International Journal of Computer and Information Technology (ISSN: 2279 – 0764)
Volume 03 – Issue 06, November 2014

Discrete Process Neural Networks and Its Application in Time Series Prediction

Xin Li
School of Computer & Information Technology
Northeast Petroleum University
Daqing 163318, China
E-mail: lixin_dq {at} 163.com

Panchi Li
School of Computer & Information Technology
Northeast Petroleum University
Daqing 163318, China
E-mail: lipanchi {at} vip.sina.com

Abstract—**Considering that inputs of a process neural network (PNN) are generally time-varying functions while the inputs of many practical problems are discrete values of multiple series, in this paper, a process neural network with discrete inputs is presented to provide improved forecasting results for solving the complex time series prediction. The proposed model first makes the discrete input series carry out Walsh transformation, and then submits the transformed series to the network for training, which can solve the problem of space-time aggregation operation of PNN. In order to examine the effectiveness of the proposed method, the two examples are employed. First, the developed model is tested on the Mackey-Glass time series and has comparison with the results in literatures, and then, taking the actual data of sunspots during 1749-2007 as examples, the number of sunspots is predicted and the suitability of the developed method is examined in comparison with the other models to show its superiority. The proposed method provides a new way for the space environment prediction in future.**

Keywords- Process neuron; Process neural networks; Learning algorithm; Time series predication; sunspot number

I. INTRODUCTION (HEADING 1)

More than half a century, the research on neural networks (NNs) has been rapidly developed by people's continuous efforts, which is widely applied to various subjects. Especially, that a multi-layer neural network was proved to be a consistent approximator for continuous function in 1989 laid a theoretical foundation for neural networks and accelerated the development of neural networks. There are a lot of NNs architectures in the literature that work well when the number of inputs is relatively small, however when the complexity of the problem grows or the number of inputs increases, their performance decreases very quickly. In recent years, many new models and algorithms of NNs have emerged one after another. In general, neural networks have been maturing in addressing the issues on L_2 space map either in theory or in practical applications. On the other hand, the inputs of a system are generally time-related process in practical problems, such as the process of chemical reaction, the process of stock market volatility, etc. It has attracted considerable attention that NNs are used to solve the above problems. However, because of the time-dependent samples containing the large volume of data, the traditional neural networks are difficult to solve the problems of large sample learning and generalizing. Thus, new models are expected to solve these problems. Aiming at the problems that the inputs of many systems are continuous functions of time, and that the outputs of some control signals depend on the spatial aggregation of input functions, and are related to accumulation effect of time, a process neural network model is proposed by [1]. For the training of process neural networks, [2,3] give a learning algorithm based on orthogonal function basis expansion. Using the orthogonality of base function, aggregation operation in time field can be simplified effectively. However, for many practical problems, there are no accurate mathematical models or difficult to find them for a system, and the accurate analytic formulas of functions are difficult to determine. Therefore, it is necessary to study the learning algorithm of PNN with the discrete inputs.

In this paper, a learning algorithm of process neural networks with discrete input is presented based on discrete Walsh transformation. In order to explain effectiveness of the proposed method, the two complicated time series examples are used. First, the presented method is tested on the Mackey-Glass time series; then, take the actual data of sunspots during 1749-2007 as examples, to predict the number of sunspots. The results indicate that the presented method does not only solve those problems of large sample learning and generalizing, but also improve the approximation and prediction ability of networks.

II. PROCESS NEURAL NETWORK MODEL

A. Process Neuron

Process neuron is mainly composed of three operations, including weighting, aggregation and activation. The difference between process neuron and traditional neuron is that the inputs and weights of process neuron change with time, which can be functions of time. The aggregation operation can be the multi-input aggregation of space, and can be the accumulation of time. Therefore, it is the extension of traditional neuron in time-domain. The traditional neuron can be considered as the special case of process neuron.

This work was supported by the Youth Foundation of Northeast Petroleum University (Grant No. 2013NQ119) and the National Natural Science Foundation of China (Grant No. 61170132).

The structure of process neuron is shown in Fig.1, and the Eq. (1) is its input-output relationship of process neuron.

$$Y = f(\int_0^T \sum_{i=1}^{n} w_i(t)x_i(t)dt - \theta) \tag{1}$$

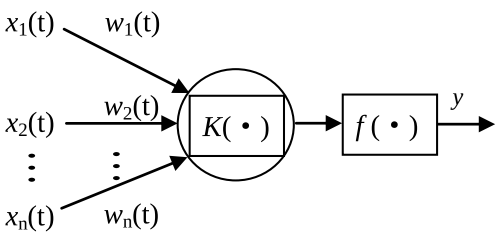

Figure 1. Process neuron.

B. *Process Neural Network Model*

Process neural network (PNN) is a network that is composed of several process neurons based on a certain topology. The topological structure is shown in Fig.2.

