Temperature Prediction of Hydrogen Producing Reactor Using SVM Regression with PSO

Pan Minqiang[1], Zeng Dehuai[1,2], Xu Gang[2,3]

1. School of Mechanical and Automotive Engineering, South China University of Technology, Guangzhou, China, 510640
2. School of Mechatronics and Control Engineering, Shenzhen University, Shenzhen, China, 518060
3. Shenzhen Key laboratory of mould advanced manufacture, Shenzhen, China, 518060

Email: mexmpan@126.com

***Abstract*—Temperature forecasting of hydrogen-producing reactor is a complicated problem due to its nonlinearity and the small quantity of training data. Support vector machine (SVM) has been successfully employed to solve regression problem of nonlinearity and small sample. The determination for hyper-parameters including kernel parameters and the regularization is important to the performance of SVM. Particle Swarm Optimization (PSO) is a method for finding a solution of stochastic global optimizer based on swarm intelligence. Using the interaction of particles, PSO searches the solution space intelligently and finds out the best one. Thus, the proposed forecasting model based on the global optimization of PSO and local accurate searching of SVM is applied to forecast hydrogen-producing reactor temperature in this paper. Practical example results indicate that the application of the PSO-SVM method to temperature forecasting of hydrogen-producing reactor is feasible and effective. And to prove the effectiveness of the model, other existing methods are used to compare with the result of SVM. The results show that the model is effective and highly accurate in the forecasting of hydrogen-producing reactor temperature.**

***Index Terms*—Support vector machine, Particle swarm optimization, parameter selection, prediction**

I. INTRODUCTION

The need for alternative energy, which is clean without emission of pollutants and has high energy efficiency, gradually increases due to the exhaustion of fossil-fuel. Hydrogen is abundantly available in the universe and possesses the highest energy content per unit of weight compared to any of the known fuels, but today near to 95% of hydrogen is produced from fossil-based materials such as methane and naphtha [1-3]. In order to support hydrogen economy, renewable and clean energy source for hydrogen production is demanded. Bio-ethanol Steam Reforming (BESR) is considered as a promising route for Hydrogen production from renewable sources. It is environmentally friendly, has the potential of being carbon neutral and is amenable to a distributed energy generation strategy. Because of the advantages it offers, there have been many studies focusing on BESR over different catalytic system.

Micro-structured multi-channel reactors are much more suitable for the distributed production of hydrogen compared to conventional systems. They are characterized by three dimensional structures in the sub-millimeter range. The highly endothermic reforming reaction needs an important energy supply, which may lead to the development of pronounced axial and radial temperature profiles in the reactor [4]. Temperature control of hydrogen-producing reactor is an important factor for hydrogen production efficiency. However, Temperature time series prediction is a complicated problem due to its nonlinearity and the small quantity of training data.

Numerous studies have focused on the accurate prediction of reactor temperature by using statistical approaches and artificial intelligence approaches. However, the relationship among the hydrogen production operational parameters is complex and highly nonlinear, and the data collected from multi-channel reactor are also quite noisy. Neural network modeling has been shown to reproduce nonlinear data very well while some inherent drawbacks, e.g., the multiple local minima problem, the choice of the number of hidden units and the danger of over fitting [5], etc., would make it difficult to put the ANN into some practice. Superior forecasting accuracy can be gained with a small quantity of training data by using grey model. However, grey model only depicts a monotonously increasing or decreasing process with time as exponential law, and the temperature of hydrogen production reactor is not monotonously increasing or decreasing with time as exponential law because of the operational parameters influence. So a certain error is always generated in forecasting of hydrogen producing reactor based on GM.

