Advanced Self-organizing Neural Networks with Fuzzy Polynomial Neurons : Analysis and Design

Sung-Kwun Oh and Dong-Yoon Lee

Abstract - We propose a new category of neurofuzzy networks- Self-organizing Neural Networks(SONN) with fuzzy polynomial neurons(FPNs) and discuss a comprehensive design methodology supporting their development. Two kinds of SONN architectures, namely a basic SONN and a modified SONN architecture are dicussed. Each of them comes with two types such as the generic and the advanced type. SONN dwells on the ideas of fuzzy rule-based computing and neural networks. Simulation involves a series of synthetic as well as experimental data used across various neurofuzzy systems. A comparative analysis is included as well.

Keywords - Self-organizing Neural Networks(SONN), Fuzzy Polynomial Neuron(FPN), fuzzy rule-based computing, GMDH, design methodology

1. Introduction

The challenging quest for constructing models of systems that come with significant approximation and generalization abilities as well as are easy to comprehend has been within the community for decades. While neural networks, fuzzy sets and evolutionary computing have augmented a field of modeling quite immensely, they have also given rise to a number of new methodological issues and increased awareness about tradeoffs one has to make in system modeling. When the dimensionality of the model goes up, so do the difficulties. In particular, when dealing with high-order nonlinear and multivariable equations of the model, we require a vast amount of data for estimating all its parameters. The Group Method of Data Handling (GMDH)[1] is one of the approaches that help alleviate the problem. GMDH is an analysis technique for identifying nonlinear relationships between system's inputs and outputs. The networks of polynomials built with GMDH have fewer nodes than standard artificial neural networks(ANN), but their nodes are more flexible[2]. The GMDH algorithm is carried out to generate an optimal architecture through a successive generation of both the nodes at each layer and the layers themselves by using the partial descriptions of the data.

Fuzzy sets emphasized the aspect of transparency of the models and a role of a model designer whose prior knowledge about the system may be very helpful in facilitating all identification pursuits. On the other hand, to build models of high approximation capabilities, there is a need for advanced tools. The art of modeling is to reconcile these two tendencies and find a workable synergistic environment. In this paper, we study a new neurofuzzy topology, called a Self-organizing Neural Network(SONN). SONN is a network resulting from the fusion of the extended GMDH algorithm and a fuzzy inference system. We introduce a complete learning scheme, discuss a way of growing the SONN and provide with a series of comprehensive experimental studies.

2. The SONN with FPNs

We introduce a fuzzy polynomial neuron(FPN). This neuron, regarded as a generic type of the processing unit, dwells on the concepts of fuzzy sets and neural networks. Fuzzy sets realize a linguistic interface by linking the external world - numeric data with the processing unit. Neurocomputing manifests in the form of a local polynomial unit realizing a nonlinear processing. The FPN encapsulates a family of nonlinear "if-then" rules. When arranged together, FPNs build a Self-organizing Neural Network (SONN). In this Fig. 1, the FPN consists of two basic functional modules. The first one, labeled by F, is a collection of fuzzy sets (here $\{A_l\}$ and $\{B_k\}$) that form an interface between the input numeric variables and the processing part realized by the neuron. Here x_q and x_p denote input variables. The second module (denoted here by P) is about the function – based nonlinear (polynomial) processing. This nonlinear processing involves some input variables (x_i and x_j).

The FPN realizes a family of multiple-input single-output rules. Each rule, refer again to Fig. 1, reads in the form

This work was supported by a grant No. R02-2000-00284 from Korea Science & Engineering Foundation.

Manuscript received; June 4, 2001 accepted: Oct. 25, 2001.

Sung-Kwun Oh is with Department of Electrical, Electronic and information Engineering, Wonkwang University, 344-2, Shinyong-Dong, Iksan, Chon-Buk, 570-749, Korea

Dong-Yoon Lee is with Department of Information Engineering, Joongbu University, San 2-25, Majeon-Ly, Chubu-Myon, Geumsan-Country, Chung-Nam, 312-702, Korea.

