Preprints of the
9th International Symposium on Advanced Control of Chemical Processes
The International Federation of Automatic Control
June 7-10, 2015, Whistler, British Columbia, Canada

Time-series prediction modelling based on an efficient self-organization learning neural network

Gang Yang[1, 2], Hui Yang[1, 2], Lizhen Dai*[1, 2]

*{1. School of Electrical and Electronic Engineering, East China Jiaotong University, Nanchang 330013, China;
2. Key Laboratory of Advanced Control & Optimization of Jiangxi Province, Nanchang 330013, China;
e-mails: dr.hankyang@gmail.com; yhshuo@263.net; dr.alicedai@gmail.com}*

Abstract: For solving some process control engineering problems which can be treated as a time-series, a fast and accurate self-organization learning strategy is proposed based on the significance evaluation of hidden neurons with respect to the network output. This approach is introduced to optimize the architecture and parameters of span-lateral inhibition neural network (S-LINN) simultaneously. The insignificant neuron(s) will be pruned automated step by step based on the determination of significance index. The proposed self-organizing approach has been tested on one time-series prediction benchmark problem. Simulation results demonstrate that the proposed method has good exploration and exploitation capabilities in terms of searching the optimal structure and parameters for S-LINN.

Keywords: time-series, self-organization, spatial-lateral inhibition neural network, significance.

1. INTRODUCTION

With the development and application of a variety of sensing equipments in the industrial processes, a large number of real-time monitoring data were obtained. Actually, many process problems can be treated as a time-series to solve the solutions. Especially in recent years, many methods based on data driven to predict or estimate the time-series problems have been reported. Since the significant advantage of Artificial Neural Networks (ANNs) can "learn" from samples, it has been widely used in many actual applications with great success. Without any doubt, the time-series prediction problem is one of the most adapt at handling problems of ANNs. The use of ANNs is mainly based on the network architecture and learning algorithm, which define the function and efficiency respectively. In many applications, the structure of ANNs is fixed in advance, which may can not provide the optimal performance of ANNs with a larger or smaller network size. And it is also difficult for the operators to determine the appropriate size of ANNs. It has been shown that, a network with a larger size will affect the generalization ability owing to the redundant neurons, while the too small network may lack sufficient approximation capability and generalization ability due to its limited information processing power. (Xin, 1999; Leung et al., 2003; Jinn-Tsong et al., 2006)

To solve the above problem, many self-organizing approaches that including constructive, destructive and hybrid algorithms (for example, see Jinn-Tsong et al., 2006; Huang et al., 2005; Han and Qiao, 2010, 2013; and references therein) have been studied for various ANNs to obtain the network structure automatically. Among the aforementioned works, Tsai et al. proposed(Jinn-Tsong et al., 2006) a hybrid Taguchi-genetic algorithm (HTGA) combining the traditional genetic algorithm and Taguchi method to tune both structure and parameters of a feedforward neural network; Huang et al. proposed (Huang et al., 2005) a generalized growing and pruning sequential learning algorithm for RBF networks (GGAP-RBF) for function approximation based on the idea of significance of neurons. Qiao et al. proposed (Han and Qiao, 2010) a growing and pruning algorithm to optimize the structure of a fuzzy neural network (GP-FNN) with radial basis function neurons, and the structure-learning relied on the sensitivity analysis of the output and the parameter-learning based on gradient decent method performed concurrently. A constructing-and-pruning approach based on the calculation of contribution ratios of hidden neurons for feedforward neural network is also proposed (Han and Qiao, 2013) (CP-NN) to optimize the network structure. Besides, in some other self-organizing algorithms, such as VNP (Engelbrecht, 2001), Xing-Hu's method (Xing and Hu, 2009) and N2PS (Augasta and Kathirvalavakumar, 2011), both the input and hidden neurons may be pruned during the learning process based on different criteria. Engelbrecht et al. proposed (Engelbrecht, 2001) a pruning heuristic algorithm (VNP) based on variance analysis of sensitivity information through pruning the variance nullity input and hidden neurons. Xing et al. proposed (Xing and Hu, 2009) a two-phase constructive algorithm through pruning both input and hidden neurons of MLPs based on the mutual information. Augasta et al. proposed (Augasta and Kathirvalavakumar, 2011) a pruning algorithm (N2PS) based on the determination of insignificant input and hidden neurons by the sigmoidal activation value of neurons and the weights of the outgoing connections.