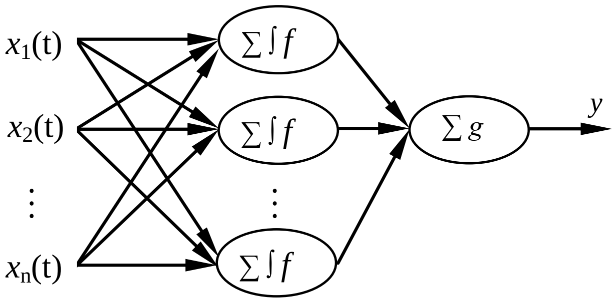

Figure 2. A process neural networks with a hidden-layer.

Here, there are n units in the input-layer; there are m units in the hidden-layer and the activation function is f ; there is a unit in the output-layer. The relationship between procedure inputs and output is as follows:

$$y = g(\sum_{j=1}^{m} v_j f(\int_0^T (\sum_{i=1}^{n} w_{ij}(t)x_i(t))dt - \theta_j^{(1)}) - \theta) \tag{2}$$

In Eq.(2), $w_{ij}(t)$ is the weight function of process neurons between input-layer and hidden-layer; v_j is the weight of process neurons between hidden-layer and output-layer; θ_j is the output-threshold of process neurons in the hidden-layer; [0,T] is the interval of time sampling.

III. Functional Approximating Capability of Process Neural Networks

Functional approximation capability is an important property of a process neural network, and it determines the applicability and the modeling capability of the process neural network for solving problems. In this section, the functional approximation capability of a process neural network will be discussed in detailed by the following definitions and theorems.

Definition1 Suppose that $K(\cdot)$: $\mathbf{R}^n \rightarrow V \subset \mathbf{R}$ is a random continuous function from $\mathbf{R}^n$ to $\mathbf{R}$, and is denoted as $K \in C(\mathbf{R}^n)$. Define functional class $\sum^n(K)=\{f:U \rightarrow V \mid f(x(t)) = \int_0^T K(x(t))\mathrm{d}t$ $x(t) \in U \subset \boldsymbol{R}^n$, $f(x) \in V \subset \boldsymbol{R}\}$.

Definition2 Suppose that $X(t) = (x_1(t), x_2(t), \ldots, x_n(t))^{\mathrm{T}}$, where $x_i(t) \in C[0,T]$. If $|x_i(t_1) - x_i(t_2)| \le L|t_1 - t_2|$ with $L \ge 0$ for any $t_1, t_2 \in [0,T]$, $x_i(t)$ is called to satisfy Lipschitz condition, if $\|X(t_1) - X(t_2)\| \le L_x |t_1 - t_2|$ with $L_x \ge 0$, $X(t)$ is called to satisfy Lipschitz condition, and if $\|K(X(t_1)) - K(X(t_2))\| \le L_K \|X(t_1) - X(t_2)\|$, $K(\cdot) \in C(\boldsymbol{R}^n)$ is called to satisfy Lipschitz condition.

The research on the traditional neural network has already proved the following well-known approximation theorem.

Theorem 1 [4] For any continuous function $g \in C(\mathbf{R}^n)$ there exists a feedforward neural network with only one hidden layer, which can approximate g with any accuracy.

Theorem 2 Traditional neural network is a special case of process neural network.

Proof: In $y = g(\sum_{j=1}^{m} v_j f(\int_0^T (\sum_{i=1}^{n} w_{ij}(t)x_i(t))dt - \theta_j^{(1)}) - \theta)$ if let $T = 0$, $x_i(t) = x_i$ and $w_{ij}(t) = w_{ij}$, then it can be simplified as

$$y = g(\sum_{j=1}^{m} v_j f(\sum_{i=1}^{n} w_{ij}x_i - \theta_j^{(1)}) - \theta) \tag{3}$$

This is a time-invariant traditional feed forward neural network with a single hidden layer. Thus, the proof is completed.

Theorem 3 For any continuous functional $G(x(t)) \in \sum^n(K)$ defined by Definition 1 and any $\varepsilon>0$, if $G(x(t))$ satisfies Lipschitz condition, then there exists a process neural network P such that $\|G(x(t)) - P(x(t))\| < \varepsilon$.

Proof: For any $G \in \sum^n(K)$, we have

$$G(x(t)) = \int_0^T K(x(t))\mathrm{d}t \tag{4}$$

Without loss of generality, let $T=1$, K is regarded as the composite function with respect to t, and the integral interval is divided into N equal parts, here $t_i=i/N$ (i=1, 2, …, N) is the partition point, then

$$G(x(t))=\sum_{i=1}^{N}\int_{t_{i-1}}^{t_i}K(x(t))\mathrm{d}t \quad (5)$$

Let functional $\tilde{G}(x(t))=\frac{1}{N}\sum_{i=1}^{N}K(x(t_i))$ be the approximation of $G(x(t))$, then

$$\begin{aligned}|G(x(t))-\tilde{G}(x(t))|&=|\sum_{i=1}^{N}\int_{t_{i-1}}^{t_i}K(x(t))dt-\frac{1}{N}\sum_{i=1}^{N}K(x(t_i))|\leq\\ &\sum_{i=1}^{N}|\int_{t_{i-1}}^{t_i}K(x(t))\mathrm{d}t-\frac{1}{N}K(x(t_i))|\end{aligned} \quad (6)$$

