A SELF-ORGANIZING RECURRENT NEURAL NETWORK BASED ON DYNAMIC ANALYSIS

Qili Chen[1] Junfei Qiao[2] and Yi Ming Zou[3]

[1]Beijing Information Science and Technology University, Beijing, China
[2]Beijing University of Technology, Beijing, China
[3]University of Wisconsin-Milwaukee, Milwaukee, USA

ABSTRACT

A recurrent neural network with a self-organizing structure based on the dynamic analysis of a task is presented in this paper. The stability of the recurrent neural network is guaranteed by design. A dynamic analysis method to sequence the subsystems of the recurrent neural network according to the fitness between the subsystems and the target system is developed. The network is trained with the network's structure self-organized by dynamically activating subsystems of the network according to tasks. The experiments showed the proposed network is capable of activating appropriate subsystems to approximate different nonlinear dynamic systems regardless of the inputs. When the network was applied to the problem of simultaneously soft measuring the chemical oxygen demand (COD) and NH3-N in wastewater treatment process, it showed its ability of avoiding the coupling influence of the two parameters and thus achieved a more desirable outcome.

KEYWORDS

Recurrent Neural Network, Dynamic Analysis, Self-organizing

1. INTRODUCTION

Recurrent neural networks (RNNs) have a wide range of applications in approximating complex dynamic systems[1-5]. Different discrete time recurrent neural networks have been appeared in the literature. The classical fully recurrent network[6,7] is composed of a single layer of fully interconnected neurons. several such recurrent layers are combined to obtain a richer architecture[8]. Other cases of recurrent networks are the external feedback representations [9], the higher-order recurrent neural networks[10], and the block-structured recurrent neural networks[11].

To reduce the complexity of fully connected recurrent networks, a simplified network structure was proposed in [12], in which the feedback connections are grouped into pairs. In addition to many exceptional properties, this type of network architecture reduces the computational and storage burden of the more complex recurrent networks significantly. The stability problem was subsquencially considered in [13] for the special case where the 2x2 matrix of each mutrally connected pair of variables is scaled orthogonal. For a network to be stable, eigenvalues of the corresponding matrix must be inside the unit circle on the complex plane. The approaches used in both [12] and [13] take advantage of the 2x2 block diagonal form of the matrix and derive the

David C. Wyld et al. (Eds) : CCSEIT, AIAP, DMDB, ICBB, CNSA - 2017
pp. 53– 66, 2017. © CS & IT-CSCP 2017 DOI : 10.5121/csit.2017.70606

conditions which ensure the eigenvalues lay within the unit circle. When each of the 2x2 diagonal blocks of the block diagonal matrix is scaled orthogonal, the condition is immediately clear, which allows the possibility for a more efficient algorithm in [13].

Our goal in this paper is to design a recurrent neural network with a self-organizing structure based on dynamic analysis. The key problems we need to address properly are the stability of the recurrent network and the self-organizing structure.

In contrary to recurrent neural networks, there exist many self-organizing algorithms for feed forward neural networks. Most of these algorithms work by adding new neurons or deleting existing neurons based on sensitivity analysis (SA) for the purpose of optimizing the network structures [14-18]. However they cannot be adapted easily to the RNN case, since growing or pruning neurons will change the dynamics of an RNN. RNNs require that both the structure stability and the dynamic stability be guaranteed. So the RNNs have their own special self-organizing ways.

In general, we can separate the existing self-organizing RNNs into two types. One consists of networks with self-organized structures by changing the behaviours of individual neurons. These methods use the unsupervised algorithms. The echo state network (ESN) is a typical example of this type of networks. It has a big reservoir of neurons. Researches have proposed some self-organizing way of this reservoir. A biologically motivated learning rule based on neural intrinsic plasticity was used to optimize reservoirs of analog neurons in [19-22]. The self-organizing RNNs introduced in [23] combines three distinct forms of local plasticity for the learning of spatio-temporal patterns in the inputs while maintains the networks dynamics in a healthy regime. The above self-organizing RNNs are based on the idea of maximizing available information at each internal neuron in a self-organized way by changing the behaviours of individual neurons.

Others consists of local recurrent global feed forward neural networks with growing and pruning algorithms which require supervised algorithms. The local recurrent global feedforward models proposed in [3,24-26] can easily be equipped with self-organizing structures. A self-structuring neural network control method was proposed in [24] which keeps the dynamics of the network when the structure changed. A way to combine training and pruning for the construction of a recurrent radial basis function network (RRBFN) based on recursive least square (RLS) learning was discussed in [27]. All above methods focus on the structure stability. A growing and pruning method, which adjusts the structures of RNNs by modifying the subsystems, was introduced in [28]. This method, to the authors' knowledge, is the first time to consider the dynamic stability during the self-organizing process. However, the proposed approach in [28] considers only one error performance index, and thus limits its ability of finding the best structure to approximate the target systems.

The local recurrent global feedforward neural network we propose here contains two hidden layers. The first hidden layer, which is a feedback layer, carries the structure of the network introduced in [12]. The second hidden layer, for the purpose of increasing the dynamic characteristic of the network, ramifies some of the restrictions occur if only one hidden layer is used in the network, and thus makes the network closer to a fully connected network. To equip the network with a self-organizing structure, we developed an algorithm based on dynamic analysis of the task and the network. Experiments showed that our proposed network has many advantages over the existing RNNs. With the stability guaranteed, the two layer structure adds versatility to the network with minimum complexity added to the network in comparison with the fully connected ones, and one neural network can be used for approximating different nonlinear dynamic systems.

