2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Using Self-organizing Incremental Neural Network (SOINN) For Radial Basis Function Networks

Jie Lu, Furao Shen*, and Jinxi Zhao

Abstract— **This paper presents a batch learning algorithm and an online learning algorithm for radial basis function networks based on the self-organizing incremental neural network (SOINN), together referred to as SOINN-RBF. The batch SOINN-RBF is a combination of SOINN and least square algorithm. It achieves a comparable performance with SVM for regression. The online SOINN-RBF is based on the self-adaption procedure of SOINN and adopts the growing and pruning strategy of the minimal resource allocation network (MRAN). The growing and pruning criteria use the redefined *significance*, which is originally introduced by the growing and pruning algorithm for RBF (GGAP-RBF). Simulation results for both artificial and real-world data sets show that, comparing with other online algorithms, the online SOINN-RBF has comparable approximation accuracy, network compactness and better learning efficiency.**

I. Introduction

Radial basis function networks are special-designed, three-layer neural networks. The hidden (second) layer maps the input samples into a new high-dimensional space and the output layer performs linear operations in this high-dimensional space. According to Cover's theory, a low-dimensional pattern-classification problem is more likely to be linearly separable when casted in a high-dimensional space [1]. The RBF network performs well in many applications such as time series prediction [2] and face recognition [3]. These applications are based on the excellent properties of the RBF network.

The hidden layer consists of the hidden units, which are the representative nodes picked from input samples, can be chosen appropriately by methods like K-means. The weights of the linear connections between the hidden layer and output layer can be trained by the least squares (LS) methods. The K-means and LS algorithms are simple to apply and achieve good enough performances. But they all have a drawback, the entire training data must be ready before the training procedure. They can't handle the situation that the input data come in serially. Several algorithms have been proposed to overcome this drawback – [4][5][6][7].

Platt firstly proposed the Resource-Allocating Network, referred to as RAN [4]. RAN starts with no hidden unit and allocates a new one whenever an *unusual* sample arrives. The *unusual* means a input sample is far away from the hidden units and causes a big output error. Otherwise the network updates the parameters of existing units using standard least mean square (LMS) algorithms. Kadirkamanathan and Niranjan enhanced RAN by replacing the LMS algorithm with the extended Kalman filter(EKF) [5]. The resulting network is called RANEKF and converges faster than RAN. RANEKF's network is also more compact because during the growing procedure, the faster the convergence is the less hidden units will be allocated later. But both RAN and RANEKF never remove the inactive units, making the network unnecessarily large. Yingwei et al. introduced a pruning strategy to remove those units that contribute relatively little to the whole network[6]. This network is the minimal resource allocation network (M-RAN) which also uses a sliding window to smooth the growing and pruning procedure.

Huang et al. proposed the generalized growing and pruning algorithm for RBF (GGAP-RBF) [7] which outperforms all the algorithm listed above in terms of generalization error, network compactness and learning speed. This algorithm uses the normalized *significance* in the growing and pruning criteria and updates only the nearest unit instead of the whole network. The *significance* of a hidden unit is its average information over the whole input space. This means that the distribution of the input space needs to be learned before the online(sequential) learning producer, which dose't conform with the definition of online learning. What makes it worse is that the distribution is in the form of joint distribution of each dimension, making the implement of this algorithm very unfriendly. Therefore, in this paper, we category GGAP-RBF as a semi-online learning algorithm.

The SOINN-RBF algorithms proposed in this paper are based on the Self Organization Incremental Neural Network (SOINN)[8][9].SOINN itself is an online learning algorithm which obtains an excellent topology representation of the input space. The output network of SOINN is just suitable for the hidden layer of RBF network. By simply replacing K-means with SOINN in the standard RBF network we get the batch SOINN-RBF. In the online SOINN-RBF we adopt sliding window [6] technology to smooth the growing and pruning procedure, use EKF [5] to train the parameter of output layer and redefine the *significance* to ensure the online learning. We compare the performance of SOINN-RBF with SVM for regression [10], MRAN and GGAP-RBF in three problems, namely Double Moon–an artificial data set, Thyroid–a case from PROBEN1 and Insurance–a data set used in the CoIL 2000 Challenge.

The rest of paper is organized as follows. Section II gives brief descriptions of RBF, SOINN and batch SOINN-RBF. Section III gives details of online SOINN-RBF. Section IV

Jie Lu (www.jerrylu@gmail.com), Furao Shen (frshen@nju.edu.cn) and Jinxi Zhao (jxzhao@nju.edu.cn) are with Robotic Intelligence and Neural Computing Laboratory(RINC.Lab), the State Key Laboratory for Novel Software Technology at School of Computer Science and Technology, Nanjing University, Nanjing, Jiangsu, 210046, P.R.China. Corresponding author is Furao Shen.