Because $K(x(t))$ is continuous with respect to t, by the interval mean value theorem, there exists $\xi_i\in[(i-1)/N,i/N]$ such that

$$\int_{t_{i-1}}^{t_i}K(x(t))dt=\frac{1}{N}K(x(\xi_i)) \quad (7)$$

Therefore,

$$\begin{aligned}&|G(x(t))-\tilde{G}(x(t))|\leq\frac{1}{N}\sum_{i=1}^{N}|K(x(t_i))-K(x(\xi_i))|\leq\\ &\frac{1}{N}\sum_{i=1}^{N}L_K\,||x(t_i)-x(\xi_i)||\leq\frac{1}{N}\sum_{i=1}^{N}L_K L_x\,|t_i-\xi_i|\leq\frac{L_K L_x}{N}\end{aligned} \quad (8)$$

where L_K and L_x are respectively Lipschitz constants of $K(x)$ about x and $x(t)$ about t. Therefore,

$$G(x(t))=\int_0^T K(x(t))\mathrm{d}t=\frac{1}{N}\sum_{i=0}^{N}K(x(t_i))+O(\frac{1}{N}) \quad (9)$$

Let $x(t_i)=x^{(i)}$. Because $K(x^{(i)})$:$\mathbf{R}^n\rightarrow V$ is the continuous function in $C(\mathbf{R}^n)$, according to Theorem 1 , it can be approximated by a traditional neural network, and based on Theorem 2, this traditional feed forward neural network can certainly be replaced by a process neural network P_i, i.e.

$$|K(x^{(i)})-P_i(x^{(i)})|<\varepsilon_i \quad (10)$$

where ε_i>0 is an arbitrarily small value, i=1, 2, …, N. Might as well let $\varepsilon_i<\varepsilon/(2N)$, from Eq.(9), there exists N_0, when $N>N_0$, we have

$$|G(x(t))-\frac{1}{N}\sum_{i=0}^{N}K(x(t_i))|<\frac{\varepsilon}{2} \quad (11)$$

Let

$$P(x(t))=\frac{1}{N}\sum_{i=0}^{N}P_i(x(t_i)) \quad (12)$$

Then

$$\begin{aligned}&|G(x(t))-P(x(t))|\\ &=|G(x(t))-\frac{1}{N}\sum_{i=0}^{N}K(x(t_i))+\frac{1}{N}\sum_{i=0}^{N}K(x(t_i))-\frac{1}{N}\sum_{i=0}^{N}P_i(x(t_i))|\\ &\leq|G(x(t))-\frac{1}{N}\sum_{i=0}^{N}K(x(t_i))|+|\frac{1}{N}\sum_{i=0}^{N}K(x(t_i))-\frac{1}{N}\sum_{i=0}^{N}P_i(x(t_i))|\\ &<\frac{\varepsilon}{2}+\frac{\varepsilon}{2}=\varepsilon\end{aligned} \quad (13)$$

$P(x(t))$ is solved.

IV. The Learning Algorithm of PNN with Discrete Inputs

For many practical problems, the inputs of systems are generally the time-dependent processes, such as the concentration change process of chemical reaction, the process of stock market volatility, the change process of temperature, humidity, fertilizer and light intensity for crops growth, etc. These processes are generally complex non-linear functions of time, and difficult to be determined by mathematical expressions. Hence, the application of the algorithm in [5] is limited. At present, the method to solve the limitation is that, firstly, curve fitting is adopted to acquire the approximate analytic formulas of input functions from discrete series, and then these mathematical functions are submitted to PNN [6]. There are mainly two disadvantages of this method. Firstly, the method exist fitting error, which has an influence on the precision of prediction. Secondly, the method increases the computational complexity of the algorithm. Therefore, a novel learning algorithm based on discrete Walsh transformation is proposed in this paper. The algorithm directly submits the discrete series of the system to the network inputs, which overcomes the limitation of the algorithm in [5].

A. *Walsh Transformation*

It can be seen from [7] that, any square-intergrable function in the interval [0,1]can be expanded as Walsh series. If the given function is a periodic function with a period of 1, the function can be expanded as Walsh series in the interval (-∞,+∞), and the series have general convergence. That is

$$f(t) = \sum_{n=0}^{\infty} F_n wal(n,t) \tag{14}$$

$$F_n = \int_0^1 f(t) wal(n,t) dt \tag{15}$$

The above Eq.(14) and Eq.(15) are a Walsh transformation pair of continuous functions. In the discrete condition that there are N sampling points, the Walsh transformation pair is as follows

$$x_i = \sum_{k=0}^{N-1} X_k wal(i, \frac{k}{2^n}) \ \ i = 0,1,\cdots,N-1, N = 2^n \tag{16}$$

$$X_k = \frac{1}{N} \sum_{i=0}^{N-1} x_i wal(i, \frac{k}{2^n}) \ \ k = 0,1,\cdots,N-1, N = 2^n \tag{17}$$

Using Eq.(16) and Eq.(17), the input information can implement discrete Walsh transformation.