Recently, a novel type of learning machine, called support vector machine (SVM), has been receiving increasing attention. SVM was developed by Vapnik and his coworkers in 1995 [6], and it is based on the structure risk minimization (SRM) principle that seeks to minimize an upper bound of the generalization error consisting of the sum of the training error and a confidence interval. There are some remarkable characteristics of SVM, such as good generalization performance, the absence of local minima and sparse representation of solution. SVM has

doi:10.4304/jcp.5.3.388-393

been widely applied from its original application of pattern recognition to the extended application of regression estimation [7]. On the other hand, applications of support vector regression (SVR) extended by SVM such as forecasting of financial market, estimation of power load and prediction of highway traffic flow, have been also under development and have shown many breakthroughs and excellent performances. The time-varying properties of SVR applications resemble the time dependency of hydrogen producing reactor temperature prediction, combined with many successful results of SVR predictions encourage our research in using SVR for temperature forecasting.

The selection of parameters of a SVM model is important to the accuracy of the forecasting. However, most SVM practitioners select these parameters empirically by trying a finite number of values and keeping those that provide the least testing error. This procedure requires a grid search over the space of parameter values and needs to locate the interval of feasible solution and a suitable sampling step. Because of the computational complexity, grid search is only suitable for the adjustment of very few parameters [8]. In further researches, some intelligent algorithms such as evolution algorithms (EA) and genetic algorithms (GA) were employed to choose the parameters of a SVM model, and the improved model offers a superior performance to ordinary regression SVM model [9,10]. Though the GA is less time-consuming and can obtain the optimal solution well [11], the operation of genetic algorithm is difficult, the different types and rates of crossover and mutation need to be set for different optimal problem.

Instead of using GA, a new technology: particle swarm optimization (PSO) motivated by social behavior of organisms such as bird flocking and fish schooling is tried in this study [12]. The PSO is an evolutionary computation technique based on swarm intelligence. It follows a collaborative population-based search, which models over the social behavior of bird flocking. The PSO system combines experiences form both self and neighboring and attempts to balance exploration and exploitation. The PSO has many advantages over other heuristic techniques, e.g., it can be used effectively to exploit the distributed and parallel computing capabilities, to escape local optima, and to implement in a few lines of computer codes. The proposed method is applied to tuning kernels and regularization parameters of SVMs.

This paper is organized as follows: Section 2 introduces the background of SVM regression forecasting method. Parameters tuning of SVM based on PSO is introduced in Section 3. Section 4 proposes temperature forecasting model based on SVR-PSO and testifies the performance of the proposed model with the real data sets from hydrogen producing reactor. Finally, the conclusion is drawn in Section 5.

II. SVM REGRESSION FORECASTING METHOD

The method estimates regression function of SVM. The basic principle of it is to map data in the input space to a high dimensional feature space by using a nonlinear mapping. Then, a linear mapping is made in the high dimensional space.

Suppose a set of data (x_i, y_i), i=1,2,…,n, $x_i \in R^n$ are given as input, $y_i \in R^n$ are the corresponding output. SVM regression theory is to find a nonlinear map from input space to output space and map the data to a higher dimensional feature space through the map, then the following estimate function is used to make linear regression:

$$f(x) = w \cdot \varphi(x) + b \tag{1}$$

where $\varphi(x)$ denotes the high-dimensional feature space, w denotes the weight vector and b denotes the bias term. The problem of the function approximate is equivalent with the minimizing the following problem:

$$\begin{aligned} R_{reg}[f] &= R_{emp}[f] + \lambda\|\omega\|^2 \\ &= C\frac{1}{l}\sum_{i=1}^{l} L_\varepsilon(y, f(x)) + \lambda\|\omega\|^2 \end{aligned} \tag{2}$$

$R_{reg}[f]$ is objective function and l is the number of the sample. $L_\varepsilon(y, f(x))$ is called loss function and λ is adjusting constant meter. The following loss function can be gained concerning with the rarefaction character of the linear ε-insensitive loss function.

$$L_\varepsilon(y, f(x)) = \begin{cases} |y - f(x)| - \varepsilon & |y - f(x)| \geq \varepsilon \\ 0 & |y - f(x)| < \varepsilon \end{cases} \tag{3}$$

