A Polynomial Neural network with Controllable Precision and Human-Readable Topology II: Accelerated Approach Based on Expanded Layer

Gang Liu and Jing Wang

Abstract—How about converting Taylor series to a network to solve the black-box nature of Neural Networks? Controllable and readable polynomial neural network (Gang transform or CR-PNN) is the Taylor expansion in the form of network, which is about ten times more efficient than typical BPNN for forward-propagation. Additionally, we can control the approximation precision and explain the internal structure of the network; thus, it is used for prediction and system identification. However, as the network depth increases, the computational complexity increases. Here, we presented an accelerated method based on an expanded order to optimize CR-PNN. The running speed of the structure of CR-PNN II is significantly higher than CR-PNN I under preserving the properties of CR-PNN I.

Index Terms—CR-PNN, Taylor series, neural network, approximation, regression, system identification

I. INTRODUCTION

O data, artificial neural networks (ANNs) have been widely applied to a variety of engineering fields. As a “learning machine”, ANNs implement the mapping of some complex systems [3]. However, models constructed using ANNs encounter difficulty in preserving the physical explanations of real-world problems. Usually, ANNs are considered as a class of “nonparametric” approaches due to its “black box” nature. The further development of neural networks in system identification has been greatly constrained (see Fig. 1).

As to explain the parameters of models constructed using ANNs, many methods have been proposed. In the literature [4], Bao-Gang Hu et al. categorized the studies in this direction into two types of strategies: 1) Incorporating prior information into the design of ANNs [5, 6]. 2) Extracting knowledge or rules embedded within neural networks [7-9]. For the first strategy, there are thirteen schemes described in [4], such as adjusting the search space and search steps, partitioning a task into independent subtasks [10], and imposing functionals as the constrained learning algorithms (CLA) [11]. Although these schemes have achieved successful results in some specific professional issues, it lacks general applicability.

In recent years, some studies attempted to explain ANNs using Taylor series or polynomial [12, 13]. It is also troublesome. How about converting Taylor series to a network? Besides, we found an interesting phenomenon. Series expanded around 0 of function \( f(x) \) is as follows.

\[
f(x) = f(0) + f'(0)x + \frac{f''(0)}{2!}x^2 + \cdots + \frac{f^{(n)}(0)}{n!}x^n + \cdots
\]

Derivatives in Taylor series expansion have likely a similar effect as error back-propagation and chain rule [14]. Then, considering Jiushao Qin or Horner Rule, we designed CR-PNN I (see Fig. 2) [2]. Although it is about ten times more efficient than typical BPNN for forward-propagation, the computational complexity increases with increasing the network depth. Therefore, based on the expanded layer, we presented CR-PNN II to optimize high order.

II. RELATED WORK

In this section, we show previous works on CR-PNN I and highlight the proven properties of CR-PNN I [2].

This earlier version from Gang Liu is only used to scoop CR-PNN before peer-review. We will submit it to a journal after correction.

Gang Liu is a doctoral student of Institute of Robotics and Intelligent Systems, School of Mechanical Engineering, Xi'an Jiao Tong University, Xi'an 710049 China (e-mail: gangliu.6677@gmail.com). [Put forward CR-PNN II]. Jing Wang is an associate professor of Institute of Robotics and Intelligent Systems, School of Mechanical Engineering, Xi'an Jiao Tong University, Xi'an 710049 China (e-mail: wangpeile@gmail.com) [Offer guidance to this study].
1. CR-PNN I, in fact, essentially is the Taylor expansion in the form of network.
2. Derivatives in Taylor expansion are imitated by error back-propagation algorithm.
3. The number of layers represents precision. We can approach the real system step by step.
4. To some extent, CR-PNN I is morphologically similar to statistical regression. However, the computational complexity of statistical regression does increase exponentially with the polynomial order. For this condition, CR-PNN I requires only increasing one layer, whether the present layers are.
5. The computation complexity of the CR-PNN structure is far lower than the typical ANNs structure in the same effect.
6. The generalization capability of CR-RNN matches and often exceeds that of BPNN due to targeted control of precision.
7. Because CR-PNN I essentially is the Taylor series, it could generate relation spectrum or polynomial spectrum inheriting Taylor series for system analysis. Due to this peculiarity, we also called CR-PNN as “Gang transform” that transforms the unknown system to Taylor series network [15, 16].

