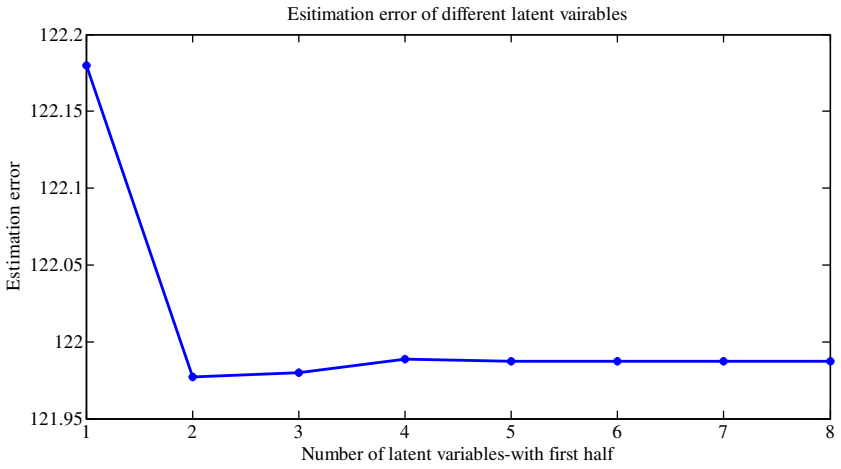


Fig. 6. Estimation error of different latent variables: with first half variables.

the assumption that we do not know any information of components in X, the two different parts can be obtained by the equally splitting strategy. Note that both two data sets should be sufficient to build a base learner, which is necessary for co-training. Moreover, when this requirement is fulfilled, co-training can always help. It is quite fair to train both learners by splitting the variables evenly, which also makes it easier to incorporate the final regressors together by simply averaging them. However, a more sophisticated method for splitting the whole variables into two different parts is still under investigation and should be considered as one of the important issues in future work, which may have impacts on the performance of the co-training algorithm. In the current paper, the whole variables are simply divided through the equally splitting strategy. Two labeled data sets $L_1 : \{X^1, Y\} = \{(x_1^1, y_1), (x_2^1, y_2), \cdots, (x_{|L|}^1, y_{|L|})\}$ and $L_2 : \{X^2, Y\} = \{(x_1^2, y_1), (x_2^2, y_2), \cdots, (x_{|L|}^2, y_{|L|})\}$ represent two different views for the original data set. Then two PLS base learners h_1 and h_2 with a certain diversity can be separately trained by using the two labeled data sets L_1 and L_2. During the learning iterations, one learner is used to update the other in each loop. Rather than sharing the same continually updating labeled data set L as the conventional co-training strategy, the newly labeled data sample in the co-training PLS method used to train one regressor will not be used to train the other one. As the iteration proceeds, this strategy ensures the regressors updated not only on two data sets with growing difference but also on different parts of them. The discrepancy between the two regressors, which is of great value to the co-training learning strategy, can be well guaranteed. Zhou has proved that given the initial labeled data, if we can train two learners which have large differences, the learners can be improved by exploiting the unlabeled data through the co-training process [26]. Therefore, this strategy is to maximize the difference of the two learners, which proves to be useful in our simulations in Section 4.

To choose appropriate unlabeled data samples, the labeling confidence is estimated through the method proposed by Zhou and Li. In order to label the data sample in each step, we can simply take the sample which makes the refined regressor hold the highest consistency with the labeled data as the most confident one. That is to say, the sample that maximizes Δ_u will be selected as the new labeled one:

$$\Delta_u = \frac{1}{|L|}\left(\sum_{x_i \in L}(y_i - h(x_i))^2 - \sum_{x_i \in L}(y_i - h'(x_i))^2\right) \tag{6}$$

Then the new labeled data are put into L_1 or L_2 according to the regressor it is labeled by. The detailed procedures of co-training PLS are listed below.

Algorithm: co-training PLS

Input: labeled example set L (consists by variables x_l and their correspond label y), unlabeled example set U (contains only x_u), maximum number of learning iterations T

Process:

Equally divide x_l and x_u into two parts respectively: x_l^1, x_l^2 and x_u^1, x_u^2.