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

compare SOINN-RBF with SVM for regression, MRAN and GGAP-RBF. The conclusions are summarized in section V.

II. BATCH SOINN-RBF

The typical structure of the RBF networks is shown in Fig. 1. Let $x \in \mathcal{R}^m$ be the input sample and $y \in \mathcal{R}$ be the output. The network can be considered as a mapping from $\mathcal{R}^m$ to $\mathcal{R}$ which is written as

$$f(x) = \sum_{k=1}^{K} w_k \varphi(x, \mu_k). \tag{1}$$

We chooses the Gaussian kernel as the radial function,

$$\varphi(x, \mu_k) = exp\left(-\frac{1}{2\sigma_k^2}||x - \mu_k||^2\right), \text{k=1,2,...,K} \tag{2}$$

where μ_k is the center of hidden unit k and σ_k is the radius of hidden unit k.

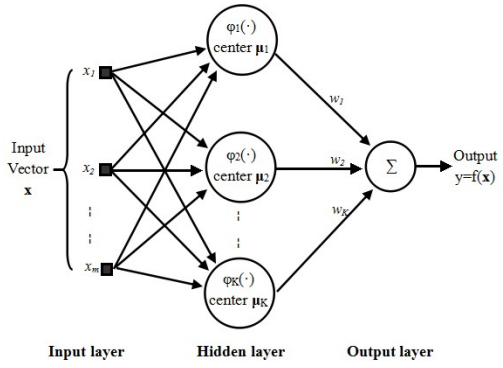

Fig. 1. Structure of a typical RBF network

We can see from function (1) and (2) that the network is characterized by the hidden units' centers, radii and weights. A sample from the input space is expanded to the higher space by measuring the *distance* to each hidden units and the output is a linear combination of these *distance*. Therefore, the key of learning an RBF network is to choose an appropriate set of hidden units that can approximate the distribution of the training data.

The algorithm SOINN, which is based on Growing Neural Gas (GNG) [10] and Self Organizing Map (SOM) [11], is adequate to this work. For online data with a complex geometrical distribution, it can form a topological representation of the input data in a self-organizing way. In addition, the network's size doesn't need to be assigned manually and the algorithm is very robust to noises. Thus, we choose SOINN to learn the hidden units of the RBF network.

The main procedure of SOINN is the maintenance of the network's units and their connections. When a input sample arrives, it firstly find the two nearest units and respectively compare the distances with their radii. If the new-coming sample is within any of the unit's dominant area, the connection between the two units is established and the nearest one will be pulled towards the sample. Otherwise, a new unit is added to the network. Fig. 2 shows these two situations. After this procedure the radii of these two units are adjusted to make sure they can cover all the connected neighbors. As the samples keep coming in, the network grows to fit the distribution of the input data. By introducing additional strategies like removing the oldest connections to break early-inappropriate connections and removing the isolated nodes to eliminate the noises, the resulting network is a SOINN network.

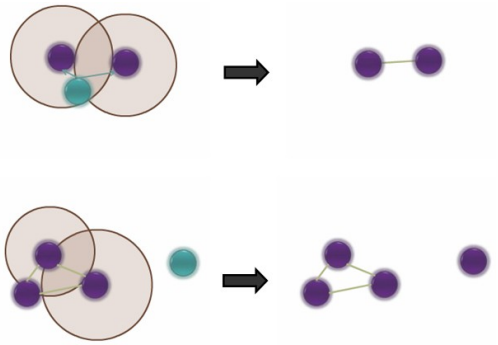

Fig. 2. Two situations of SOINN's growing procedure.The picture above shows the situation that when a input sample lies within the radii of two nearest units, the connection between these two units will be established and the nearest one will be pulled to the input sample. The following picture shows the other situation, a new unit will be allocated when the input sample is far from its two nearest units.

