Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Optimizing radial basis function neural network based on rough sets and affinity propagation clustering algorithm*

Xin-zheng XU†[1], Shi-fei DING[1,2], Zhong-zhi SHI[2], Hong ZHU[1]

([1]*School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221116, China*)

([2]*Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China*)

†E-mail: xuxinzh@163.com

Received June 25, 2011; Revision accepted Oct. 25, 2011; Crosschecked Dec. 29, 2011

Abstract: A novel method based on rough sets (RS) and the affinity propagation (AP) clustering algorithm is developed to optimize a radial basis function neural network (RBFNN). First, attribute reduction (AR) based on RS theory, as a preprocessor of RBFNN, is presented to eliminate noise and redundant attributes of datasets while determining the number of neurons in the input layer of RBFNN. Second, an AP clustering algorithm is proposed to search for the centers and their widths without a priori knowledge about the number of clusters. These parameters are transferred to the RBF units of RBFNN as the centers and widths of the RBF function. Then the weights connecting the hidden layer and output layer are evaluated and adjusted using the least square method (LSM) according to the output of the RBF units and desired output. Experimental results show that the proposed method has a more powerful generalization capability than conventional methods for an RBFNN.

Key words: Radial basis function neural network (RBFNN), Rough sets, Affinity propagation, Clustering
doi:10.1631/jzus.C1100176 **Document code:** A **CLC number:** TP183

1 Introduction

The radial basis function neural network (RBFNN), as a type of feedforward neural network (NN), has recently attracted extensive research interest because of its simple architecture, high approximation and regularization capability, and good local specialization and global generalization ability. RBFNN has proved to be able to approximate any reasonable continuous function mapping with a satisfactory level of accuracy (Zhang and Zhang, 2004). To date, RBFNN has been widely used in function approximation, pattern recognition, data classification, control, time series prediction, and nonlinear system identification (Guerra and Coelho, 2008; Jing *et al*., 2008; Lee and Ko, 2009; Beyhan and Alci, 2010; Jayasree *et al*., 2010; Hou and Han, 2011).

However, optimizing the structure of RBFNN remains challenging. The parameters of RBFNN involve the numbers of neurons in the input layer, hidden layer, and output layer, RBF centers and widths of neurons in the hidden layer, and linear weights connecting the hidden layer and output layer. Each neuron in the hidden layer of RBFNN produces a radically symmetric response around a node parameter vector called a 'center' (Du *et al*., 2010). As is well known, the performance of RBFNN critically relies on the selection of RBF centers. Several different learning approaches have been presented to determine the RBF centers of neurons in the hidden layer. First, the conventional strategy is a clustering technique including *k*-means clustering, fuzzy *k*-means clustering, and hierarchical clustering,

* Project supported by the National Natural Science Foundation of China (Nos. 41074003 and 60975039), the Opening Foundation of the Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences (No. IIP2010-1), and the Youth Science Foundation of China University of Mining and Technology (Nos. 2008A045 and 2009A053)

through which RBF centers are selected randomly from input data and adjusted constantly by the clustering algorithms until they no longer change (Zhang and Zhang, 2004; Park *et al*., 2011; Shen *et al*., 2011). However, these algorithms cannot determine the number of hidden neurons when no prior knowledge is provided, and usually result in a large number of RBF centers. Secondly, orthogonal least squares (OLS) is regarded as a popular approach to selecting RBF centers, based on their contributions to maximizing the model error reduction ratio (Chen *et al*., 1991; Su *et al*., 2009). Furthermore, the recursive OLS algorithm and the fast recursive algorithm (FRA) have also been presented for the selection of RBF centers (Yu *et al*., 1997; Du *et al*., 2008). Although these algorithms are of fast convergent speed, they cannot avoid local minima and easily produce suboptimal solutions. Finally, in recent years, evolution algorithms, such as the genetic algorithm (GA) and differential evolution (DE), have been adopted widely to optimize RBF centers (Wu *et al*., 2008; Qu *et al*., 2009; Coelho and Santos, 2011). Although the more reduced structure can be achieved through evolution algorithms, a better generalization performance cannot usually be guaranteed.

