IEICE TRANS. FUNDAMENTALS, VOL. E81–A, NO. 3 MARCH 1998

PAPER Special Section of Selected Papers from the 10th Karuizawa Workshop on Circuits and Systems

A Cascade Form Predictor of Neural and FIR Filters and Its Minimum Size Estimation Based on Nonlinearity Analysis of Time Series

Ashraf A.M.KHALAF† *and* Kenji NAKAYAMA††, *Members*

SUMMARY Time series prediction is very important technology in a wide variety of fields. The actual time series contains both linear and nonlinear properties. The amplitude of the time series to be predicted is usually continuous value. For these reasons, we combine nonlinear and linear predictors in a cascade form. The nonlinear prediction problem is reduced to a pattern classification. A set of the past samples $x(n-1),\ldots,x(n-N)$ is transformed into the output, which is the prediction of the next coming sample $x(n)$. So, we employ a multi-layer neural network with a sigmoidal hidden layer and a single linear output neuron for the nonlinear prediction. It is called a Nonlinear Sub-Predictor (NSP). The NSP is trained by the supervised learning algorithm using the sample $x(n)$ as a target. However, it is rather difficult to generate the continuous amplitude and to predict linear property. So, we employ a linear predictor after the NSP. An FIR filter is used for this purpose, which is called a Linear Sub-Predictor (LSP). The LSP is trained by the supervised learning algorithm using also $x(n)$ as a target. In order to estimate the minimum size of the proposed predictor, we analyze the nonlinearity of the time series of interest. The prediction is equal to mapping a set of past samples to the next coming sample. The multi-layer neural network is good for this kind of pattern mapping. Still, difficult mappings may exist when several sets of very similar patterns are mapped onto very different samples. The degree of difficulty of the mapping is closely related to the nonlinearity. The necessary number of the past samples used for prediction is determined by this nonlinearity. The difficult mapping requires a large number of the past samples. Computer simulations using the sunspot data and the artificially generated discrete amplitude data have demonstrated the efficiency of the proposed predictor and the nonlinearity analysis.

key words: *cascade form predictor, time series prediction, multi-layer neural networks, FIR filters, nonlinear prediction, nonlinearity analysis, input dimension estimation*

1. Introduction

It is well known that linear filters are insufficient to deal with nonlinear time series processing (i.e. predicting, modeling, and characterizations). On the other hand, neural networks are useful for nonlinear adaptive signal processing. They have many important properties such as nonlinearity built into their structures, input-output mapping capability, and adaptivity. So, neural networks have been applied successfully in a variety of signal and information processing fields. One of these fields is the nonlinear time series prediction [1]–[8], and others. Neural networks were first applied to time series prediction by Lapedes and Farber (1987) [1].

In [1], a multi-layer neural network was used to predict the real-world data, sunspot time series, with the encouragable results. The number of the hidden neurons were estimated based on their own method [1] and the input dimension discussed in [9] was used to optimize the network size. Although, the network size was optimized, the convergence time was very long. The approach proposed in [4] can provide appreciate convergence speed at the expense of the network size. A large size, complex numbered multi-layer (two hidden layers) network with local feedback in its hidden neurons was used. The computer simulations were made only for a computer generated discrete amplitude time series. Many types of neural network structures have been introduced in [3] and it is a good reference for the time series analysis, prediction, modeling, and characterization.

In practice, many of the time series include both nonlinear and linear properties. Furthermore, the amplitude of the time series is usually continuous. Therefore, it is useful to use a combined structure of linear and nonlinear models to deal with such signals. A combined structures were proposed in [2] and [10] for different tasks. In [2], a pipelined recurrent neural network trained by a real-time recurrent learning algorithm and the taped-delay-line filter trained by LMS algorithm was proposed for speech signal prediction. Its efficiency and capability for the adaptive signal processing were demonstrated by studying the one-step prediction of speech signal [2] and adaptive differential pulse-code modulation of speech signals [11]. However, the real-time recurrent learning algorithm has a major limitation. Its computational complexity is very large specially for a large network size, which is needed for difficult task. Another hybrid structure of linear filters and neural network uses radial-basis-function was proposed in [10] for channel equalization.

In this paper, we propose a cascade form predictor, which consists of the following sub-predictors [12], [13]:

Manuscript received July 1, 1997.
Manuscript revised October 1, 1997.

†The author is with Graduate School of Natural Science and Technology, Kanazawa University, Kanazawa-shi, 920–8667 Japan.
††The author is with the Department of Electrical and Computer Engineering, Faculty of Engineering, Kanazawa University, Kanazawa-shi, 920–8667 Japan.

(1) A nonlinear sub-predictor (NSP), which consists of a multi-layer (ML) neural network with a nonlinear hidden layer and a linear output neuron.
(2) A linear sub-predictor (LSP), which is a conventional finite-impulse-response (FIR) filter.

A nonlinearity analysis method for the time series is proposed in order to estimate the minimum effective combination of the input samples and the hidden neurons. Relation between the network size and the learning performance will be discussed. Computer simulation using discrete amplitude time series and the real sunspot data will be demonstrated.

In this paper, we deal with one-step prediction. However, the proposed predictor and analysis method can be applied to more general prediction problems.

2. A Cascade Structure of Neural Network-FIR Predictor

2.1 Network Structure

Figure 1 shows the proposed predictor structure. The actual time series contains both linear and nonlinear properties and its amplitude is usually continuous value. For these reasons, we combine nonlinear and linear predictors in a cascade form.

