2009 Ninth International Conference on Intelligent Systems Design and Applications

Backpropagation-Based Non Linear PCA for Biomedical Applications

Alberto Landi, Paolo Piaggi
Dep. of Electrical Systems and Automation,
University of Pisa, Pisa, Italy
landi@dsea.unipi.it, piaggi@gmail.com

Giovanni Pioggia
Institute of Clinical Physiology,
National Council of Research, Pisa, Italy
giovanni.pioggia@ifc.cnr.it

Abstract

Machine learning methodologies such as artificial neural networks (ANN), fuzzy logic or genetic programming, as well as principal component analysis (PCA) and intelligent control have been recently introduced in medicine. ANNs imitate the structure and workings of the human brain by means of mathematical models able to adapt several parameters. ANNs learn the input/output behavior of a system through a supervised or an unsupervised learning algorithm. In this work, we present and demonstrate a new pre-processing algorithm able to improve the performance of an ANN in the processing of biomedical datasets. The algorithm was tested analyzing lung function and fractional exhaled nitric oxide differences in the breath in children with allergic bronchial asthma and in normal population. Classification obtained using non linear PCA based on the new algorithm shows a better precision in separating asthmatic and control subjects.

1. Introduction

Advances in information and communication technologies are greatly contributing to the evaluation of human clinical situations. Broadening the healthcare spectrum, the artificial intelligence systems, better described as clinical decision support systems, are today found supporting medication prescribing, in clinical laboratories and educational settings, for clinical surveillance, or in data-rich areas like the intensive care setting. Pattern classification systems are generally used to search for a discrimination among signals and a decision among a number of possible categories. Many techniques are nowadays used for this purpose in medicine, but recently the processing architectures are often performed by models inspired by biology, such as genetic algorithms, Artificial Neural Networks (ANN), Support Vector Machines (SVM) and Fuzzy Logic-based (FC) modelling. From a general point of view, a human monitoring complex system consists of different modules cooperating in order to perform data acquisition from multiple sensors, data analysis through several techniques and data redirection. Often the Principal Component Analysis (PCA) techniques is adopted to pre-process data in order to reduce the dimensionality of a data set and to identify underlying variables. In this paper we describe, assess and discuss a novel non linear algorithm inspired by a supervised learning methodologies known as backpropagation able to implement a non linear PCA.

The methodology was evaluated analyzing lung function and fractional exhaled nitric oxide differences in the breath in children with allergic bronchial asthma and in normal population. Lung function and fractional exhaled nitric oxide have been proposed as markers for the evaluation of airway inflammation in asthma [1].

2. PCA

Specifically, PCA looks for u, a linear combination of the x_i, and an associated vector a, with $u = a \cdot x$ so that

$$\left\langle \left\| x - a \cdot u \right\|^2 \right\rangle \qquad (1)$$

is minimized, where the operator < > denotes a sample mean. Here u is called the first principal component (PC) (or score) while a, called the first eigenvector (or loading), is the first eigenvector of the data covariance or correlation matrix. Together u and a make up the first PCA mode. In essence, a given data set is approximated by a straight line (oriented in the direction of a), which accounts for the maximum amount of variance in the data; pictorially, in a scatterplot of the data, the straight line found by PCA

978-0-7695-3872-3/09 $26.00 © 2009 IEEE
DOI 10.1109/ISDA.2009.176

passes through the ‘‘middle’’ of the data set. From the residual, $x - a \cdot u$, the second PCA mode (orthogonal to the previous PCA mode) can similarly be extracted and so on for the higher modes. In practice, the common algorithms for PCA extract all modes simultaneously [2], [3]. By retaining only the leading modes, PCA has been commonly used to reduce the dimensionality of the data set and to extract the main patterns from it.

3. Backpropagation-Based Non Linear PCA

3.1. Neural Networks

The most widely used ANN models are feed forward networks known as multilayer perceptrons (MLP) [4], [5], which perform nonlinear regression and classification. The basic architecture consists of a layer of input neurons x_i (a ‘‘neuron’’ is simply a variable in ANN jargon) linked to a layer or more of ”hidden” neurons, which are, in turn, linked to a layer of output neurons y_j.