AP clustering, as a new clustering technique based on message passing, has been proposed by Frey and Dueck (2007). The AP clustering algorithm has been shown to be very useful for many applications in face images, gene expression, and text summarization. Unlike previous methods, AP simultaneously considers all data points as potential exemplars, and recursively transmits real-valued messages along the edges of the network until a good set of centers is generated (Xia *et al*., 2008).

Based on the above analysis, in this paper we propose a novel optimizing method for RBFNN based on rough sets (RS) and the AP clustering algorithm. To determine the centers and widths of RBF units, different from the conventional methods, the proposed method can find the centers and widths of clusters automatically, without defining the number of clusters through the AP clustering algorithm. Then these parameters are transferred to the hidden layer of RBFNN as the centers and widths of RBF units. Meanwhile, the number of the neurons of RBF units equals the number of RBF units. In addition, to eliminate noise and redundant attributes of datasets, attribute reduction (AR) based on RS is used to reduce the datasets and determine the number of the neurons in the input layer.

2 Structure of RBFNN

RBFNN consists of three different layers with a feedforward architecture: input layer, hidden layer, and output layer. The structure of RBFNN is shown in Fig. 1. Here the input layer is a set of m neurons, which accept the elements of an m-dimensional input vector. The input neurons are fully connected to a hidden layer that is composed of h hidden neurons, called RBF units. RBF units are connected directly to all the elements in the output layer, which activates the response of the neural network to the input pattern.

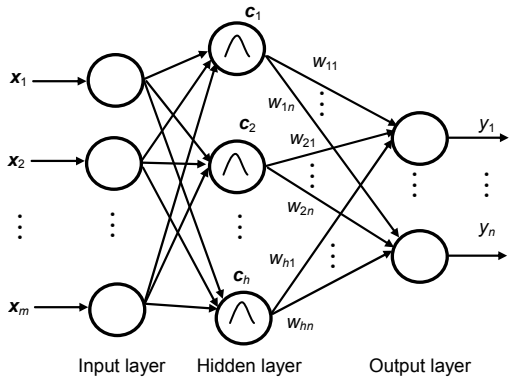

Fig. 1 The structure of a radial basis function neural network (RBFNN)

Of particular note is the hidden layer, in which each RBF unit contains an RBF. The Gaussian function is most frequently used as the RBF basis function:

$$\phi_i(\boldsymbol{x}) = \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{c}_i\|^2}{2\sigma_i^2}\right), \quad i=1,2,\ldots,h, \qquad (1)$$

where $\|\boldsymbol{x}-\boldsymbol{c}_i\|$ is the Euclidean distance between input vector $\boldsymbol{x}$ and center $\boldsymbol{c}_i$, and σ_i represents the width of the ith RBF unit. The Gaussian function, which has the property of noise suppression, is bounded, strictly positive, and continuous. Thus, for a given input pattern, only the RBF units whose centers are close to the input pattern will produce nonzero activation values to the input pattern (Xia *et al*., 2008).

The output of RBFNN is a linear combination of the responses of RBF units in the hidden layer. The values for the output neurons can be calculated as

$$y_k(\boldsymbol{x}) = \sum_{j=1}^{h} w_{jk}\phi_j(\boldsymbol{x}), \quad k = 1,2,\cdots,n, \qquad (2)$$

where $y_k(\boldsymbol{x})$ is the output of the kth neuron in the output layer, w_{jk} represents the weight from the jth neuron of the hidden layer to the kth neuron of the output layer, and n is the number of neurons in the output layer.

3 The proposed method

When an RBFNN is constructed, it is important to determine its main parameters including the number of neurons in the input layer, and centers and widths of RBF units in the hidden layer. In this section, a novel optimizing method is proposed for the RBFNN to determine these parameters of RBFNN.

3.1 Attribute reduction

In general, the number of neurons in the input layer is equal to the dimensionality of the input vector. However, the given datasets frequently involve some noise data and redundant attributes, which will weaken the generalization ability of the classifier and increase its training time. Thus, it is primarily important to eliminate noise data and redundant attributes before datasets are inputted to the classifier. To solve these problems, in this study, rough sets (RS) are adopted as the preprocessor to eliminate noise data and redundant attributes. Then the number of neurons in the input layer is equal to the number of attributions after AR.