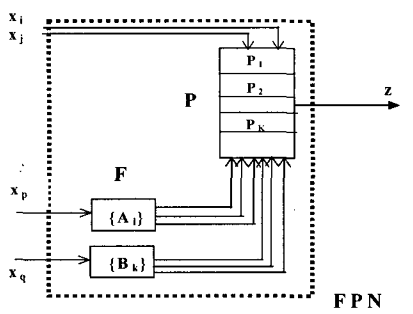


Fig. 1 A general topology of the generic FPN module (F: fuzzy set-based processing part, P: the polynomial form of mapping)

$$\text{if } x_p \text{ is } A_l \text{ and } x_q \text{ is } B_k \text{ then } y \text{ is } P_{lk}(x_i, x_j, \mathbf{a}_{lk}) \qquad (1)$$

where $\mathbf{a}_{lk}$ is a vector of the parameters of the conclusion part of the rule. Alluding to the input variables of the FPN, especially a way in which they interact with the two functional blocks there, we use the notation FPN (x_p, x_q; x_i, x_j). The processing of the FPN is governed by the following expressions that are in line of the rule-based computing existing in the literature.

a) The activation of the rule "K" is computed as an *and*-combination of the activations of the fuzzy sets standing in the rule. This combination of the subconditions is realized through any t-norm with the minimum and product. Denote the resulting activation level by □-.
b) The activation levels of the rules contribute to the output of the FPN computed as a weighted average of the individual condition parts(functional transformations) P_K (K=(1,k))

$$z=\sum_{K=1}^{\text{all rules}} \mu_K P_K(x_i,x_j,\mathbf{a}_K) \Big/ \sum_{K=1}^{\text{all rules}} \mu_K = \sum_{K=1}^{\text{all rules}} \tilde{\mu}_K P_K(x_i,x_j,\mathbf{a}_K) \qquad (2)$$

3. The topology of the SONN

Proceeding with the overall SONN architecture, see Fig. 2., essential design decisions have to be made with regard to the number of input variables and the order of the polynomial occurring in the conclusion part of the rule. Following these criteria, we distinguish between two fundamental *types* of the SONN architectures. Moreover, for each type of the topology we identify two *cases*.

(a) Basic SONN architecture – The number of the input variables of the fuzzy rules in the FPN node is kept the same in each layer of the network.

Case 1. The order of the polynomial in the conclusion part of the fuzzy rules is the same in all the nodes of each layer

Case 2. The order of such polynomial in the nodes of the 2nd layer or higher is different from the one occurring in the rules located in the 1st layer.

(b) Modified SONN architecture – The number of the input variables of the fuzzy rules in the FPN differs across the layers of the network

Case 1. The order of the polynomial in the conclusion part of the fuzzy rules is the same in all the nodes of each layer

Case 2. The order of such polynomial in the nodes of the 2nd layer or higher is different from the one occurring in the rules located in the 1st layer.

And also for each case, we consider two kinds of input vector formats in the conclusion part of the fuzzy rules of the 1st layer, namely i) selected inputs and ii) entire system inputs.

i) The input variables of the consequence part of the fuzzy rules are same as the input variables of premise part.
ii) The input variables of the consequence part of the fuzzy rules in a node of the 1st layer are same as the system input variables and the input variables of the consequence part of the fuzzy rules in a node of the 2nd layer or higher are same as the input variables of premise part.

If there are less than three input variables, the generic SONN algorithm does not generate a highly versatile structure. To alleviate this problem, the advanced type of the architecture is taken into consideration that can be treated as the modified version of the generic type of the topology of the network as shown in Fig. 2. Accordingly we identify also two types as the following.

i) Advanced SONN: in case that the no. of system input variables is less than three, the advanced type is used
ii) Generic SONN: in case that the no. of system input variables is four or higher, the generic type is used.