As the hidden units have been learned by SOINN, the parameters of the output layer will be trained by the method of least square. The mapping from the hidden layer to the output layer can be written as

$$R\hat{w} = Y \tag{3}$$

where

$$R = [\Phi(x_1), \Phi(x_2), ..., \Phi(x_n)]^T, \tag{4}$$

$$\Phi(x_i) = [\varphi(x_i, \mu_1), \varphi(x_i, \mu_2), ..., \varphi(x_i, \mu_K)], \tag{5}$$

$$\hat{w} = [w_1, w_2, ..., w_K]^T, \tag{6}$$

$$Y = [y_1, y_2, ..., y_n]^T. \tag{7}$$

R is the hidden layer's output of all input samples. $\hat{w}$ are hidden units' weights. Y is the true outputs of all inputs. By multiplying both side of (3) with R^T we get

$$R^T R\hat{w} = R^T Y. \tag{8}$$

The solution of function (8) is the least square algorithm's output:

$$\hat{w} = (R^T R)^{-1} R^T Y, \tag{9}$$

With the hidden units learned by SOINN and the linear regression parameters trained by LS algorithm, we get the batch SOINN-RBF algorithm.

BATCH-SOINN-RBF(DATA)

1 [μ_{hidden},σ_{hidden}] = SOINN(data)
  ▷ $\mu_{hidden} = [\mu_1, \mu_2, ...\mu_K]^T$,$\sigma_{hidden} = [\sigma_1, \sigma_2, ...\sigma_K]^T$
2 w_{hidden} = LS(data,μ_{hidden},σ_{hidden})
  ▷ $w_{hidden} = [w_1, w_2, ...w_K]^T$

III. THE ONLINE SOINN-RBF

This section describes the details of online SOINN-RBF with the definition of the normalized *significance* first, following by the growing and pruning procedures. The pseudo-code of online SOINN-RBF is proposed in the end.

A. Normalized Significance

Supposing that function (1) is the correct input-output mapping of the network, the error caused by removing the hidden unit k is

$$E(x)_k = |w_k|\varphi_k(x, \mu_k). \tag{10}$$

The error over all n input samples is

$$E(x) = \frac{1}{n}\sum_{i=1}^{n} |w_k|\varphi_k(x_i, \mu_k). \tag{11}$$

The *significance* of the unit k is just the error caused by removing it. However in an online learning algorithm, the learned data should not be preserved. In this paper, we use the hidden units to approximate the distribution of input samples. Thus the normalized significance is written as

$$Sig_k = E_k \approx \frac{\sum_{i=1}^{K} m_i |w_k|\varphi_k(\mu_i, \mu_k)}{\sum_{i=1}^{K} m_i}, \tag{12}$$

where m_i is the number of input samples that find unit i as the nearest unit.

In online SOINN-RBF, every time when a new sample arrives, all hidden units' *significance* should be calculated again. The computation cost is $O(K^2)$ every iteration. This will slow down the algorithm when K becomes large. Fortunately there are two features we can take advantage of to reduce the computation cost. The first is the steeply shape of the Gaussian kernel $exp(-x^2/2\sigma^2)$, which approaches zero faster when $x \gg \sigma$. The units that are far away from unit k will not be taken into consideration. The second is the connections of the hidden units which learned by SOINN. They store the information of the units' neighborhood and can be used to fetch the nearby units. Therefore the *significance* used in this paper is

$$Sig_k = \frac{\sum_{i=1}^{K} connection(i,k) m_i |w_k|\varphi_k(\mu_i, \mu_k)}{\sum_{i=1}^{K} m_i}, \tag{13}$$

in which the $connection(i,k)$ equals 1 if unit i and unit k are connected, otherwise 0. In fact, if $connection(i,k)$ equals 0, there is no calculation in implementation. The computation cost falls to $O(d * K)$, where d is the average quantities of connected neighbors to an unit, which is much smaller than K.

B. Growing Procedure

Like the RAN[4] algorithm, SOINN-RBF needs to allocate new hidden units when the current network can't represent the new inputs. The criterion used in RAN is $\|x_n - \mu_{nr}\| > \epsilon_n \& |e_n| > e_{min}$ where x_n is a new input sample, μ_{nr} is the nearest unit to x_n, ϵ_n is the threshold of distance and e_n is the output error caused by x_n, e_{min} is the threshold of error. The meaning of this criterion is that if the distance from x_n to μ_{nr} is bigger than the threshold ϵ_n and the error is bigger than e_{min}, a new unit will be allocated. Otherwise RAN adjusts the parameters with LMS algorithms. Noticing that the criterion of SOINN is also based on the distance testing and by combining the error criterion with it we get the basic version of our growing criterion in SOINN-RBF.

$$\begin{cases} \|x_n - \mu_{nr1}\| > \sigma_{nr1} \text{ or } \|x_n - \mu_{nr2}\| > \sigma_{nr2} \\ |e_n| > e_{min} \end{cases} \tag{14}$$

where $nr1$ and $nr2$ are the nearest and the second-nearest units to x_n and

$$e_n = y_n - f(x_n). \tag{15}$$

In RAN, ϵ_n is not a constant value and decays exponentially. The exponential decaying of the distance criterion adds only the large-radius units at the beginning and as the samples keep coming in more units with smaller radii are allocated to fine tune the approximation. We adopt this strategy to enhance the distance criterion of SOINN-RBF as $\hat{\sigma_{nr1}} = (1 + e^{-n/\tau})\sigma_{nr1}$ and $\hat{\sigma_{nr2}} = (1 + e^{-n/\tau})\sigma_{nr2}$.