Lemma 1: The sum of N discrete values of the product of two Walsh functions with different frequency in the interval [0,1] is equal to zero. As follows

$$\sum_{i=0}^{N-1} wal(j, \frac{i}{N}) wal(k, \frac{i}{N}) = 0 \quad j \neq k \quad j,k \leq n \quad N = 2^n \tag{18}$$

Lemma 2: The sum of N discrete values of the product of two Walsh functions with same frequency in the interval [0,1] is equal to N. As follows

$$\sum_{i=0}^{N-1} wal^2(j, \frac{i}{N}) = N \qquad j \leq n \quad N = 2^n \tag{19}$$

Theorem 4: For any discrete series $x_i, w_i (i = 0,1,\cdots,2^n - 1)$ in the interval [0,1] of two continuous functions $X(t), W(t)$, the following integral formula is tenable.

$$\int_0^1 X(t)W(t)dt = \lim_{N\to\infty} \sum_{i=0}^{N-1} wal(x_i) wal(w_i) \tag{20}$$

where $N = 2^n$, $x_i = X(t_i)$, $w_i = W(t_i)$ and $wal(\bullet_i)$ and $\bullet_i$ are the Walsh transformation pairs.

Proof: Suppose $t_i = \frac{i}{N}$ $(i = 0,1,\cdots,N-1)$ are $N = 2^n$ equal-division points in the interval [0,1]. According to Eq.(17),

$$\sum_{i=0}^{N-1} wal(x_i) wal(w_i) = \sum_{i=0}^{N-1} wal(X(t_i)) wal(W(t_i))$$

$$= \sum_{i=0}^{N-1} (\frac{1}{N} \sum_{j=0}^{N-1} X(t_j) wal(j, \frac{i}{N}))(\frac{1}{N} \sum_{k=0}^{N-1} W(t_k) wal(k, \frac{i}{N}))$$

$$= \frac{1}{N^2} \sum_{j=0}^{N-1} \sum_{i=0}^{N-1} X(t_j) W(t_j) wal^2(j, \frac{i}{N})$$

$$+ \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{\substack{j,k=0 \\ j \neq k}}^{N-1} X(t_i) W(t_j) wal(j, \frac{i}{N}) wal(k, \frac{i}{N})$$

From **Lemma 1** and **Lemma 2**, such results can be given as follows

$$\begin{cases} \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{\substack{j,k=0 \\ j \neq k}}^{N-1} X(t_i) W(t_j) wal(j, \frac{i}{N}) wal(k, \frac{i}{N}) = 0 \\ \frac{1}{N} \sum_{i=0}^{N-1} wal^2(j, \frac{i}{N}) = 1 \end{cases}, \quad \text{Hence,}$$

$$\sum_{i=0}^{N-1} wal(x_i) wal(w_i) = \sum_{i=0}^{N-1} X(t_i) W(t_i) \frac{1}{N} = \sum_{i=0}^{N-1} X(t_i) W(t_i) \Delta t_i$$

When N tends to infinity, from the limitation of the above formula and the definition of definite integral, the following integral formula is obtained.

$$\int_0^1 X(t)W(t)dt = \lim_{N\to\infty} \sum_{i=0}^{N-1} X(t_i) W(t_i) \Delta t_i$$

$$= \lim_{N\to\infty} \sum_{i=0}^{N-1} wal(x_i) wal(w_i), \ (N = 2^n)$$

B. *Learning Process*

Give K learning samples with sequence length of 2^p as follows. If the sequence length is not equal to 2^p, it can be obtained by interpolating.

$$[x_{i1}(t_j), x_{i2}(t_j), \cdots, x_{in}(t_j), d_i] \tag{21}$$

where $i = 1, 2, \cdots, K$, $j = 1, 2, \cdots, 2^p - 1$.

In Eq. (21), p is any natural number that meets the precision requirement. d_i is the expected output. Let $N = 2^p$, by Walsh transforming, such discrete series is obtained as follows

$$[wal(x_{i1}(t_j)), wal(x_{i2}(t_j)), \ldots, wal(x_{in}(t_j)), \ d_i] \tag{22}$$

where $i = 1,2,\cdots,K$ $j = 0,1,\cdots,N-1$.

For the process neural network described in Fig.2, according to the above theorem, the output of Eq. (2) can be simplified as follows

$$y_k = \sum_{j=1}^{m} v_j f\left(\sum_{i=1}^{n}\sum_{l=0}^{N-1} wal(x_{ki}(t_l))wal(w_{ij}(t_l)) - \theta_j\right) \tag{23}$$

The error function of the network can be taken as

$$\begin{aligned} E &= \sum_{k=1}^{K}(y_k - d_k)^2 \\ &= \sum_{k=1}^{K}\left(\sum_{j=1}^{m} v_j f\left(\sum_{i=1}^{n}\sum_{l=0}^{N-1} wal(x_{ki}(t_l))wal(w_{ij}(t_l)) - \theta_j\right) - d_k\right)^2 \end{aligned} \tag{24}$$

In Eqs.(23-24), the $wal(x_{ki}(t_l))$ is the Walsh transformation series of the ith component sequence of the kth learning sample. The traditional BP algorithm is adopted to train the process neural network. The learning rules of network weights may be described as follow

$$v_j = v_j + \alpha\Delta v_j,\ j = 1,2,\cdots,m \tag{25}$$

$$w_{ij}^{(l)} = w_{ij}^{(l)} + \beta\Delta w_{ij}^{(l)},\ i = 1,2,\cdots,n,\ j = 1,2,\cdots,m,\ l = 1,2,\cdots,N-1 \tag{26}$$

$$\theta_j = \theta_j + \gamma\Delta\theta_j,\ j = 1,2,\cdots,m \tag{27}$$

where, α, β, γ are learning ratios.