Pruning, one of the destructive self-organizing approaches, is a useful and traditional method for optimizing network from the assumed initial architecture. The optimized network with smaller size may have higher learning accuracy than the network with initial size. Many techniques have been used to pruning ANNs, such as, the mutual information based

methods(Xing and Hu, 2009), evolutionary pruning methods(Jinn-Tsong et al., 2006), sensitivity analysis based methods(Han and Qiao, 2010; Engelbrecht, 2001) and significance measure based methods(Huang et al., 2005; Augasta and Kathirvalavakumar, 2011), and so on.

Actually, the process of pruning irrelevant input neurons in the method of pruning both irrelevant input and hidden neurons can be regard as the principal component analysis (PCA) of samples, which can be separated from the self-organizing of ANNs. All the above methods for optimizing ANNs can optimize the structure and/or parameters through different learning strategies and criteria to obtain better performances than the networks with fixed size. However, these works are mainly based on the traditional networks such as MLPs, RBF networks. Span-lateral Inhibition neural network (S-LINN) (Yang et al., 2011, 2013) is a multi-layer network inspired by the characteristics of neocortical neurons: the pyramidal neurons can span all the thickness of the cortex, while the excitatory interneurons will inhibit the surrounding neurons through the lateral inhibitory connections. It has been proved that, S-LINN outperforms some other neural network methods in terms of the approximation and generalization performances. So, it is considered that the S-LINN with self-organizing capability can have better performances than that of some existing neural networks.

Simple and efficient method is preferred in the real-world applications. Aiming at developing an efficient and accurate self-organizing method for S-LINN to solve the time-series-like engineering problems, a new self-organizing pruning algorithm is presented in this paper to optimize the structure and parameters concurrently to predict future values. The proposed method is based on the significant measure of hidden neurons over the network output. From this method, the insignificant hidden neurons, whose significance index (*SI*) is below the assigned percent of mean significance index of total hidden neurons, will be pruned one-by-one or chunk-by-chunk until the learning performance in terms of cost function is good enough.

The paper is organized in to five sections. Section 2 briefly reviews the S-LINN model and the span-output-weight and feedforward-output-weight sequential learning (SFSL) algorithm. In Section 3, the self-organizing pruning algorithm based on significant measure for optimizing the structure and parameters of S-LINN is proposed. Section 4 presents the experimental and simulation results which show the superior performance of the self-organizing S-LINN compared to other neural network methods by implementing it on Mackey-Glass chaotic time-series prediction problem. Finally, section 5 concludes the paper with a discussion about the results presented and some perspectives for future investigation.

2. REVIEW OF S-LINN AND SEQUENTIAL LEARNING

2.1 Brief review of S-LINN

Inspired by the biotomy and neurobiology findings, an artificial neural network with special connections, which was named span-lateral neural network (S-LINN) (Yang et al., 2013) and shown in Fig.1, was developed to solve real world regression and classification applications. Based on the connect modes of excitatory pyramidal neurons and inhibitory interneurons within neocortex, the S-LINN has a multilayered structure. It consists of a number of neural layers of a similar structure cascaded one after another, which including feedforward and spatial span connections between different layers and also with recurrent lateral inhibitory connections among neurons in each hidden layers. One of the main structural features of S-LINN is the span connection between different neuronal types from two non-adjacent horizontal layers, which corresponds to the canonical circuitry of pyramidal neurons in the upper layers Ⅱ and Ⅲ of neocortex. It means that, these connections may form cortical columns and span all the thickness of the cortex (Mountcastle, 1997). And the second major structural principle of S-LINN is the introducing of lateral inhibitory connections between adjacent hidden neurons (within the same layer), which is used to organize inhibitory circuits and enhance the contrast of perception areas. The detailed description of S-LINN and the architecture of lateral inhibitory connection in hidden layers can be referenced in (Yang et al., 2011, 2013), and it is omitted here.