RS theory, first introduced by Pawlak in 1982 (Pawlak *et al.*, 1995), has proved to be a powerful tool to deal with vagueness and uncertainty of information. RS theory is a mathematical approach to managing vague and uncertain data or problems related to information systems, indiscernibility relations and classification, attribute dependence and approximation accuracy, reduction and core attribute sets, and decision rules (Cheng *et al*., 2010). AR based on RS is a process of finding an optimal subset of all attributes according to a certain criterion so that the attribute subset is good enough to represent the classification relation of data. This preserves the quality of sorting after AR and requires no human input or domain knowledge other than the given datasets. The algorithm for AR of the input vector is as follows.

Algorithm 1 AR based on RS theory

Input: the datasets including sample attributes and classes.

Output: the reduced datasets.

Step 1: Initialization. Partition condition attributes and decision attribution of the datasets. The iteration variable i=1.

Step 2: Remove the ith condition attribute of datasets. According to the remaining condition attributes and decision attribution, analyze if the decision attribute indicates correctly and unambiguously the classification of condition attributes. If not, the ith condition attribute cannot be removed. Otherwise, this attribute is marked with the deleting label.

Step 3: i=i+1. If i is not larger than the number of the original condition attributes, return to step 2. Otherwise, remove the attributes of the original datasets with the deleting labels, and then the remaining attributes and decision attribution comprise the reduced datasets, and exit Algorithm 1.

3.2 Determining the centers and widths of RBF units

Of all the parameters, the centers of RBF units are the most important. Once the centers of RBF units are determined, the RBFNN is regarded as a linear transformation from the input to the output. Considering that the centers of clustering need to be evaluated in advance in the conventional methods, in this study we propose a novel method to search automatically for the centers and widths of RBF units using the AP clustering algorithm without defining the number of clustering centers.

In view of the advantage of AP clustering, we use it to search automatically for the centers of RBF units. The algorithm determining the number of centers and the widths of RBF units by AP clustering is shown as follows.

Algorithm 2 AP clustering for determining the centers and widths of RBF units

Input:
DS(N): the datasets including N samples.
$S(i, j)$: the similarity of point i to point j.
$P(j)$: the preference of point j, which indicates if point j is chosen as a cluster center.
Output: centers and widths of RBF units by AP clustering.

Step 1: Initialization. Define the maximum number of iterations MaxN and damping factor lam. Initialize similarity matrix $\boldsymbol{S}_{N\times N}$ (S_{ij}=$S(i, j)$) according to the similarity of the data points.

Step 2: Calculate the responsibility and availability according to the following equations:

$$R(i,k)=S(i,k)-\max_{j}\{A(i,j)+S(i,j)\},\quad j=1,2,\cdots,N,\ \ j\neq k, \tag{3}$$

$$A(i,k)=\min\left\{0,\ R(k,k)+\sum_{j}\max(0,R(j,k))\right\},\quad j=1,2,\cdots,N,\ \ j\neq i,k, \tag{4}$$

$$R(k,k)=P(k)-\max_{j}\{A(k,j)+S(k,j)\},\quad j=1,2,\cdots,N,\ \ j\neq k, \tag{5}$$

where $R(i, k)$ represents the responsibility from data point i to candidate exemplar point k, and $A(i, k)$ is the availability from candidate exemplar point k to point i.

Step 3: Evaluation. Judge if point k is the center of clustering, using the following rule:

$$R(k,k)+A(k,k)>0. \tag{6}$$

Step 4: Update the $R(i, k)$ and $A(i, k)$:

$$R(i,k)=(1-\text{lam})\cdot R(i,k)+\text{lam}\cdot R(i-1,k), \tag{7}$$

$$A(i,k)=(1-\text{lam})\cdot A(i,k)+\text{lam}\cdot A(i-1,k). \tag{8}$$

Step 5: If the terminal conditions of AP clustering are met, go to step 6; otherwise, return to step 2.

Step 6: Record the centers and their number according to the results of AP clustering, and calculate the widths of the clustering centers:

$$\text{widths}(i)=\sqrt{\frac{1}{\text{NC}_i}\sum_{j=1}^{\text{NC}_i}\|\boldsymbol{x}_j-\boldsymbol{c}_i\|^2}, \tag{9}$$

where $\boldsymbol{c}_i$ is the vector of the ith center, NC_i represents the number of data points in the ith center, $\boldsymbol{x}_j$ is the jth data point, and $\|\cdot\|$ represents the Euclidean distance between two data points. Then exit Algorithm 2.