$L_1 = \{(x_1^1, y_1), (x_2^1, y_2), \cdots, (x_{|L|}^1, y_{|L|})\}$; $L_2 = \{(x_1^2, y_1), (x_2^2, y_2), \cdots, (x_{|L|}^2, y_{|L|})\}$;

Repeat for T rounds:

For $j \in \{1,2\}$ do

$h_j = pls(L_j)$

For each $x_u \in U$ do

$y_u = h_j(x_u^j)$

$h_j' = pls(L_j \cup \{x_u^j, y_u\})$

$\Delta_{x_u} = \sum_{x_i \in L}\left((y_i - h_j(x_i^j))^2 - (y_i - h_j'(x_i^j))^2\right)/|L|$

End of for

If there exists an $\Delta_{x_u} > 0$

$x_n^j = \arg\max_{x_u \in U} \Delta_{x_u}; y_n = h_j(x_n^j)$

Then $U = U - x_n$

$\pi_j = \{(x_n^j, y_n)\}$

$U = U$

Else $\pi_j = \varnothing$

End of for

$L_{3-j} = L_{3-j} \cup \pi_j$

If neither of L_1 and L_2 changes then exit

End of repeat

Output: new labeled dataset L_1 and L_2.

However, L_1 and L_2 contain only half of x variables each, based on which the performance of the PLS model may not be well expressed. Instead, we can use more x variables to build the PLS regression model. Rather than only updating L_1 and L_2 in each iteration loop, the full

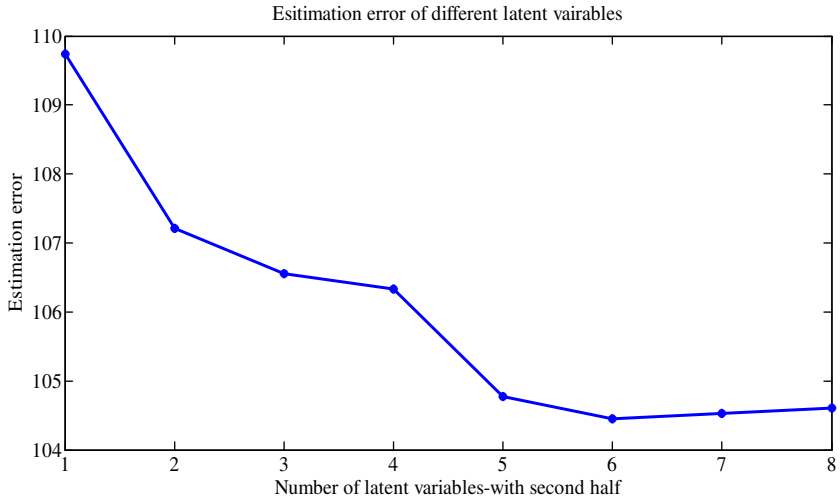
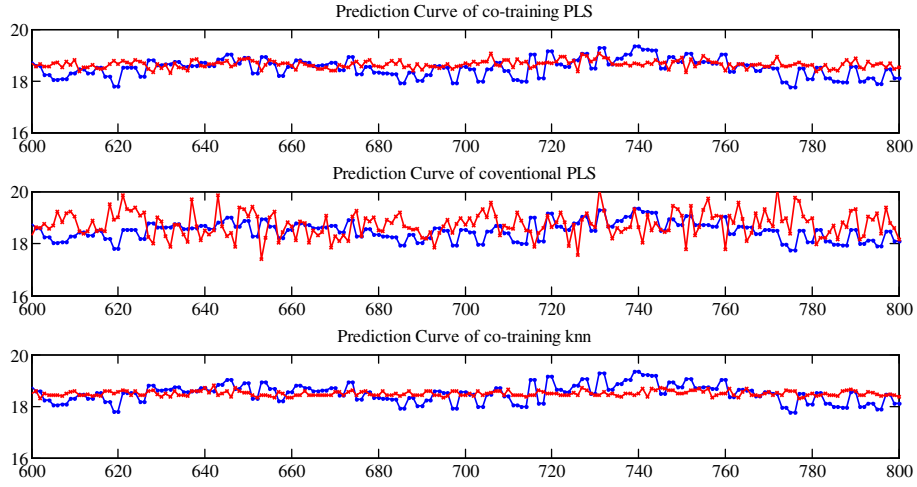
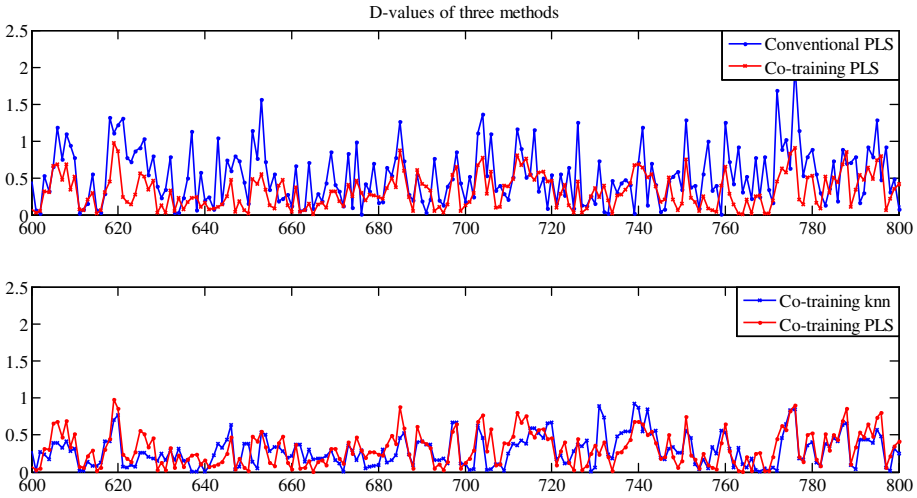