This paper also enhances the error by multiplying an *significance* of x_n

$$\hat{e_n} = \frac{\sum_{i=1}^{K} m_i \varphi_i(x_n, \mu_i)}{\sum_{i=1}^{K} m_i} e_n. \tag{16}$$

The *significance* idea of (16) is introduced by GGAP-RBF [7] and here we use the hidden units instead of the pre-learned inputs distribution.

In order to further smooth the growing procedure, the sliding window technology of [6] is used to resist the impulsive errors

$$e_{avg} = \frac{1}{W}\sum_{i=n-(W-1)}^{n} |\hat{e_i}| \tag{17}$$

where W is the size of sliding window and e_{avg} is the average error of the sliding window, which ensures that only when the network is really unsuitable can the network grow.

Thus, the growing criterion in this paper is

$$\begin{cases} \|x_n - \mu_{nr1}\| > \hat{\sigma_{nr1}} \text{ or } \|x_n - \mu_{nr2}\| > \hat{\sigma_{nr1}} \\ \hat{e_n} > e_{min} \\ e_{avg} > e_{min}^{'} \end{cases} \tag{18}$$

where $e_{min}^{'}$ is threshold of average error. If these criteria are satisfied, a new unit will be allocated with parameters

$$\begin{cases} \mu_{K+1} = x_n \\ \sigma_{K+1} = \|x_n - \mu_{nr1}\| \\ w_{K+1} = e_n \end{cases} \tag{19}$$

Otherwise the parameters will be updated. This paper updates the weights using EKF [5] and updates the center of the nearest unit using the method used in SOINN. The radii of the two winners are also updated to cover all the connected neighbors. The reason why we don't use EKF to update all the parameters is that EKF is very memory-consuming, making the algorithm much slower than its theoretical speed, especially when the dimension of inputs is high.

C. Pruning Procedure

The pruning strategy is the key to make a compact network. The algorithm can also accelerate by removing the inactive units. Like MRAN [6], a sliding window is used in this procedure to ensure that the pruned units are definitely insignificant. Therefore it enhances the smoothness of the pruning procedure and the stableness of the online learning algorithm. The pruning procedure is as follows, if the *significance* of unit k is less than a threshold for several consecutive observations, unit k will be labeled as inactive and should be removed. Given the threshold sig_{min} and the window length W , unit k will be pruned if

$$Sig_k < sig_{min} \tag{20}$$

is satisfied for W consecutive times.

D. The Online SOINN-RBF Algorithm

The online SOINN-RBF algorithm is described as follows:

ONLINE-SOINN-RBF(DATA)

▷ Initializing:
$\mu_1 \leftarrow x_1, \mu_2 \leftarrow x_2$
$\sigma_1 \leftarrow \|x_1 - x_2\|, \sigma_2 \leftarrow \|x_1 - x_2\|$
w_1,w_2 are trained by function (8)
$m_1 \leftarrow 1, m_2 \leftarrow 1, K \leftarrow 2$
$connection \leftarrow eye(2, 2)$
▷ Online Learning:
for $i \leftarrow 3$ **to** n
  do ▷ Compute the network's output:
    $f(x_i) \leftarrow \sum_{k=1}^{K} w_k \varphi(x_i, \mu_k)$
    ▷ Find the two nearest units to x_n:
    $nr1 \leftarrow argmin_{k \in [1:K]} \|x_n - \mu_k\|$
    $nr2 \leftarrow argmin_{k \in [1:K]/nr1} \|x_n - \mu_k\|$
    ▷ Compute the output error:
    $e_i \leftarrow y_i - f(x_i)$
    if criterion (18) is satisfied
      then ▷ Allocate a new hidden unit:
        $K \leftarrow K + 1$
        $\mu_K \leftarrow x_n$,$\sigma_K \leftarrow \|x_n - \mu_{nr1}\|$
        $w_K \leftarrow e_n$, $w_K \leftarrow 1$
        extend connection with zeros.
      else ▷ Update the winners parameters:
        $connection(nr1, nr2) \leftarrow 1$
        $connection(nr2, nr1) \leftarrow 1$
        $\mu_{nr1} \leftarrow (m_{nr1}\mu_(nr1) + x_i)/(m_{nr1} + 1)$
        $\sigma_{nr1} \leftarrow max_{k:connect\ to\ nr1} \|\mu_{nr1} - \mu_k\|$
        $\sigma_{nr2} \leftarrow max_{k:connect\ to\ nr2} \|\mu_{nr2} - \mu_k\|$
        adjust $w_1, w_2, ...w_K$ using EKF.
    ▷ Check the criteria for pruning:
    calculate hidden units' *significances* using (13)
    if unit k satisfies criterion (20) for W consecutive observations
      then remove the hidden unit k
        reduce the dimension of EKF
        reduce the dimension of connection