3.3 Description of the proposed method

The proposed algorithm for training the RBFNN is stated below.

Algorithm 3 The proposed algorithm for training the RBFNN
Input: the datasets including sample attributes and classes.
Output: the results of RBFNN classification.

Step 1: AR. Reduce the datasets through AR based on RS. Determine the number of neurons of the input layer according to the number of reduced attributions.

Step 2: AP clustering. Calculate the centers and widths of clusters, and transfer these parameters to the RBF units.

Step 3: Calculate the output of the hidden layer according to the centers and widths of the RBF units.

Step 4: Evaluate the weights between the hidden layer and output layer using the least square method (LSM) according to the output of the RBF units and the desired output.

Step 5: Compute the correct rate of RBFNN classification. Then exit Algorithm 3.

4 Experimental results

In this section we present three groups of experiments on several datasets from the UCI Machine Learning Repository (Blake and Merz, 1998). The first is used to illustrate the characteristics of the proposed method, and the other two are presented to compare the classification ability of our method with that of other methods.

4.1 Experiment 1

Four datasets including Iris, Wine, Crx, and Zoo are used. Table 1 shows the sample size, the number of classes, and the number of attributions of these four datasets.

Initially, the data including training data and testing data are processed through AR based on RS. The numbers of attributions before and after AR are listed in Table 2. The number of attributions of Iris

datasets does not change obviously, as the values of condition attributions of Iris datasets are all real, and thus in the AR algorithm the datasets are considered to have more patterns so that their attributions cannot be reduced obviously. In contrast, the numbers of attributions of other datasets are reduced obviously.

Next, the datasets after AR are equally split into training datasets and testing datasets. Training sets are used to train the RBFNN and testing sets are used to verify the generalization ability of the network. In this phase, we first use the AP clustering algorithm to determine the number, centers, and widths of RBF units. As we know, among all the parameters of AP, the most important is the preference (P), which directly influences the number of clusters. Note that P is negative. In our experiments, we select three different values of P to deal with the datasets, including the median of similarity matrix $\boldsymbol{S}$ (denoted as median($\boldsymbol{S}$)), its half (median($\boldsymbol{S}$)/2), and its double (2*median($\boldsymbol{S}$)). The results of clustering using the different values of P are shown in Table 3. When P increases, the number of clustering centers becomes smaller, regardless of the condition (with or without AR). The number of clustering centers with AR is almost always smaller than that without AR (except when P=2*median($\boldsymbol{S}$)), which indicates that with AR some noise and redundant attributes of the datasets can be eliminated. The running time of the AP algorithm with AR is less than that without AR when P=median($\boldsymbol{S}$)/2. In other conditions, however, the running time of the AP algorithm with AR is more than that without AR.

To illustrate more clearly the convergence of the AP algorithm, we provide Figs. 2 and 3 to describe the iteration process of the AP algorithm under varying conditions. Comparison of Figs. 2 and 3 shows that the performance of the AP algorithm increases when P increases. Meanwhile, the AP algorithm with AR almost always has better fitness (net similarity) than that without AR, especially when testing Wine and Crx datasets. This indicates that AP clustering benefits from AR.

Table 1 Parameters of the four datasets

Dataset	Number of classes	Number of samples	Number of attributions
Iris	3	150	4
Wine	3	178	13
Crx	2	690	15
Zoo	7	214	16

Table 2 Attribute reduction (AR) based on rough sets

Dataset	Number of attributions	
	Before AR	After AR
Iris	4	3
Wine	13	2
Crx	15	5
Zoo	16	6

Table 3 The results of affinity propagation clustering for three different values of preference (P)