The organization of this paper is as follows. In Section 2, we describe the RNN structure proposed in this paper. In Section 3-5, we introduce a self-organizing method to design the structure of proposed recurrent neural network and analysis the state stability of the network. In Section 6, we provide the computer simulations to validate the effectiveness of the proposed network, and we apply the network to an identification problem in an industrial process. Finally, we provide a brief conclusion in Section 7.

2. STRUCTURE OF THE PROPOSED NEURAL NETWORK

The network considered in this paper is a discrete-time dynamic neural network with *m* inputs and *n* outputs. It has two hidden layers which are the feedback hidden layer and the self-organizing hidden layer. A separate read-out layer maps different parts of the state space to the desired outputs. Though the structure of the network is similar to a multilayer perceptron, it is dynamic in contrary to a multilayer perceptron, since the existence of the feedback neurons. The network structure is shown in Figure 1.

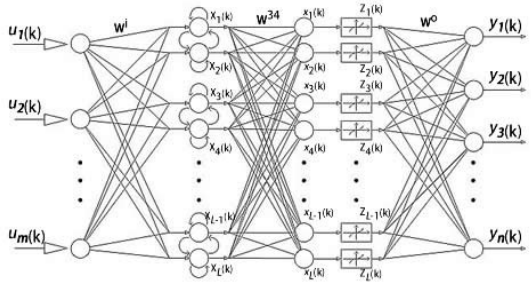

Figure 1. The structure of RNN

Layer 1: Input layer. The main function of the neurons in this layer is to transmit the input data to the neurons in layer 2 by a weight matrix W^i. The input signal is $U(k)=[u_1(k), u_2(k),\ldots\ldots, u_m(k)]^T$, $k=1,2,\ldots\ldots, N$. The weight matrix W^i performs a similar role as in static feedforward networks: W^i and the activation functions are responsible for the approximation properties of the model.

Layer 2: Self-feedback hidden layer. The self-feedback matrix is expressed by a block diagonal matrix $W^h=\text{diag}(W_1^h, W_2^h, \ldots\ldots, W_{L/2}^h)$, where the feedback connections for each pair of mutually connected neurons are given by blocks of the form:

$$W_i^h = \begin{bmatrix} w_{i(1,1)}^h & w_{i(1,2)}^h \\ w_{i(2,1)}^h & w_{i(2,2)}^h \end{bmatrix}, i = 1, 2, \cdots, L/2.$$

The state vector of Layer 2 is $X(k) = [X_1(k), X_2(k),\ldots\ldots, X_L(k)]^T$, $k=1,2,\ldots\ldots, N$. The weights given by W^h are responsible for the model's dynamics and memory function.

Layer 3: Self-organizing hidden layer. Where W^{34} is the connecting weight matrix between Layer 2 and Layer 3, and $x(k) = [x_1(k), x_2(k),\ldots\ldots, x_L(k)]^T$ is the state vector of Layer 3. The weight matrix W^{34}, together with the activation function, are responsible for approximation properties. The matrix W^{34} performs a dynamic transferring role and enriches the dynamic characteristics of the network. In this layer, the self-organizing process realized by activating the subsystems of the network. The *i*th subsystem in the network is described by the following:

$$v_i(k+1) = f(x_i(k+1)) = f(W_i^{34}(W^h X(k) + W^i U(k)))$$

where W_i^{34} is the ith row vector of W^{34}, and the $v_i(k)$ is the output. The activation function $f_i(\bullet)$ of the ith subsystem is defined as:

$$f_i(x) = \begin{cases} \dfrac{e^x - e^{-x}}{e^x + e^{-x}}, & i\text{th subsystem is activated,} \\ 0, & i\text{th subsystem is unactivated.} \end{cases}$$

Layer 4: Output layer. The output signal is $y(k)=[y_1(k), y_2(k),\ldots\ldots, y_n(k)]^T$, $k=1,2,\ldots\ldots, N$. And the W^o is the connecting weight matrix between Layer 3 and Layer 4. The activation function is a linear activation function.

So the proposed RNN model can be described as:

$$X(k+1) = W^h X(k) + W^i U(k)$$
$$y(k) = W^o f(W^{34} X(k))$$

where the notation are: L represents the number of total subsystems, $X \in R^L$ is the neural network state vector, $U \in R^m$ and $y \in R^n$ are the input and output vectors, respectively, $W^i \in R^{L \times m}$, $W^h \in R^{L \times L}$, $W^{34} \in R^{L \times L}$, and $W^o \in R^{n \times L}$.