The nonlinear prediction problem can be reduced to a pattern classification. A set of the past samples $x(n-1), \ldots, x(n-N)$ is transformed into the output, which is the prediction of the next coming sample $x(n)$. So, as a first stage of the predictor, we employ a multi-layer neural network which is good for this kind of pattern mapping. It is called a nonlinear sub-predictor (NSP) in this paper. It consists of a sigmoidal hidden layer and a single linear output neuron. The NSP is trained by the supervised learning algorithm using the sample $x(n)$ to be predicted as a target. This means the NSP itself is trained as a single predictor.

However, it is rather difficult to generate the continuous amplitude and to predict linear property. So, we employ a linear predictor after the NSP in order to compensate for the linear relation between the input samples and the target. A finite impulse response (FIR) filter is used for this purpose, which will be called a Linear Sub-Predictor (LSP). The LSP is trained by using $x(n)$ as a target. Thus, the same target is used for both the NSP and the LSP.

In order to confirm the efficiency of the proposed structure, the modified models, described in Sect. 4, are used for comparison in computer simulation.

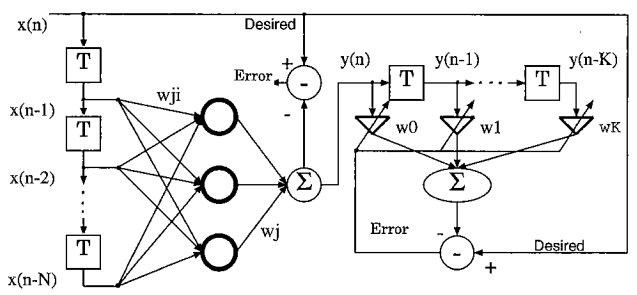


Fig. 1 Structure of the proposed predictor.

2.2 Network Operation

A set of past N samples of the input signal, $x(n-1), x(n-2), \ldots, x(n-N)$ are applied to the NSP and the current sample, $x(n)$ is used as the desired response for both the NSP and the LSP. N is the estimated input dimension.

The reason why we use $x(n)$ as a target for the NSP is explained as follows: First, it is difficult to obtain the target only for the nonlinear prediction. It may require separation of nonlinear and linear properties of the time series. Second, since the NSP has the linear output neuron, the linear prediction is also possible to some extent. Thus, the NSP output can approach the final target $x(n)$.

The LSP is an FIR filter with K taps. The weights of the sub-predictors are adjusted on a pattern-by-pattern basis. The NSP trained by the conventional Back-propagation algorithm, and the LSP is trained by the LMS algorithm.

2.3 System Equations of NSP

The output of the j th hidden neuron, $y_j(n)$ at the nth time is expressed by

$$u_j(n) = \sum_{i=1}^{N} w_{ji} x(n-i) + \theta_j(n) \tag{1}$$

$$y_j(n) = f_h(u_j(n)), \qquad j = 1, 2, \ldots, L, \tag{2}$$

where w_{ji} is the connection weight from the i th input neuron to the j th hidden neuron and $\theta_j(n)$ is its bias. The activation function, f_h used in the hidden layer is a sigmoid function of the form:

$$f_h(x) = \frac{1}{1+\exp(-x)} \tag{3}$$

The output layer contains only one linear neuron. Its output value at the nth time can be expressed by:

$$u(n) = \sum_{j} w_j y_j(n) + \theta(n), \tag{4}$$

$$y(n) = f_o(u(n)) = u(n) \tag{5}$$

w_j is the connection weight from the j th hidden neuron to the output neuron. The error of the output unit at the nth time is

$$e_{NSP}(n) = d(n) - y(n) \tag{6}$$

where $d(n)$ is the desired response at the nth time. In Fig. 1, $x(n)$ is employed as $d(n)$. The instantaneous squared error of the network is

$$\xi(n) = \frac{1}{2} e_{NSP}^2(n) \tag{7}$$

The cost function which has been used as the performance measure is the sum of the squared error over an epoch. It can be written as follows:

$$E_{NSP} = \sum_{n=1}^{M} \xi(n), \tag{8}$$

where M is the total number of samples in one epoch.

The process of adjusting the weights will be summarized in Appendix A.

3. Nonlinearity Analysis of Time Series

In order to estimate the minimum size of the proposed predictor, we analyze nonlinearity of the time series of interest. The prediction is equal to mapping a set of the past samples to the next sample to be predicted. The multi-layer neural network is good for this kind of pattern mapping. Still, difficult mapping can exist, which includes the following: Several sets of very similar patterns are mapped into very different samples. The degree of the difficulty of the mapping is closely related to the nonlinearity. The necessary number of the past samples used for prediction, that is the number of the inputs of the NSP, is determined by this nonlinearity. The difficult mapping requires a large number of the past samples. Furthermore, the number of taps of the LSP is determined by the linearity remained at the NSP output.

In this section, we introduce a measure to obtain the effective minimum combination of the input samples and the hidden neurons which enables the network to achieve its convergence faster than the other networks.

3.1 Linear Prediction

In order to make nonlinearity of time series clear, linear prediction is first briefly explained. Let $x(n)$ be a sample of the time series at the nth sampling point. If the following relation can be held for any sampling time n, then $x(n)$ can be predicted using the past N samples. This is the linear prediction.

$$x(n) = \sum_{i=1}^{N} a_i x(n-i) + e_p(n), \tag{9}$$

$$| e_p(n) | \ll \| X_{n-1} \|, \tag{10}$$

$$X_{n-1} = [x(n-1), x(n-2), \ldots, x(n-N)]^t \tag{11}$$

where, a_i are constants and $\| \cdot \|$ is Euclidean norm.