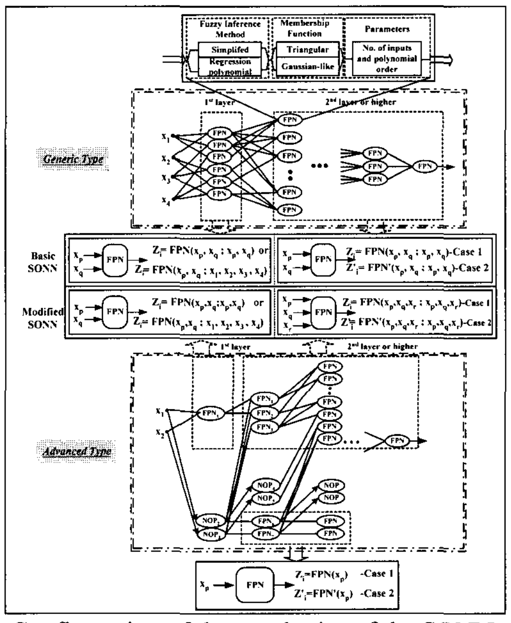


Fig. 2 Configuration of the topologies of the SONN and the design alternatives available within a single FPN

In the Figure, the "NOP$_A$ node" means the Ath node of the current layer that is the same as the node of the corresponding previous layer(NOP denotes no operation). An arrow to the NOP node is used to show that the corresponding same node moves from the previous layer to the current layer.

4. The design of the SONN

The SONN comes from a highly versatile architecture both in the flexibility of the individual nodes (that are essentially banks of nonlinear "if-then" rules) as well as the interconnectivity between the nodes and organization of the layers. The learning of SONNs dwells on the extended GMDH algorithm. When developing an FPN, we choose the input variables of a node from N input variables x_1, x_2, ..., x_N. The total number of the nodes in the current layer depends on the number of the selected input variables that come from the nodes situated in the preceding layer. This results in $k = N!/(N-r)!r!$ nodes, where r is the number of the selected nodes (input variables). Based on the number of the selected nodes (input variables) and the order of the polynomial, we build various architectures of the SONNs. The parameters of the conclusion part of the FPN are estimated by solving a standard mean-squared problem. While the structure of the SONN is regular in terms of the arrangement of the processing units into layers, the network itself exhibits interesting self-organization capabilities meaning that the structure can grow to minimize the performance index. The growth has two dimensions, namely the number of the layers and the size of each layer. The layers are added as required (the network grows in its depth). Each layer may have a different number of nodes. Here we proceed with a selection strategy and pick up the best nodes in the layer and ignoring the rest. In this sense the width of the network is not specified in advance (expect for the predefined upper limit) and can vary from layer to layer. We use the predefined number W of nodes (width of the layer) characterized by the best predictive performance (based on the values of the performance index). The outputs of the retained nodes (FPNs) serve as inputs to the next layer (generation) of the network. The termination condition that controls the growth of the model consists of two components, that is the performance index and a size of the network (expressed in terms of the maximal number of the layers). This size of the network has been experimentally found to form a sound compromise between the high accuracy of the resulting model and its complexity as well as generalization abilities.

5. A stability measure of the topology of the SONN

The question arises as to the selection of the proper structure of the SONN. Obviously, the performance of the network on the training and testing set are two important aspects one should take into account. The values of PI$_s$ and EPI$_s$, refer to Table 1, are good indicators of the approximation and generalization capabilities of the SONN. We also introduce the following ratio

$$\kappa = \mathrm{EPI}_s/\mathrm{PI}_s \tag{3}$$

This index can be viewed as a measure of "stability" of the model expressing how much the network's performance deteriorates over the testing data. Our choice of the structure is then guided by the minimal value of the ratio. Additionally, if the values of κ do not change very much over a family of the SONN architectures, the selection may not seem to be critical(intuitively the minimal size of the network will be our preference). One should stress, however, that the usefulness of the above index may be limited if the values of the PI vary significantly.