Fig. 7. Estimation errors of different numbers of latent variables-with second half.



(a) Prediction curve of three methods when proportion of labeled data is 10%



(b) D-values of three methods when the proportion of labeled data is 10%

Fig. 8. The prediction curve and *D*-values of co-training PLS, conventional PLS, and co-training *k*NN; the proportion of labeled data is 10%.

domain of the x variables in the two different viewers can be updated for PLS modeling, which are denoted as L_3 and L_4. The pseudo-code for the co-training PLS model can be updated as follows.

Algorithm: Co-training PLS with full domain of input variables

Input: labeled example set L (consists by variables x_l and their correspond label y), unlabeled example set U (contains only x_u), maximum number of learning iterations T

Process:

Equally divide x_l and x_u into two parts respectively: x_l^1, x_l^2 and x_u^1, x_u^2.

$L_1 = \{(x_1^1, y_1), (x_2^1, y_2), \cdots, (x_{|L|}^1, y_{|L|})\}$; $L_2 = \{(x_1^2, y_1), (x_2^2, y_2), \cdots, (x_{|L|}^2, y_{|L|})\}$;

$L_3 = L = \{(x_1, y_1), (x_2, y_2), \cdots, (x_{|L|}, y_{|L|})\}$; $L_4 = L = \{(x_1, y_1), (x_2, y_2), \cdots, (x_{|L|}, y_{|L|})\}$;

Repeat for T rounds:

For $j \in \{1,2\}$ do

$h_j = pls(L_j)$

For each $x_u \in U$ do

$y_u = h_j(x_u^j)$

$h_j' = pls(L_j \cup \{x_u^j, y_u\})$

$\Delta_{x_u} = \sum_{x_i \in L} \left((y_i - h_j(x_i^j))^2 - (y_i - h_j'(x_i^j))^2 \right) / |L|$

End of for

If there exists an $\Delta_{x_u} > 0$

$x_n^j = \arg\max_{x_u \in U} \Delta_{x_u}; y_n = h_j(x_n^j)$

Then $U = U - x_n$

$\pi_j = \{(x_n^j, y_n)\}$; $\pi_j' = \{(x_n, y_n)\}$

$U = U$

Else $\pi_j = \varnothing$; $\pi_j' = \varnothing$

End of for

$L_{3-j} = L_{3-j} \cup \pi_j$

$L_{3-j} = L_{3-j} \cup \pi_j'$

If neither of L_1 and L_2 changes then exit

End of repeat

Output: new labeled dataset L_3 and L_4.