For property 7, we would like to correct our previous paper [2]. We withdraw the conclusion that it has a globally optimal solution, and we hope to be able to prove it strictly mathematically or to give a better solution for CR-PNN, such as using the genetic algorithm to optimize CR-PNN like BPNN [17]. Here, we emphasize the essence of CR-PNN structure or Gang transform without considering the methods of optimizing general NNs. Although we cannot ensure the globally optimal solution, the property 3 endows CR-PNN greater generalization capability relative to typical ANNs.

We guess the multivariate relation spectrum is influenced by two factors:

a. Correlation of input variable: An input variable can be represented by a Taylor series for another input variable, especially for the higher order model.
b. Because we cannot ensure the globally optimal solution, the generating relation spectrum maybe not the optimal relation spectrum.

Despite of the above insufficient, the trained model can be “read” by the relation spectrum and used for system analysis. In the future, when issue a and b about the network are resolved, Gang transform may open a whole new perspective in general complex system analysis.

In this paper, while preserving the above properties, we optimize the higher-order terms and further reduce the computational complexity.

III. CR-PNN II

CR-PNN I increase the order of the approximate network by increasing the layer, resulting in some redundant terms when the number of layers beyond the input data dimension. CR-PNN II aims to solve this problem and is applied to the situation where the number of layers exceeds the input data dimension when using CR-PNN I.

A. Network structure of CR-PNN II

According to Figure 2, the initial order is 1, increased by one order for each increasing hidden layer. In other words, the primary role of layer is to increase the order of CR-PNN. When the number of layers is greater than the input data dimension, increasing the number of layers only increases the order of variables while no new interaction. Therefore, in CR-PNN II, we simply increase the order of variables by adding an expanded layer.

Figure 3 shows the network structure of CR-PNN II. It mainly consists of expanded layer, Taylor layer and output layer. Among them, based on the combination of layer order, to avoid missing items, the highest order of expanded layer $c$ is the number of Taylor layer $l$ plus 2. The forward propagation is as follows.

\[ \text{Expanded layer: } A^i = W_{0,i}X \odot X^c, \quad c \in [1, l + 2], c \in \mathbb{N} + \]  \hspace{1cm} (2)

\[ \text{Taylor layer: } A^i = W_{i-j}^{l-1}A^{i-1} \odot X, \quad i \in [2, l+1], \quad i \in \mathbb{N} + \]  \hspace{1cm} (3)

\[ \text{Output layer: } Y = A^{i+2} = W_{i-j+1}^{l+1}A^{i+1} \]  \hspace{1cm} (4)

Where $A^{i-j}$ and $A^i$ denote input and output of the current layer, respectively. $W_{i-j}^{l-1}$ denote the weight matrix. $X$ and $Y$
Fig. 3. CR-PNN II. Based on the combination of layer order, to avoid missing items, the highest order of expanded layer \( c \) is equal to the number of Taylor layer \( l \) plus 2. \( \{x_1, x_2, \ldots, x_n\} \in X, \{y_1, y_2, \ldots, y_m\} \in Y \)

denote input and output of the network, respectively. \( j \in [1, l + 2], j \in N^+, l \geq n, l \in N^+, n \in N^+ \)

According to Eq. (2), Eq. (3), and Eq. (4), the order of network \( L \) is determined as follows.

\[
L = l + c + 1
\]  

(5)

when \( c \) be equal to \( l + 2 \), the network obtains the highest order \( 2l + 3 \). Therefore, for a given network order, the algorithm to calculate the number of layers was as follows.

**Algorithm:** Calculation of number of layers

Require:
- \( n \): the input data dimension
- \( L \): the given network order

Calculation:
- \( n \to l \)
- while \( L > 2l + 3 \)
  - \( l + 1 \to l \)
- end

Number of layers: \( l + 2 \)

**B. Learning rule of CR-PNN II**

We describe an error back-propagation based learning rule for CR-PNN II [14].

The error-backpropagation is described as follows.

\[
\text{Overall error: } dA^{i+2} = \hat{Y} - Y
\]  

(6)

**IV. Experimental Results**

This study is based on our previous research on CR-PNN I [2]. Therefore, CR-PNN II was compared with CR-PNN I in terms of complex multivariate function approximation and computation complexity.