In Eq.(3), the loss equals zero if the error of forecasting value is less than ε, otherwise the loss equals value beyond ε. Empirical risk function is

$$R_{emp}^{\varepsilon}[f] = \frac{1}{l}\sum_{i=1}^{l} |y - f(x)|_\varepsilon \tag{4}$$

According to statistic theory, the regression function is determined by minimizing the following functions:

$$\text{Min } \psi(w, \xi, \xi^*) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \tag{5}$$

$$\text{s.t.} \begin{cases} y_i - w \cdot \varphi(x) - b \leq \varepsilon + \xi_i^* \\ w \cdot \varphi(x) + b - y_i \leq \varepsilon + \xi_i \\ \xi_i, \xi_i^* \geq 0 \end{cases}$$

C is used to equalize the complicated item of the model and the parameters of the item of training error. ξ_i and ξ_i^* are relaxation factors and ε is insensitive loss function.

This constrained optimization problem is solved using the following Lagrangian form:

$$\begin{aligned} \text{Max } H(\alpha, \alpha^*) = &-\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(x_i, x_j) \\ &+ \sum_{i=1}^{l}\alpha_i^*(y_i - \varepsilon) - \sum_{i=1}^{l}\alpha_i(y_i - \varepsilon) \end{aligned} \tag{6}$$

$$\text{s.t.} \begin{cases} \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0, \\ \alpha_i, \alpha_i^* \in [0, C] \end{cases} \tag{7}$$

Where α_i and α_i^* are the so-called Lagrangian multipliers, represent solutions to the above quadratic problem that acts as forces pushing predictions towards target value y_i. Only the non-zero values of the Lagrange multipliers in Eq. (7) are useful in forecasting the regression line and are known as support vectors. For all points inside the e-tube, the Lagrange multipliers equal to zero do not contribute to the regression function. Only if the requirement $|y\text{-}f(x)|\geq\varepsilon$ is fulfilled, Lagrange multipliers may be non-zero values and used as support vectors.

By the Lagrange multipliers α_i and α_i^* calculated, an optimal desired weight vector of the regression hyperplane is obtained, that is:

$$w^* = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)K(x_i, x) \tag{8}$$

Now, we have solved the value of w in terms of the Lagrange multipliers. For the variable b, it can be computed by applying Karush-Kuhn-Tucker (KKT) conditions which, in this case, implies that the product of the Lagrange multipliers and constrains has to equal zero:

$$\begin{cases}\alpha_i(w\cdot\varphi(x)+b-y_i+\varepsilon+\xi_i)=0\\ \alpha_i^*(y_i-w\cdot\varphi(x)-b+\varepsilon+\xi_i^*)=0\end{cases} \tag{9}$$

and

$$\begin{cases}(C-\alpha_i)\xi_i=0\\ (C-\alpha_i^*)\xi_i^*)=0\end{cases} \tag{10}$$

Since $\alpha_i, \alpha_i^*=0$ and $\xi_i^*=0$ for $\alpha_i^*\in(0,C)$, b can be computed as follows:

$$\begin{cases}b=y_i-w\cdot\varphi(x)-\varepsilon, \text{ for } \alpha_i\in(0,C)\\ b=y_i-w\cdot\varphi(x)+\varepsilon, \text{ for } \alpha_i^*\in(0,C)\end{cases}$$

Hence, the regression function is:

$$f(x)=\sum_{i=1}^{l}(\alpha_i-\alpha_i^*)K(x_i,x)+b \tag{11}$$

In Eq. (11), $K(x_i, x)$ is called the kernel function. The value of the kernel function equals the inner product of $\phi(x_i)$ and $\phi(x)$ which are produced by mapping two vectors x_i and x into the higher dimensional feature space, that is $K(x_i,x)=\phi(x_i)\cdot\phi(x)$. In SVM, four common kernels function types are given as follows:

RBF kernel: $K(x_i,x)=\exp(-\|x-x_i\|^2/\sigma^2)$

Sigmoid kernel $K(x_i,x)=\tanh(kx_i^Tx+\theta)$

Polynomial kernel: $K(x_i,x)=(x_i^Tx+r)^d$

Linear kernel: $K(x_i,x)=x_i^Tx$

The RBF kernel nonlinearly maps the samples into the high dimensional space, so it can handle nonlinear problem. Furthermore, the linear kernel is a special case of the RBF. The sigmoid kernel behaves like the RBF for certain parameters; however, it is not valid under some parameters. The polynomial has more parameters than the RBF kernel, optimizing parameters will become complex using polynomial basis function or sigmoid function because two variables need determination in polynomial basis function and sigmoid function, respectively. the RBF function has less numerical difficulties. While RBF kernel values are $0 < K(x_i, x)\leq 1$, polynomial kernel value may go to infinity or zero when the degree is large. In addition, polynomial kernel takes a longer time in the training stage and is reported to produce worse results than the RBF kernel in the previous studies. Thus, in this work, RBF is used in the SVM.

For the nonlinear SVR, its generalization performance depends on a good setting of hyper-parameters C, ε and the RBF kernel parameter σ. Parameter C determines penalties to estimation errors. A large C assigns higher penalties to errors so that the regression is trained to minimize error with lower generalization while a small C assigns fewer penalties to errors; this allows the minimization of margin with errors, thus higher generalization ability. Parameter ε controls the width of the ε-insensitive zone, used to fit the training data. The value of ε can affect the number of support vectors used to construct the regression function.

Here, C, σ and ε are user-determined parameters, which may affect SVR generalization performance, the selection of the parameters plays an important role in the performance of SVM. Therefore, these parameters need to be properly optimized to minimize the generalization error. In this paper, there parameters are automatically tuned using the PSO in the training phase.

III. Parameters Tuning of SVM based on PSO

PSO is a stochastic optimization technique introduced recently by Kennedy and Eberhart, which is inspired by social behavior of bird flocking and fish schooling [12, 13]. Similar to other evolutionary computation algorithms such as genetic algorithms, PSO is a population-based search method that exploits a population of individuals to probe promising region of the search space. In this context, the population is called swarm and the individuals are called particles. During the search process in the d-dimensional solution space, each particle (i.e., candidate solution) will adjust its flying velocity and position according to its own flying experience as well as the experiences of the other companion particles of the swarm. The global variant of PSO the best position ever attained by all individuals of the swarm is communicated to all the particles.

During last decade many studies focused on this method and almost all of them, strongly confirmed the abilities of this newly proposed optimization technique [12-18]. Abilities such as fast convergence, finding global optimum in presence of many local optima, simple programming and adaptability with constrained problems. PSO has showed to be promising for solving various engineering problems such as automatic control [16], antenna design [17], and inverse problems [18]. The general principles for the PSO algorithm are stated as follows:

Let us consider a swarm of size n. Each particle P_i (i = 1, 2, . . . , n) from the swarm is characterized by: 1) its current position $X_i(k)\in R^d$, which refers to a candidate solution of the optimization problem at iteration k; 2) its

velocity $V_i(k) \in R^d$; and 3) the best position $P_{besti}(k) \in R^d$ that is identified during its past trajectory. Let $G_{besti}(k) \in R^d$ be the best global position found over all trajectories that are traveled by the articles of the swarm. Each of *n* particles fly through the *d*-dimensional search space R^d with a velocity $V_i(k)$, which is dynamically adjusted according to its personal previous best solution $P_{besti}(k)$ and the previous global solution $G_{besti}(k)$ of the entire swarm. The velocity updates are calculated as a linear combination of position and velocity vectors. The particles interact and move according to the following equations