Note that L_3 and L_4 are corresponding to data set L_1 and L_2 separately. In each step, we update L_3 if L_1 changes, and update L_4 when L_2 changes. If L_1 and L_2 share the new labeled sample to refresh themselves in each learning iteration as the traditional co-training does, both L_3 and L_4 will be updated simultaneously in the training stage, too. The simulation comparison in the case study demonstrates that our method has certain superiority.

Based on the new co-training PLS model, two regressors are trained separately on L_3 and L_4, named as h_3 and h_4. For soft sensing of a new data sample x_{new}, the quality prediction value of x_{new} can be calculated as follows

$$\hat{y}_{new} = (h_3(x_{new}) + h_4(x_{new}))/2 \tag{7}$$

To evaluate the prediction performance of the co-training soft sensor, the conventionally used root mean squared error (RMSE) index is defined as follows:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (\hat{y}_i - y_i)^2}{n}} \quad i = 1, 2, \ldots n \tag{8}$$

Where n represents the number of test samples, y_i and $\hat{y}_i$ are real and predicted values, respectively.

4. Case studies

In this section, two case studies are provided for performance evaluation of the co-training PLS model based soft sensor. One is a numerical example and the other one is the well-known Tennessee Eastman benchmark process.

4.1. Numerical example

This is a single output system, with the detailed model structure given as follows:

$$\begin{aligned} X &= [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}] = M * T + E \\ y &= x_1 + 0.4\sin(x_2 x_6 x_9 x_{12}) + 0.1 x_3 x_7 x_{11} + x_4 + x_5 + x_8 + x_{10} \\ &\quad + 0.5 x_2 x_9 + e \end{aligned} \tag{9}$$

where $T \in R^{n \times 8}$ is the latent variable matrix, each variable follows the Gaussian distribution; thus, $t_i \sim N(0,1)$, $i = 1, 2L\ 8$, n is the number of data samples, M is a random 12×8 matrix, and the added noise is assumed to be a white noise with covariance $0.01I$.

In order to obtain a confident result, a total of 100 simulation times are carried out under different circumstances with diverse labeled data proportions. In each case, 300 data samples are generated, among which 100 samples are used as the training data set and the rest 200 are used as testing data set. During each time of the 100 simulations, labeled data are randomly chosen from the training data set, and the rest of them are denoted as unlabeled data. The latent variable number inside the training process of co-training PLS is chosen to be 4. The number of latent variables in the final building of models is selected as 8, which is the same as the latent variable number used in this example. Here, different proportions of the labeled data set have been studied, which are between 15% and 45%. Particularly, one realization of the prediction curve of 200 samples, and their corresponding D-values are provided in Figs. 1 and 2, under the situations that the labeled data proportion are 15% and 40%. The red lines represent the prediction curves of co-training PLS, conventional PLS, and co-training kNN in Figs. 1(a) and 2(a). The blue line represents real values of outputs here. The index of D-values in Figs. 1(b) and 2(b) is defined as follows.

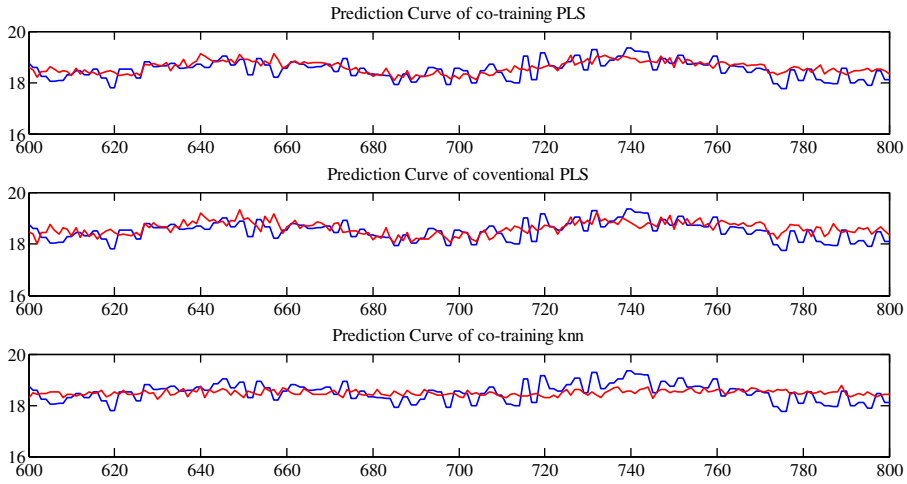
$$D\text{-values} = |\text{predict values} - \text{real values}| \tag{10}$$

Table 1 gives the RMSEs of sharing and not sharing new unlabeled samples in the co-training process. It shows that the proposed algorithm is superior.