From the structure, we have that $x = W^{34}X$, so the neural network functions are described by:

$$x(k+1) = W^{34} W^h (W^{34})^{-1} x(k) + W^i U(k)$$
$$y(k) = W^o f(x(k)).$$

Let $P = W^{34} W^h (W^{34})^{-1}$, we see that this network is equivalent to a fully connected recurrent neural network with 3 layers such that the state equations can be represented by:

$$x(k+1) = Px(k) + W^i U(k)$$
$$y(k) = W^o f(x(k)).$$

From this presentation of the proposed neural network, one can see that the neurons in the hidden layer could be fully connected. So though our network here is a special case of the full feedback Wiener-type recurrent neural network (WRNN)[25], it offers a number of significant features. One of which is the stability of the network can easily be analysed, and this will be discussed in Section 3.

3. SELF-ORGANIZING RECURRENT NEURAL NETWORK

The self-organizing algorithm presented in this paper is based on a dynamic analysis scheme. Two key problems, how to organize the dynamical subsystems to work and which dynamical subsystems are selected to work, need to be resolved. Our approach is the following. Firstly, the dynamics of the system and all the subsystems are analysed and the subsystems fitness are computed outline. Then, depending on the task, the network self-organizes its structure online by activating the best-fit subsystems one by one and the weights of the output layer are trained for the purpose of approximating the target system.

3.1. Initialize the network

Assume that L is sufficiently large for tasks. Set $f(\bullet) = 0$ in layer 3 and set W^o to be the zero vector, i.e. no subsystems are activated at the beginning and thus the outputs of neural network are also zeros. The rests of the weights W^i, W^h, W^{34} are randomly initialized.

To ensure the stability of the network, a synaptic normalization (SN)[23] was used to update W^h. This SN proportionally adjusts the feedback connections to a neuron. Specifically, feedbacks weights are normalized according to:

$$w^h_{i(1,1)} = \frac{w^{h\prime}_{i(1,1)}}{|w^{h\prime}_{i(1,1)}| + |w^{h\prime}_{i(2,1)}| + 1} \qquad w^h_{i(1,2)} = \frac{w^{h\prime}_{i(1,2)}}{|w^{h\prime}_{i(1,2)}| + |w^{h\prime}_{i(2,2)}| + 1}$$

$$w^h_{i(2,1)} = \frac{w^{h\prime}_{i(2,1)}}{|w^{h\prime}_{i(1,1)}| + |w^{h\prime}_{i(2,1)}| + 1} \qquad w^h_{i(2,2)} = \frac{w^{h\prime}_{i(2,2)}}{|w^{h\prime}_{i(1,2)}| + |w^{h\prime}_{i(2,2)}| + 1}.$$

Where $w^{h\prime}_i$ is the randomly initialized value for the weight w^h_i.

3.2. Dynamic analysis

Let the input be zero and let the initial network state $X(0)$ be given by an arbitrary L dimensional vector. Different subsystems have different dynamics. Some outputs of subsystems of network are shown in Figure 2. We can see that the outputs of the subsystems behave as many dynamical subsystems with different time-scale dynamics. Supporting multiple time-scales is equivalent to a transmission hidden layer having individual neurons with different contractive dynamics. The contractive dynamic is governed by the contraction coefficients of the state transition function.

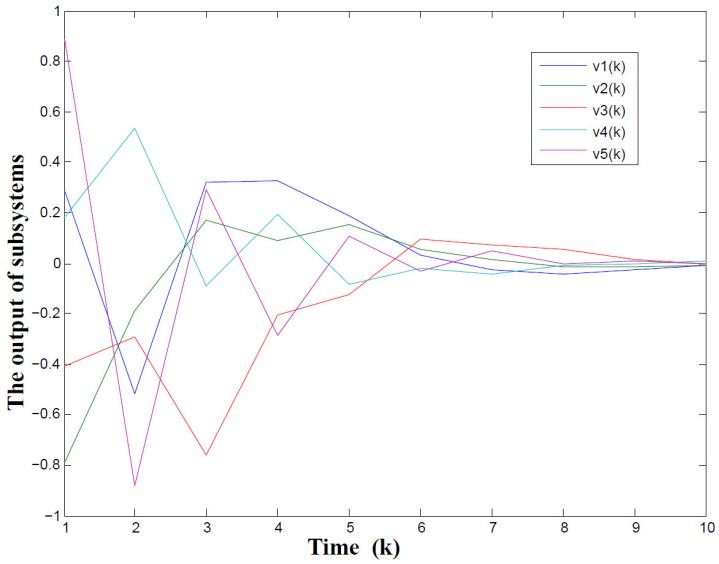

Figure 2. The outputs of the subsystems

Among all dynamic subsystems of network, some subsystems are more fit for approximating the given goal system. The following is a competitive learning algorithm for the selection of subsystems.

Step 1: Giving the goal system a single pulse input signal to produce an output response $O(k)=[O_1(k), O_2(k), \ldots\ldots, O_n(k)]^T$, $k=1, 2, \ldots\ldots, N$, which can be regarded as representatives for the dynamic characteristics of the system. For a chaotic time sequence system or an autonomous system, the outputs of the system $O(k)=[O_1(k), O_2(k), \ldots\ldots, O_n(k)]^T$, $k=1, 2, \ldots\ldots, N$ are the first N steps of the state of the system. If the magnitude of $O(k)$ is large, the outputs need to be transformed into the interval (-1,1) by a Min-Max normalization method. The transformed outputs are marked as $d(k)=[d_1(k), d_2(k), \ldots\ldots, d_n(k)]^T$, $k=1, 2, \ldots\ldots, N$.