IV. Simulation Results

In this section, the performance comparison of SOINN-RBF with three other popular algorithms (SVM for regression, GGAP-RBF, and MRAN) are presented for three problems: 1) Double Moon, 2) Thyroid and 3) Insurance.

The learning accuracy measurement used in our experiments is root mean square error, defined as

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \|y_i - f(x_i)\|^2} \tag{21}$$

A. Double Moon

Double Moon is artificial data set for nonlinear pattern recognition algorithms. Fig. 3.a shows the distribution of it. It consists of a pair of half moons facing each other. The radius of each moon is 10 and the width of band area is 6. The bias in horizontal direction is 10 and in vertical direction is -6. This data set has 2000 train samples and 4000 test samples.

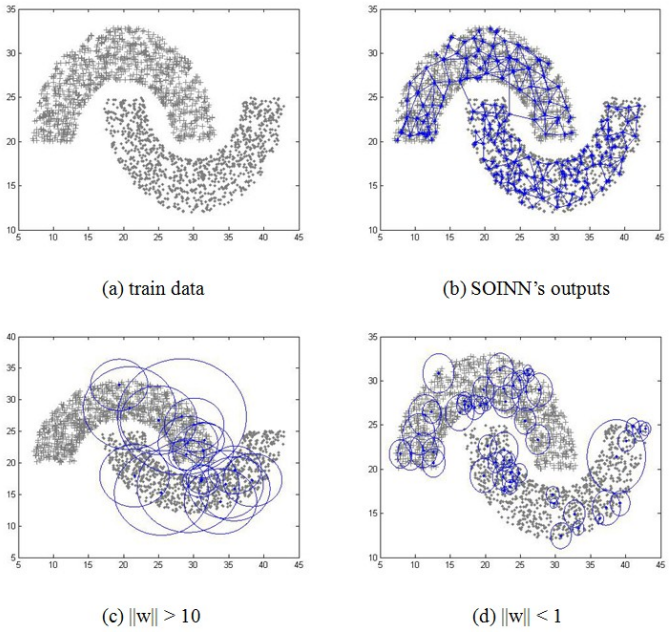

Fig. 3. Batch SOINN-RBF's results:(a) shows the distribution of train data. The blue pattern in (b) is the hidden units and their connections learned by SOINN. (c) is the units with $|w|$ bigger than 10. The size of circle represent unit's radius. (d) is the units with $|w|$ smaller than 1.

The parameters of online SOINN-RBF for this data set chosen as: $e_{min} = 0.001$, $e^{'}_{min} = 0.0005$, $sig_{min} = 0.0001$, $\tau = 10$, $W = 60$, $Q_0 = 0.001$, $P_0 = 0.96$.

The parameters of GGAP-RBF for this data set is: $e_{min} = 0.0001$, $\epsilon_{max} = 2$, $\epsilon_{min} = 0.01$, $\kappa = 0.4$, $\gamma = 0.999$, $Q_0 = 0.00001$, $P_0 = 0.99$. And the parameters of MRAN is: $e_{min} = 0.0001$, $e^{'}_{min} = 0.005$, $\epsilon_{max} = 2$, $\epsilon_{min} = 0.01$, $\kappa = 0.15$, $\gamma = 0.999$, $W = 150$, $Q_0 = 0.001$, $P_0 = 0.95$. The distribution information used in GGAP-RBF is learned before learning procedure. The first dimension is a gaussian distribution with center at 25 and the sigma of 15, the second

dimension is also a gaussian distribution with center at 20 and sigma of 10.