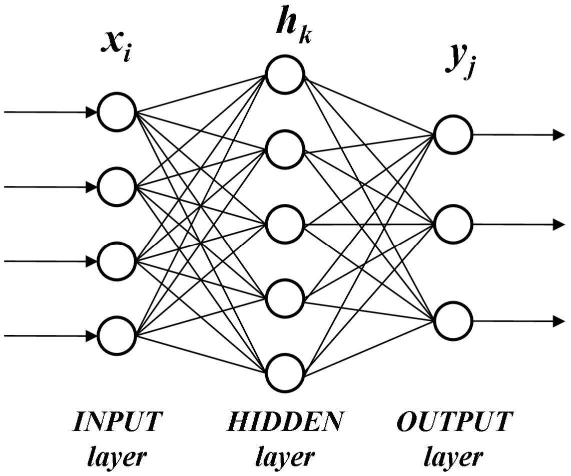

Figure 1. Schematic diagram of the multilayer perceptron neural network

In Figure 1, there is only one layer of hidden neurons h_k. A transfer function (an “activation” function in ANN jargon) maps from the inputs to the hidden neurons. There is a variety of choices for the transfer function, the sigmoid function being a common one, i.e.,

$$h_k = \left(1 + e^{-\sigma_i}\right)^{-1} \quad (2)$$

where $\sigma_i = w_{i0} + \sum_i \left(w_{ki} \cdot x_i\right)$, w_{i0} and w_{ki} are the bias and the weight parameters respectively.

The sigmoid function $e^{-\sigma_i}$ has two asymptotic values of 0 and 1 as $\sigma_i \to -\infty$ and $\sigma_i \to +\infty$ respectively and they can be viewed as representing the two states of a neuron (at rest or activated), depending on the strength of the excitation σ_i. In the same way, the output neurons y_j are calculated by applying the sigmoid function on the excitation σ_k of the preceding layer,

$$y_j = \left(1 + e^{-\sigma_k}\right)^{-1} \quad (3)$$

Given observed data y_{oj}, the optimal values for the weight parameters are found by training the ANN, i.e., performing a nonlinear optimization, where the cost function or objective function

$$E = \left\langle \sum_j \left(y_j - y_{oj}\right)^2 \right\rangle \quad (4)$$

is minimized, where E is the mean squared error (MSE) of the output. The ANN has found a set of nonlinear regression relations $y_j = f_j(x)$.

To approximate a set of continuous functions f_j, only one layer of hidden neurons is enough, provided enough hidden neurons are used in that layer [6], [7].

3.2. Backpropagation algorithm

The best-known algorithm for the neural network training is *backpropagation* [8]. In backpropagation, the gradient vector of the error surface of entire neural network is calculated. This vector points along the line of steepest descent from the current point so, as the algorithm moves along it, the error will decrease. A sequence of such “moves” (slowing near the bottom of the surface) will eventually find a minimum of some sort. The step size of the algorithm is proportional to the slope (so that it settles down in a minimum) and to a special constant: the learning rate η which is time-varying, getting smaller as the algorithm progresses. As shown in Figure 2, after an input pattern has been applied to the input layer of the network, it is propagated through each upper layer, until an output is generated. This output pattern is then compared to the desired output, and an error signal is computed for each output unit. The signals are then transmitted backward from the output layer to each unit in the intermediate layer that contributes directly to the output. However, each unit in the intermediate layer receives only a portion of the total error signal, based roughly on the relative contribution of each unit to the original output. This process repeats, layer by layer, until each unit in the network has received an error signal that describes its relative contribution to the total error.

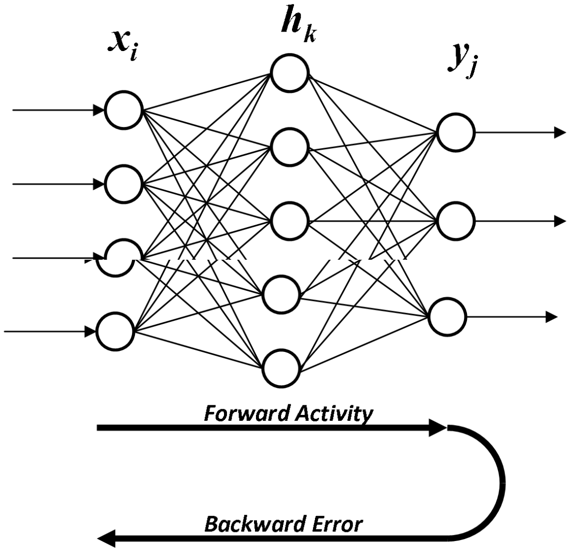

Figure 2. The two phases of backpropagation algorithm

3.3. The proposed algorithm

Consider the generic schema of five layered MLP of
Fig.3.