Step 2: Giving the neural network a single pulse input signals to produce output responses of the network's subsystems $v(k)=[v_1(k), v_2(k), \ldots\ldots, v_L(k)]^T$, $k=1, 2, \ldots\ldots, N$. The subsystems must have converged in N steps.

Step 3: Compute the fitness matrix $F \in R^{n \times L}$. The fitness value $F(i, j)$ is the inverse of

$$FV(i,j) = \frac{1}{N}\sum_{k=1}^{N}[v_j(k) - d_i(k) - \frac{1}{N}\sum_{p=1}^{N}(v_j(p) - d_i(p))]^2 .$$

If $F(i, j)$ is the max value of the i row of F matrix, the subsystems v_j can be the best fit subsystem for approximating the output d_i.

4. SELF-ORGANIZING ALGORITHM

The proposed self-organizing RNN organizes its structure online by activating the subsystems one by one. Different dynamic systems are approximated by different linear combinations of the subsystems. The network can also improve its approximation ability by training the weights. Note that this is an online algorithm. Only the weights of the connections to the output readout neurons are modified by training.

The following self-organizing algorithm is used to organize the structure of the network for approximating target systems. For the multiple outputs systems, we separate the outputs to $d_i(k)$, $k=1, 2, \ldots\ldots, N$ and then approximate each of the systems $d_i(k)$, $k=1, 2, \ldots\ldots, n$ separately by following steps.

Step 1: Initialize the network and create a diagonal auxiliary matrix $\Psi^{-1}(0)$ of size L by L, where L is the number of total subsystems. Define a forgetting rate ζ and activate the best fit subsystem v_j (change the activation function $f_j(\bullet)$ of the subsystem and set the weight W_{ji}^o between neural network's output $y_i(k)$ and the best fit subsystem output $v_j(k)$). The vector W_i^o (ith row of output connection weights matrix W^o) only has one nonzero element W_{ji}^o at the beginning.

Step 2: If there are new samples, then give a new sample $[u(k), d_i(k)]$, and train the vector W_i^o, else stop.

Step 3: Compute the error $e_i(k)$, where $e_i(k)$ is defined as

$$e_i(k) = \sqrt{\frac{\sum_{s=1}^{k}(d_i(s) - y_i(s))^2}{k}} .$$

Step 4: If the error $e_i(k) < \varepsilon$, go to step 2, else to step 5 , where ε is a given small value or is equal to $a*l(k)$, where a is the coefficient of $l(k)$ and $l(k)$ is the number of activated subsystems in the hidden layer at time k.

Step 5: If the number of inactivated subsystems is zero, go to step 2, else to step 6.

Step 6: Activate the best fit subsystem among the inactive subsystems.

Step 7: Train the vector W_i^o. Then go to step 2.

In this paper, the following algorithm [29] was used to train W_i^o in step 2 and step 7.

(i) $\mathbf{u}(k) = \Psi^1(k-1)\mathbf{v}(k)$ [this $\mathbf{u}$ is not related to the input $u(k)$ and $\mathbf{v}(k) = [v_1(k), \ldots\ldots, v_L(k)]$ is the outputs of total subsystems. The outputs of inactivated subsystems are all zero.]

(ii) $$\mathbf{q}(k) = \frac{\mathbf{1}}{\zeta + \mathbf{v}(k)^{\mathrm{T}}\mathbf{u}(k)}\mathbf{u}(k)$$ [comment: T indicates transpose]

(iii) $y_i(k)=w_i^o(k+1)^T v(k)$

(iv) $e_i(k)=d_i^+(k)-y_i(k)$ [comment: one-dimensional teacher output $d_i(k-1) =: d_i^+(k)$]

(v) $W_i^o(k)= W_i^o(k-1)+\mathbf{q}(k)\, e_i(k)$

(vi) $\Psi^1(k)=\zeta^{-1}(\Psi^1(k-1)- \mathbf{q}(k)[\, v(k)^{\mathrm{T}}\Psi^1(k-1])$.

5. NETWORK STATE STABILITY ANALYSIS

As dynamic systems, RNNs require stability analysis frequently. Global recurrent systems lead to difficulties in state monitoring as well as large computation task.

The proposed RNN in this paper is composed of many subsystems. This built-in structure enables us to work with each local subsystem, and thus greatly reduce the computational complexity.

Every subsystem $v_i(k)$ is a nonlinear mapping of the linear dynamic system $x_i(k+1)$, where $x_i(k+1)$ is the linear combination of L subsystems like this:

$$\sum_i : X_i(k+1) = \begin{bmatrix} w^h_{i(1,1)} & w^h_{i(1,2)} \\ w^h_{i(2,1)} & w^h_{i(2,2)} \end{bmatrix} X_i(k) + w_i^i U(k)$$

Theorem 1: If the weights $w_{i(m,n)}{}^h$; m=1, 2; n=1, 2; i=1, 2, ……, L are normalized by the SN mechanism given in section 3.1, then the neural network will be stable.

