Scientific Research Publishing

International Journal of Geosciences, 2018, 9, 397-413
http://www.scirp.org/journal/ijg
ISSN Online: 2156-8367
ISSN Print: 2156-8359

# Application of Neural Networks in Probabilistic Forecasting of Earthquakes in the Southern California Region

**Vitor H. A. Dias[1], Andrés R. R. Papa[1,2]**

[1]Geophysics Department, Observatório Nacional, Rio de Janeiro, Brazil
[2]Physics Institute, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, Brazil
Email: vhadrock@gmail.com, papa@on.br

**How to cite this paper:** Dias, V.H.A. and Papa, A.R.R. (2018) Application of Neural Networks in Probabilistic Forecasting of Earthquakes in the Southern California Region. *International Journal of Geosciences*, **9**, 397-413.
https://doi.org/10.4236/ijg.2018.96025

**Received:** January 12, 2018
**Accepted:** June 25, 2018
**Published:** June 28, 2018



## Abstract

During the last few decades, many statistical physicists have devoted research efforts to the study of the problem of earthquakes. The purpose of this work is to apply methods of Statistical Physics and network systems based on "neurons" in the study of seismological events. Data from the Advanced National Seismic System (ANSS) of Southern California were used to verify the relationship between time differences between consecutive seismic events with magnitudes greater than 3.0, 3.5, 4.0 and 4.5 through the modeling of neural networks. The problem we are analyzing is time differences between seismological events and how these data can be adopted as a time series with non linear characteristic. We are therefore using the multilayer perceptron neural network system with a backpropagation learning algorithm, because its characteristics allow for the analysis of non-linear data in order to obtain statistical results regarding the probabilistic forecast of tremor occurrence.

## Keywords

Neurons, Multi-Layer Perceptron, Backpropagation, Prediction

## 1. Introduction

Earthquakes, one of nature's many different phenomena, are the cause of huge catastrophes in their places of occurrence. Such catastrophes are characterized by the physical destruction of cities (houses, buildings, urban roads, etc.) and consequently large numbers of human victims. The extent of the damage affects thousands of people and many cities around where the tremor occurred, reaching thousands of square kilometers. Interestingly, neural networks have been shown to be useful when applied in different areas such as recognition of word

patterns [1], speech recognition [2], among others.

Much research is being done today in Seismology to better understand the dynamics of earthquakes, such as: the study of volcanic activity as a precursor of tremors [3]; a better understanding of the Earth's structure and activities occurring in internal layers of the Earth, such as the relationship of geological faults with earthquakes [4]; the observation of the effects of tsunamis caused by earthquakes [5]; analysis of the malfunctions that a tremor can cause for better prevention [6]; to name a few.

Several prediction-related works have been carried out throughout history with the aim of relating earthquakes to their probability of occurrence [7]. It has been demonstrated that earthquakes can be artificially triggered by the injection of fluids, and in addition that many earthquakes in California and Nevada occur at depths accessible by drill. It was [8] found that tremors include several premonitory events such as crust movements as well as anomalous changes in phenomena affecting for example slope, fluid pressure, electric and magnetic fields, radon emission and even the number of small tremors that could result in stronger tremors, while in [9] it was verified the distribution of time intervals between successive tremors as a predictor, and in [10] it was studied the relationship of complex network systems to the position of a quake and the modeling of earthquakes related to real data [11].

Considering the relationship between earthquakes and neural networks we have some work related to the modeling of neural networks oriented to the understanding of earthquakes [12], which analyzes in a neural network probabilistic for prediction of earthquakes based on the parameters of the law of Gutenberg-Richter, [13] that analyzes the possibility of predicting earthquakes in location and time by introducing eight different seismological indicators, [14] that realizes predictions of earthquakes in the northeast of the red sea based on neural networks, [15], who performs earthquake prediction in Chile, relating data from the neural network input with Bath and Omori-Utsu's parameters associated with high seismic activity.

However, the prediction of earthquakes continues to be difficult, and much effort will certainly be devoted to solving this problem. For this paper, in order to estimate the possibility of a quake occurrence and time differences between events, the seismological data for analysis was taken from the Advanced National Seismic System (ANSS) catalog in Southern California.

The rest of the paper is organized as follows: in Section 2 we describe the study area; Section 3 describes the database used and the way in which the data were prepared; in Section 4, we present the type of network we use and in Section 5, the learning process. In Section 6, the results are described and, finally, Section 7 presents the conclusions.