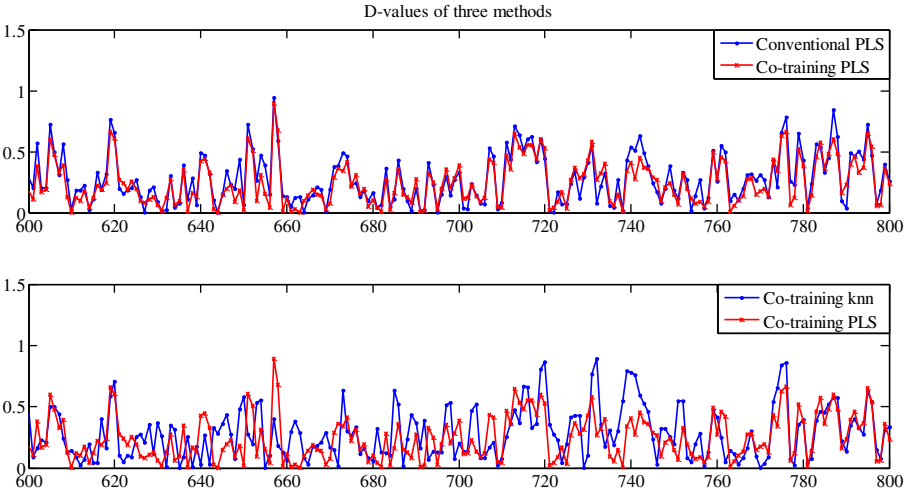
As co-training kNN is the most well-known algorithm, it is taken into comparison with the new method in this paper. Parameters are set according to Zhou's paper [29], except that unlabeled examples used in each training iteration are from the full data set rather than randomly selected 100 samples, which makes it fair to compare with the co-training PLS in both time efficiency and RMSE. Fig. 3 shows the averaged RMSE values of the 100 simulation times for co-training PLS, co-training kNN, and traditional PLS model based soft sensors under different cases. Table 2 shows the time efficiency of 100 total simulation times of these two methods. The simulation was executed on a computer with 3.4GHz Intel Core i3 processor and 8 Gb of memory. It can be seen that the prediction performance has been greatly improved by the co-training PLS method, particularly when the proportion of labeled data is quite small. Furthermore, co-training PLS outperforms co-training kNN all the time, both in time efficiency and prediction performance.

4.2. Tennessee Eastman process

The Tennessee Eastman (TE) process is a well-known benchmark that has been extensively used for testing process monitoring, control, and soft sensor modeling methods. This process consists of five operating units: a condenser, a compressor, a reactor, a separator, and a



(a) Prediction curve of three methods when proportion of labeled data is 30%



(b) D-values of three methods when the proportion of labeled data is 30%

Fig. 9. The prediction curve and *D*-values of co-training PLS, conventional PLS, and co-training *k*NN; the proportion of labeled data is 30%.