Proof: By Gershgorin Circle Theorem, if λ is an eigenvalue of the aforementioned 2x2 matrix, then λ satisfies

$$|\lambda - w^h_{i(1,1)}| \le |w^h_{i(2,1)}|,$$

or

$$|\lambda - w^h_{i(2,2)}| \leq |w^h_{i(1,2)}|.$$

If λ satisfies the first inequality, then

$$\begin{aligned} |\lambda| &= |\lambda - w^h_{i(1,1)} + w^h_{i(1,1)}| \\ &\leq |\lambda - w^h_{i(1,1)}| + |w^h_{i(1,1)}| \\ &\leq |w^h_{i(2,1)}| + |w^h_{i(1,1)}| \\ &= \frac{|w^{h\prime}_{i(1,1)}|}{|w^{h\prime}_{i(1,1)}|+|w^{h\prime}_{i(2,1)}|+1} + \frac{|w^{h\prime}_{i(2,1)}|}{|w^{h\prime}_{i(1,1)}|+|w^{h\prime}_{i(2,1)}|+1} \\ &< 1. \end{aligned}$$

Similarly, the second inequality also leads to $\lambda < 1$. It is known that if the eigenvalues of a linear dynamic system laying inside the unit circle on the complex plane, then the system is stable. So, under the assumption of the theorem, all systems $\sum_i$ are stable and thus the network is stable.

6. EXAMPLES

The purpose of this section is to demonstrate the capability of the proposed network using simulations.

6.1. Predicting the Mackey-Glass Sequence

Mackey-Glass Sequence is a classical benchmark problem [29,30] defined by the following differential equation:

$$\frac{\partial u(t)}{\partial t} = \frac{0.2u(t-\alpha)}{1+u(t-\alpha)^{10}} - 0.1u(t)$$

This task consists of a next-step prediction of a discrete version. It was simulated using the dde23 solver for delay differential equations from the commercial toolbox Matlab. This solver allows one to specify the absolute accuracy; it was set to 1e-16. A step size of 1.0 was used. The resulting time series were shifted by 1 and passed through a tanh function so that they fell into a range of (-0.5, 0.3). From all data series thus generated, the first 1000 steps were discarded to get rid of initial washouts.

We carried out three series of experiments here. Firstly, we did a multiple 6-fold cross validation for deciding the number of the maximum subsystems. Secondly, we validated the self-organizing ability of the network. Finally, we validated the stability of proposed algorithm.

Set the system parameter to be 17, we generated 4500 time steps of the series, of which the first 4000 time steps were used for training and the last 500 time steps were used for testing [18]. An initial transient of 1000 time steps was discarded before training the readout. Every element of the Mackey-Glass sequence was shifted by -1 and fed through the $tanh$ function. Set the coefficient *a* of the ε to 0.02 and train the samples one by one. We ran the experiments with different maximum number of the subsystems *L* for *L* = 2, 5, 20, 30, 50, 100, and repeat each one100 times. The averages of the results are shown in Table 1 (The number of activated subsystems in the second column is the average value of 100 times experiments). The table shows that on an average, 5 or 6 subsystems were activated for the Mackey-Glass time sequence predicting task. Longer training times will be needed if there are more subsystems in the hidden layer. However, 20 subsystems are sufficient for this task. We also performed similar experiments for a RNN

with a fixed structure. The relationships between RMSE and the number of the subsystems are shown in Figure 3. These results also show that the 5 ~7 subsystems are sufficient for this task.

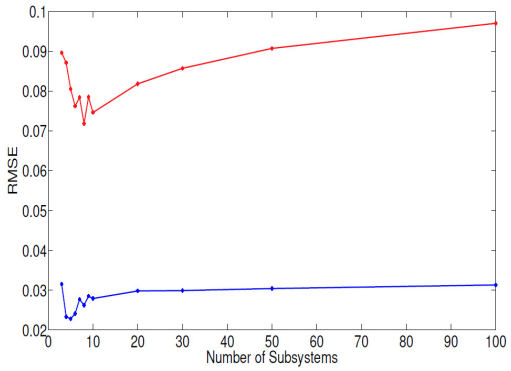

Figure 3. The The red (respectively, blue) line is the average training RSME, (respectively, testing RMSE) for different number of hidden subsystems in fix-structuring neural networks.

Therefore, we set L =20, kept the other parameters, repeated the experiments 100 times, and compared the results with the other online self-organizing time sequential algorithms. The results are given in Table 2 (In this table, the number of nodes (average) is the number of neurons in the network with the structure 1-20-7-1, where the number 20 indicates there were 20 total subsystems in the second layer and the number 7 indicates 7 activated subsystems).

Table 1. Experiments results of different maximum number of the subsystems.