TABLE I
SIMULATION PERFORMANCE FOR DOUBLE MOON

Process	Algorithm	RMSE	Time(s)[a]	♯Neurons
Batch	SOINN-RBF	0.0443	2.90	289
	SVM-R[b]	0.0264	11.47	1590
Online	SOINN-RBF	0.1280	8.54	101
	GGAP-RBF	0.1187	22.73	95
	MRAN[c]	0.2235	76.45	212

[a] CPU:Intel i5-3470. Memory:4G. Platform:matlab R2012b.
[b] libsvm [12] with parameters '-s 4'
[c] MRAN is out of work.

Table I shows that the batch SOINN-RBF achieves comparable performance with SVM in approximation accuracy and performs better in learning speed and network compactness, even though SVM is run in C. The graphic representation of the network is shown in Fig. 3, from which we may explain the good performance of the batch SOINN-RBF. We can see from Fig. 3.b that the pattern of the input samples has been recognized accurately by SOINN. And this accurate pattern is the hidden units of the SOINN-RBF network. By comparing Fig. 3.c with Fig. 3.d, a remarkable phenomenon should be noticed: the units with big weights are the large ones and the units with small weights are the tiny ones. This is consistent with expecting result of exponential decaying used in distance criterion: the large units are main deciders of the network and the tiny ones fine tune the network. The batch procedure and the online procedure are identical in this respect. Thus, the elaborately-chose hidden units plus the well-tuned weights make the batch SOINN-RBF accurate and efficient.

To simplify the compare of online algorithms, GGAP-RBF is grouped as online learning algorithm. Table I shows that the online SOINN-RBF obtains similar results in approximation accuracy and network compactness with GGAP-RBF, yet with a faster speed. The online SOINN-RBF outperforms MRAN in all respects. The main reason of the effectiveness is that the online SOINN-RBF uses EKF to update the weights only, while others use EKF to update all the parameters. EKF is a memory-consuming algorithm. When input distribution is complex or the inputs' dimension is high, it will slow down the speed. Fig. 4. shows the growing procedures of hidden units. Fig. 5. shows the learning speed comparison of these online learning algorithms.

B. Thyroid

The thyroid is a data set from PROBEN1[1]. It consists of 7200 examples and 3772 out of which are for training. Each example is featured by 21 attributes and has 1 output. The parameters of online SOINN-RBF for this data set chosen as: $e_{min} = 0.0025$, $e^{'}_{min} = 0.025$, $sig_{min} = 0.0001$, $\tau = 200$, $W = 30$, $Q_0 = 0.001$, $P_0 = 0.99$.

[1]http://tracer.lcc.uma.es/problems/ann/ann.html

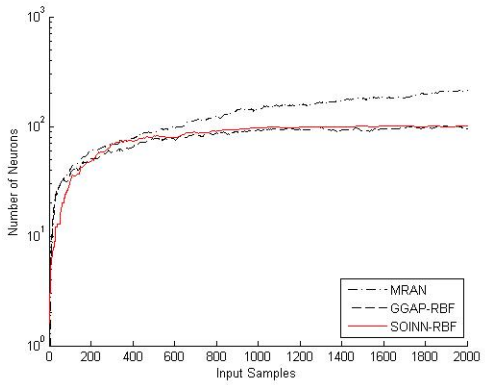

Fig. 4. Hidden units updating process for Double moon

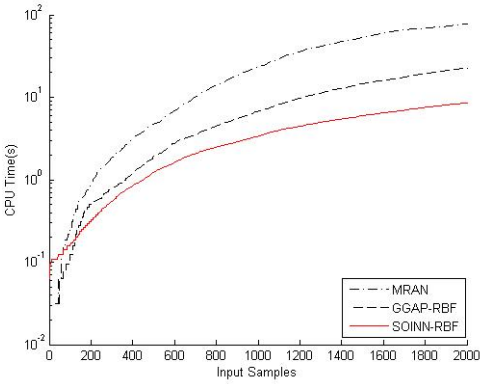

Fig. 5. Learning speed comparison for Double moon.

The parameters of GGAP-RBF for this data set is: $e_{min} = 0.0001$, $\epsilon_{max} = 11.5$, $\epsilon_{min} = 1.4$, $\kappa = 0.6$, $\gamma = 0.995$, $Q_0 = 0.00001$, $P_0 = 0.99$. The distribution information of the 21 attributes is learned before learning procedure. And the parameters of MRAN is: $e_{min} = 0.001$, $e^{'}_{min} = 0.01$, $\epsilon_{max} = 10$, $\epsilon_{min} = 0.8$, $\kappa = 0.6$, $\gamma = 0.995$, $W = 40$, $Q_0 = 0.001$, $P_0 = 0.95$.