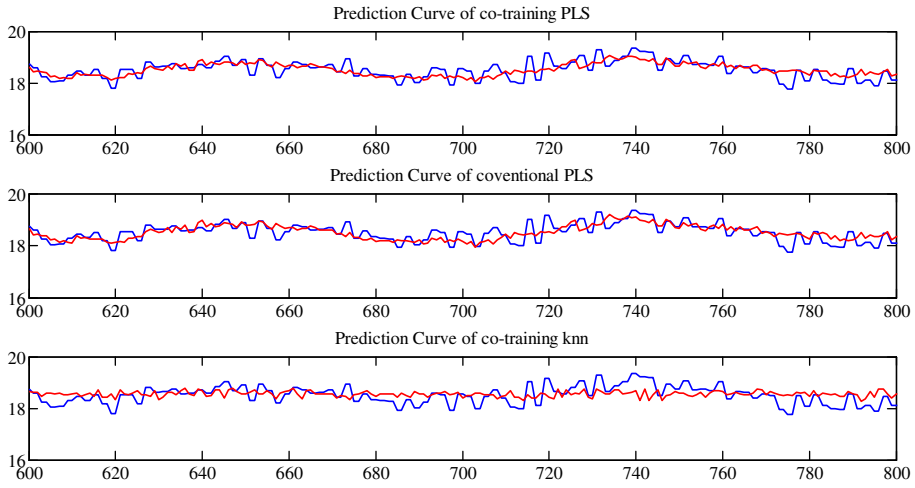
stripper as shown in Fig. 4. A total of 53 variables are incorporated in this process, which contain 41 measured variables and 12 manipulated variables. Among the 41 measured variables, 22 of them are easy to measure while the other 19 component variables are difficult to measure. Therefore, in order to predict those component variables, soft sensors are needed. A total of 16 easy-to-measure variables listed in Table 3 are chosen as input variables of the soft sensor, and the prediction performance of the co-training PLS algorithm is tested on a single output variable (corresponding to component E in the purge gas stream). The training data set is a collection of 500 normal samples, while the testing data set is a collection of 960 samples.

Fig. 5 shows the result of the estimation error for different number of latent variables based on the leave-one-out cross validation (LOOCV) method. Based on this result, the number of latent variables in the final model building stage is selected as 8. The LOOCV is also executed on the different parts of TE data. Figs. 6 and 7 show the outcome of LOOCV on the front half and latter half. As a result, the latent variables numbers inside co-training are selected as 2 and 6. Labeled data used in this simulation are randomly selected from the training data set, and the remaining data are automatically used as unlabeled data. The numbers of the labeled data are selected as 20, 30, 40, 50, 60, 70, 80, 90, and 100 in different experiments, and the corresponding unlabeled data are 180, 170, 160, 150, 140, 130, 120, 110, and 100. The iteration rounds is set as 80. A total of 50 simulations have been carried out in order to provide a more confident result.

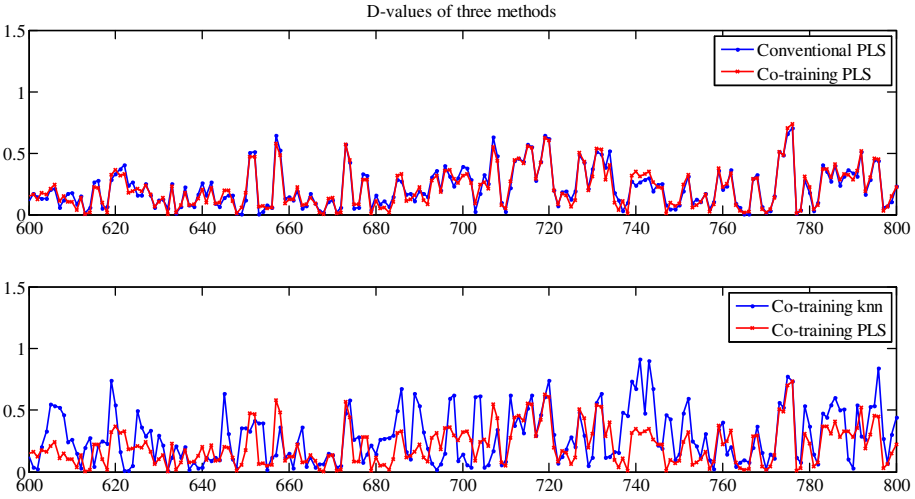
Detailed prediction results and *D*-values of co-training PLS, co-training *k*NN, and conventional PLS model based soft sensors under three conditions are provided in Figs. 8–10. To be clear, only those samples between 600 and 800 are selected in the figures. Fig. 11 shows the averaged RMSE values of these three algorithms, under different proportions of labeled data samples. It is apparent that the co-training PLS model based soft sensor has greatly improved the prediction performance over traditional PLS, particularly when the number of labeled data samples is quite small. Moreover, it also outperforms the co-training *k*NN algorithm in this case. As the number of labeled data samples increases, the performance difference between the co-training and traditional PLS models tends to be smaller. Actually, when the proportion of labeled data sample is nearly 50%, it can hardly make a clear difference between co-training PLS and traditional PLS methods.