**A. Complex Multivariate Function Approximation**

To evaluate the effectiveness of CR-PNN II, we randomly generated two complex five-variates functions. Function 1
contains 4737 items, and function 2 contains 2772 items. The coefficients of items are as shown in figure 4, and the corresponding item can be found in supporting materials (GitHub: https://github.com/liugang1234567/CR-PNN#cr-pnn).

We randomly selected five input data and calculated the output of function 1 and function 2 separately. The input data were defined by:

\[
\begin{align*}
x_1(t) &= \sin(2t) \\
x_2(t) &= \sin(3t) \\
x_3(t) &= \sin(5t) \\
x_4(t) &= \sin(7t + 20) \\
x_5(t) &= \sin(11t)
\end{align*}
\] (11)

We trained CR-PNN II from 7th order to 14th order for function 1 and function 2. All networks run 50 with different initial parameters. Figure 5 illustrates the visualization results of complex multivariate function approximation. We randomly selected one output result from each network order. CR-PNN II showed good performance for approaching complex multivariate functions.

Besides, to compare the differences between CR-PNN I and CR-PNN II in convergent property, we also trained CR-PNN I from 7 order to 14 order for 50 runs with different initial parameters. Some examples of output were shown.
parameters, and employ the commonly used mean squared error (MSE) as the measurement of approximation. The MSE is defined as

$$MSE = \frac{1}{K} \sum_{k=1}^{K} (y_k - \hat{y}_k)^2$$  \hspace{1cm} (12)$$

Where $y_k$ and $\hat{y}_k$ denote the output of function and network respectively, and $K$ is the number of data.

Figure 1 shows the results of function approximation using CR-PNN I and CR-PNN II. As expected for CR-PNN I, as the network order is increased, the MSE has decreased accordingly. However, CR-PNN II did not show this property. This may be because the items of CR-PNN II is generated by the combination of the corresponding network nodes, and the different items require different approximation priority of the expanded layer and Taylor layer. Therefore, the structure of CR-PNN II is easily trapped in a local optimum relative to CR-PNN I. However, after finding the global optimal solution using the global optimum algorithm, the structure of CR-PNN II will show better real-time performance in online applications (see the next section).

**B. Computation Complexity**

Because the structure of CR-PNN is very simple compared with typical ANNs, the computational complexity can be easy to express.

CR-PNN I: $O\left( (L-1) \left( \frac{1}{2}n^2 + n + 1 \right) + m(n+1) \right)$  \hspace{1cm} (13)$$

CR-PNN II:

$$O\left( (n+1)^2 + (L-1)(n+1) + l \left( \frac{1}{2}n^2 + n + 1 \right) + m(n+1) \right)$$  \hspace{1cm} (14)$$

Where $n$ and $m$ express the dimension of input and output respectively, $L$ expresses the network order, and $l$ expresses the number of Taylor layer. Therefore, the structure of CR-PNN II was reduced by $O\left( \left[ L - (l+2) \right]^2 \right)$ . Figure 8...
showed a comparison of CR-PNN I and CR-PNN II for the number of network layers in the same network order.

In addition, this result was further confirmed through experiments. We measure the efficiency of the structure of CR-PNN II by the running time. The experiments are carried out using MATLAB 2019b on a 2.2-GHz laptop PC.

Measuring the running time in the forward-propagation and back-propagation is the best and most direct way to evaluate the computation complexity of network structure. We measured the running time for 1000 epoch and 1000 forward-propagation with 5000 samples, respectively. The experimental results are in agreement with the theory as expected (see Fig. 7). This means that CR-PNN II is faster than CR-PNN I in the online application.

V. CONCLUSION

Some studies attempted to explain ANNs using Taylor series or polynomial [12, 13]. We think it may be too troublesome. Therefore, we converted Taylor series to a neural network, called CR-PNN I. Thus, models constructed using CR-PNN I can be “read”. Additionally, we have demonstrated that CR-PNN I can adjust the network order to increase the generalization ability, and the running speed of CR-PNN I is significantly higher than the structure of typical ANNs [2].

Here, we presented CR-PNN II to optimize the higher-order terms and further reduce the computational complexity. CR-PNN II mainly includes expanded layer, Taylor layer, and output layer. Among them, the expanded layer is used to increase the order of the network, and Taylor layer is similar to the hidden layer of CR-PNN I. We compared CR-PNN II with CR-PNN I in terms of complex multivariate function approximation and computation complexity. The results demonstrated that CR-PNN II showed good property of multivariate function approximation, yet, was easily stuck in a local optimum. However, the running speed of the structure of CR-PNN II is significantly higher than CR-PNN I. For the issue of local optimum, we can optimize CR-PNN II like BPNN using the previous method, such as the genetic algorithm [17].