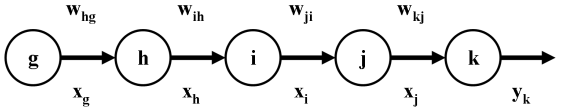

Figure 3. Example of five layered MLP

Denote by $x_{g,h,i,j}$ the output values of layer g, h, i and j respectively, y_k the output of the final layer k, w_{BA} the weights vector between consecutive layers A and B (with A and B corresponding to hg, ih, ji and kj respectively). In addition,

$$\begin{cases} \sigma_B = w_{B0} + \sum_B \left(w_{BA} \cdot x_A \right) \\ y_B = s\left(\sigma_B\right) \end{cases} \tag{5}$$

The number of neural units of the last layer k and first layer g is assumed equal. The mean squared error (MSE) of the entire network is

$$E_k = \frac{1}{2} \sum_k \left(y_k - x_g \right)^2 \tag{6}$$

The weight correction (for weights between j-k layers) is:

$$\Delta w_{kj} \overset{def}{=} \eta \frac{\delta E_k}{\delta w_{kj}} = \eta \frac{\delta E_k}{\delta \sigma_k} \frac{\delta \sigma_k}{\delta w_{kj}} = \eta \Delta_k x_j \tag{7}$$

which is the Hebbian learning rule, where η is the learning rate and

$$\begin{aligned} \Delta_k &\overset{def}{=} \frac{\delta E_k}{\delta \sigma_k} = \frac{1}{2} \frac{\delta}{\delta \sigma_k} \left(\sum_k \left(y_k - x_g \right)^2 \right) = \left(y_k - x_g \right) \frac{\delta y_k}{\delta \sigma_k} = \\ &= \left(y_k - x_g \right) \frac{\delta}{\delta \sigma_k} \left(s\left(\sigma_k\right) \right) = \left(y_k - x_g \right) s'\left(\sigma_k\right) \end{aligned} \tag{8}$$

Substituting (8) in (7), the weight correction for the output layer k is:

$$\Delta w_{kj} = \frac{\delta E_k}{\delta \sigma_k} = \eta \cdot \Delta_k \cdot x_j = \eta \cdot \left(y_k - x_g \right) \cdot s'\left(\sigma_k\right) \cdot x_j \tag{9}$$

Regarding the weight correction for layer j, we have:

$$\Delta w_{ji} = \eta \cdot \Delta_j \cdot x_i \tag{10}$$

where

$$\begin{aligned} \Delta_j &= \frac{\delta E_k}{\delta \sigma_j} = \frac{\delta E_k}{\delta x_j} \frac{\delta x_j}{\delta \sigma_j} = \frac{\delta E_k}{\delta x_j} s'\left(\sigma_j\right) = \\ &= \left(\sum_k \left(\frac{\delta E_k}{\delta \sigma_k} \frac{\delta \sigma_k}{\delta x_j} \right) \right) s'\left(\sigma_j\right) = \left(\sum_k \left(\Delta_k w_{kj} \right) \right) s'\left(\sigma_k\right) \end{aligned} \tag{11}$$

for the chain rule in case of several variables:

$$\begin{cases} f(x) = f\left(g_1(x), g_2(x), \cdots, g_n(x) \right) \\ \dfrac{\delta f}{\delta x} = \sum_{i=1}^{n} \left(\dfrac{\delta f}{\delta g_i} \dfrac{\delta g_i}{\delta x} \right) \end{cases} \tag{12}$$