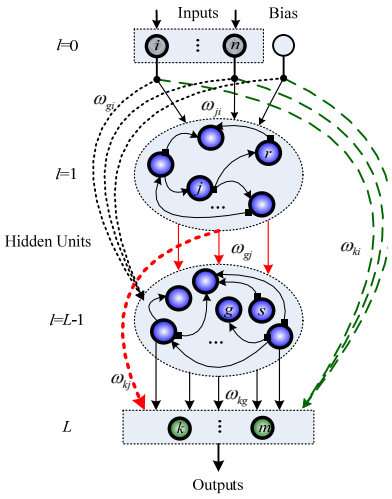

Fig. 1. Structural diagram of (*L*+1)-layer S-LINN.

The mathematical representation of (*L*+1)-layer S-LINN is given by,

$$y_i^p = \sum_{i=1}^{n_0} \omega_{ki}^0 x_i^p + b_0 + \sum_{l=1}^{L-2}\left(\sum_{j=1}^{n_l} \omega_{kj}^l o_j^l + b_l\right) + \sum_{g=1}^{n_{L-1}} \omega_{kg}^{L-1} o_g^{L-1} + b_{L-1} \quad (1)$$

where $x^p{}_i$ denotes the i-th input of p-th sample, $y^p{}_i$ indicates the network output corresponding to $x^p{}_i$, b_l is the bias, $\omega^l{}_{kj}$ is used to indicate the weight connects the j-th neuron from l-th layer to the k-th output neuron, $p \in [1, N]$ is the index of samples, n_l is the number of neurons from l-th layer. o^l_j denotes the output of j-th neuron form l-th layer, which is

$$o_j^l = f^l\left[\xi_j^l - \ddot{\xi}_j^l\right] = f^l\left[\sum_{l=0}^{l-1}\left[\omega_{ij}^l o_i^l\right] - \sum_{r=1, r\neq j}^{n_l} v_{rj}\left(o_r^l - \theta_{rj}\right)\right] \quad (2)$$

where, ξ^l_j is the inner product of input and weight, including signals received from the (l-1)-th layer neurons and the

spatial span connections from the range of [0, *l*-2] layers, while ξ_j^l is the inhibitory input after lateral inhibited by surrounding neurons. θ_{rj} is the inhibiting threshold value of *j*-th neuron due to *r*-th neuron; v_{rj} is the lateral inhibitory coefficient, which is a real number in [0, 1], with "0" meaning no inhibition and "1" meaning full inhibition.

2.2 SFSL sequential learning algorithm

The application of ANNs is based on the approximation, which means that the aim of training ANN is to find optimal parameters such that

$$t_i^p = y_i^p = \sum_{i=1}^{n_0} \omega_{ki}^{0*} x_i^p + b_0^* + \sum_{l=1}^{L-2} \left(\sum_{j=1}^{n_l} \omega_{kj}^{l*} o_j^l + b_l^* \right) + \sum_{g=1}^{n_{L-1}} \omega_{kg}^{L-1*} o_g^{L-1} + b_{L-1}^* \tag{3}$$

where $t^p{}_i$ indicates the desired output corresponding to the input $x^p{}_i$, ω^* and b^* are optimal parameters such that $t^p{}_i = y^p{}_i$.

For training S-LINN effectively, a span-output-weight and feedforward-output-weight sequential learning (SFSL) algorithm is proposed (Yang and Qiao, 2014), which is based on dividing the adjustable parameters into *span-output-weights* ($\omega^0{}_{ki} \ldots \omega^l{}_{kj}$) and *feedforward-output-weights* ($\omega^{L\text{-}2}{}_{gq}$ and $\omega^{L\text{-}1}{}_{kg}$).

For simplicity, the bias neuron of each layer can be considered as a simple neuron. Thence, the above *N* equations of Equation (3) can be written compactly as:

$$\Omega\Theta + \Psi H = T \tag{4}$$

where, $\Omega = \left[\boldsymbol{\omega}_{ki}^0 \quad \cdots \quad \boldsymbol{\omega}_{kj}^l \quad \cdots \quad \boldsymbol{\omega}_{kp}^{L-2} \right]$, $\Psi = \left[\boldsymbol{\omega}_{kg}^{L-1} \right]$, $\Theta = \left[\boldsymbol{x}_i \quad \cdots \quad \boldsymbol{o}_j^l \quad \cdots \quad \boldsymbol{o}_p^{L-2} \right]^T$, $H = \left[\boldsymbol{o}_g^{L-1} \right]$.