5. Conclusions

In this paper, a co-training PLS model for soft sensor modeling has been constructed under the circumstance that the numbers of labeled and unlabeled samples are imbalanced. The effectiveness of co-training strategy based soft sensor has been validated through a



(a) Prediction curve of three methods when proportion of labeled data is 50%



(b) D-values of three methods when the proportion of labeled data is 50%

Fig. 10. The prediction curve and *D*-values of co-training PLS, conventional PLS, and co-training *k*NN; the proportion of labeled data is 50%.

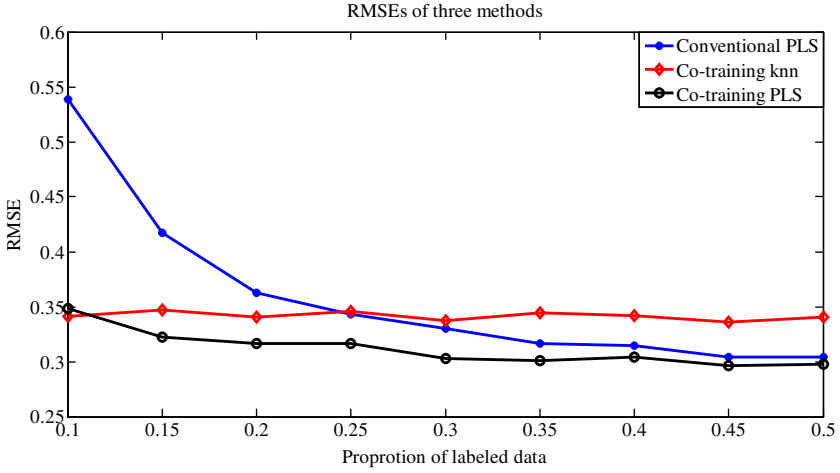


Fig. 11. RMSE values of different proportions of labeled data.

numerical example and the TE benchmark process. Compared to traditional PLS and co-training *k*NN model based soft sensors, the co-training PLS model based soft sensor has provided more satisfactory performances. Although only the basic PLS model has been combined with the co-training modeling strategy, the idea can be extended to other commonly used soft sensor modeling methods, such as principal component regression, artificial neural networks, support vector regressions, etc. Since the diversity between different learners has been considered as an important factor in the co-training method, it may be a good choice if different structures of the data model are introduced for co-training model development.

Conflict of interest

The authors declare that they have no conflict of interest.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (NSFC) (61370029), and National Project 973 (2012CB720500), and the Alexander von Humboldt Foundation.