In the nonlinear time series, Eq. (9), is not satisfied using a finite N. In other words, in Eq. (9), the similar past samples predict the similar output samples. Of course, the different past samples may predict the similar output samples. It depends on the combination of the samples. However, the similar past samples never predict the different output samples by Eq. (9). In the nonlinear time series, the latter case can occur. Although, by increasing the number of the past samples N, Eq. (9) can be held to some extent, there is some limitation of network realization.

Thus, in the next section, we propose an analysis method for the nonlinearity of the time series based on the variance of $x(n)$, which corresponds to the similar sets of past samples $\{x(n-1), x(n-2), \ldots, x(n-N)\}$.

3.2 Input-Output Mapping

Impossible Mapping: A set of the N samples X_n is mapped onto the next coming sample $x(n)$ as

$$X_n \Rightarrow x(n), \qquad n = 1, 2, \ldots, M \tag{12}$$

where M is the total number of mappings in one epoch, and

$$X_n = [x(n-1), x(n-2), \ldots, x(n-N)]^t \tag{13}$$

X_n is conceptually called "a set of the past samples." Mathematically, it is represented by "a vector."

We consider two different mappings as

$$X_i \Rightarrow x(i) \tag{14}$$

$$X_j \Rightarrow x(j) \tag{15}$$

If the above two different mappings satisfy the following relation:

$$X_i = X_j \qquad x(i) \neq x(j). \tag{16}$$

then, they can not be realized by the multi-layer neural network at the same time. If such mappings are exist, the network will fail to converge at all. This problem can be overcome by increasing the number of the input samples N.

Difficult Mapping: In this case, the two patterns are similar to each other to some extent, and their targets are different from each other. It can be expressed as:

$$X_i \approx X_j, \qquad x(i) \neq x(j) \tag{17}$$

Although this mapping is basically possible, it is still a difficult mapping. Convergence may be possible, however, it may often take a very long training time. The key question is how to evaluate the degree of this difficulty. We introduce a nonlinearity analysis method for this purpose.

In order to measure the similarity among the sets of the past samples, we employ the Euclidean distance among them as:

$$d_{ij} = \|X_i - X_j\|, \qquad i \neq j \tag{18}$$

Similar sets are selected based on d_{ij} using some threshold I. If the Euclidean distance between X_i and X_j satisfies

$$d_{ij} \leq I, \tag{19}$$

then they are selected as a similar pair. Threshold value, I is determined by

$$I = \alpha A_x \tag{20}$$

$$A_x = \frac{1}{M} \sum_{n=1}^{M} | x(n) | \tag{21}$$

A process of selecting sets of X_i is as follows: Let the number of X_i sets to be M, that is,$\{X_1, X_2, \ldots, X_M\}$. One of these sets, X_k is selected and find the other X_i, $i \neq k$ which satisfies

$$d_{ki} \leq I \tag{22}$$

X_i is selected as the similar member of X_k. A set of these members is denoted by Ω_k. Thus,

$$X_i \in \Omega_k, \qquad d_{ki} \leq I \tag{23}$$

$$X_i \notin \Omega_k, \qquad d_{ki} > I \tag{24}$$

$$1 \leq i \leq M \text{ and } i \neq k$$

Ω_k is obtained for all data $X_1 \sim X_M$.

Next, the difference between $x(i)$ and $x(j)$, that is, $\|x(i) - x(j)\|$, is investigated, where both X_i and X_j are included in the same Ω_k. Let $x_k(i)$ be the corresponding output for the input sample set $X_i \in \Omega_k$. The variance of $x_k(i)$ is used to estimate the difference among $x_k(i)$.

$$\mu_k = \frac{1}{Q_k} \sum_i x_k(i), \qquad X_i \in \Omega_k \tag{25}$$

$$\sigma_k^2 = \frac{1}{Q_k} \sum_i (x_k(i) - \mu_k)^2, \qquad X_i \in \Omega_k \tag{26}$$

where Q_k is the number of elements of Ω_k. Furthermore, an average of σ_k^2 over all Ω_k is used to estimate the difficulty of mapping, that is, the degree of nonlinearity of the entire time series.

$$\overline{\sigma^2} = \frac{1}{M} \sum_{k=1}^{M} \sigma_k^2 \tag{27}$$

3.3 Estimation of Input Dimension of NSP

A large $\overline{\sigma^2}$ means the similar X_i is mapped onto the different $x(i)$, the mapping of this time series is difficult, in other words nonlinearity is high. On the other hand, if $\overline{\sigma^2}$ is small, the similar X_i are mapped onto the similar $x(i)$, then the mapping is easy, and the nonlinearity is low.

Although $\overline{\sigma^2}$ is large for some number of the past samples N, used in prediction, $\overline{\sigma^2}$ can be decreased by increasing N. Thus, the necessary number of the past samples, that is the input samples of the NSP is determined by $\overline{\sigma^2}$. The threshold I should be appropriately determined.

There is another nonlinearity. X_i and X_j, whose distance $\|X_i - X_j\|$ is large, are mapped onto the similar samples $x(i)$ and $x(j)$, that is $\|x(i) - x(j)\|$ is small. This problem belongs to pattern classification, which is an easy problem for the multi-layer neural networks.

4. Comparison with Modified Models

In Sect. 2, we have proposed the cascade form predictor structure. Some questions may arise about the order of the combination of the linear and nonlinear processings. Therefore, some modifications are considered here.

In Fig. 2, the LSP is divided into two parts, and the NSP is sandwiched between them. The same number of free parameters as in Fig. 1 are used. It will be called a sandwich model. We also consider another model in which the LSP and NSP are arranged in the reverse order compared with the proposed predictor in Fig. 1. We call this model as a reverse order model. The necessity of using this structure is to answer the question of which is better to use LSP or NSP as the first stage.