In the first phase, the span-output-weights were estimated through solving the least-square solution of Equation (4), namely

$$\left(\Omega^T \right)^* = \left(\Theta^T \right)^\dagger \left(T - \Psi H \right)^T \tag{5}$$

where, $\left(\Theta^T \right)^\dagger$ is the Moore-Penrose generalized inverse of matrix Θ^T; $\left(\Omega^T \right)^*$ is the trained span-output-weight, which is equal to the minimum norm least-squares solution of the linear system $\left(\Omega\Theta \right)^T = \left(T - \Psi H \right)^T$.

In the second phase of SFSL, the feedforward-output-weights were trained by the gradient-based algorithm such as:

$$\frac{\partial E}{\partial \omega_{kg}^{L-1}} = \sum_{k=1}^{m} \frac{\partial E}{\partial \vec{o}_{pk}} \cdot \frac{\partial \vec{o}_{pk}}{\partial \omega_{kg}^{L-1}}$$
$$\frac{\partial E}{\partial \omega_{gq}^{L-2}} = \sum_{k=1}^{m} \frac{\partial E}{\partial \vec{o}_{pk}} \cdot \frac{\partial \vec{o}_{pk}}{\partial \vec{o}_{pg}^{L-1}} \cdot \frac{\partial \vec{o}_{pg}^{L-1}}{\partial \omega_{gq}^{L-2}} \tag{6}$$

where $\vec{o}_{pk}$ and $\vec{o}_{pg}^{L-1}$ are feedforward-outputs of *L*-th and (*L*-1)-th hidden layer neurons respectively; ω_{gq}^{L-2} denotes the feedforward weight connecting the *q*-th hidden neuron of (*L*-2)-th layer and the *g*-th neuron from (*L*-1)-th layer, and ω_{kg}^{L-1} links the *g*-th hidden neuron from (*L*-1)-th layer and the *k*-th output unit respectively; *E* is the cost function which is give by

$$E = \frac{1}{2} \sum_{q=1}^{N} \left(T - \Omega^* \Theta - \Psi H \right)^2 \tag{7}$$

In the minimization procedure of *J*, vector $\boldsymbol{\omega}$ is iteratively adjusted as follows:

$$\boldsymbol{\omega}(c+1) = \boldsymbol{\omega}(c) - \eta \frac{\partial E}{\partial \omega} \tag{8}$$

Here η is the learning rate; vector $\boldsymbol{\omega}$ is the set of weights and bias parameters, *c* is the iterative index.

3. SELF-ORGANIZING PRUNING ALGORITHM FOR S-LINN

In this section, the algorithm of optimizing the structure and parameters of S-LINN has been proposed. The proposed learning algorithm (SOSLINN) comprises a structure learning phase and a parameter learning phase.

In this algorithm, structure learning is based on the significance evaluation of hidden neurons used to determine whether a hidden neuron should be pruned to trimming the network. Parameter learning is based on SFSL algorithm. The SFSL algorithm minimizes a given cost function by adjusting the weights and bias before and after pruning process. In the following part, the structure learning and parameter learning are detailed. Here, *c* is the iteration index, C_1, C_2 and *C* are the maximum number of iterations during initial training, self-organization learning and total learning respectively, ε_1, ε_2 and ε are tolerance error corresponding to C_1, C_2 and *C* respectively,

Step 1: Initial training of S-LINN based on SFSL algorithm.

The first step is the initial training of given network with assumed larger size. The SFSL algorithm in section 2 is used to optimize the parameters.

Step 2: Self-organization of architecture based on the significance evaluation of hidden neurons.

From the mathematical representation of S-LINN and other ANNs, it can be find that the output of network is determined directly by the output of hidden neurons $o^l{}_j$ and the corresponding weights connect to the output neurons ω_{kj}. So, the significance index of *j*-th hidden neurons from *l*-th layer for the network output was defined as $SI^l{}_j$, which is give by

$$SI_j^l = \sum_{k=1}^{m} \left| o_j^l \cdot \omega_{kj} \right| \tag{9}$$

And, the pruning rule is that

$$u_j^l : \begin{cases} significant, & SI_j^l \geq \rho \\ insignificant, & SI_j^l < \rho \end{cases} \quad (10)$$

where ρ is the threshold of pruning, which is given by

$$\rho = \kappa \times \frac{\sum_{j=1}^{n_l} SI_j^l}{n_l} \quad (11)$$

where, κ is the significance coefficient. From the setting of κ values, the size of insignificant hidden neurons will be controlled. And then, the pruning of hidden neuron can be implemented one-by-one or chunk-by-chunk in one step according to the actual situation. Namely, the most insignificant neuron(s) may be pruned based on SI_j^l.