Let $\sum_{i=1}^{n}\sum_{l=0}^{N-1} wal(x_{ki}(t_l))wal(w_{ij}(t_l)) - \theta_j = u_{kj}$, and then there are

$$\Delta v_j = -\frac{\partial E}{\partial v_j} = -2\sum_{k=1}^{K}\left(\sum_{j=1}^{m} v_j f(u_{kj}) - d_k\right) f'(u_{kj}) \tag{28}$$

$$\Delta w_{ij}^{(l)} = -\frac{\partial E}{\partial w_{ij}^{(l)}} = -2\sum_{k=1}^{K}\left(\sum_{j=1}^{m} v_j f(u_{kj}) - d_k\right) f'(u_{kj}) wal(x_{ki}(t_l))) \tag{29}$$

$$\Delta\theta_j = -\frac{\partial E}{\partial \theta_j} = -2\sum_{k=1}^{K}\left(\sum_{j=1}^{m} v_j f(u_{kj}) - d_k\right) f'(u_{kj})(-1) \tag{30}$$

If the activation function is chosen as sigmoid function, then

$$f'(u) = f(u)(1 - f(u)) \tag{31}$$

C. Algorithm Description

The learning algorithm is described as follows.

Step1 The process data of inputs are carried out discrete Walsh transformation;

Step2 Give the maximal iteration times *M*; set the current iteration times *s*=0;

Step3 Initialize the network's connection weights and activation thresholds $v_j, w_{ij}^{(l)}, \theta_j$;

Step4 Calculate the output and the error according to Eqs.(23)-(24);

Step5 Modify the connection weights and the thresholds according to Eqs.(25)-(30); s+1→s; If $s < M$ go to **Step 4**;

Step6 Output the learning result and stop.

There are two methods to determine the input information of process neural networks: one is the theoretical model based on practical application systems or the analytic function based on statistical law. In this situation, the discrete values that meet the precision requirement can be acquired by analytic functions; the other is to obtain discrete process data based on experiment sampling that can reflect the relationship between inputs and outputs. After the Walsh transformation, using the above algorithm can avoid the complex integral operation, and can improve the efficiency of network training.

V. THE PREDICTION OF MACKEY-GLASS TIME SERIES

In this section, the actual application of the proposed method in this paper is illuminated and its effectiveness is verified with the prediction of disordered Mackey-Glass time series [8], which is widely researched in literatures. Mackey-Glass time series is generated by the following formula

$$x(t+1) - x(t) = a\frac{x(t-\tau)}{1 + x^{10}(t-\tau)} - bx(t) \tag{32}$$

where t and τ are integers, and the other parameters are given as $a = 0.2$, $b = 0.1$, $\tau = 17$.

According to Eq. (32), we can obtain the discrete series $\{x(i)\}_{i=51}^{1074}$ with 1024 items of data. By dividing the continuous 24 items of data into 3 groups, 8 items of data in each group, which are used as three discrete input series of PNN, and taking the 25th data as the corresponding expected output of the network, we can obtain 1000 groups of samples. The first 800 groups are used as the training sample set, and the next 200 groups as test sample set. According to "trial method", a PNN with the topology of "3-10-1"is constructed. The learning rate of the network is set to 0.003, and the maximum iteration steps to 10000. The training results are as follows: the maximum error is 0.06434, the minimum error is 0.00001, the mean square error is 0.02134, and the average relative error is 0.67652%. By applying the well-trained network to the test sample set, the prediction results are as follows: the maximum error is 0.06774, the minimum error is 0.00003, the mean square error is 0.02546, and the average relative error is

1.43848%. The comparison between the prediction values and the real values is shown in Fig.3.

For the prediction of Mackey-Glass time series, in [9], by applying process neural network with time-varying threshold functions to the sample set with 106 discrete series data, the average relative error of self-recognition results reaches to 1.52%. However, in our model, the average relative error is only 0.67652% for the training sample set with 800 discrete series data, and only 1.43848% for the testing sample set with 200 discrete series data. The simulation result shows that the proposed perdition method of time series based on discrete PNN in the paper has both the better generalization capability and the prediction ability.

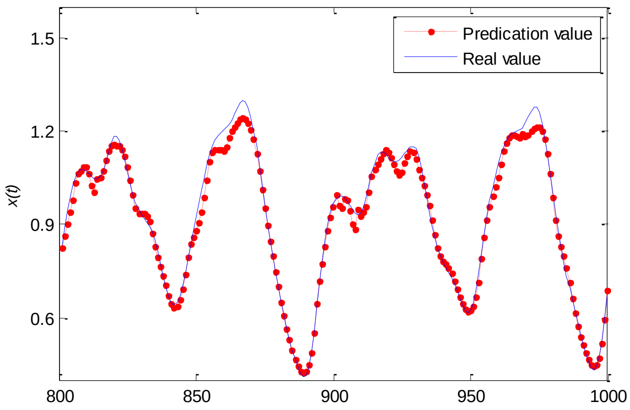

Figure 3. The comparison between prediction result and actual result.

