2001년도 대한전기학회 하계학술대회 논문집 2001.7.18-20

퍼지 및 다항식 뉴론에 기반한 새로운 동적퍼셉트론 구조

김동원, 박호성, 오성권
원광대학교 공과대학 제어계측공학과

Fuzzy and Polynomial Neuron Based Novel Dynamic Perceptron Architecture

Dong-Won Kim, Ho-Sung Park, Sung-Kwun Oh
School of Electrical and Electronic Engineering Wonkwang Univ.

Abstract - In this study, we introduce and investigate a class of dynamic perceptron architectures, discuss a comprehensive design methodology and carry out a series of numeric experiments. The proposed dynamic perceptron architectures are called as Polynomial Neural Networks(PNN). PNN is a flexible neural architecture whose topology is developed through learning. In particular, the number of layers of the PNN is not fixed in advance but is generated on the fly. In this sense, PNN is a self-organizing network. PNN has two kinds of networks, Polynomial Neuron(PN)-based and Fuzzy Polynomial Neuron(FPN)-based networks, according to a polynomial structure. The essence of the design procedure of PN-based Self-organizing Polynomial Neural Networks(SOPNN) dwells on the Group Method of Data Handling (GMDH)[1]. Each node of the SOPNN exhibits a high level of flexibility and realizes a polynomial type of mapping (linear, quadratic, and cubic) between input and output variables. FPN-based SOPNN dwells on the ideas of fuzzy rule-based computing and neural networks. Simulations involve a series of synthetic as well as experimental data used across various neurofuzzy systems. A detailed comparative analysis is included as well.

1. Introduction

Recently, a lot of attention has been directed to advanced techniques of system modeling. Neural networks, fuzzy sets and evolutionary computing have augmented a field of modeling quite immensely, they have also gave rise to a number of new methodological issues and increased awareness about tradeoffs one has to make in system modeling. The art of modeling is to reconcile these two tendencies and find a workable synergistic environment. In this study, we introduce a new class of Self-organizing Polynomial Neural Networks (SOPNN), namely Polynomial Neuron(PN) based SOPNN and Fuzzy Polynomial Neuron(FPN) based SOPNN.

In a PN based SOPNN, this network comes with a high level of flexibility as each node can have a different number of input variables as well as exploit a different order of the polynomial (say, linear, quadratic, cubic, etc.). In comparison to well-known neural networks whose topologies are commonly prior to all detailed (parametric) learning, the SOPNN architecture is not fixed in advance but becomes fully optimized. Especially, the number of layers of the SOPNN architecture can be modified with new layers added, if required. FPN based SOPNN is a network resulting from the fusion of the extended GMDH algorithm and a fuzzy inference system. Each node of the SOPNN, that is a fuzzy polynomial neuron (FPN) operates as a compact fuzzy inference system. By exploiting several types of regression polynomials in the conclusion part of the rules, the architecture of SOPNN can be easily changed to adapt to system environment.

In this study, we provide with a general taxonomy of the SOPNNs, discuss detailed learning schemes and include detailed experimental studies.

2. The architecture of fuzzy and polynomial neuron based dynamic perceptron

2.1 PN based SOPNN

We introduce two kinds of polynomial neuron based SOPNN structures, namely the basic and the modified SOPNN. In what follows, we discuss their architectural details. More specifically, the main features of these architectures are as follows

(a) Basic PN based SOPNN structure - The number of input variables of PDs is same in every layer.

Case 1. The polynomial order of PDs is the same in each layer of the network.

Case 2. The polynomial order of PDs in the 2^{nd} layer or higher has a different or modified type in comparison with the one of PDs in the 1^{st} layer.

(b) Modified PN based SOPNN structure - The number of input variables of PDs varies from layer to layer.

Case 1. The polynomial order of PDs is same in every layer.

Case 2. The polynomial order of PDs in the 2^{nd} layer or higher has a different or modified type in comparison with the one of PDs in the 1^{st} layer.