Step 3: Consequent training of parameters based on SFSL algorithm for the pruned network from Step 2.

In summary, the main idea of SOSLINN is the determination of insignificant hidden neurons through the significant measure. The hidden neuron may be pruned in this process if and only if the *SI* value is under ρ. As mentioned above, the significance of hidden neurons will be judged during the tuning process, and the insignificant neuron(s) will be trimmed according to the setting and needing. The method of this significance evaluation is flexible and effective, and the architecture and parameters of NN can be optimized synchronously. So, this self-organization pruning algorithm can be used to training S-LINN.

4. PERFORMANCE EVALUATION OF SOSLNN

Mackey-Glass Chaotic time-series(Mackey and Glass, 1977) prediction is one classical benchmark problem that has been used for evaluating ANN models. (Liang et al., 2006; Huang et al., 2005; Rez et al., 2007; Wu and Er, 2000) The proposed SOSLINN model was tested by applying it to the prediction of the well-known Mackey-Glass time series. The discrete version of the time series is governed (Mackey and Glass, 1977) by:

$$x(t+1) = (1-a)x(t) + \frac{bx(t-\tau)}{1+x^{10}(x-\tau)} \quad (12)$$

where the parameters $\alpha = 0.1$, $b = 0.2$, $\tau = 17$, and the initial condition $x(0) = 1.2$ are chosen to be the same as those of Ref.(Wu and Er, 2000).

The goal of this experimental is to predict the future value $x(t+\Delta)$ from the known values of $\{x(t), x(t-\Delta), \ldots, x(t-(n-1)\Delta)\}$ based on SOSLINN. As in Ref.(Liang et al., 2006), the values $\Delta = 6$ and $n = 4$ are used to reconstruct the phase-space. Then, the prediction model can be summarized as

$$x(t+\Delta) = f\left(x(t), x(t-6), x(t-12), x(t-18)\right) \quad (13)$$

For the purpose of training, we extract 1000 data points between $t = 124$ and $t = 1123$ to prepare the input-output sample data in the structure given by equation (13), where the first 500 samples are as the training samples and the other samples are used to test the prediction accuracy. Here, the tolerance errors during different learning process are setting as $\varepsilon 1$ = 1e-3, $\varepsilon 2$ = $\varepsilon 3$ = 1e-7, which responding to the maximum iterations are $T1$ = 1e+3, $T2$ = 1e+2, $T3$ = 4e+3 respectively. The learning rate is $\eta = 0.085$, the momentum term $\alpha = 0.8$. The initial size of S-LINN is a three-layer network with 10 hidden neurons. 10-fold cross validation is employed independently, and the average result was reported. After self-organizing learning, the number of hidden neuron was keeping in 5(±2).

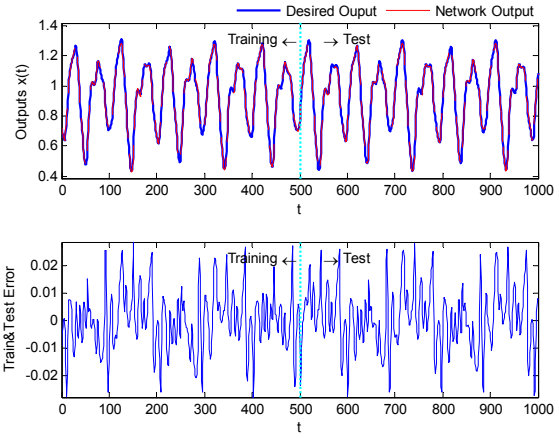

Fig. 2. CaseA: Outputs and learning errors of Mackey-Glass time series prediction based on SOSLINN (10/7)

The outputs and errors of Mackey-Glass time series prediction based on SOSLINN is shown in Fig. 2, which has 7 hidden neurons after self-organization learning. The dynamic process of hidden neuron adjustment and the learning error were given by Fig.3. Fig.4 is the mean square error curve during the learning process.