Dataset	Number of classes	Number of clustering centers					
		Without AR			With AR		
		P=median($\boldsymbol{S}$)/2	P=median($\boldsymbol{S}$)	P=2*median($\boldsymbol{S}$)	P=median($\boldsymbol{S}$)/2	P=median($\boldsymbol{S}$)	P=2*median($\boldsymbol{S}$)
Iris	3	11	9	4	10	7	5
Wine	3	28	14	7	15	9	7
Crx	2	68	44	26	59	40	28
Zoo	7	17	9	7	11	8	8
Dataset	Number of classes	Time (s)					
		Without AR			With AR		
		P=median($\boldsymbol{S}$)/2	P=median($\boldsymbol{S}$)	P=2*median($\boldsymbol{S}$)	P=median($\boldsymbol{S}$)/2	P=median($\boldsymbol{S}$)	P=2*median($\boldsymbol{S}$)
Iris	3	5.625	5.125	4.609	4.797	5.172	5.391
Wine	3	5.750	6.656	4.250	4.813	4.437	4.422
Crx	2	31.266	14.594	15.547	15.547	35.922	18.516
Zoo	7	15.750	7.094	7.002	11.235	9.062	8.782
Dataset	Number of classes	Number of iterations					
		Without AR			With AR		
		P=median($\boldsymbol{S}$)/2	P=median($\boldsymbol{S}$)	P=2*median($\boldsymbol{S}$)	P=median($\boldsymbol{S}$)/2	P=median($\boldsymbol{S}$)	P=2*median($\boldsymbol{S}$)
Iris	3	181	154	138	152	158	178
Wine	3	169	220	135	157	140	141
Crx	2	184	170	164	176	432	217
Zoo	7	578	253	235	427	341	337

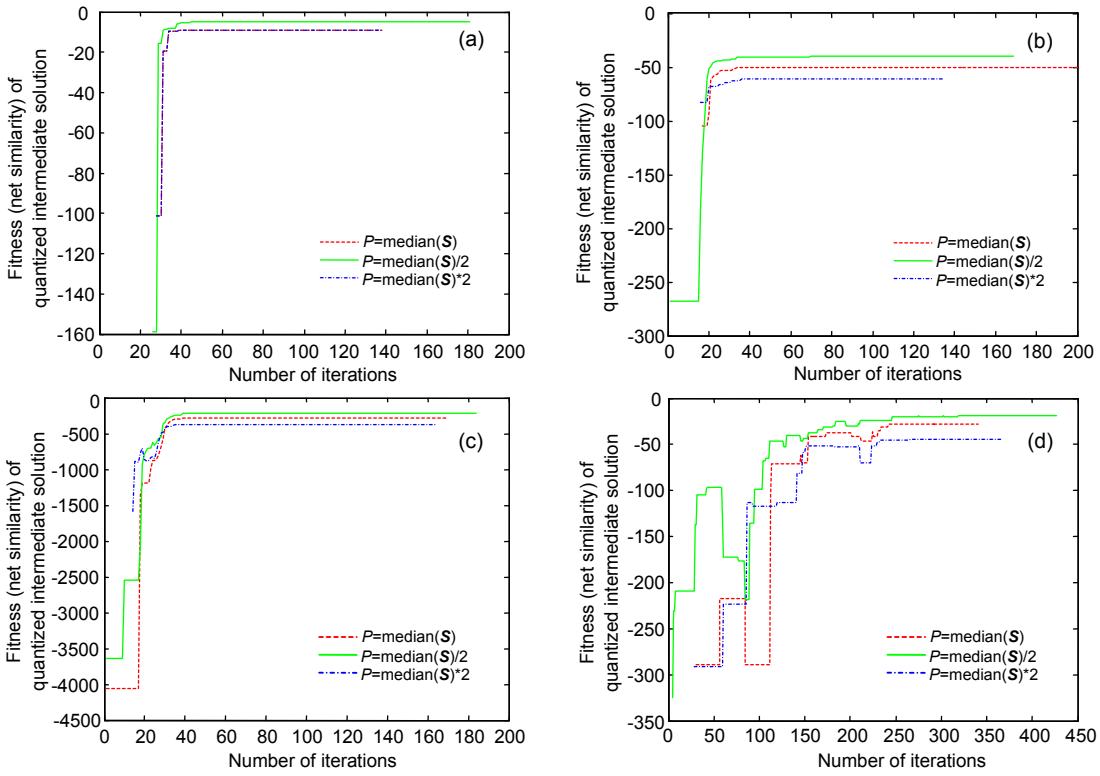

Fig. 2 The iteration process of the affinity propagation algorithm without attribute reduction
(a) Iris datasets; (b) Wine datasets; (c) Crx datasets; (d) Zoo datasets