In a similar way, the weight correction for layer i, we have

$$\Delta w_{ih} = \eta \cdot \Delta_i \cdot x_h \tag{13}$$

where

$$\begin{aligned}\Delta_i &= \frac{\delta E_k}{\delta \sigma_i} = \frac{\delta E_k}{\delta x_i}\frac{\delta x_i}{\delta \sigma_i} = \frac{\delta E_k}{\delta x_i} s'(\sigma_i) = \\ &= \left(\sum_k \left(\frac{\delta E_k}{\delta \sigma_k}\frac{\delta \sigma_k}{\delta x_i}\right)\right) s'(\sigma_i) = \left(\sum_k \left(\Delta_k \frac{\delta \sigma_k}{\delta x_i}\right)\right) s'(\sigma_i)\end{aligned} \tag{14}$$

and

$$\frac{\delta \sigma_k}{\delta x_i} = \frac{\delta \sigma_k}{\delta x_j}\frac{\delta x_j}{\delta x_i} = w_{kj}\frac{\delta x_j}{\delta \sigma_j}\frac{\delta \sigma_j}{\delta x_i} = w_{kj} s'(\sigma_i) w_{ji} \tag{15}$$

Replacing (15) in (14), we obtain

$$\Delta_i = \left(\sum_k \left(\Delta_k w_{kj} w_{ji} \cdot s'(\sigma_j)\right)\right) \cdot s'(\sigma_i) \tag{16}$$

The weight correction for layer *h* is

$$\Delta w_{hg} = \eta \cdot \Delta_h \cdot x_g \tag{17}$$

where

$$\begin{aligned}\Delta_h &= \frac{\delta E_k}{\delta \sigma_h} = \frac{\delta E_k}{\delta x_h}\frac{\delta x_h}{\delta \sigma_h} = \frac{\delta E_k}{\delta x_h} s'(\sigma_h) = \\ &= \left(\sum_k \left(\frac{\delta E_k}{\delta \sigma_k}\frac{\delta \sigma_k}{\delta x_h}\right)\right) s'(\sigma_h) = \left(\sum_k \left(\Delta_k \frac{\delta \sigma_k}{\delta x_h}\right)\right) s'(\sigma_h)\end{aligned} \tag{18}$$

and

$$\begin{aligned}\frac{\delta \sigma_k}{\delta x_h} &= \frac{\delta \sigma_k}{\delta x_j}\frac{\delta x_j}{\delta x_h} = w_{kj}\left(s'(\sigma_j) w_{ji}\right)\frac{\delta x_i}{\delta x_k} = \\ &= w_{kj}\left(s'(\sigma_j) w_{ji}\right)\left(\frac{\delta x_i}{\delta \sigma_i}\frac{\delta \sigma_i}{\delta x_h}\right) = w_{kj}\left(s'(\sigma_j) w_{ji}\right)\left(s'(\sigma_i) w_{ih}\right)\end{aligned} \tag{19}$$

Finally, replacing (19) in (18), we obtain

$$\Delta_h = \left(\sum_k \left(\Delta_k \cdot w_{kj} \cdot w_{ji} \cdot s'(\sigma_j) \cdot w_{ih} \cdot s'(\sigma_i)\right)\right) s'(\sigma_h) \tag{20}$$

To summarize, we get the following Δ formula:

$$\begin{aligned}\Delta_k &= s'(\sigma_k)\left(y_k - x_g\right) \\ \Delta_j &= s'(\sigma_j)\left(\sum_k \left(\Delta_k \cdot w_{kj}\right)\right) \\ \Delta_i &= s'(\sigma_i)\left(\sum_k \left(\overbrace{\Delta_k \cdot w_{kj}} \cdot w_{ji} \cdot s'(\sigma_j)\right)\right) \\ \Delta_h &= s'(\sigma_h)\left(\sum_k \left(\overbrace{\Delta_k \cdot w_{kj} \cdot w_{ji} \cdot s'(\sigma_j)} \cdot w_{ih} \cdot s'(\sigma_i)\right)\right)\end{aligned} \tag{21}$$

Note that the structure of the Hebbian learning rule at each layer contains an embedded term common to the previous layer.

Our approach represents an extension of the backpropagation algorithm tipically associated to a standard three-layer MLP. It is based on the recursive form (21) that can be applied to a generalized MLP with an arbitrary number of layers.

3.4. Nonlinear PCA

Consider the network shown in Figure 4.