VI. The Prediction of Sunspots Time Series

The predication of sunspot number is a typical predication problem of time series. Sunspots are the dark spots that often appear in the solar photosphere, which is the basic symbol of solar activities. As it is known to all that the sunspot activity has great influence on the human environment, people have already paid attention to it for a long time. Sunspot number is an important index of sunspots, which reflects the intensity changes of solar activities and has great influence on the earth. Many phenomena are related to the number of sunspots, such as geomagnetic variation, atmospheric motion, climate anomaly, variability of the ocean, etc. [9,10]. Therefore, it is significant to predicate the number of sunspots accurately in future. Since the sunspot activity has certain regularities, physicists have already built some prediction models and obtained some achievements in the past years. In fact, human beings have already accumulated much observation data about sunspot activity during the long process of the sun observation. Obviously, they are the typical time-varying process records. Because it is highly complex that the number of sunspots changes with time, there exists great difficulties when we use modeling to describe the variation of sunspot number [11,12].

The data of sunspots include monthly data and daily data. If the months in a year are considered as continuous time-domain intervals, and each quarterly datum is considered as a discrete sampling point, thus, we can conveniently build a prediction model of sunspot activity with process neural network method with discrete inputs. First, the time is divided into several sections, e.g., one year is considered as an interval, and then the observation results in the time interval are considered as a time-varying function. The observation results in the previous time interval are considered as the network inputs, and the observation results in the next time interval are considered as the network outputs. Accordingly, we can construct the training samples and finish the training of network, so as to build a prediction model.

A. Predication Scheme of the yearly average of sunspots

The experiment adopts the quarterly and yearly data of continuous 259 years (1749-2007) of sunspots [13]. The quarterly data is the average number of sunspots in a quarter; the yearly data is the average number of sunspots in four quarters. The predication scheme in this paper is described as follows: Use the quarterly data of the first n years to predicate the yearly data of the $n+1$ th year. For example, we use the quarterly data of years during 1850-1852 to predicate the average number of sunspots in 1853, where $n=3$.

B. Sample construction

The original data of sunspots series are shown in Table 1.

TABLE I. The Original Data of Sunspots Series

Years	Quarterly data				Yearly data
	1st	*2nd*	*3rd*	*4th*	
1749	63.53	74.73	79.00	106.43	80.9
1750	64.83	92.77	93.20	68.13	83.4
1751	52.93	55.93	49.87	31.90	47.7
…	…	…	…	…	…
2007	10.67	9.07	6.03	4.23	7.50

According to the predication scheme, we can use the original data of sunspots to construct the sample data for network training and predication. Take $n=5$ for an example, and parts of samples are shown in Table 2.

TABLE II. Partial Data of Sunspots Series ($n=5$)

No.	Inputs(the quarterly data)					Outputs
	$x_{11\sim14}$	$x_{21\sim24}$	$x_{31\sim34}$	$x_{41\sim44}$	$x_{51\sim54}$	y
1	1749th	1750th	1751th	1752th	1753th	1754th
2	1750th	1751th	1752th	1753th	1754th	1755th
3	1751th	1752th	1753th	1754th	1755th	1756th
4	1752th	1753th	1754th	1755th	1756th	1757th
5	1753th	1754th	1755th	1756th	1757th	1758th
6	1754th	1755th	1756th	1757th	1758th	1759th
…	…	…	…	…	…	…

No.	Inputs(the quarterly data)					Outputs
	$x_{11\sim14}$	$x_{21\sim24}$	$x_{31\sim34}$	$x_{41\sim44}$	$x_{51\sim54}$	y
254	2002th	2003th	2004th	2005th	2006th	2007th

C. Network Structure

According to the predication scheme, the inputs of the process neural network are n discrete value of sunspots quarterly series; the output is a single value; so there are n input nodes for each sub-network; the number of discrete sampling points is L=4, which is generally determined by experience or experiment. In this paper the number of hidden-layer nodes is $2n$ and the number of output-layer nodes is 1.

D. Network Training

In all samples, the data of previous 180 years (1749-1928) are chosen to finish the network training, which makes the network approximate the complex mapping relationship of sunspots series between different intervals. The data of next 59 years (1929-2007) are chosen to test the generalization ability of network. Obviously, the value of input node n of each network will have effect on the performance of network. Denote the maximum error, the minimum error, mean square error and average relative error of network as $E_{\max}$, $E_{\min}$, E_{avg}, E_{rel}; the step number of training is 10^4. When $n = 3, 5, 7, 9$, the training results of network are shown in Table3.