L	**# activated subsystems (l) (average value)**	**Time**	**Training RMSE**	**Testing RMSE**
2	2	0.2936	0.0650	0.0340
5	5	0.3017	0.0502	0.0294
20	5.69	0.3106	0.0457	0.0270
30	5.78	0.3125	0.0458	0.0272
50	5.57	0.3349	0.0431	0.0287.
100	5.56	0.5311	0.0433	0.0283

Table 2. Comparison with other online sequential algorithms

Algorithms	**Time**	**Training RMSE**	**Testing RMSE**	**# nodes**
Proposed method (Average) (Min)	 0.2982 0.2928	 0.0448 0.0275	 0.0275 0.0138	 29 27
OS-ELM(sigmoid)[18]	7.1148	0.0177	0.0183	120
OS-ELM(RBF)[18]	10.0603	0.0184	0.0186	120
GGAP-RBF[17]	24.326	0.0700	0.0368	13
MRAN[17]	57.205	0.1101	0.0337	16
RANEKF[17]	62.674	0.0726	0.0240	23
RAN[17]	58.127	0.1006	0.0466	39

The experiments show that the proposed algorithm is a super-fast online learning algorithm. The OS-ELM of [18] algorithm has the best training RMSE, but the structure of the network is complicated. 120 nodes are needed in the network. The GGAP-RBF [17] can generate a small

network, but the training RMSE and test RMSE are not good. It can be seen that our method provides an overall improvement over the compared methods.

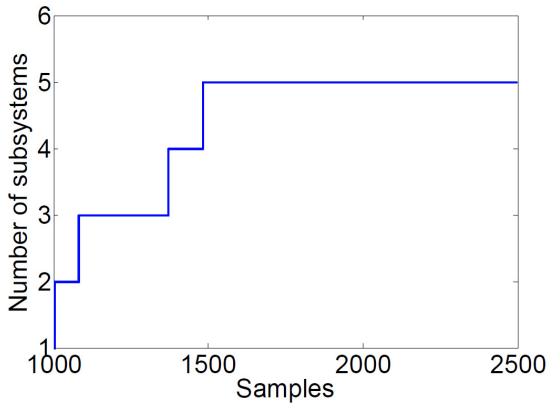

Figure 4. The process of the adjusting

Both the changes of RMSE and the changes of the number of activated subsystems changes during the training process were monitored in one of the experiments. The results are shown in Figure 4. The results show that the network structure's convergence was guaranteed by the proposed self-organizing algorithm.

Note that the time cost of proposed online algorithm depends on the number of total subsystems and the dimension of outputs of system. The larger of the number of the subsystems and the larger of the size of the dimension of the output system, the larger of the time cost in the training process.

6.2. Soft-sensing problem

In recent decades, wastewater problem has become one of the major environmental concerns. Treating wastewater at source is critical. In order to minimize microbial risk and optimize the treatment operation, many variables must be controlled. Biochemical oxygen demand (BOD), chemical oxygen demand (COD), PH level and nutrient levels are the most important ones. Although wastewater quality parameters can be measured by laboratory analysis, a significant time delay, which may range from a matter of minutes to a few days, is usually unavoidable. This limits the effectiveness of operation of effluent quality. Thus, a water quality prediction model is highly desirable for wastewater treatment.

Wastewater treatment process (WWTP) is a highly nonlinear dynamic process. Subject to large disturbances, where different physical (such as settling) and biological phenomena are taking place. It is especially difficult to measure many parameters of WWTP online. These effluent parameters are COD and NH_3-N, which indirectly represent the water organic pollution degree by DO consumption through microorganism metabolism (DO is an important index accords with the practical self-purification situation and the routes of most waste water treatment processes). The measuring of COD is coupled to the NH_3-N. The experiment used the proposed recurrent neural network to predict the COD and NH_3-N simultaneously.

The data was from the water quality testing daily sheet of a small-sized waste water treatment plant. The data includes information on influent COD, influent SS, influent NH_3-N, influent TN, influent TP, PH, and other indices. Only the six mentioned here were used to predict the effluent COD and NH_3-N. We used the proposed recurrent neural network to model waste water treatment

process with the inputs being the value of the above specified six variables, and the outputs being the effluent COD and NH_3-N.

Because of the instability of real system, instead of the real WWTP, the Benchmark Simulation Model 1(BSM1) was used to analyse the dynamic of the waste water treatment process. The initialized network structure contained 30 subsystems. The fitness of all subsystems for approximating COD and NH_3-N in WWTP are shown in Figure 5. First the dependence of the subsystems sequencing on the analysis of the dynamics was obtained, and then the network was used to approximate the effluent COD and NH_3-N. The training error for COD was 0.0136 and the training error for NH_3-N was 0.0312.

The number of activated subsystems for the approximation of the effluent COD is depicted on the left of the Figure 6 and the number of activated subsystems for the approximation of the effluent NH_3-N is depicted on the right of the Figure 6. The number of activated subsystems N increased with time and reached a fit number. The final sequence numbers of subsystems for approximating COD were 1, 28, 9, 6, 18, 7, 21, 13, 23. The final sequence numbers of subsystems for approximating NH_3-N were 13, 21, 18, 24, 5, 8. Different quality parameter are needed to active different subsystems. This avoids the interaction of dynamic between different quality parameters.