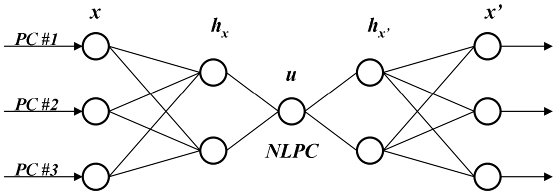

Figure 4. Schematic diagram of the ANN model for calculating the nonlinear principal component

The non linear PCA (NLPCA) was basically achieved with a standard feed forward ANN with four layers of transfer functions, mapping from the inputs to the outputs. There are two layers of hidden neurons sandwiched between the input layer *x* (which consists on the first three linear PC) on the left and the output layer *x'* on the right. Next to the input layer is the encoding hidden layer, followed by the "bottleneck" layer (with a single neuron *u*), which is then followed by the decoding hidden layer. [8]

In plain words, the NLPCA network is composed of two cascaded standard three-layer feed forward ANNs.

The first three-layer network maps from the inputs *x* through a first hidden layer to the bottleneck layer with only one neuron *u*, i.e., a nonlinear mapping *u*=*f*(*x*).

The next three-layer feed forward ANN inversely maps from the nonlinear PC (NLPC) *u* back to the original three-dimensional *x* space, with the objective that the outputs $x'=g(u)$ be as close as possible to the inputs x.

Note that $g(u)$ nonlinearly generates a curve in the x space and hence a one-dimensional (1-D) approximation of the original data.

The MSE of this approximation was minimized using the cost function $\left\langle \|x-x'\|^2 \right\rangle$ and the backpropagation algorithm described in section 3.3.

A number of runs (i.e., training of ANN) mapping from x to x' using random initial parameters (i.e. weight and bias parameters) were performed: the run attaining the lowest value of MSE was then selected as the final solution. In order to avoid overfitting, 20% of the data were randomly selected as validation data and withheld from the training set of the ANN: runs where MSE was larger for the validation data set than for training data set were rejected to avoid overfitted solutions.

The number of the hidden neurons was determined through a trial-and-error process and following a general principle of parsimony, because no commonly accepted theory exists for predetermining the optimal number of neurons in the hidden layers: in particular, several runs with increasing number of neurons were made. As a result of this step, MSE did not decrease appreciably by increasing the number of neurons in hidden layers over two.

Squeezing the input information through a bottleneck layer with only one neuron accomplished the dimensional reduction. In effect, the linear relation $u = a \cdot x$ in PCA is now generalized to $u=f(x)$, where f can be any nonlinear continuous function representable by a feed forward ANN mapping from the input layer to the bottleneck layer; and instead of $\left\langle \|x-a\cdot u\|^2 \right\rangle$, $\left\langle \|x-g(u)\|^2 \right\rangle$ was minimized.

The residual, $x - g(u)$, represents the input of the same network, to extract the second NLPCA mode, and so on for the higher modes.

The classical PCA is indeed a linear version of this NLPCA: it can be easily verified by replacing all the sigmoidal transfer functions with the identity function, thereby removing the nonlinear modeling capability of the NLPCA. Then the forward map to u involves only a linear combination of the original variables as in the linear PCA.

4. Pattern recognition model and experimental results

A machine learning task was applied in order to define and validate the algorithm in classifying biomedical data. The task was realized by means of a Kohonen Self-Organizing Map (KSOM) [3]. A KSOM maps the original space into a two-dimensional net of neurons, in such a way that close neurons respond to similar signals, in order to solve classification tasks and to find structures in a dataset. KSOMs are unsupervised neural networks, i.e. they exploit similarities of samples apart from the class which they belong to. In the unsupervised training process, the synaptic weight vectors of the artificial neurons of the KSOM are adapted by means of the training data set examples, in such a way that the KSOM supplies as good a representation as possible of the training data set.

The lung function variables (FEV1, FVC, FEF25-75%, FEV1/FVC and TTV) and the fractional exhaled nitric oxide (FeNO) of asthmatic and control subjects were monitored in 89 assessments. The dataset consists in a 89x6 matrix.

Data were prefiltered by PCA with only the three leading modes retained. PCA modes 1, 2, and 3 accounted for 37.9%, 29.4%, and 16%, respectively, of the variance of the original data. The first three PCs (PC1, PC2, and PC3) were used as the input x for the NLPCA network.

In terms of variance explained, the first NLPCA mode explained 52.4% of the variance versus 37.9% explained by the first PCA mode. The residual between data and the first non linear mode were put as input into the same network to extract the second NLPCA mode. A scatterplot of the asthmatic and control subjects in the principal component space is shown in Figure 5: the first NLPCA mode is plotted by red circles.