We emphasize that the structure of CR-PNN II showed high running speed in online applications under preserving the properties of CR-PNN I.

REFERENCES

[1] Z. Y. Ran and B. G. Hu, "Parameter Identifiability in Statistical Machine Learning: A Review," (in English), Neural Computation, Review vol. 29, no. 5, pp. 1151-1203, May 2017.
[2] G. Liu and J. Wang, "A Polynomial Neural Network with Controllable Precision and Human-Readable Topology for Prediction and System Identification," arXiv e-prints, p. arXiv:2004.03955Accessed on: April 01, 2020Available: https://ui.adsabs.harvard.edu/abs/2020arXiv200403955L
[3] K. Hornic, "Multilayer feedforward networks are universal approximators," vol. 2, no. 5, pp. 359-366, 1989.
[4] B. G. Hu, H. B. Qu, Y. Wang, and S. H. Yang, "A generalized-constraint neural network model: Associating partially known relationships for nonlinear regressions," (in English), Information Sciences, Article vol. 179, no. 12, pp. 1929-1943, May 2009.
[5] R. M. Golden, "Neural networks: A comprehensive foundation - Haykin,S.,” (in English), Journal of Mathematical Psychology, Book Review vol. 41, no. 3, pp. 287-292, Sep 1997.
[6] S. Milanic, S. Strmcnik, D. Sel, N. Hvala, and R. Karba, "Incorporating prior knowledge into artificial neural networks - an industrial case study," (in English), Neurocomputing, Article vol. 62, pp. 131-151, Dec 2004.
[7] L. M. Fu, "RULE GENERATION FROM NEURAL NETWORKS," (in English), Ieee Transactions on Systems Man and Cybernetics, Article vol. 24, no. 8, pp. 1114-1124, Aug 1994.
[8] C. W. Omlin and C. L. Giles, "Rule revision with recurrent neural networks," (in English), Neural Networks and Data Engineering, Article vol. 8, no. 1, pp. 183-188, Feb 1996.
[9] A. B. Tickle, R. Andrews, M. Golea, and J. Diederich, "The truth will come to light: Directions and challenges in extracting the knowledge embedded within trained artificial neural networks," (in English), Ieee Transactions on Neural Networks, Article vol. 9, no. 6, pp. 1057-1068, Nov 1998.
[10] R. Waibel, "Modular Construction of Time-Delay Neural Networks for Speech Recognition," (in English), Neural Computation, Article vol. 1, no. 1, pp. 39-46, Spr 1989.
[11] F. Han, Q. H. Ling, and D. S. Huang, "Modified constrained learning algorithms incorporating additional functional constraints into neural networks," (in English), Information Sciences, Article vol. 178, no. 3, pp. 907-919, Feb 2008.
[12] C. Xi, B. Khomtchouk, N. Marlio, and P. Mohanty, "Polynomial Regression As an Alternative to Neural Nets," (in English), arXiv, pp. 28 pp.-28 pp.., 13 2018.
[13] T. Poggio, "On optimal nonlinear associative recall," Biological Cybernetics, vol. 19, no. 4, pp. 201-209, 1975.
[14] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "LEARNING REPRESENTATIONS BY BACK-PROPAGATING ERRORS," (in English), Nature, Article vol. 323, no. 6088, pp. 533-536, Oct 1986.
[15] G. Liu and J. Wang, "A Relation Spectrum Inheriting Taylor Series: Muscle Synergy and Coupling for Hand," arXiv e-prints, p. arXiv:2004.11910Accessed on: April 01, 2020Available: https://ui.adsabs.harvard.edu/abs/2020arXiv200411910L
[16] G. Liu and J. Wang, "A Novel System Analysis Methodology: Transform Method, Relation Spectrum, and System Filter," Preprints 2020, p. 2020040538Accessed on: April 30, 2020, doi: 10.20944/preprints202004.0538.v1 Available: https://www.preprints.org/manuscript/202004.0538/v1
[17] F. Yu and X. Z. Xu, "A short-term load forecasting model of natural gas based on optimized genetic algorithm and improved BP neural network," (in English), Applied Energy, Article vol. 134, pp. 102-113, Dec 2014.