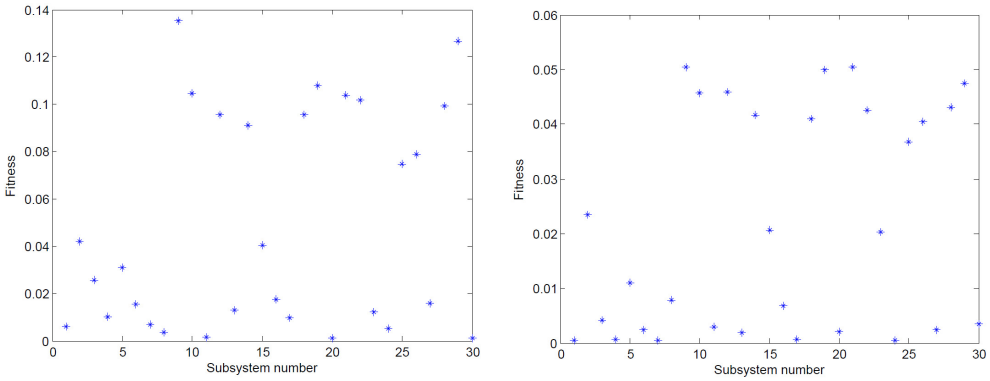

Figure 5. Fitness of all subsystems dynamics for COD(L),NH_3-N(R) dynamics in WWTP

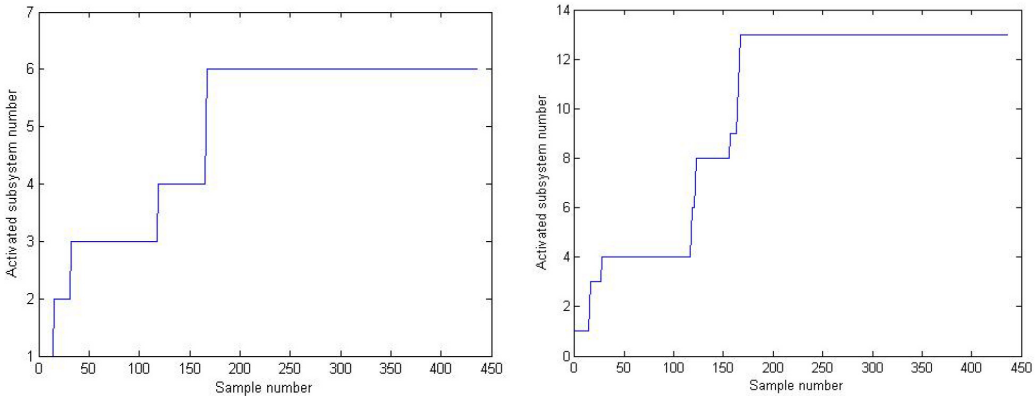

Figure 6. Subsystem changes of approximating COD(L) and NH_3-N(R)

7. CONCLUSIONS

A new approach is proposed for creating a self-organizing recurrent neural network. The structure of this neural network is automatically organizing based on dynamic analysis. Comparing with the existing self-organizing recurrent neural networks, the self-organizing recurrent neural network proposed here has the following advantages: 1) It can simplify and accelerate the structure optimization process. 2) It is capable of solving multiple coupling problems. Due to the fact that different water quality models had different dynamic characteristics, neural networks with fixed structures face difficulties in approximating them because of the coupling among different factors. The proposed neural network models the multiple parameter modeling needs separately by activating different subsystems simultaneously, and thus is able to avoid the coupling and obtains a better approximating accuracy. The effectiveness and performance of the proposed neural network were demonstrated by applying it to solving simple task and multi-task problems. The experimental results provided the supporting evidences for the above claims.

ACKNOWLEDGEMENTS

This work was supported by the Open Research Project of The Beijing Key Laboratory of High Dynamic Navigation Technology under grant No. HDN2017005. The first author acknowledges the support of China Scholar Council, which enabled her to visit the Department of Mathematical Sciences at the University of Wisconsin-Milwaukee as a student for one year. She also wishes to thank the University of Wisconsin-Milwaukee and its faculty for the hospitality she received during her visit.

REFERENCES

[1] Song C., Gao H., & Zheng W Xing, (2009) “A new approach to stability analysis of discrete-time recurrent neural networks with time-varying delay”, Neurocomputing, Vol.72, No. 10-12, pp 2563-2568.

[2] He Y., Wu M. & She J., (2006) “An improved global asymptotic stability criterion for delayed cellular neural networks”, IEEE Transaction on Neural Networks, Vol. 17, No.1, pp 250-252.

[3] Patan K., (2007) “Stability analysis and the stabilization of a class of discretetime dynamic neural networks”, IEEE Transaction on Neural Networks, Vol. 18, pp 660-673.

[4] Mahmoud, Magdi S & Sunni, Fouad M. AL, (2012) “Stability of discrete recurrent neural networks with interval delays: global results”, International Journal of System Dynamics Applications , Vol. 1, No. 2, pp 1-14.

[5] Liu S. & Cao J., (2011) “Global exponential stability of discrete-time recurrent neural network for solving quadratic programming problems subject to linear constraints ”, Neurocomputing, Vol. 74, pp 3494-3501.

[6] Williams, Ronald J & Zipser, David, (1989) “A learning algorithm for continually running fully recurrent neural networks”, Neural Computing, Vol. 1, pp 270-280.

[7] Pearlmutter B. A., (1995) “Gradient calculations for dynamic recurrent neural networks: a survey”, IEEE Transaction on Neural Networks, Vol. 6, No. 5, pp 1-20.

[8] Puskorius G.V. & Feldkamp L. A., (1994) “Neurocontrol of nonlinear dynamical systems with kalman fitter-trained recurrent networks”, IEEE Transaction on Neural Networks, Vol. 5, No. 2, pp 279-297.