TABLE III. THE TRAINING RESULTS OF SUNSPOTS SERIES PREDICATION

n	$E_{\max}$	$E_{\min}$	E_{avg}	E_{rel}
3	3.7218	0.0016	0.6358	2.3136%
5	2.2353	0.0011	0.5984	1.5812%
7	3.0536	0.0023	0.6413	2.0513%
9	2.8198	0.0019	0.6257	1.9632%

As shown in Table 3, when $n = 5$, the performance of network is optimal. The comparison of the training results is shown in Fig.4. Therefore, the structure of the networks is determined as 5-10-1.

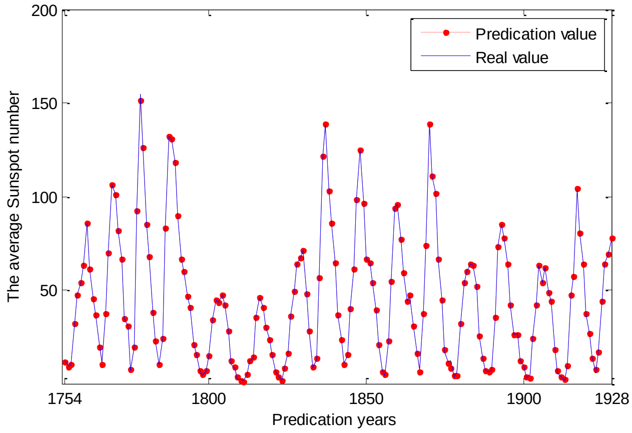

Figure 4. The comparison between the training results and the actual results during 1754-1928.

E. Predication Results

Use the sample data during 1929-2007 to test the network that has been trained, the predication results are $E_{\max}$ =2.3525; $E_{\min}$ =0.0015; E_{avg} =0.4886; E_{rel} =1.7629%. The comparison of the predication results with the actual results are shown in Fig.5.

It is clear from Fig.5 that the predication results accurately show the change trends of the actual results. Although there is a relatively larger predication error in some years in which the sunspot number is in the peak or the trough of the curves, the predication result is very satisfactory in the vast majority of other years. It can be seen that, the relative errors of this model is only 1.7629%. However, for the method in [9], the average relative error reaches to 2.30%. In PNN model, the process inputs are submitted that contains a large number of as decision making information, which enhance the approximation and predication capabilities of PNN. In addition, because the discrete series are directly submitted to PNN, this method is more practical than the one in [9] where the discrete series are transformed the continuous functions by fitting method before submitting to PNN.

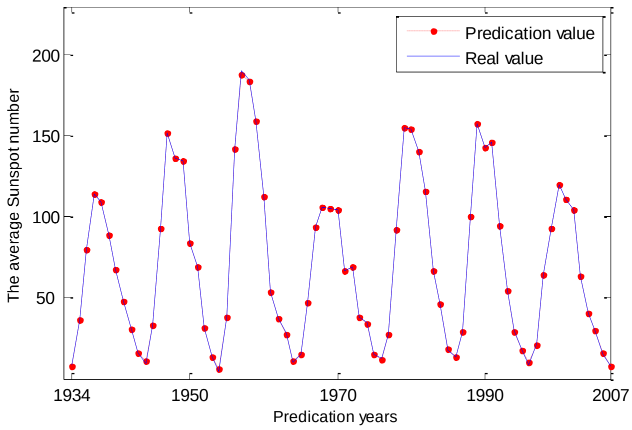

Figure 5. The comparison of the predication results with the actual results during 1934-2007.

F. *Predication Scheme of the monthly average of sunspots*

The experiment adopts the monthly and daily data of continuous 159 years (1849-2007) of sunspots [13]. The daily data is the number of sunspots in a day; the monthly data is the average number of sunspots in a month. The predication scheme is described as follows: Use the daily data of the *k*-th month in the previous *n* years to predicate the average value of the *k*-th month in the (*n*+1)-th year. For example, we use the daily data of March during 1850-1854 to predicate the monthly average value of March in 1855. Here $n=5$, $k=3$.

According to the predication scheme, the original data of sunspots can be used to construct the sample data for networks training and predication. Taking $n=5$, $k=3$ for example, the parts of samples are shown in Table4.

TABLE IV. THE PARTIAL DATA OF SUNSPOTS SERIES ($n=5$, $k=3$)

No.	**Inputs(daily data in March)**						**Outputs (monthly data)**
1	1849.3	1850.3	1851.3	1852.3	1853.3		1854.3
2	1850.3	1851.3	1852.3	1853.3	1854.3		1855.3
…	…	…	…	…	…		…
154	2002.3	2003.3	2004.3	2005.3	2006.3		2007.3

According to the predication scheme, the inputs of networks are n discrete series of daily sunspots data; the output is single value; so there are n input nodes; the number of hidden-layer nodes is set to $2n$ and the number of output-layer nodes is set to 1.

In all samples, the data in March of first 100 years (1849-1948) are chosen to train networks. The data in March of next 59 years (1949-2007) are chosen to test the generalization ability of networks. Obviously, the number of input nodes will have effect on the performance of networks. Denote the maximum error, the minimum error, mean square error and average relative error of network as $E_{\max}$, $E_{\min}$, E_{avg}, E_{rel}; the step number of training is 10^4. When $n=3,5,7,9$, the training results of networks are shown in Table5.