In the proposed model, we do not use the LSP in front of the NSP, because the LSP does not work well for the nonlinear time series. This point will be investigated through computer simulation.

Figure 3 shows a structure of a multi-layer neural network with direct linear connections from the input layer to the output [3, p.28]. Nonlinear hidden neurons and a linear output neuron are used. It has been stated that: "... this architecture can extract the linearly predictable part early in the learning process and free up the nonlinear resources to be employed where they are really needed" [3].

We have chosen this architecture for comparison. Because, this network also try to predict both nonlinear and linear properties using the different structure, by mixing the linear and nonlinear processings in the same network. The network size is chosen to have a very close number of free parameters as that of Figs. 1 and 2.

5. Computer Simulation

5.1 Nonlinear Time Series

Computer simulations have been done for a one-step ahead prediction task for both computer generated and real world time series.

- Artificially Generated Time Series:

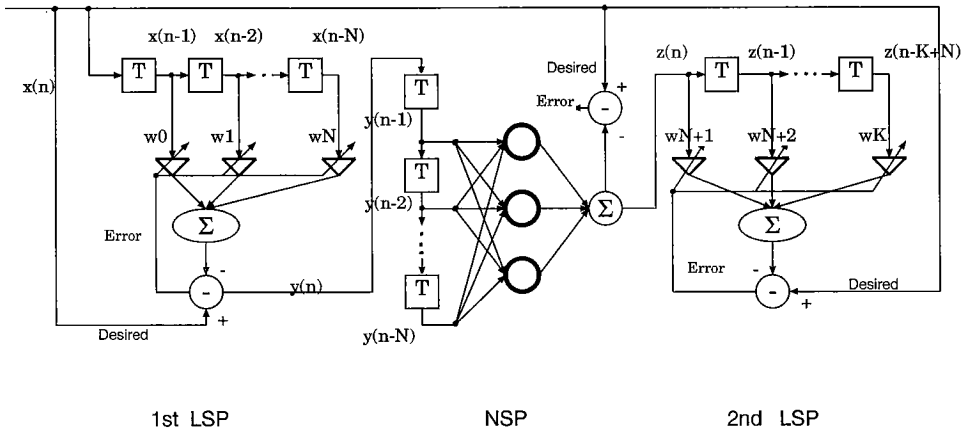


Fig. 2 Sandwich structure: The same size as Fig. 1, but the LSP is split into two parts and NSP is sandwiched between them.

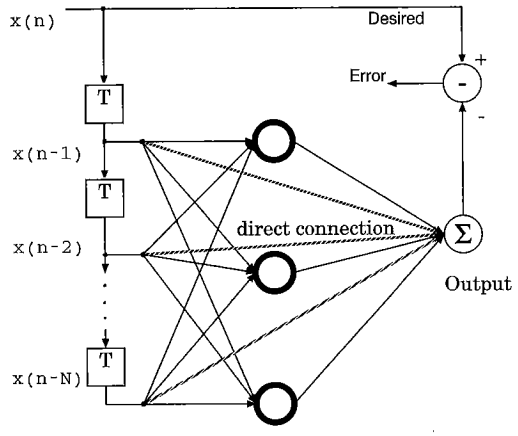


Fig. 3 Multi-layer neural network with direct linear connections between each input unit and the output (ML-WDC) [3].

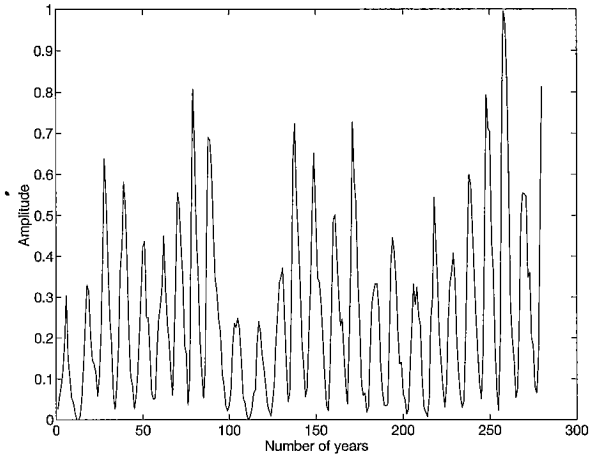


Fig. 4 Sunspot time series from 1700 to 1979.

Discrete amplitude signals used in [4], are generated by the following formula:

$$x(n) = \left(\sum_{k=1}^{P} x(n-k) \right) \bmod J. \tag{28}$$

Two examples of the time series using $J = 3$, $P = 5$, and $J = 7$, $P = 3$ respectively are generated. The samples inside the { } represent one period (one epoch) of the time series.

Example 1: $J = 3$, $P = 5$:
$x(n) = \ldots, \{2, 0, 2, 1, 1, 0, 1, 2\}, 2, 0, 2, \ldots$
Example 2: $J = 7$, $P = 3$:
$x(n) = \ldots, \{3, 0, 4, 0, 4, 1, 5, 3, 2, 3, 1, 6, 3, 3, 5, 4, 5,$
$0, 2, 0, 2, 4, 6, 5, 1, 5, 4, 3, 5, 5, 6, 2, 6, 0, 1, 0, 1, 2, 3, 6,$
$4, 6, 2, 5, 6, 6, 3, 1\}, 3, 0, 4, 0, 4, \ldots$

The length of the time series in Examples 1 and 2 are 8 and 48, respectively. Therefore, the nonlinearity of Example 2 is higher than that of Example 1.