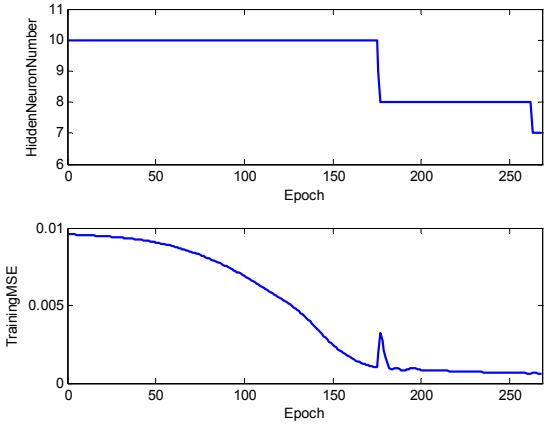

Fig. 3. CaseA: The self-organizing pruning process of SOSLINN and the corresponding learning error curve (10/7)

Figs. 2-4 indicate that the self-organization strategy SOSLINN not only optimize the architecture online, but also improve the performance during the process of structure adjusting and parameters learning, which can predict the Mackey-Glass chaotic time-series with expected accuracy.

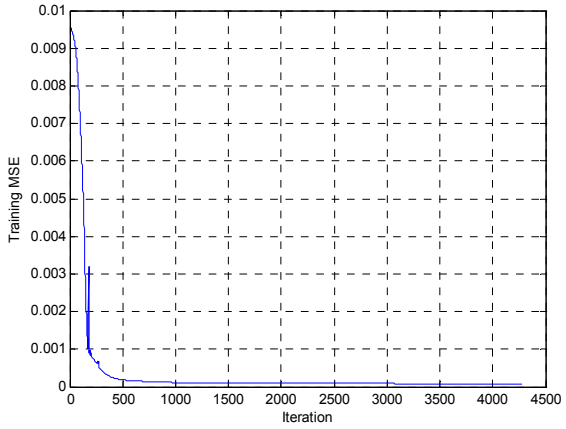

Fig. 4. CaseA: The training mse curve of SOSLINN (10/7)

Table 1. Comparisons of the SOSLINN with other algorithms ($\Delta = 6, x(0) = 1.2$).

Methods	Neuron number	Training RMSE	Testing RMSE
OLS(Wang et al., 2009)	13	0.0158	0.0163
FAOS-PFNN(Wang et al., 2009)	11	0.0073	0.0127
RBF-AFS(Cho and Wang, 1996)	21	0.0107	0.0128
OS-ELM* (Sigmoid) (Liang et al., 2006)	120	0.0177	0.0183
DFNN(Wu and Er, 2000)	5	0.0132	0.0131
GGAP-RBF(Huang et al., 2005)*	13	0.0700	0.0368
S-LINN	5	0.0056	0.0109
S-LINN	3	0.0080	0.0126
RBFAFS(Cho and Wang, 1996)	23rules	0.0096	0.0114
SOSLINN(10)	7	0.0091	0.0093
	6	**0.0065**	**0.0061**
	5	0.0093	0.0095

Note that: * means that the results were not explained clearly or the samples are not same with this paper, which were for reference only.

The RMSE of predicting Mackey-Glass chaotic time series by SOSLINN and other NN methods were reported in Table 1. Here, the performances of three folds with the biggest network size after self-organizing learning were reported, namely there are 7, 6 and 5 hidden neurons in the trained network. The comparison results of Table 1 in terms of the prediction performance among the various models including the proposed SOSLINN, which reveals that the proposed method produces much smaller prediction errors (Training RMSE = 0.0065, Testing RMSE = 0.0061) than those of the other methods. Furthermore, compared with the S-LINN with fix architecture, the SOSLINN has higher generalization ability. For example, the S-LINN with fixed 5 hidden neurons can achieve the performance with Training RMSE = 0.0056 and Testing RMSE = 0.0109, while the same 5 hidden neurons was persisted after an initial S-LINN with 10 hidden neurons was self-organizing learned will have the performance of Training RMSE = 0.0093 and Testing RMSE 0.0095. It means that, when has the similar training accuracy, the generalization accuracy will be improved 13%. The results show the good statistical properties of the SOSLNN. It also can be noticed that the approach based on S-LINN outperformed all the other models.