6. Simulation results

We illustrate the performance of the network and elaborate on its development by experimenting with data coming from the gas furnace process. The time series data(296 input-output pairs) resulting from the gas furnace process has been intensively studied in the previous literatures[3-8]. The delayed terms of methane gas flow rate, $u(t)$ and carbon dioxide density, $y(t)$ are used as three types of system input variables(System Inputs(SI): 2,3,4) with vector formats such as [u(t-3), y(t-1)], [u(t-2), y(t-2), y(t-1)] and [u(t-2), u(t-1), y(t-2), y(t-1)]. And as output variable, $y(t)$ is used. The generic type of SONN is utilized for SI-4 and the advanced type of SONN, for SI-2 and SI-3. The performance of two types of SONN architectures is far better both in the sense of its prediction and approximation abilities than other works studied in the literatures[3-8] as shown in Table 1. Where PI$_s$(EPI$_s$) is defined as the mean square errors(MSE). The results of computations of stability index κ are also summarized in Table 1. In light of the values reported there, a preferred architecture of the network is selected and its detailed topology is visualized in Fig. 3. The shadowed nodes indicate neurons which have the optimal polynomial in each layer(the optimal being expressed from the viewpoint of PI as well as EPI). And the values of the performance index vis-à-vis number of layers of the advanced and modified SONN related to the preferred architectures of the network are shown in Fig. 4. When using Gaussian fuzzy sets, the minimal value of the performance index, that is PI$_s$=0.021, EPI$_s$=0.124 are obtained by using 3 inputs in the 1st layer and 2 inputs in the 2nd layer or higher($3 \rightarrow 2$ node inputs with Type 4). As shown in Fig. 4, four types such as Type 1(Constant), Type 2(Linear), Type 3(Quadratic), and Type 4(Modified quadratic) are utilized as the polynomial type of the consequence part of the fuzzy rules.

In Fig. 3, ○ : The A^th node of the each corresponding layer used for the generation of the output $\hat{y}$. ◉: The A^th node of the each corresponding layer used for the generation of the output $\hat{y}$ and indicates the optimal node in each layer. ◌: The A^th node of the each corresponding layer that is not used for the generation of the output $\hat{y}$. ◌ : The A^th node of the each corresponding layer that is not used for the generation of the output $\hat{y}$ and indicates the optimal node in each layer. ——: The solid line used for the generation of the output $\hat{y}$, ----- : The dashed line that is not used for the generation of the output $\hat{y}$.

Table 1 Comparison of identification error with previous fuzzy models (PI- performance index over the entire data set, PI_s- performance index on the training data, EPI_s – performance index on the testing data)

Model				MSE			Stability index
				PI	PI_s	EPI_s	κ
Tong's model[3]				0.469			
Sugeno and Yasukawa's model[4]				0.190			
Xu's model[5]				0.328			
Pedrycz's model[6]				0.320			
Oh and Pedrycz's model[7]				0.123	0.020	0.271	
Kim, et al.'s model[8]					0.034	0.244	
Our model	Advanced (SI:2)	Basic	Case1	0.119	0.016	0.266	16.625
			Case2	0.114	0.016	0.265	16.56
		Modified	Case1	0.091	0.013	0.267	20.53
			Case2	0.09	0.013	0.272	20.92
	Advanced (SI:3)	Basic	Case1	0.040	0.013	0.123	9.461
			Case2	0.047	0.017	0.144	8.470
		Modified	**Case1**	**0.049**	**0.021**	**0.124**	**5.904**
			Case2	0.052	0.020	0.130	6.5
	Generic (SI:4)	Basic	Case1	0.049	0.016	0.116	7.25
			Case2	0.047	0.016	0.128	8
		Modified	Case1	0.057	0.016	0.133	8.312
			Case2	0.059	0.018	0.131	7.278

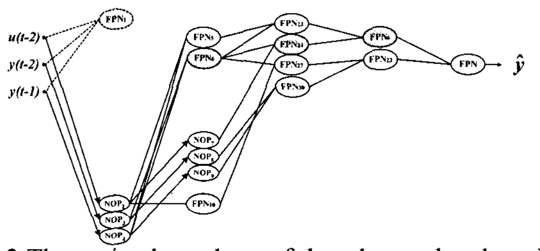
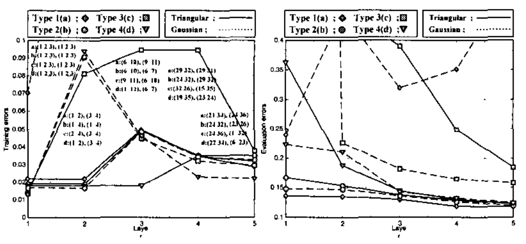


Fig. 3 The optimal topology of the advanced and modified SONN in Case 1(SI : 3, 3→2 node inputs, Type 4 and Gaussian MF)



(a) In case of training (b) In case of evaluation

Fig. 4 Performance index of the advanced and modified SONN in Case 1 (SI : 3)

7. Conclusions

In this paper, we have introduced an idea of a self-organizing neural networks (SONN) with fuzzy polynomial neurons, studied its properties and came up with a design procedure. Extensive experimental studies produced superb results. Some general observations can be summarized as follows: (i)with a properly selected type of membership functions and the organization of the layers, SONN performs better than other models, (ii) the architecture of the SONN is not fully predetermined and can be generated (adjusted) during learning, (iii) one can reduce a conflict between overfitting and generalization abilities of the model, and (iv) the diversity of the SONN architectures helps determine the best alternative for the problem at hand.