• Sunspot Data:
The yearly sunspot time series, shown in Fig. 4, is used as a benchmark for many years by many researchers. We have used the record of sunspot data from 1700 to 1920 for learning process and the data from 1921 to 1979 for testing process. The same data was used in [1] and [9]. In Fig. 4, on the horizontal axis, 1 corresponds to the year 1700 and 280 corresponds to 1979.

5.2 Nonlinearity Analysis

Nonlinearity of the time series are analyzed based on the average variance $\overline{\sigma^2}$ using $I = A_x$, $0.8\,A_x$ and $0.5\,A_x$. I, A_x and $\overline{\sigma^2}$ are defined by Eqs. (20), (21), and (27), respectively. The values of I are determined by experience. At the present, we do not have a general rule how to determine I. However, another important point is the universality of the value of I. That is, is it possible to use the same threshold for any nonlinear time series? We want to confirm this point in this paper.

Tables 1 through 3 show the relations among the

Table 1 Average variance for Example 1.

No. of Input Samples N		2	3
$I = 0.5\,A_x$	$\overline{\sigma^2}$	0	0
$I = 0.8\,A_x$	$\overline{\sigma^2}$	0	0
$I = A_x$	$\overline{\sigma^2}$	0.5474	0

Table 2 Average variance for Example 2.

Input Samples N		3	4	5	6
$I = 0.5\,A_x$	$\overline{\sigma^2}$	0.5127	0	0	0
$I = 0.8\,A_x$	$\overline{\sigma^2}$	1.8187	1.1399	0.0312	0
$I = A_x$	$\overline{\sigma^2}$	2.9858	1.1399	1.1615	0.3958

Table 3 Average variance for Sunspot example.

Input Samples N		8	9	10	12
$I = 0.5\,A_x$	$\overline{\sigma^2}$	0.00002	0	0	0
$I = 0.8\,A_x$	$\overline{\sigma^2}$	0.0014	0.00003	0.00001	0
$I = A_x$	$\overline{\sigma^2}$	0.0023	0.000484	0.000064	0

average variance $\overline{\sigma^2}$, the threshold I and the number of the past samples N, that is the input samples of the NSP. By increasing the number of the input samples, $\overline{\sigma^2}$ can be decreased. $\overline{\sigma^2} = 0$ means that all Ω_k are empty or $\{x_{ki} \mid X_i \in \Omega_k\}$ take the same value.

5.3 Network Size Estimation

Network size will be estimated based on the nonlinearity analysis shown in Tables 1 through 3. For this purpose, we must know relations among a pair of I and $\overline{\sigma^2}$, the convergence speed and the prediction error. However, these relations are complicated. So, we first analyze their relations, and then estimate the appropriate threshold I and the variance $\overline{\sigma^2}$ for both the convergence speed and the prediction error.

In Tables 2 and 3, if we select $I = 0.5\,A_x$, then the number of the input samples $N = 4$ and 9 are enough to make $\overline{\sigma^2}$ zero. However, performance of the NSP is not good. So, we use $I = A_x$ for Example 1 and $I = 0.8\,A_x$ for both Examples 2 and 3. Thus, $N = 3$, 6 and 12 for Examples 1, 2, and 3, respectively. In simulation, they are slightly changed in order to confirm the efficiency of the selected number N.

The next stage is how to determine the number of the hidden neurons. It will be determined by complexity of pattern classification discussed in [14], [15], that is the another aspect of nonlinearity described in Sect. 3.2. The conventional methods can be applied to this problem. We also want to compare with the other methods [1], [3], [4]. The number of the hidden neurons is determined from this point. In Examples 1, 2 and 3, (Input-Hidden-Output)=3-2-1, 6-7-1 and 12-8-1, and their slight modification are employed.

Furthermore, we must estimate the order of the LSP. For linear prediction, the conventional methods can be also applied. However, if we separate a training and an actual prediction phases, a most important point is generalization. Even though the error in the training phase can be well decreased, if the prediction error for the testing data is drastically increased, this means the predictor over fits only to the training data. Thus, the order of the LSP should be determined taking the generalization into account. This point is also investigated through computer simulation.

5.4 Learning Curves and Generalization

A. Artificially Generated Time Series:

In Figs. 5, 6, and 7, the vertical axis represents the sum of squared error over one epoch of the time series and the horizontal axis represents the time (iterations) in epochs. Figure 5 shows the learning curves of Example 1, using the proposed predictor of the NSP with 2-3-1 and 3-2-1 in Figs. 5 (a) and (b), respectively. LSP taps are 6 and 7 in both figures (a) and (b).

In this example, the minimum number of the input samples is 3 to make $\overline{\sigma^2} = 0$ with $I = A_x$. So, the efficiency of the input samples and hidden neurons has been investigated. In this figure, the prediction errors are shown using (1) for the NSP only, (2) and (3) for the proposed predictor (NSP + LSP) with LSP having 6 and 7 taps, respectively. From these results, we can confirm the followings:

· 3 input samples can provide faster convergence than 2 input samples even the same network size is used in both cases.

· Using the LSP provides faster convergence compared with the case of using only the NSP. The NSP is equivalent to the model proposed in [1].

· The FIR filter with 7 taps is better than that with 6 taps. However, this point should be more investigated from the viewpoint of generalization, as it will be shown in the later example.

Figure 6 shows the results of Example 2. In this case, the minimum number of the input samples is 6 to make $\overline{\sigma^2} = 0$ with $I = 0.8\,A_x$. Therefore, the NSP with 6-7-1 and 5-8-1 are compared. The difference between using 5 and 6 input samples is also evident. The other properties are similar to the first example.