TABLE II
SIMULATION PERFORMANCE FOR THYROID

Process	Algorithm	RMSE	Time(s)	♯Neurons
Batch	SOINN-RBF	0.3055	12.73	366
	SVM-R[a]	0.2999	20.11	1999
Online	SOINN-RBF	0.3589	3.90	29
	GGAP-RBF	0.3522	109.20	29
	MRAN	0.4422	1238.46	90

[a] libsvm with parameters '-s 4 -c 300'

The results of the simulation are shown in Table II. The batch SOINN-RBF gains the similar approximation accuracy with SVM for regression. The online SOINN-RBF achieves an amazing learning speed with the comparable accuracy and compactness. Fig. 6. shows the growing procedures of hidden units. Fig. 7. shows the learning speed comparison of these online learning algorithms.

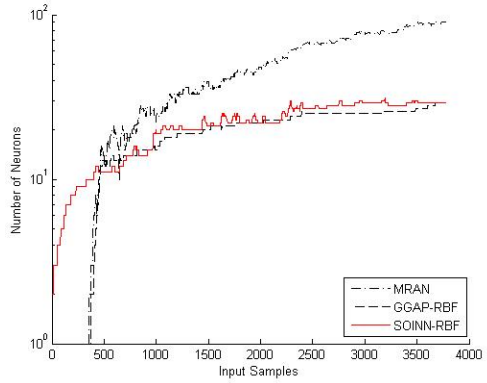

Fig. 6. Hidden units updating process for Thyroid.

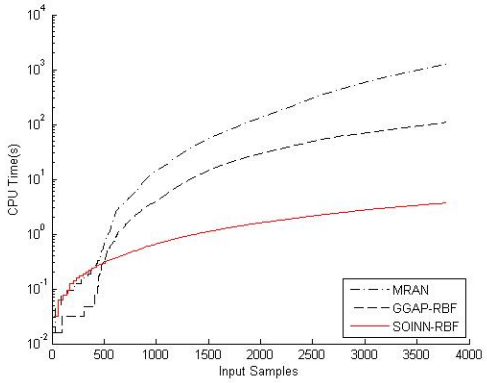

Fig. 7. Learning speed comparison for Thyroid.

C. Insurance

The Insurance benchmark is a data set used in the CoIL 2000 Challenge which contains information on customers of an insurance company. The data consists of 86 variables and the last one is the output label. The input attributes are normalized to range $[0, 1]$ in this experiment. This data set has 5822 samples for training and 4000 samples for testing. It is collected for the classification on weather a person would be interest in buying a caravan insurance policy, 1 means yes and 0 means no. So the output are divided into two class by

$$label = \begin{cases} 1 & \text{if } f(x) > 0.5 \\ 0 & \text{if } f(x) < 0.5 \end{cases} \tag{22}$$

for classification.

The parameters of online SOINN-RBF for this data set chosen as: $e_{min} = 0.001$, $e^{'}_{min} = 0.01$, $sig_{min} = 0.0001$, $\tau = 800$, $W = 60$, $Q_0 = 0.0001$, $P_0 = 0.96$. And the parameters of GGAP-RBF for this data set is: $e_{min} = 0.0001$, $\epsilon_{max} = 15$, $\epsilon_{min} = 2.5$, $\kappa = 0.6$, $\gamma = 0.995$, $Q_0 = 0.00001$, $P_0 = 0.9$. The distribution information of the 85 attributes is learned before learning procedure. The MRAN algorithm can not work appropriately for this problem.

Table III shows the simulation results of the simulation performance for the Insurance data set. An addition column with classification accuracy information is added to this table. We can see that all these algorithms achieves the identical accuracy. The online SOINN-RBF is much faster than GGAP-RBF, even though it has more hidden units.

TABLE III
SIMULATION PERFORMANCE FOR INSURANCE

Process	Algorithm	RMSE	Accuracy	Time(s)	♯Neurons
Batch	SOINN-RBF	0.2344	94.05%	29.54	368
	SVM-R[a]	0.2378	94.05%[b]	8.08	1257
Online	SOINN-RBF	0.2422	94.05%	20.68	107
	GGAP-RBF	0.2326	94.05%	2890.32	23
	MRAN[c]	—	—	—	—

[a] libsvm with parameters '-s 3 -c 300'
[b] libsvm with parameters '-s 0'
[c] Out Of Work

Fig. 8. shows the growing procedures of hidden units. Fig. 9. shows the learning speed comparison of these online learning algorithms. From this figure we can see the *dimensional disaster* phenomenon of the GGAP-BRF. In contrast, the online SOINN-RBF still maintains a fast learning speed and obtains a smooth learning curve.