Table 2. The performances before and after self-organization, and after training of three independent runs

Method (# Hidden Neuron)		Training MSE	Training RMSE	Testing MSE	Testing RMSE
Case A	S-LINN1(10)	0.0010	0.0316	0.0012	0.0342
	S-LINN2(7)	0.0005	0.0234	0.0008	0.0290
	S-LINN3(7)	0.0001	0.0091	0.0001	0.0093
Case B	S-LINN1(10)	0.0010	0.0309	0.0012	0.0347
	S-LINN2(6)	0.0007	0.0265	0.0432	0.2077
	S-LINN3(6)	**0.000037**	**0.0061**	**0.000042**	**0.0065**
Case C	S-LINN1(10)	0.0009	0.0303	0.0016	0.0398
	S-LINN2(5)	0.0012	0.0349	0.0067	0.0818
	S-LINN3(5)	0.0001	0.0093	0.0001	0.0095

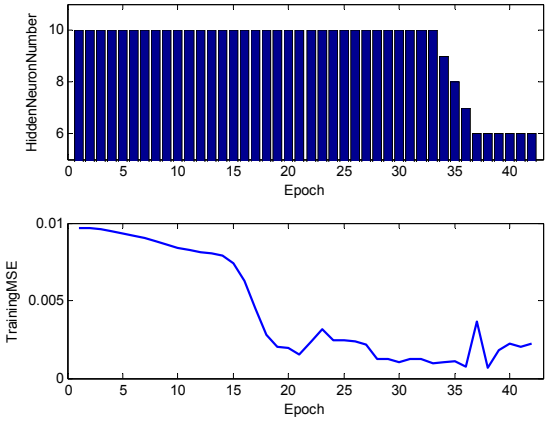

Fig. 5. CaseB: The self-organizing pruning process of SOSLINN and the corresponding learning error curve (10/6)

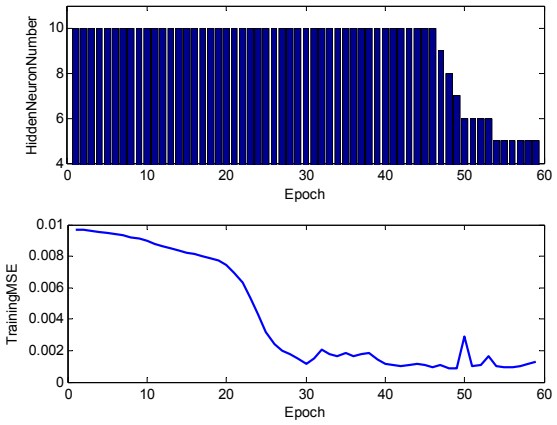

Fig. 6. CaseC: The self-organizing pruning process of SOSLINN and the corresponding learning error curve (10/5)

The results of three independent experiments based on SOSLINN (CaseA, CaseB and CaseC) were shown in Table 2, which respect to the experiments shown in Table 1 respectively. Here, S-LINN1 indicates the network before

self-organization, S-LINN2 means the network after self-organizing pruning, and S-LINN3 denotes the network after learning. Figs. 5-6 showed the dynamic pruning process and the corresponding learning error of CaseB and CaseC. From Table 2, it can be seen that the performance is increasing during the three time points. In spite of the testing RMSE of CaseC after the architecture was self-organizing pruned is a little worse than the network without pruned, namely 0.0818>0.0398, the performance after self-organization is superior to the initial network clearly, e.g., 0.0095<<0.0398. The results demonstrated that the self-organizing strategy is efficient and operable.

5. CONCLUSIONS

In order to solve the real applications which can be treated as time-series problems, an efficient self-organization optimization approach is proposed to optimize the S-LINN, which is based on the significance evaluation of hidden neurons with respect to the network output. The effectiveness and the superiority of the self-organizing pruning algorithm have been demonstrated in the Mackey-Glass time-series benchmark problem. The results show that, the corresponding algorithm used to optimize an S-LINN can find a set of weights for predict the Mackey-Glass chaotic time-series.