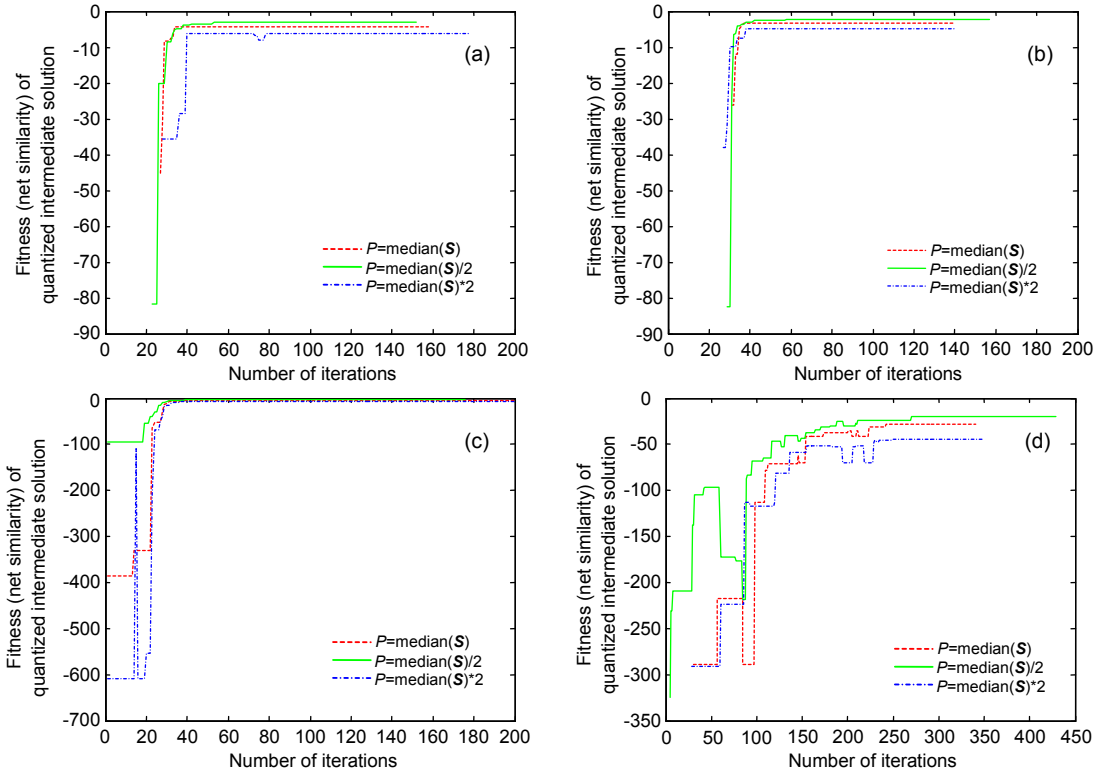

Fig. 3 The iteration process of the affinity propagation algorithm with attribute reduction
(a) Iris datasets; (b) Wine datasets; (c) Crx datasets; (d) Zoo datasets

According to the results of the AP clustering algorithm, we can calculate the centers, their widths and numbers, and transfer these parameters to the RBF units of RBFNN. According to Eq. (1), we can evaluate the output of RBF units. In the next phase of training, the main work is to estimate the weights connecting the hidden layer and output layer, depending on the output of the RBF units. Then the weights are adjusted according to LSM, depending on the mean-square error between the output of RBFNN and the expected output of the samples. Table 4 lists the results of calculating the centers of the RBF units, achieved by the proposed method and *k*-means clustering. The proposed method gives better results than *k*-means clustering. Meanwhile, the classification accuracy rate of RBFNN optimized by the proposed method with AR is very close to that without AR.

Table 4 The classification accuracy rate of RBFNN

Method	Classification accuracy rate (%)			
	Iris	Wine	Crx	Zoo
AP without AR				
P=media(s)/2	96.00	93.26	87.83	88.24
P=media(s)	96.00	93.26	88.98	88.24
P=media(s)*2	93.33	94.38	88.12	94.12
AP with AR				
P=media(s)/2	96.00	89.89	87.29	86.27
P=media(s)	96.00	91.01	88.41	88.24
P=media(s)*2	94.67	93.26	87.83	92.17
k-means without AR	86.67	86.52	86.97	84.31
k-means with AR	93.33	88.76	86.38	82.35

AP: affinity propagation; AR: attribute reduction

4.2 Experiment 2

The second group contains three datasets: the Pima Indian diabetes problem (dataset 1), the Wisconsin breast cancer problem (dataset 2), and the Ionosphere data classification problem (dataset 3) (Mao and Huang, 2005). The first two datasets are tested by cross-validation to estimate the classification accuracy of RBFNN. Table 5 lists the classification accuracy results.