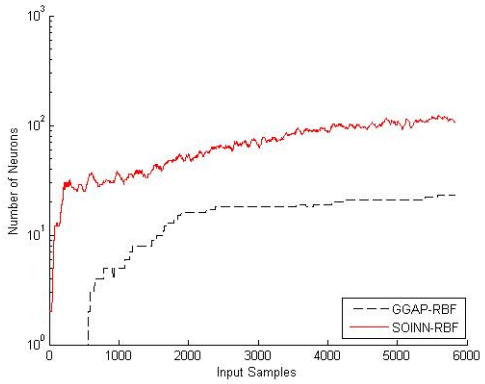

Fig. 8. Hidden units updating process for Insurance.

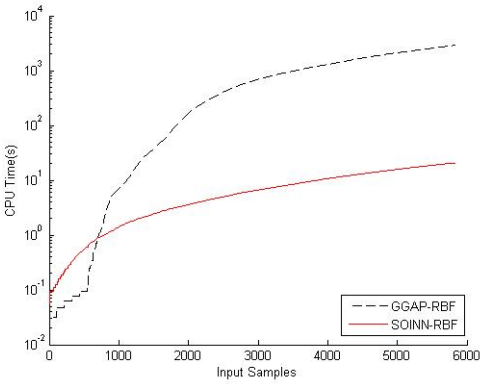

Fig. 9. Learning speed comparison for Insurance.

V. CONCLUSIONS

Two efficient learning algorithms for the radial basis function network are proposed in this paper. They are called SOINN-RBF and all based on the excellent topological learning ability of SOINN. The batch SOINN-RBF is the combination of SOINN and the least square algorithm, which achieves comparable performance with SVM for regression. The accurate pattern gained by SOINN is the main contributor to the batch SOINN-RBF's nice performance. The online SOINN-RBF is basically a variation of SOINN with the redefined growing and pruning criteria and the back-propagation error information. Only the weights of the units are adjusted by EKF algorithm instead of the whole network. This strategy reduces the memory usage and the computing cost, therefore accelerate the algorithm, which is especially obvious when the input data is in high dimension.

Although online SOINN-RBF has gained a fine performance in learning speed, it seems to have some prices. The approximation accuracy and the network compactness are not as good as GGAP-RBF. How to accelerate an online algorithm and meanwhile maintaining approximation accuracy is an challenging topic for further research.

REFERENCES

[1] Cover Thomas M. "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition." *Electronic Computers, IEEE Transactions on* 3 (1965): 326-334.

[2] Mller K-R., et al. "Predicting time series with support vector machines." *Artificial Neural NetworkstICANN'97.* Springer Berlin Heidelberg, 1997. 999-1004.

[3] Er Meng Joo, et al. "Face recognition with radial basis function (RBF) neural networks." *Neural Networks, IEEE Transactions on* 13.3 (2002): 697-710.

[4] Platt John. "A resource-allocating network for function interpolation." *Neural computation* 3.2 (1991): 213-225.

[5] Kadirkamanathan Visakan and Mahesan Niranjan. "A function estimation approach to sequential learning with neural networks." *Neural Computation* 5.6 (1993): 954-975.

[6] Yingwei Lu, Narashiman Sundararajan, and Paramasivan Saratchandran. "Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm." *Neural Networks, IEEE Transactions on* 9.2 (1998): 308-318.

[7] Huang, Guang-Bin, Paramasivan Saratchandran, and Narasimhan Sundararajan. "Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm." *Neural Networks, IEEE Transactions on* 16.1 (2005): 57-67.

[8] Furao Shen, Osamu Hasegawa. "An on-line learning mechanism for unsupervised classification and topology representation." *IEEE Computer Society International Conference on Computer Vision and Pattern Recognition (CVPR 2005), San Diego, CA, USA*. 2005.

[9] Furao Shen, Osamu Hasegawa. "An incremental network for on-line unsupervised classification and topology learning." *Neural Networks* 19.1 (2006): 90-106.

[10] Fritzke, Bernd. "A growing neural gas network learns topologies." *Advances in neural information processing systems* 7 (1995): 625-632.

[11] Kohonen, Teuvo. "The self-organizing map." *Proceedings of the IEEE* 78.9 (1990): 1464-1480.

[12] Chih-Chung Chang and Chih-Jen Lin. "LIBSVM : a library for support vector machines." *ACM Transactions on Intelligent Systems and Technology,* 2:27:1–27:27, 2011.