[9] Narendra K.S. & Parthasarathy K., (1990) "Identification and control of dynamical systems using neural networks", IEEE Transaction on Neural Networks, Vol. 1, No. 1, pp 4-27.

[10] Kosmatopoulos E.B., Polycarpou, M.M., Christodoulou M. A. & Ioannou P.A., (1995) "Higher-order neural network structures for identification of dynamical systems", IEEE Transaction on Neural Networks, Vol. 6, No. 2, pp 422-431.

[11] Santini S. Del Bimbo A. & Jain R., (1995) "Block-structured recurrent neural networks", Neural Networks, Vol. 8, No. 1, pp 135-147.

[12] Sivakumar S.C., Robertson W. & Phillips W.J., (1999) "On-Line stabilization of block-diagonal recurrent neural networks", IEEE Transaction on Neural Networks, Vol. 10, No. 1, pp 167-175.

[13] Mastorocostas P.A. & Theocharis J.B., (2006) "A stable learning algorithm for block-diagonal recurrent neural networks: application to the analysis of lung sounds", IEEE Transactions on Systems, Man, and Cyebrnetics- Part B: Cyberneitics, Vol. 36, No. 2, pp 242-254.

[14] Coyle D., Prasad G. & McGinnity T.M., (2010) "Faster Self-organising Fuzzy Neural Network Training and Improved Autonomy with Time-Delayed Synapses for Locally Recurrent Learning", In: Temel (ed.), System and Circuit Design for Biologically-Inspired Learning, IGI-global, pp 156-183.

[15] Coyle D., Prasad G. & McGinnity T.M., (2009) "Faster self-organizing fuzzy neural network training and a hyperparameter analysis for a brain computer interface", IEEE Transactions on Systems, Man and Cybernetics (Part B), Vol. 39, No. 6, pp 1458-1471.

[16] Han H., Chen Q. & Qiao J., (2010) "Research on an online self-organizing radial basis function neural network", Neural Computing and Applications, Vol. 19, No. 5, pp 667-676.

[17] Huang G., Saratchandran P. & Sundararajan N., (2005) "A generalized growing and pruning RBF neural network for function approximation", IEEE Transaction on Neural Networks, Vol. 16, No. 1, pp 57-67.

[18] Liang N. & Huang G., (2006) "A Fast and Accurate Online Sequential Learning Algorithm for Feedforward Networks", IEEE Transaction on Neural Networks, Vol. 17, No. 6, pp 1411-1423.

[19] Boedecker J., Obst O. & Mayer N.M., (2009) "Initialization and self-organized optimization of recurrent neural network connectivity", HFSP Jornal, Vol. 3, No. 5, pp 340-349.

[20] Dasgupta S. Worgotter F. & Manoonpong P., (2012) "Information theoretic selforganised adaptation in reservoirs for temporal memory tasks", Engineering Applications of Neural Networks Communications in Computer and Information Science, Vol. 311, pp 31-40.

[21] Dasgupta S. Worgotter F. & Manoonpong P., (2013) "Information dynamics based self-adaptive reservoir for delay temporal memory tasks.", Evolving Systems, Vol. 4, No. 4, pp 235-249.

[22] Jochen J.S., (2007) "Online reservoir adaptation by intrinsic plasticity for backpropagation decorrelation and echo state learning", Neural Networks, Vol. 20, No. 3, pp 353-364.

[23] Andreea L., Gordon P. & Jochen T., (2009) "SORN: A Self-organizing recurrent neural network", Frontiers in Computational Neuroscience, Vol. 3, No. 2009, pp 1-9.

[24] Park J.H., Huh S.H. & Seo S.J., (2005) "Direct adaptive controller for nonaffine nonlinear systems using self-structuring neural networks", IEEE Transactions on neural networks, Vol. 16, No. 2, pp 414-422.

[25] Hsu Y.L. & Wang J.S., (2008) "A Wiener-type recurrent neural network and its control strategy for nonlinear dynamic applications", Journal of Process Control, Vol. 19, pp 942-953.

[26] Tsoi A.C. & Back A.D., (1997) "Discrete time recurrent neural network architectures, a unifying review", Neurocomputing, Vol. 15, No. 3-4, pp 183-223.

[27] Chi S.L. & Ah C.T., (2005) "Combined learning and pruning for recurrent radial basis function networks based on recursive least square algorithms", Neural Computing and Application, Vol. 15, pp 62-78.

[28] Chen Q. Chai W. & Qiao J., (2011) "A stable online self-constructing recurrent neural network", Advances in Neural Networks, Lecture Notes in Computer Science, Vol. 6677, pp 122-131.

[29] Jaeger H. & Hass H., (2004) "Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication", Science , Vol. 304 , pp 78-80.

[30] Mackey M.C. & Glass L., (1997) "Oscillation and chaos in physiological control systems", Science, Vol. 197, pp 287-289.

AUTHORS

Qili Chen received the Ph. D. degree in Beijing university of technology, China, in 2014, the M.E. from the Beijing University of Technology, Beijing, China, in 2010, respectively. She visited the University of Wisconsin-Milwaukee in 2012. Currently she is working in the Beijing information science of technology university. Her current research interests include recurrent neural networks, modeling and control in complex dynamic process.