References

[1] L. Fortuna, S. Graziani, A. Rizzo, M.G. Xibilia, Soft sensors for monitoring and control of industrial processes, Springer, 2007.
[2] Z. Ge, F. Gao, Z. Song, Batch process monitoring based on support vector data description method, J. Process Control 21 (2011) 949–959.
[3] P. Kadlec, B. Gabrys, S. Strandt, Data-driven soft sensors in the process industry, Comput. Chem. Eng. 33 (2009) 795–814.
[4] S. Khatibisepehr, B. Huang, S. Khare, Design of inferential sensors in the process industry: a review of Bayesian methods, J. Process Control 23 (2013) 1575–1596.
[5] Z. Ge, Z. Song, F. Gao, Review of recent research on data-based process monitoring, Ind. Eng. Chem. Res. 52 (2013) 3543–3562.
[6] S. Kim, R. Okajima, M. Kano, S. Hasebe, Development of soft-sensor using locally weighted PLS with adaptive similarity measure, Chemom. Intell. Lab. Syst. 124 (2013) 43–49.
[7] Z. Ge, B. Huang, Z. Song, Mixture semisupervised principal component regression model and soft sensor application, AIChE J. 60 (2014) 533–545.
[8] H. Kaneko, K. Funatsu, Adaptive soft sensor based on online support vector regression and Bayesian ensemble learning for various states in chemical plants, Chemom. Intell. Lab. Syst. 137 (2014) 57–66.
[9] B.M. Shahshahani, D.A. Landgrebe, The effect of unlabeled samples in reducing the small sample size problem and mitigating the Hughes phenomenon, IEEE Trans. Geosci. Remote Sens. 32 (1994) 1087–1095.
[10] X. Zhu, A.B. Goldberg, Introduction to semi-supervised learning, Synthesis lectures on artificial intelligence and machine learning, 32009 1–130.
[11] Z. Ge, Z. Song, Semisupervised Bayesian method for soft sensor modeling with unlabeled data samples, AIChE J. 57 (2011) 2109–2119.
[12] H. Shin, T. Hou, K. Park, C.-K. Park, S. Choi, Prediction of movement direction in crude oil prices based on semi-supervised learning, Decis. Support. Syst. 55 (2013) 348–358.
[13] A. Søgaard, Semi-supervised learning and domain adaptation in natural language processing, Synthesis Lectures on Human Language Technologies, 62013 1–103.
[14] D. Pierce, C. Cardie, Limitations of co-training for natural language learning from large datasets, Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing 2001, pp. 1–9.
[15] M. Steedman, M. Osborne, A. Sarkar, S. Clark, R. Hwa, J. Hockenmaier, P. Ruhlen, S. Baker, J. Crim, Bootstrapping statistical parsers from small datasets, Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics, vol. 1, Association for Computational Linguistics 2003, pp. 331–338.
[16] Z.-H. Zhou, K.-J. Chen, H.-B. Dai, Enhancing relevance feedback in image retrieval using unlabeled data, ACM Trans. Inf. Syst. 24 (2006) 219–244.
[17] Z.-H. Zhou, K.-J. Chen, Y. Jiang, Exploiting unlabeled data in content-based image retrieval, Machine Learning: ECML 2004. , Springer, 2004 525–536.
[18] J. Liu, D.-S. Chen, J.-F. Shen, Development of self-validating soft sensors using fast moving window partial least squares, Ind. Eng. Chem. Res. 49 (2010) 11530–11546.
[19] S. Wold, M. Sjöström, L. Eriksson, PLS-regression: a basic tool of chemometrics, Chemom. Intell. Lab. Syst. 58 (2001) 109–130.
[20] F. Ahmed, L.-H. Kim, Y.-K. Yeo, Statistical data modeling based on partial least squares: application to melt index predictions in high density polyethylene processes to achieve energy-saving operation, Korean J. Chem. Eng. 30 (2013) 11–19.
[21] B. Lin, B. Recke, J.K. Knudsen, S.B. Jørgensen, A systematic approach for soft sensor development, Comput. Chem. Eng. 31 (2007) 419–425.
[22] A. Blum, T. Mitchell, Combining labeled and unlabeled data with co-training, Proceedings of the eleventh annual conference on Computational learning theory, ACM 1998, pp. 92–100.
[23] K. Nigam, R. Ghani, Analyzing the effectiveness and applicability of co-training, Proceedings of the ninth international conference on Information and knowledge management, ACM 2000, pp. 86–93.
[24] S. Goldman, Y. Zhou, Enhancing Supervised Learning with Unlabeled Data.
[25] Z.-H. Zhou, M. Li, Tri-training: exploiting unlabeled data using three classifiers, IEEE Trans. Knowl. Data Eng. 17 (2005) 1529–1541.
[26] W. Wang, Z.-H. Zhou, Analyzing co-training style algorithms, Machine Learning: ECML 2007. , Springer, 2007 454–465.
[27] M. Zhang, J. Tang, X. Zhang, X. Xue, Addressing Cold Start in Recommender Systems: A Semi-supervised Co-training Algorithm, 2014.
[28] Y. Yamashita, K. Sasagawa, Co-learning with a locally weighted partial least squares for soft sensors of nonlinear processes, 2014.
[29] Z.-H. Zhou, M. Li, Semi-Supervised Regression with Co-Training, IJCAI, 2005 908–916.