In conclusion, the presented SOSLINN not only adjusts the network architecture adaptively, but also has the advantages of high approximation accuracy and good generalization performance. The proposed approach performs well and stably. However, the data of numerical simulation without any noise, which is inconsistent with the actual situation. The data with noise from complex system may affect the efficiency and stability of the proposed approach. Therefore, the stability analysis and other important issues should be considered in the future research.

ACKNOLEDGEMENT

This work was partially supported by the National Natural Science Foundation of China under Grants No. 51174091 and No. 61164013, the Special Program for National Basic Research of China under Grant No. 2014CB360502, the Education Department Foundation of Jiangxi Province under Grant No. GJJ14402, and Transport Department Science and Technology Foundation of Jiangxi Province under Grant No. 2014X0015. In addition, the authors would like to thank the anonymous reviewers for their constructive comments and suggestions which helped to improve the quality of this paper.

REFERENCES

Augasta, M., and Kathirvalavakumar, T. (2011). A novel pruning algorithm for optimizing feedforward neural network of classification problems (Vol. 34, pp. 241-258): Springer Netherlands.

Cho, K.B., and Wang, B.H. (1996). Radial basis function based adaptive fuzzy systems and their applications to system identification and prediction. *Fuzzy Sets and Systems, 83*(3), 325-339.

Engelbrecht, A.P. (2001). A new pruning heuristic based on variance analysis of sensitivity information. *IEEE Transactions on Neural Networks, 12*(6), 1386-1399.

Han, H., and Qiao, J. (2010). A Self-Organizing fuzzy neural network based on a Growing-and-Pruning algorithm. *IEEE Transactions on Fuzzy Systems, 18*(6), 1129-1143.

Han, H., and Qiao, J. (2013). A structure optimisation algorithm for feedforward neural network construction. *Neurocomputing, 99*, 347-357.

Huang, G., Saratchandran, P., and Sundararajan, N. (2005). A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation. *IEEE Transactions on Neural Networks, 16*(1), 57-67.

Jinn-Tsong, T., Jyh-Horng, C., and Tung-Kuan, L. (2006). Tuning the structure and parameters of a neural network by using hybrid Taguchi-genetic algorithm. *IEEE Transactions on Neural Networks, 17*(1), 69-80.

Leung, F.H.F., Lam, H.K., Ling, S.H., and Tam, P.K.S. (2003). Tuning of the structure and parameters of a neural network using an improved genetic algorithm. *IEEE Transactions on Neural Networks, 14*(1), 79-88.

Liang, N.Y., Huang, G.B., Saratchandran, P., and Sundararajan, N. (2006). A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Transactions on Neural Networks, 17*(6), 1411-1423.

Mackey, M.C., and Glass, L. (1977). Oscillation and chaos in physiological control systems. *Science, 197*(4300), 287-289.

Mountcastle, V.B. (1997). The columnar organization of the neocortex. *Brain, 120*(4), 701-722.

Rez, E.G.M., Najim, K., and Ikonen, E. (2007). Forecasting time series with a new architecture for polynomial artificial neural network. *Applied Soft Computing, 7*(4), 1209-1216.

Wang, N., Er, M.J., and Meng, X.Y. (2009). A fast and accurate online self-organizing scheme for parsimonious fuzzy neural networks. *Neurocomputing, 72*(16-18), 3818-3829.

Wu, S., and Er, M.J. (2000). Dynamic fuzzy neural networks - a novel approach to function approximation. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 30*(2), 358-364.

Xin, Y. (1999). Evolving artificial neural networks. *Proceedings of the IEEE, 87*(9), 1423-1447.

Xing, H., and Hu, B. (2009). Two-Phase construction of multilayer perceptrons using information theory. *IEEE Transactions on Neural Networks, 20*(4), 715-721.

Yang, G., Qiao, J., Bo, Y., and Han, H. (2013). A novel lateral inhibition neural network based on neocortex topology. *Control and Decision, 28*(11), 1702-1706.

Yang, G., Qiao, J., and Bo, Y. (2011). Research on artificial neural networks with spatial architecture based on span connection and lateral inhibition mechanism. *International Journal of Computational Science and Engineering, 6*(1-2), 86-95.

Yang, G., and Qiao, J.F. (2014). A fast and efficient two-phase sequential learning algorithm for spatial architecture neural network. *Applied Soft Computing, 25*, 129-138.