2.2 FPN based SOPNN

The topology of the FPN based SOPNN implies the ensuing learning mechanisms; in the description below we indicate some of these learning issues that permeate the overall architecture. First, the network is homogeneous in the sense it is constructed with the use of the FPNs. It is also heterogeneous in the sense that FPNs can be very different and this contributes to the generality of the architecture. The network may contain a number of hidden layers each of them of a different size (number of nodes). The nodes may have a different number of inputs and this triggers a certain pattern of connectivity of the network. The FPN itself promotes a number of interesting design options, see Fig. 1. These alternatives concern a choice of the membership functions, the type of the conclusion (consequence) part, and the associated order of the polynomial realizing a conclusion part of the rule.

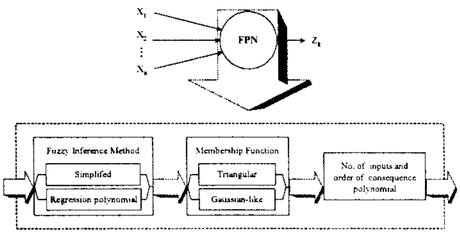

Fig. 1. The design alternatives available within a single FPN

Basic FPN based SOPNN and Modified FPN based SOPNN are as follows. Moreover, for each type of the topology we identify two cases.

(a) Basic FPN based SOPNN 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 polynomial order of the consequence part of the fuzzy rules is same in the nodes of every layer of the network

Case 2. The polynomial order of the consequence part of the fuzzy rules in the nodes of the second layer or higher is different in comparison to the order of the nonlinearity encountered in the nodes in the first layer.

(b) Modified FPN based SOPNN 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 2^{nd} layer or higher is different from the one occurring in the rules located in the 1st layer.

2.2.1 The fuzzy polynomial neuron (FPN)

As shown in Fig. 2, the FPN consists of two basic functional modules. The first one, labeled by F, is a collection of fuzzy sets that form an interface between the input numeric variables and the processing part realized by the neuron. In this figure, 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). Quite commonly, we will be using a polynomial form of the nonlinearity, hence the name of the fuzzy polynomial processing unit.

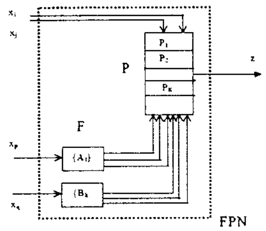

Fig. 2. A general topology of the generic FPN module; note its fuzzy set-based processing part (the module denoted by F) and the polynomial form of mapping (P)

3. The algorithm of the self-organizing polynomial neural networks

The SOPNN comes a highly versatile architecture both in the flexibility of the individual nodes as well as the interconnectivity between the nodes and organization of the layers. Overall, the framework of the design procedure of the SOPNN comes as a sequence of the following steps

[Step 1] Determine systems input variables.

[Step 2] Using available experimental data, form a training and testing data set.

[Step 3] Choose a structure of the SOPNN.

[Step 4] Determine the number of input variables and the order of the polynomial.

▶ The number of input variables and the order of the polynomial in the PN based SOPNN

▶ The number of input variables and the polynomial order of the consequence part of the fuzzy rules in the FPN based SOPNN

[Step 5] Select nodes with the best predictive capabilities.

[Step 6] Check the stopping criterion.

[Step 7] Determine new input variables for the next layer.

4. Simulation studies

In this section, we illustrate the development of the SOPNN and show its performance for a number of well-known and widely used datasets. The first one is a time series of gas furnace (Box-Jenkins data) which was studied previously in [2-7]. The delayed terms of methane gas flow rate, $u(t)$ and carbon dioxide density, $y(t)$ are used as system input variables. And as output variable, $y(t)$ is used. The number of system inputs(SI), inputs, and output used to design an optimal model from gas furnace process data are $u(t-2), u(t-1), y(t-2), y(t-1) : y(t)$

The experiments were completed for four

fundamental architectures of the SOPNNs and the results are shown in a series of figures.