TABLE V. THE TRAINING RESULTS OF SUNSPOTS IN MARCH(1849-1948)

n	$E_{\max}$	$E_{\min}$	E_{avg}	E_{rel}
3	10.4670	0.0119	3.1049	0.11%
5	12.6838	0.0157	3.9552	0.22%
7	17.6565	0.0187	4.4907	0.40%
9	20.1948	0.0359	6.3106	0.99%

It is clear from the Table5 that, the performance of networks is the most optimal when n=3. Therefore, the structure of networks is determined as 3-6-1.

By applying the sample data in March during 1949-2007 to the trained networks, the predication results are below: $E_{\max}=15.9981$; $E_{\min}=0.0352$; $E_{avg}=4.6322$; $E_{rel}=0.21\%$. The comparison of the predication results with the actual results are shown in Fig.6

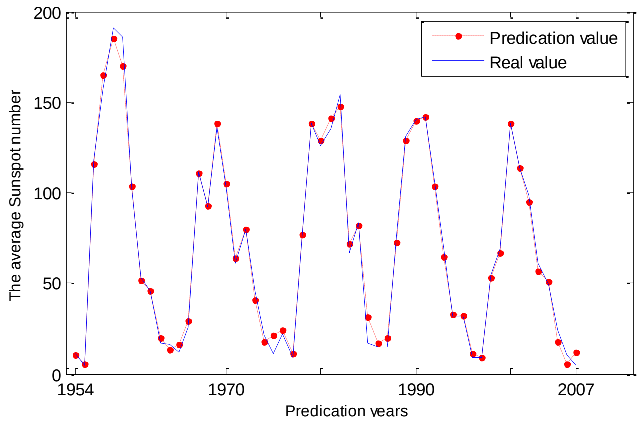

Figure 6. The comparison of the predication results with the actual results during 1954-2007.

It is clear from Fig.6 the predication results accurately show the change trends of the actual results. The experimental results show that the developed model that can use the information in first interval to predicate the information in the next interval provides a new way to complex time series predication problems.

VII. CONCLUSIONS

In this paper, a process neural network model with discrete input is presented for complex time series predication. A novel learning algorithm of PNN based on discrete Walsh transformation is proposed in order to overcome the limitation of the traditional learning algorithm in deal with the time-dependent processes. The developed model adopts procedure inputs, which shows the temporal and spatial aggregation of biological neurons and adopts discrete data series to the input of PNN, which can be applied to build models for the complex practical problems. The two complicated time series examples are examined, and the performance results in comparison with the literature show the developed model is suitable for the predication time-related problems that can use the information in first interval to predicate the information in the next interval. The analyses of the results indicate that the performances of the developed model are significantly improved if the discrete series are submitted to the networks inputs to model building. The results of study are highly encouraging and suggest that the developed model in this paper is a recommendable predication method for complex time series and it has a theoretical and practical value for the space environment prediction in future.

REFERENCES

[1] X. G. He, J. Z. Liang, Some Theoretical Issues on Procedure Neural Networks, Engineering Science, 2000, 2(12), 40-44.

[2] X. G. He, J. Z. Liang, S. H. Xu, Learning for process neural networks and its applications, Engineering Science, 2001, 3(4), 31-35.

[3] P. C. Li, Realization of system converse solution based on process neural networks and genetic algorithm, Control Theory & Applications, 2005, 22(6), 895-899.

[4] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Networks, 1991, 4(2), 251-257.

[5] Y. H. Dong, M. Shao, X. Y. Tai, An adaptive counter propagation network based on soft competition, Pattern Recognition Letters, 2008, 29(7), 938-949.

[6] M. Ghiassi, H. Saidane, D. K. Zimbra, A dynamic artificial neural network model for forecasting time series events, International Journal of Forecasting, 2005, 21(2), 341-362.

[7] Y. B. Han, A prediction method of peak amplitude and time of sunspot number, Chinese Science Bulletin, 2000, 45(4), 356-360.

[8] S. H. Xu, X. G. He, F. H. Shang, Research and application of process neural network with two hidden-layer based on expansion of basis function, Control and Decision, 2004, 19(1), 36-40.

[9] G. Ding, S. S. Zhong, Sunspot number prediction based on process neural network with time-varying threshold functions, Acta Physica Sinica, 2007, 56(2), 1224-1230.

[10] H. J. Zhao, H. F. Liang, L. S. Zhan, A method of predicting the smoothed monthly mean sunspot numbers in solar cycle 23, Chinese Astronomy and Astrophysics, 2004, 28(1), 67-73.

[11] K. W. Lau, Q. H. Wu, Local prediction of non-linear time series using support vector regression, Pattern Recognition, 2008, 41(5), 1556-1564.

[12] L. S. Zhan, H. J. Zhao, H. F. Liang, Prediction of sunspot numbers in the declining phase of solar cycle 23, New Astronomy, 2003, 8(5), 449-456.

[13] The Solar Influences Data Analysis Center of the Royal Observatory of Belgium, http://sidc.oma.be/sunspot-data/, 2009.