$$V_i(k+1) = w(k) \cdot V_i(k) + C_1 \cdot R_1(k) \cdot (P_{besti}(k) - X_i(k)) + C_2 \cdot R_2(k) \cdot (G_{besti}(k) - X_i(k)) \quad (12)$$

$$X_i(k+1) = X_i(k) + V_i(k+1) \quad (13)$$

Where $V_i(k+1)$ is the velocity of $(k+1)^{th}$ iteration of i^{th} individual, $V_i(k)$ is the velocity of k^{th} iteration of i^{th} individual, $w(k)$ is the inertial weight used as a tradeoff between global and local exploration capabilities of the swarm. Small values of *w* result in more rapid convergence usually on a suboptimal position, while a too large value may prevent divergence. Typical implementations of the PSO adapt the value of during the training stage, e.g., linearly decreasing it from 1.0 to near 0 over the execution. C_1, C_2 are two acceleration constants regulating the relative velocities with respect to the best global and local positions, respectively. $R_1(\cdot)$ and $R_2(\cdot)$ are random variables that are drawn from a uniform distribution in the range [0, 1] to provide a stochastic weighting of the different components participating in the particle velocity definition.

The inertia weight is set to the following equation:

$$w(k) = w_{\max} - \frac{w_{\max} - w_{\min}}{k_{\max}} \cdot k \quad (14)$$

where $w_{\max}$ is the initial weight, $w_{\min}$ is the final weight, $k_{\max}$ is the maximum number of iterations or generation, and *k* is the current iteration number.

Eq.(12) allows the computation of the velocity at iteration *k*+1 for each particle in the swarm by combining linearly its current velocity (at iteration *k*) and the distances that separate the current particle position from its best previous position and the best global position, respectively. The updating of the particle position is performed with (12). Both (12) and (13) are iterated until convergence of the search process is reached. Typical convergence criteria are based on the iterative behavior of the best value of the adopted fitness function and simply on a user-defined maximum number of iterations.

PSO mentioned above is used to select parameters of SVR: the trade off variable *C*, ε and kernel parameters σ, which are three attributes of each particle. In PSO operation, the fitness function of the particles group with test cases was evaluated using the mean absolute percent error (MAPE). The performance metrics is showed by the following equation:

$$E_{MAPE} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - f(x_i)}{y_i} \right| \quad (15)$$

Where y_i is the actual output, $f(x_i)$ is the estimated value in the SVR model, *n* is the number of sets of test cases.

At the beginning of the algorithm we have ten particles in a group, and the velocity and position of them are created at random. The mean square error is the objective of the particles group. With particles searching, we can get the minimum mean square error of training SVR, and also get the value of *C*, ε and σ at that point. Thus, the computational flow of PSO technique can be described in the following steps.

·Step 1*: PSO Intialization*

(1) Set the iteration number *k* to zero. Initialize randomly the swarm (containing *m* particles) such that the position $X_i(0)$ of each particle to meet the prescribed conditions. PSO learning factors C_1=C_2=2, $w_{\max}$=0.95, w_{min}=0.35, k_{max}=1000.

(2) Set the best position of each particle with its initial position, i.e.P_{besti}=P_i (*i*=1,2,,…,*n*).

(3) Set to zero the velocity vectors V_i (*i*=1,2,,…,*n*) that are associated with the *m* particles.

(4) For each candidate particle P_i (*i*=1,2,,…,*n*), train an SVM estimator on the corresponding augmented training set and with the estimates of the regularization and kernel parameters that are conveyed by P_i. Then, compute its fitness function(s) using Eq.(15).