As shown in Table 5, our method achieves a similar performance to the method proposed by Mao and Huang (2005). Specifically, the experiment of the eight-fold on dataset 2 achieves a classification accuracy of 100%, and its average performance is superior to that of Mao and Huang (2005)'s method.

Table 5 The classification accuracy of the second group of datasets

Dataset No.	Classification accuracy (%)	
	Mao and Huang (2005)	Our method
1	78.39±5.2	78.13±4.69
2	97.4±2.2	97.96±2.04
3	98.0	98.0

4.3 Experiment 3

Cleveland data on cardiology patients are used in the third experiment. The dataset has 303 samples with 13 input attributes and one output attribute, and is divided into a training dataset and a test dataset consisting of 152 and 151 samples, respectively.

To compare our method with the methods mentioned in Mao (2002), we also perform a ten-fold cross-validation. Table 6 gives a comparison of our results with these methods, showing that our method is superior to these methods, except linear discrimination analysis and Naïve Bayes, with respect to average accuracy.

Table 6 Comparison with the methods mentioned in Mao (2002) for the Cleveland data on cardiology patients

Method	Accuracy (%)
Our method	82.12±1.34
Mao (2002)	81.8
Linear discrimination analysis	84.5
Naïve Bayes	82.5–83.4
SVM (five-fold cross-validation)	81.3
CART (classification and regression tree)	80.8
C4.5 (Quinlan's induction decision, five-fold cross-validation)	77.8

In general, the performance of our method is reasonably good. This also indicates that the AP clustering algorithm can effectively find the centers of the RBF units without a priori knowledge about the number of centers.

5 Conclusions

The aim of this paper is to introduce a novel method optimizing RBFNN based on RS theory and the AP clustering algorithm. Through AR based on RS theory, noise data and redundant attributions are

eliminated, and the number of neurons in the input layer is also reduced. Furthermore, the AP clustering algorithm, in which there is no need to define the number of clusters in advance, can determine automatically the centers and widths of the RBF units. The better performance obtained by the proposed method is shown by simulation results. In the future, we will focus on improving the performance of the AR algorithm and adjusting, dynamically, the parameters of AP clustering according to the output of RBFNN.

References

Beyhan, S., Alci, M., 2010. Stable modeling based control methods using a new RBF network. *ISA Trans.*, **49**(4): 510-518. [doi:10.1016/j.isatra.2010.04.005]

Blake, C.L., Merz, C.J., 1998. UCI Repository of Machine Learning Databases, 1998. Department of Information and Computer Science, University of California, Irvine, CA. Available from http://www.ics.uci.edu/mlearn/Machine-Learning.html [Accessed on Sept. 23, 2011].

Chen, S., Cowan, C.F.N., Grant, P.M., 1991. Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Trans. Neur. Network*, **2**(2):302-309. [doi:10.1109/72.80341]

Cheng, J.H., Chen, H.P., Lin, Y.M., 2010. A hybrid forecast marketing timing model based on probabilistic neural network, rough set and C4.5. *Exp. Syst. Appl.*, **37**(3): 1814-1820. [doi:10.1016/j.eswa.2009.07.019]

Coelho, L.S., Santos, A.A.P., 2011. A RBF neural network model with GARCH errors: application to electricity price forecasting. *Electr. Power Syst. Res.*, **81**(1):74-83. [doi:10.1016/j.epsr.2010.07.015]

Du, D.J., Fei, M.R., Li, L.X., 2008. Radial-basis-function neural network based on fast recursive algorithm and its application. *Control Theory Appl.*, **25**(5):827-830.