The predictor proposed in [4] can demonstrate comparable convergence speed for Examples 1 and 2. However, it requires a multi-layer (two hidden layers) neural network with the neurons of 3-input, 25-hidden, 25-hidden, and 3-output and a local feedback for each hidden neuron. The variables of the network are represented with complex numbers.

B. Sunspot Data:

Simulation results using the sunspot data are shown in Fig. 7. 12 input samples are minimum for $\overline{\sigma^2} = 0$ with $I = 0.8\,A_x$. 8 hidden neurons are selected based on a try-and-error method. In this figure, learning curves using only the NSP and the proposed predic-

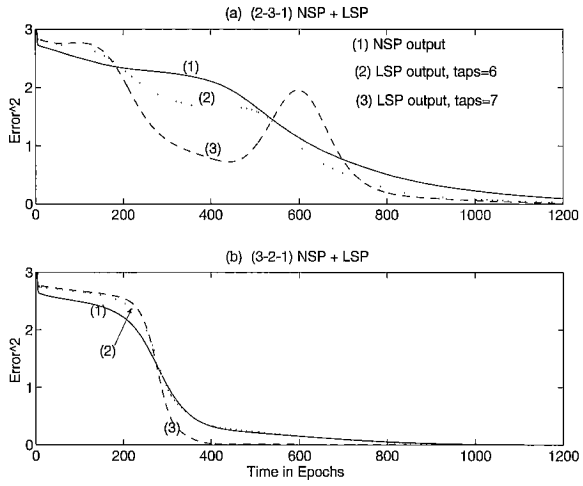


Fig. 5 Example 1: (a) Few input samples and enough hidden neurons (b) Minimum effective combination of input samples and hidden neurons.

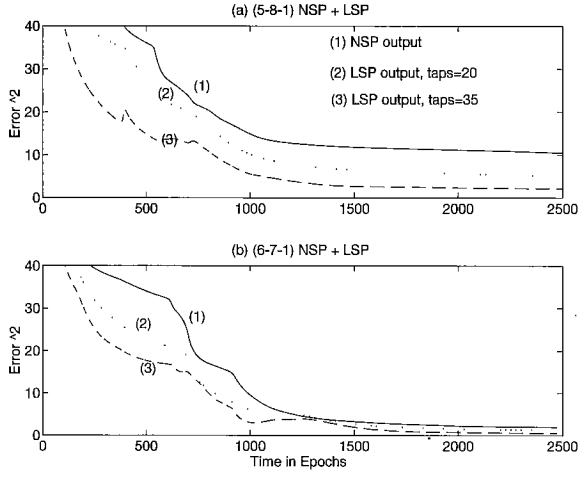


Fig. 6 Example 2: (a) Insufficient input samples, the same but not proper size as (b). (b) Minimum effective combination of input samples and hidden neurons, proper size network.

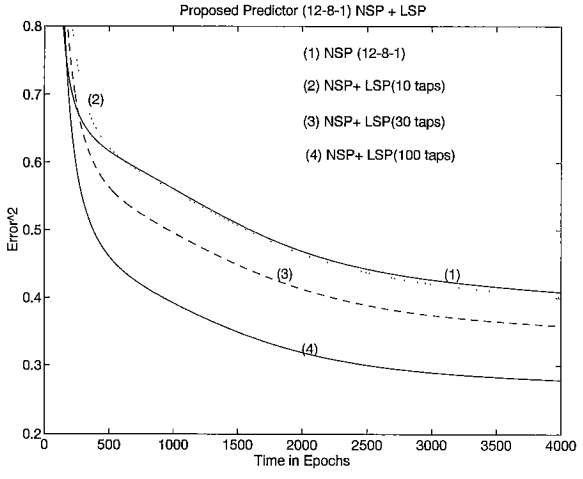


Fig. 7 Sunspot example: Learning curves of the proposed predictor. NSP consists of 12 inputs, 8 hidden neurons and one output.

Table 4 The sum of squared errors for Proposed model with different LSP taps. The last row, LSP taps=0 means the model is the NSP only. (The * points to better results at specified LSP tap).

LSP taps	Learning Phase	Testing Phase
30	0.3726	0.3129
12	0.4163	0.3005
10*	0.4227	0.2962
8	0.4281	0.3098
6	0.4281	0.3104
0	0.4087	0.4476

Table 5 The sum of squared errors for reverse order model with different LSP taps.

LSP taps	Learning Phase	Testing Phase
12	0.4404	0.5009
10	0.4396	0.5151
8	0.4643	0.4535
6*	0.4787	0.4235
5	0.4893	0.4437

Table 6 Comparison among different models.

Model Name	Learning Phase	Testing Phase
ML-WDC model (12-8-1)	0.5786	0.7152
Sandwich model LSP(5)+NSP(12-8-1) +LSP(5)	0.6514	0.4093
The reverse order model LSP(6)+NSP(12-8-1)	0.4787	0.4235
Proposed model NSP(12-8-1)+LSP(10)	0.4227	0.2962

tor (NSP+LSP) are shown. The FIR filter employs 10, 30, 100 taps for comparison.

Furthermore, the sum of squared errors for the training data at a stop point (4000 iterations in epochs) and for testing data are tabulated in Table 4. The testing data is the part of the time series from 1921 to 1979 which was not used in the learning phase. Although the LSP of large number of taps can decrease the error in the learning phase, the error for the testing data is large. This means the learning is over fitting to the training data. From the viewpoint of generalization and network size, the LSP with 10 taps is better than the others.

5.5 Comparison with Conventional and Modified Models

Table 5 shows the results of the reverse order model in the learning and testing phases. The LSP with 6 taps is better than the others. The results of different models with the specified size are listed in Table 6. The network size of the proposed and reverse order models are chosen to give the best performance in the generalization. The size of the sandwich model is taken to be almost equivalent to that of the proposed model. The ML-WDC size is slightly larger than the other models.