·Step 2*: PSO Search Process*

(1) Perform the update of each particle velocity, the best global position G_{besti} is chosen as the position exhibiting the minimal value of the considered fitness function over all explored trajectories, i.e.

$$G_{besti}(k) \in \{P_{best1}(k), P_{best2} \cdots, P_{bestm}(k)\} \left| f(G_{besti}(k)) = \min\{f(G_{best1}(k)), f(G_{best2}(k)), \cdots, f(P_{bestm}(k))\} \right. \quad (16)$$

(2) Compare the personal best of each particle to its current fitness, and set $P_{besti}(k)$ to the better performance, i.e.

$$P_{besti}(k) = \begin{cases} P_{besti}(k-1), & if \quad f(P_{besti}(k)) \le f(P_{besti}(k-1)) \\ P_{besti}(k), & if \quad f(P_{besti}(k)) > f(P_{besti}(k-1)) \end{cases} \quad (17)$$

(3) Update the speed of each particle using (1), if $V_i > V_{max}$ then $V_i = V_{max}$. If $V_i < V_{min}$ then $V_i = V_{min}$.

(4) Update the position of each particle by means of Eq.(11). If a particle goes beyond the predefined boundaries of the search space, truncate the updating by setting the position of the particle at the space boundary and reverse its search direction. This will permit to forbid the particles from further attempting to go outside the allowed search space.

(5) For each candidate particle P_i (*i*=1, 2, …, *n*), train an SVM estimator with the corresponding augmented training set and regularization and

kernel parameters. Compute the fitness function using Eq.(15)

·Step 3: *Convergence Check*

If one of the stopping criteria is satisfied, then stop, we can get the global optimum that are the value of C, ε and σ. otherwise, loop to Step 2.

·Step 4: Establish the prediction model based on the parameter values of C, ε and σ.

VI. Temperature Forecasting Model Based on SVR-PSO

A. Problem formulation

According to the changes of time series of the temperature, the current temperature is certainly linked with that of several hours ago. Thus the previous hydrogen-producing reactor temperature sequence data can be used to predict the future reactor temperature.

Assuming $T_i(t)$ is the temperature value of hydrogen-producing reactor in t moment, $v_i(t\text{-}1)$ is the temperature value in t-1 moment. The temperature value of hydrogen-producing reactor of current and previous s time period can be used to forecast the reactor temperature value in the future time period. Let $T_i(t)$, $T_i(t\text{-}1)$, …, $T_i(t\text{-}m)$be the samples input vector in t moment as $\boldsymbol{x}_i$, $T_i(t+1)$ be the samples output value y_i.

Chose n reactor temperature value samples as initial training set: $S = \{s_i | s_i = (x_i, y_i), i = 1,2,\cdots,n\}$. According to SVR algorithm, the initial prediction regression function is obtained. Then the prediction value for reactor temperature $\hat{y}_i$ is:

$$\hat{y}_i = \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) K(x_i, x) + b \tag{18}$$

B. Data Pre-processing

The samples are divided into two data sets: the training data sets and the testing data sets. Subject to random factors (e.g. transmission errors, etc.), it can not be avoided to lose data accuracy such as data errors and data loss, so the primary data preprocessing need to be implemented to correction errors. The training data is smoothly processed to eliminate singular value, and the experimental data (including training data and testing data) is normalized, which can improve generalization capability of SVR. If the data is unequal interval series, unequal interval series are transformed into equal interval series to train SVR.

In order to avoid the influence of difference between factors, the parameters of input and output are normalized as Eq.(19), the learning data are scaled into [0, 1]:

$$\bar{y}_i = \frac{2(y_i - y_{\min})}{y_{\max} - y_{\min}} - 1 \tag{19}$$

C. Reactor Temperature using SVR-PSO Model

While using PSO-SVR model for hydrogen-producing reactor temperature forecasting, in the training stage, firstly the hyper-parameters C, ε and σ of SVM model are optimized by PSO, the validation error is measured by Eq.(15), the adjusted parameters with minimum validation error are selected as the most appropriate parameters. Then, the optimal parameters are utilized to train SVM model. The testing data sets are used to examine the accuracy of the forecasting model.