The KSOM was trained with data pre-processed with PCA and Backpropagation-Based Non Linear PCA in order to discriminate asthmatic and control subjects: in particular, we used the first two modes both in linear and in non linear case. In order to check the generalization capability of the neural networks, a *k*-fold cross-validation was carried out. Cross-validation is one of the several approaches for estimating the performance of a model on future as-yet-unseen data [4].

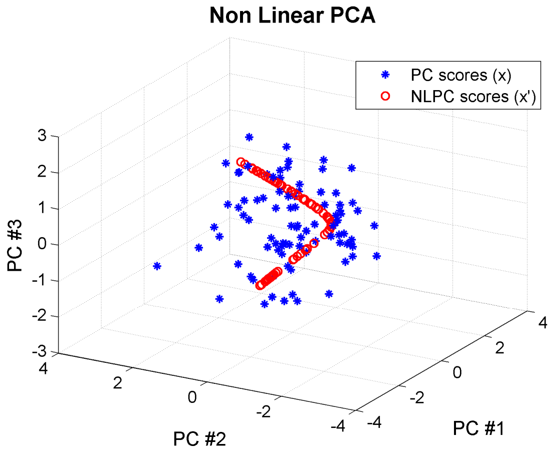

Figure 5. Scatterplot of the asthmatic and control subjects in the principal component space.

In k-fold cross-validation, the original dataset is partitioned into k subsets. For each cross-validation step, a single subset is retained as the test set, and the remaining $k - 1$ subsubsets are used as training set. The cross-validation process is then repeated k times, with each of the k subsets used exactly once as the test set. The k results from the folds then can be averaged (or otherwise combined) to produce a single estimation. In this work a 3-fold cross-validation was applied; each fold consisted of randomly selected samples, at least one for each category index was included in each fold.

As regards the two KSOMs a topological analysis showed for each test the presence of minimally overlapping zones. In order to quantify results obtained in the cross-validation, a labelling process for each KSOM and a test process allowed subjects to be classified as belonging to at least one of the two classes.

We found the KSOM trained with PCA data is able to discriminate asthmatic and control subjects classes with accuracy rates respectively of 89.2%, and 90.8%, while the KSOM trained with Backpropagation-Based Non Linear PCA of 97.1% and 94.7%.

5. Conclusions

We reported a novel pre-processing algorithm able to improve the performances of an ANN in the processing of biomedical dataset. The performances of our approach were assessed analyzing lung function and fractional exhaled nitric oxide differences in the breath of children with allergic bronchial asthma and in normal population. The comparison of results showed that our approach is able to enhance the recognition percentages of an ANN analysing biomedical data in terms of classification and misclassification, successfully demonstrating the improvement of the performances.

Acknowledgment
The authors wish to acknowledge the help of prof. Massimo Pifferi for making us available the data set used to test the non linear PCA based on the proposed new algorithm.

6. References

[1] P. Pohunek, J. O. Warner, J. Turzíková, J. Kudrmann and W. R. Roche, “Markers of eosinophilic inflammation and tissue re-modelling in children before clinically diagnosed bronchial asthma,” *Pediatr Allergy Immunol,* pp. 43-51, 2005.

[2] I. T. Jolliffe, *Principal Component Analysis*, 502 pp., Springer-Verlag, New York, 2002.

[3] R. W. Preisendorfer, *Principal Component Analysis in Meteorology and Oceanography*, Elsevier Sci., NewYork, 1988.

[4] C. M. Bishop, *Neural Networks for Pattern Recognition*, 482 pp., Clarendon, Oxford, U.K, 1995.

[5] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning internal representations by error propagation”, *Parallel Distributed Processing,* pp.318-362, Cambridge, Mass, 1986.

[6] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” *Neural Networks*, pp.359-366,1989.

[7] G. Cybenko, “Approximation by superpositions of a sigmoidal function,” *Math. Control Signals Syst.*, pp.303-314, 1989.

[8] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice Hall NY, 1989.

[9] W.W. Hsieh, “Nonlinear multivariate and time series analysis by neural network methods,” *Reviews of Geophysics*, pp. 1000-1029, 2004.