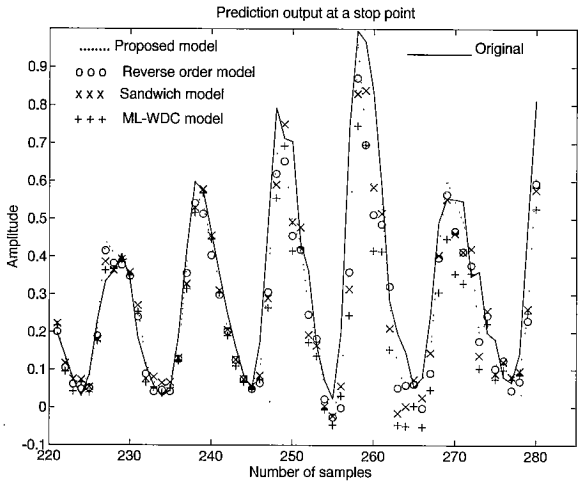


Fig. 8 Sunspot example: Prediction of sunspot data from 1921 to 1979.

From Table 6, the proposed predictor can provide the minimum prediction error in both learning and testing phases.

Figure 8 shows the output waveforms of the different models in the testing phase.

The above results are investigated more. The linear predictor can not represent the nonlinear property and it cannot predict also linear property well if nonlinearity is dominant in the time series. In the sandwich model and the reverse order model, the first LSP does not work well. The result of the ML-WDC model demonstrates that it is rather difficult to separate linearity and nonlinearity of the time series in parallel. On the other hand, the NSP can realize nonlinear mapping. Furthermore since it uses a linear output neuron, linear prediction from the hidden layer to the output is also possible to some extent as shown in curves denoted by (1) in Figs. 5, 6, and 7.

6. Conclusions

A nonlinear predictor connecting the multi-layer neural network (NSP) and the FIR filter (LSP) in a cascade form has been proposed. Both the NSP and the LSP are trained by the supervised learning algorithm using the same target. A nonlinearity analysis method for the time series has been also proposed in order to achieve the fast convergence and the small prediction error with the minimum network size. Based on the proposed nonlinearity analysis, the minimum number of the past samples, used in the prediction, can be estimated. The relation among the degree of nonlinearity, the network size, the convergence speed and the prediction error has been investigated also through the computer simulations using the artificial data and the sunspot data. The estimated number of the input samples was very meaningful for the above three factors. The models which uses a partial LSP in front of the NSP, and the other mixing linear and nonlinear processing in the multi-layer neural network have been compared with the proposed model. The proposed model has demonstrated its superiority over them in both the learning and testing phases. It has been also confirmed that the number of taps in the LSP is sensitive to generalization of the nonlinear prediction.

References

[1] A.S. Weigend and D.E. Rumelhart, "Generalization through minimal networks with application to forecasting," Proc. INTERFACE'91: Computing Science and Statistics, ed. Elaine Keramindas, pp.362–370, Springer-Verlag, 1992.

[2] S. Haykin and L. Li, "Nonlinear adaptive prediction of nonstationary signals," IEEE Trans. Signal Processing, vol.43, no.2, pp.526–535, Feb. 1995.

[3] A.S. Weigend and N.A. Gershenfeld, "Time series prediction: Forecasting the future and understanding the past," Proc. V. XV, Santa Fe Institute, 1994.

[4] M. Kinouchi and M. Hagiwara, "Learning temporal sequences by complex neurons with local feedback," Proc. ICNN'95, pp.3165–3169, 1995.

[5] T. Matsumoto, H. Hamagishi, and Y. Chonan, "A hierarchial bayes approach to nonlinear time series prediction with neural nets," Proc. ICNN'97, pp.2028–2033, 1997.

[6] T.J. Cholewo and J.M. Zurada, "Sequential network construction for time series prediction," Proc. ICNN'97, pp.2034–2038, 1997.

[7] A. Atia, N. Talaat, and S. Shaheen, "An efficient stock market forecasting model using neural networks," Proc. ICNN'97, pp.2112–2115, 1997.

[8] F.M. Thiesing and O. Vornberger, "Sales forecasting using neural networks," Proc. ICNN'97, pp.2125–2128, 1997.

[9] H. Tong and K.S. Lim, "Threshold autoregression, limit cycles and cyclical data," Journal Royal Statistical Society B, vol.42, pp.245–292, 1980.

[10] W.-T. Lee and J. Pearrson, "A hybrid linear/nonlinear approach to channel equalization problems," Advances in Neural Information Processing Systems 5, eds. S.J. Hanson, J.D. Cowan, and C. Lee Giles, U.S.A, 1993.

[11] S. Haykin and L. Li, "16 kb/s adaptive differential pulse code modulation of speech," Proc. Int. Workshop Applications Neural Networks Telecommun, Princeton, NJ, pp.132–138, 1993.

[12] A.A.M. Khalaf and K. Nakayama, "A hybrid neural predictor and its convergence analysis," Proc. of the 10th Karuizawa Workshop on Circuits and Systems, Japan, pp.357–362, April 1997.

[13] A.A.M. Khalaf, K. Nakayama, and K. Hara, "A neural-FIR predictor: Minimum size estimation based on nonlinearity analysis of input sequence," Proc. ICANN'97, Lausanne, Switzerland, pp.1047–1052, Oct. 1997.

[14] K. Hara and K. Nakayama, "Comparison of signal classification performance between multilayer neural networks and linear signal processing methods," Information Processing Society of Japan Trans., vol.38, no.2, pp.245–259, Feb. 1997.