The training data is used to train the SVR for determining hyper-parameters with the PSO algorithm, and the testing data is used to evaluate the trained SVR. The parameters of PSO are set as follows: the size of the population is 45, set learning factors C_1= C_2 = 2, the inertia weight w decreases linearly from 0.9 to 0.3, locate each particle's position and velocity of weight variables in [-1, 1] randomly.

Cross-validation is a popular technique for estimating generalization performance. In order to better evaluate the performance of the proposed approach, we also use the k-folds cross validation [6] to chosen the appropriate parameters (C, ε and σ). In this example, we use k = 5 for the number of folds. The optimal parameters of SVR searched by PSO are C = 11.8, ε = 0.7, σ = 1.4. Using the parameters determined by PSO and several groups of parameters selected randomly to establish reactor temperature prediction models based on SVR-PSO, the tuning results are given in Table 1. It can be seen from Table 1, the model established by the optimal parameters has the best prediction effect. Therefore, the practical prediction model of hydrogen production reactor based on SVR-PSO is established with C = 11.8, ε = 0.7, σ =1.4.

Tab.1 Model prediction results with different parameters

No.	C	ε	σ	Error%
1	11.8	0.7	1.4	4.89
2	8.4	0.8	0.7	6.83
3	9.5	0.6	2.4	7.45
4	3.2	0.5	4.3	8.98
5	1.2	0.4	2.8	11.93

Comparison of the forecasting results among SVR-PSO, SVR and ANN are shown in Figure 1.

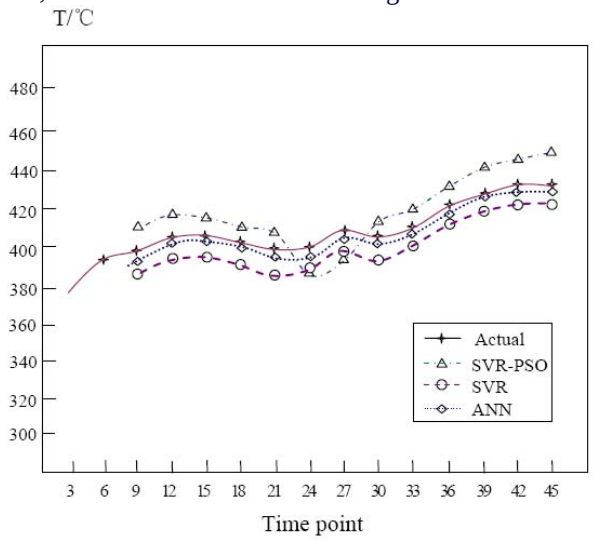

Fig. 1 Prediction results comparison of SVR-PSO, SVR and ANN

From the results of it is obvious that SVR-PSO algorithm has better prediction accuracy than that of SVR, the prediction ability of SVR-PSO is significantly stronger than that of ANN, and its prediction results are

stable. Thus prove that optimizing the model parameters of SVR by PSO is feasible and superior. SVR-PSO implements the principle of structural risk minimization in place of experiential risk minimization, which makes it have excellent generalization ability in the situation of small sample, it has no requirement in data series distribution, and can be in the wake of changing disciplinarian of data. Thus, the forecasting error of SVR-PSO is also small under the circumstances that the change of data takes on great fluctuation.

V. CONCLUSION

In this paper, we applied SVR optimized by PSO to the temperature prediction of hydrogen production reactor. A promising novel particle swarm optimization-based hyper-parameters selection for SVM regression prediction model is proposed, which avoids over-fitting or under-fitting of the SVM model occurring because of the improper determining of these parameters. PSO is a new optimization method, which not only has strong global search capability, but also is very easy to implement. So it is very suitable for parameters selection of SVM. Results show that SVR can serve as a promising alternative for existing prediction models. Compared with the results of other methods, it can be seen from the experiment that the SVR-PSO prediction model yields higher accurate rate for all data sets tested and overcomes the main shortage of artificial neural network without defining network structure and trapping in the local optimum.