Acknowledgment

This work was supported by a grant No. R02-2000-00284 from Korea Science & Engineering Foundation.

References

[1] A.G. Ivahnenko, "Polynomial theory of complex systems", *IEEE Trans. Systems, Man and Cybernetics*, SMC-12, pp. 364-378, 1971

[2] R. Hecht-Nielsen, "*Neurocomputing*", Addison-Wesley, Reading, MA, 1990.

[3] R.M. Tong, "The evaluation of fuzzy models derived from experimental data", *Fuzzy Sets and Systems*, Vol.13, pp.1-12, 1980.

[4] M. Sugeno and T. Yasukawa, "A fuzzy-logic-based approach to qualitative modeling", *IEEE Trans. Fuzzy Systems*,Vol. 1, No. 1, pp. 7-31, 1993

[5] C.W. Xu and Y. Zailu, "Fuzzy model identification self-learning for dynamic system", *IEEE Trans on Systems, Man, Cybernetics*, Vol.

[6] W. Pedrycz, "An identification algorithm in fuzzy relational system", *Fuzzy Sets and Systems*, Vol. 13, pp. 153-167, 1984

[7] S.-K. Oh and W. Pedrycz, "Identification of Fuzzy Systems by Means of an Auto-tuning Algorithm and Its Application to Nonlinear Systems", *Fuzzy Sets and Systems*, Vol. 115, No. 2, pp. 205-230, 2000

[8] E. Kim, H. Lee, M. Park and M. Park, "A simple identified Sugeno-type fuzzy model via double clustering", *Information Sciences*, Vol. 110, pp. 25-39, 1998

[9] S.-K. Oh and W. Pedrycz, "The Design of Self-organizing Polynomial Neural Networks", *Information Sciences*, 2002 (to appear).

[10] B.-J. Park, W. Pedrycz and S.-K. Oh, "Identification of Fuzzy Models with the Aid of Evolutionary Data

Granulation", *IEE proceedings-CTA,* Vol. 148, Issue 05, pp. 406-418, 2001
[11] B.-J. Park, W. Pedrycz and S.-K. Oh, "Fuzzy Polynomial Neural Networks: Hybrid Architectures of Fuzzy Modeling", *IEEE Trans. on Fuzzy Systems,* 2002 (to appear)
[12] S.-K. Oh, W. Pedrycz and H.-S. Park, "Self-organizing Networks in Modeling Experimental Data in Software Engineering", *IEE proceedings.-CDT,* 2002 (to appear)
[13] S.-K. Oh, "*Fuzzy model and control system by C programming*", NAEHA Press, Korea, 2002

Sung-Kwun Oh received the B.S., M.S., and Ph. D. degrees in the Department of Electrical Engineering from Yonsei University, Korea, in 1980, 1983, and 1993, respectively. He had worked as a postdoctoral fellow in the Department of Electrical and Computer Eng. Univ. of Manitoba, Canada, from 1996 to 1997. He is currently an Associate Professor in the School of Electrical, Electronic and Information Engineering, Wonkwang University. His research interests include fuzzy and neural systems, intelligent control, and computational intelligence. He currently serves as an Associate Editor of the journal of KIEE and ICASE.
E-mail: ohsk@wonkwang.ac.kr

Dong-Yoon Lee received the B.S. degree in the Department of Electrical Engineering from Wonkwang University, in 1987. And he received the M.S., Ph.D. degrees in the Department of Electrical Engineering from Yonsei University, in 1990, 2001, respectively. He had worked as a research engineer in Intellect, J-tek and KIST from 1990 to 2000. From March 2001 to February 2002, he was a B.K. Professor in the School of Electrical, Electronic and Information Engineering, Wonkwang University. He is currently a Professor in the School of Information Engineering, Joongbu University.