Du, D.J., Li, K., Fei, M.R., 2010. A fast multi-output RBF neural network construction method. *Neurocomputing*, **73**(10-12):2196-2202. [doi:10.1016/j.neucom.2010.01.014]

Frey, B.J., Dueck, D., 2007. Clustering by passing messages between data points. *Science*, **315**(5814):972-976. [doi:10.1126/science.1136800]

Guerra, A.F., Coelho, L., 2008. Multi-step ahead nonlinear identification of Lorenz's chaotic system using radial basis neural network with learning by clustering and particle swarm optimization. *Chaos Sol. Fract.*, **35**(5): 967-979. [doi:10.1016/j.chaos.2006.05.077]

Hou, M.Z., Han, X.L., 2011. The multidimensional function approximation based on constructive wavelet RBF neural network. *Appl. Soft Comput.*, **11**(2):2173-2177. [doi:10.1016/j.asoc.2010.07.016]

Jayasree, T., Devaraj, D., Sukanesh, R., 2010. Power quality disturbance classification using Hilbert transform and RBF networks. *Neurocomputing*, **73**(7-9):1451-1456. [doi:10.1016/j.neucom.2009.11.008]

Jing, X.Y., Yao, Y.F., Yang, J.Y., Zhang, D., 2008. A novel face recognition approach based on kernel discriminative common vectors (KDCV) feature extraction and RBF neural network. *Neurocomputing*, **71**(13-15):3044-3048. [doi:10.1016/j.neucom.2007.08.027]

Lee, C.M., Ko, C.N., 2009. Time series prediction using RBF neural networks with a nonlinear time-varying evolution PSO algorithm. *Neurocomputing*, **73**(1-3):449-460. [doi:10.1016/j.neucom.2009.07.005]

Mao, K.Z., 2002. RBF neural network center selection based on Fisher ratio class separability measure. *IEEE Trans. Neur. Networks*, **13**(5):1211-1217. [doi:10.1109/TNN.2002.1031953]

Mao, K.Z., Huang, G.B., 2005. Neuron selection for RBF neural network classifier based on data structure preserving criterion. *IEEE Trans. Neur. Networks*, **16**(6): 1531-1540. [doi:10.1109/TNN.2005.853575]

Park, H.S., Chung, Y.D., Oh, S.K., Pedrycz, W., Kim, H.K., 2011. Design of information granule-oriented RBF neural networks and its application to power supply for high-field magnet. *Eng. Appl. Artif. Intell.*, **24**(3):543-554. [doi:10.1016/j.engappai.2010.11.001]

Pawlak, Z., Grzymala-Busse, J., Slowinski, R., Ziarko, W., 1995. Rough sets. *Commun. ACM*, **38**(11):88-95. [doi:10.1145/219717.219791]

Qu, N., Mi, H., Wang, B., Ren, Y.L., 2009. Application of GA-RBF networks to the nondestructive determination of active component in pharmaceutical powder by NIR spectroscopy. *J. Taiwan Inst. Chem. Eng.*, **40**(2):162-167. [doi:10.1016/j.jtice.2008.08.002]

Shen, W., Guo, X.P., Wu, C., Wu, D.S., 2011. Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm. *Knowl.-Based Syst.*, **24**(3):378-385. [doi:10.1016/j.knosys.2010.11.001]

Su, S., Duan, X.Z., Chan, W.L., Li, Z.H., 2009. Erroneous measurement detection in substation automation system using OLS based RBF neural network. *Int. J. Electr. Power Energy Syst.*, **31**(7-8):351-355. [doi:10.1016/j.ijepes.2009.03.008]

Wu, X.J., Zhu, X.J., Cao, G.Y., Tu, H.Y., 2008. Predictive control of SOFC based on a GA-RBF neural network model. *J. Power Sources*, **179**(1):232-239. [doi:10.1016/j.jpowsour.2007.12.036]

Xia, D.Y., Wu, F., Zhang, X.Q., Zhuang, Y.T., 2008. Local and global approaches of affinity propagation clustering for large scale data. *J. Zhejiang Univ.-Sci. A*, **9**(10):1373-1381. [doi:10.1631/jzus.A0720058]

Yu, D.L., Gomm, J.B., Williams, D., 1997. A recursive orthogonal least squares algorithm for training RBF networks. *Neur. Process. Lett.*, **5**(3):167-176. [doi:10.1023/A:1009622226531]

Zhang, A.S., Zhang, L., 2004. RBF neural networks for the prediction of building interference effects. *Comput. Struct.*, **82**(27):2333-2339. [doi:10.1016/j.compstruc.2004.05.014]