4.1 PN based SOPNN

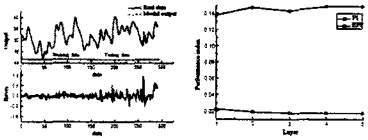

Fig. 3. Output comparisons, identification errors, and performance index of the basic SOPNN in Case 1(3 inputs ; Type 1)

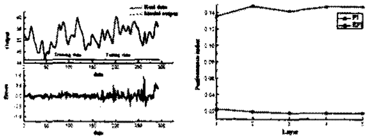

Fig. 4. Output comparisons, identification errors, and performance index of the basic SOPNN in Case 2(3 inputs ; 1st layer: Type 3, 2nd layer or higher: Type 1)

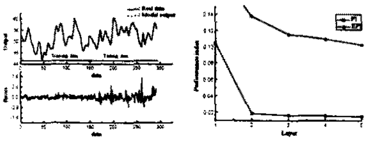

Fig. 5. Output comparisons, identification errors, and performance index of the modified SOPNN in Case 1(1st layer: 2 inputs , 2nd layer or higher: 3 inputs ; Type 2)

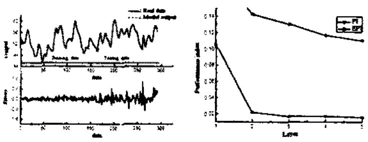

Fig. 6. Output comparisons, identification errors, and performance index of the modified SOPNN in Case 2(1st layer: 2 inputs, Type 1, 2nd layer or higher: 3 inputs, Type 2)

4.2 FPN based SOPNN

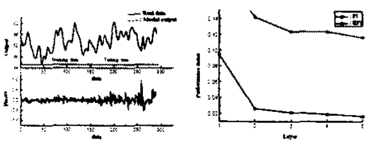

Fig. 7. Output comparisons, identification errors, and performance index of the basic SOPNN in Case 1(2 inputs ; Type 2, Gaussian MF)

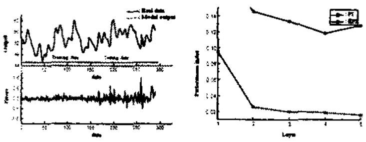

Fig. 8. Output comparisons, identification errors, and performance index of the basic SOPNN in Case 2(2 inputs ; 1st layer: Type 2, 2nd layer or higher: Type 4, Gaussian MF)

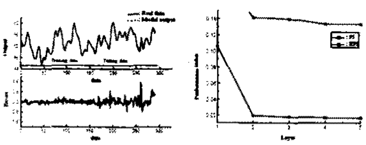

Fig. 9. Output comparisons, identification errors, and performance index of the modified SOPNN in Case 1(1st layer: 2 inputs , 2nd layer or higher: 3 inputs ; Type 1, triangular MF)

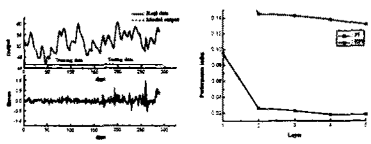

Fig. 10. Output comparisons, identification errors, and performance index of the modified SOPNN in Case 2(1st layer: 2 inputs, Type 4, 2nd layer or higher: 3 inputs, Type 1, triangular MF)

Table 1 provides a comparison of the SOPNN architectures with other models being already proposed in the literature.

Table 1. Performance of the SOPNN for different structures

Model				MSE		
				PI	PI_s	EPI_s
[2]				0.355		
[3]				0.320		
[4]				0.123	0.020	0.271
Our model	PN based	Basic	Case1	0.067	0.017	0.148
			Case2	0.064	0.017	0.147
		Modified	Case1	0.045	0.014	0.102
			Case2	0.044	0.015	0.110
	FPN based	Basic	Case1	0.049	0.016	0.116
			Case2	0.047	0.016	0.128
		Modified	Case1	0.057	0.016	0.133
			Case2	0.059	0.018	0.131

5. Concluding remarks

In this study, we introduced a class of self-organizing polynomial neural networks, discussed a diversity of their topologies, came up with a detailed design procedure, and used these networks to nonlinear system modeling.

Acknowledgement

This work was supported by grant No. 2000-1-30300-009-3 from the Basic Research Program of the Korea Science&Engineering Foundation

[References]

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

[2] M. Sugeno and T. Yasukawa, "Linguistic Modeling Based on Numerical Data", *IFSA'91 Brussels, Computer, Management & Systems Science*, pp.264-267, 1991

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

[4] 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.

[5] G.E.P. Box and F.M. Jenkins, *Time Series Analysis : Forecasting and Control*, 2nd ed. Holden-day, 1976

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

[7] E.T. Kim, M.K. Park, S.H. Ji, M. Park, "A New Approach to Fuzzy Modeling", *IEEE Trans. on Fuzzy Systems*, Vol. 5, No. 3, pp. 328-337, 1997.