REFERENCES

[1] Vizcaino AJ, Carrero A, Calles JA. Hydrogen production by ethanol steam reforming over Cu–Ni supported catalysts. Int J Hydrogen Energy 2007;32:1450–1461.

[2] Haryanto A, Fernando S, Murali N, Adhikari S. Current status of hydrogen production techniques by steam reforming of ethanol: a review. Energy Fuels 2005;19:2098–2106.

[3] Chang-Yeol Yu, Dong-Wook Lee, Sang-Jun Park, Kwan-Young Lee, Kew-Ho Lee, Study on a catalytic membrane reactor for hydrogen production from ethanol steam reforming, Int J Hydrogen Energy 2009;34(7):2947–2954.

[4] Pierre Reuse, Albert Renken, Katja Haas-Santo, Oliver Görke, Klaus Schubert, Hydrogen production for fuel cell application in an autothermal micro-channel reactor, Chemical engineering journal, 2004, 101:133-141

[5] Mohamed EA, Abdelaziz AY, Mostafa AS. A neural network-based scheme for fault diagnosis of power transformers. Elect Power Syst. Res. 2005, 75(1): 29–39.

[6] Vapnik V. The nature of statistical learning theory. New York: Springer-Verlag, 1995.

[7] V.N. Vapnik, S.E. Golowich, and A.J. Smola, Support vector machine for function approximation, regression estimation and signal procession. Adv. Neural Information Procession Syst. 1996,. 9: 281-287.

[8] Mina Sung-Hwan, Lee Jumin, Han Ingoo. Hybrid genetic algorithms and support vector machines for bankruptcy prediction. Expert Syst Appl 2006, 31(3):652–60.

[9] Frauke Friedrichs, Christian Igel, Evolutionary tuning of multiple SVM parameters , Neurocomputing, 2005, 64: 107-117

[10] F.P. Ping and C.H. Wei, Support vector machines with simulated annealing algorithms in electricity load forecasting, Energy Conversion and Management, 2005, 46: 2669-2688,.

[11] F.P. Ping and C.H. Wei, Forecasting regional electricity load based on recurrent support vector machines with genetic algorithms, *Electric Power Systems Research*, vol. 74, pp. 417-425, 2005

[12] Kennedy J, Eberhart R C, Particle swarm optimization, Proceedings of IEEE International Conference on Neutral Networks, Perth, Australia, 1995,pp. 1942-1948.

[13] Shi Y H, Eberhart R C, A modified particle swarm optimizer, IEEE International Conference on Evolutionary computation, Anchorage, Alaska, 1998,pp. 69-73.

[14] R. C. Eberhart and J. Kennedy, A new Optimizer Using Particle Swarm Theory, in proc. 6th Symp. Micro Machine and Human Science, Nagoya, Japan, 1995, pp. 34-44.

[15] Konstantinos E. Parasopoulos and Michael N. Vrahatis, On the Computation og All Global Minimizers Through Particle Swarm optimization, IEEE Trans. On Evolutionary Computation, vol. 8, No. 3, June 2004.

[16] K.E. Parsopoulos and M.N. Vrahatis. Particle swarm optimization method for constrained optimization problems. In P. Sincak, J. Vascak, V. Kvasnicka, and J. Pospichal, editors, Intelligent Technologies--Theory and Application: New Trends in Intelligent Technologies, volume 76 of Frontiers in Artificial Intelligence and Applications, pages 214--220. IOS Press, 2002.

[17] J. F. Scutte, J. A. Reinbolt, B. J. Fregly, R. T. Haftka and A. D. George, Parallel global optimization with the particle swarm algorithm, Int. Journal for Numerical Methods in Engineering 2004; 61:2296-2315.

[18] Raymond R. Tan, Hybrid evolutionary computation for the development of pollution prevention and control strategies, Journal of Cleaner Production 15 (2007) 902-906.