[15] K. Hara and K. Nakayama, "Multy-frequency signal classification by multilayer neural networks and linear filter methods," IEICE Trans., Fundamental, vol.E80-A, no.5, pp.894–902, May 1997.

[16] S. Haykin, "Neural Networks: A Comprehensive Foundation," Macmillan, New York, 1994.

[17] J. Hertz, A. Krogh, and R.G. Palmer, "Introduction to the Theory of Neural Computation," Santa Fe Institute, 1991.
[18] D.E. Rumelhart and J.L. McCelland, et al., "Parallel Distributed Processing," The MIT Press, pp.318–362, 1986.

Appendix: A Learning Process in the Cascade Form Predictor

Forward Path: Transmission of Signal through the Predictor

Step 1: The past N samples of the input signal, $x(n-1), x(n-2), \ldots, x(n-N)$ are applied to the NSP and the current sample, $x(n)$ is used as the desired response. The network computes the outputs at the hidden nodes according to Eqs. (1), (2), and (3). The output of the output neuron will be computed using Eqs. (4) and (5).

Step 2: We apply the output of the NSP and its past K values to the LSP as the input which can be expressed in a vector form as

$$Y(n) = [y(n), y(n-1), \ldots, y(n-K)]^t \quad \text{(A·1)}$$

The coefficients of the LSP can be written in a vector form as

$$W = [w_0, w_1, \ldots, w_K]^t \quad \text{(A·2)}$$

then the LSP computes its output as

$$y_{LSP}(n) = W^t Y(n) \quad \text{(A·3)}$$

Thus, the error of the LSP is computed by

$$e_{LSP}(n) = d(n) - y_{LSP}(n) \quad \text{(A·4)}$$

Backward Path: Adjustment of the Weights

Step 3: Again we write the error function of the output unit of the NSP at the nth time and the instantaneous function of its squared error as:

$$e_{NSP}(n) = d(n) - y(n) \quad \text{(A·5)}$$

$$\xi(n) = \frac{1}{2} e^2_{NSP}(n) \quad \text{(A·6)}$$

By applying the Back-Propagation (BP) algorithm for the weights updating, the correction $\Delta w_{ji}(n)$ applied to $w_{ji}(n)$ is defined by the delta rule

$$\Delta w_{ji}(n) = -\eta \frac{\partial \xi(n)}{\partial w_{ji}(n)} \quad \text{(A·7)}$$

$$\frac{\partial \xi(n)}{\partial w_{ji}(n)} = \frac{\partial \xi(n)}{\partial e_{NSP}(n)} \frac{\partial e_{NSP}(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial u_j(n)} \frac{\partial u_j(n)}{\partial w_{ji}(n)} \quad \text{(A·8)}$$

$\Delta w_j(n)$ and $\Delta w_{ji}(n)$ can be written as

$$\Delta w_j(n) = \eta \delta(n) y(n) \quad \text{(A·9)}$$

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_j(n) \quad \text{(A·10)}$$

The local gradients $\delta(n)$ and $\delta_j(n)$ for both output unit and hidden units, can be written as follows

$$\delta(n) = e_{NSP}(n) f'_o(u(n)) = e_{NSP}(n) \quad \text{(A·11)}$$

$$\delta_j(n) = y_j(n)(1 - y_j(n)) e_{NSP}(n) w_j(n) \quad \text{(A·12)}$$

Step 4: The LSP coefficients are adjusted according to the LMS algorithm as

$$W(n+1) = W(n) + \mu Y(n) e_{LSP}(n) \quad \text{(A·13)}$$

where μ is the learning rate parameter of LSP. After updating the NSP weight matrix and the LSP coefficient vector W, then repeat the above four steps for the next $(n+1)$th time until the end of an epoch.

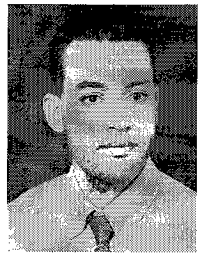
Step 5: After the end of an epoch, the cost function for both NSP and LSP are computed as the sum of the instantaneous squared errors in one epoch.

$$E_{NSP} = \sum_{n=1}^{M} \xi(n), \quad \text{(A·14)}$$

$$E_{LSP} = \sum_{n=1}^{M} \frac{1}{2} e^2_{LSP}(n) \quad \text{(A·15)}$$

where M is the total number of samples in one epoch.

Step 6: Repeat the above five steps until the cost function of either NSP or LSP reaches some reasonable value. Then stop the learning.



Ashraf A.M. Khalaf received the B.Sc. and M.Sc. degrees in Electrical Engineering from Minia University, Minia, Egypt, in 1989 and 1994, respectively. From 1989 to 1995 he was working as a Teaching Assistant in the Department of Electrical Engineering, Faculty of Engineering and Technology, Minia University. Since April, 1996 he has been a Ph.D. candidate with the Graduate School of Natural Science and Technology, Kanazawa University. His current research interests include neural networks.

Kenji Nakayama received the B.E. and Dr. degrees in Electronics Engineering from Tokyo Institute of Technology (TIT), Tokyo, Japan, in 1971 and 1983, respectively. From 1971 to 1972 he was engaged in the research on classical network theory in TIT. He was involved in NEC Corporation from 1972 to 1988, where his research subjects were filter design methodology and signal processing algorithms. He joined the Department of Electrical and Computer Engineering at Kanazawa University, in Aug. 1988, where he is currently a Professor. His current research interests include neural networks, adaptive signal processing, and signal theory. Dr. Nakayama is a senior member of IEEE and a member of INNS.