## 2. Study Area

**Figure 1** demonstrates the region in which the earthquake occurred from 1932

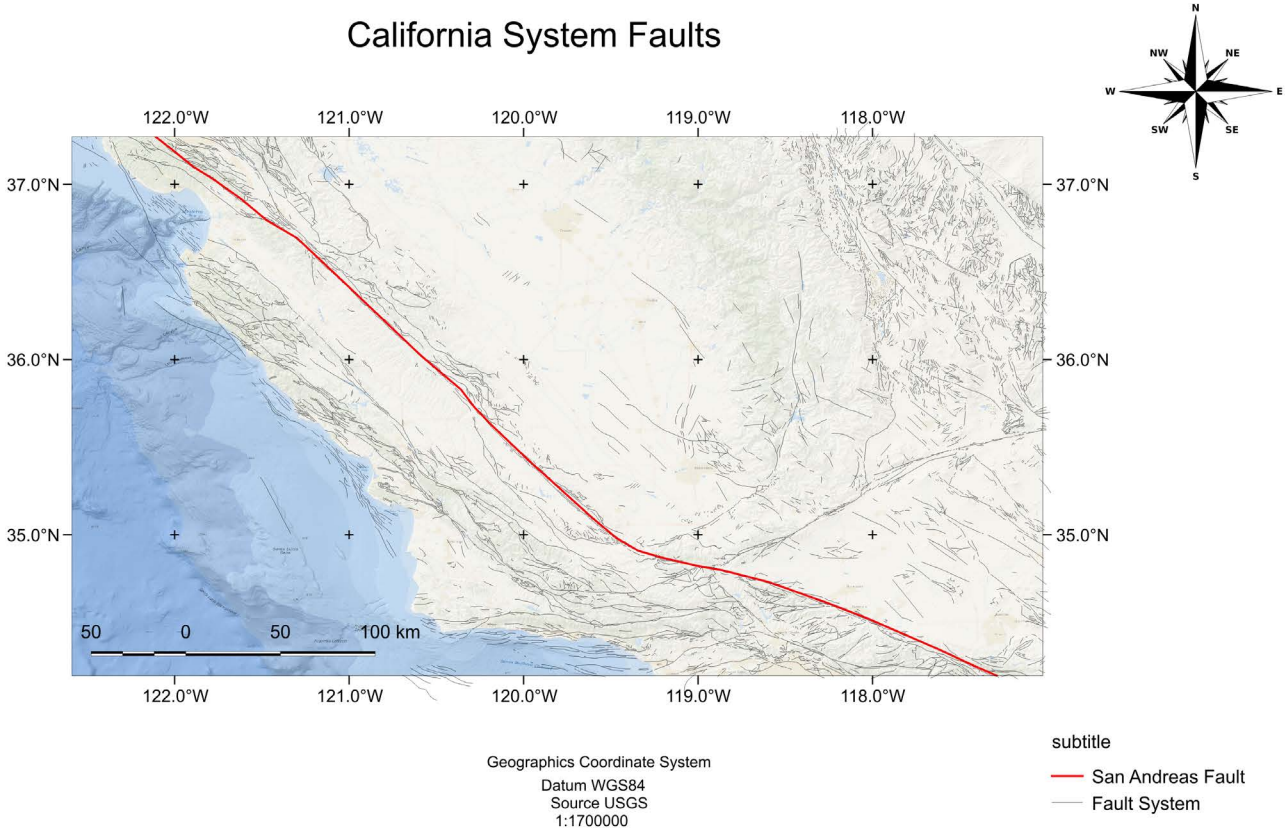



**Figure 1.** Map of California illustrating details of the faults system under study taken from usgs database (https://data.usgs.gov/datacatalog/ in Nov/2017).

to 2013, the data for which was taken from the Advanced National Seismic System (ANSS) catalog in Southern California. The choice of this region is related to the fact that the San Andreas Fault is a major cause of tremors and to the amount of existing tremor data available measured at the site. The San Andreas fault system in the region of San Francisco is a complex of faults and part of an isolated system where the Pacific plate meets the North American plate [16]. In April/1906[1] the magnitude was 7.8 which was one of the largest tremors in the region.

## 3. Data Preparation

Using neural networks for a better analysis of the seismological data, the ANSS Catalog was modified and only the time differences between the seismological events in decimal time were considered. **Figure 2** shows the table of seismic events and the time differences.

## 4. Multi-Layered Neural Networks

The multilayer perceptron network has played an important role in solving complex non-linear characteristic problems, such as, voice recognition [17], im-

[1]http://earthquake.usgs.gov/earthquakes/events/1906calif/18april/ in Dez/2017.

age [18] and time series prediction [19] [20], to name but a few. The multilayer networks are composed of three main parts: the input layer which contains input sensory units; the middle layer (or hidden layer) which can be more than one layer; and the output layer. All layers, not counting the input layer, are made up of neurons. Figure 3 represents the structure of a multilayer network that forms the multilayer perceptron (MLP) structure (4-3-2).

The algorithm called error retro propagation, or just retro propagation, is widely used in multilayer neural networks containing one or more hidden layers. The algorithm consists of two steps: propagation and backpropagation [21]. A set of standards is applied to the neural network and the input signal is propagated in each neuron of the hidden layers, thus reaching the output layer where the outputs of the network are generated in each neuron of this layer. The synaptic weights that interconnect the network layers, from the input layer, through the hidden or intermediate layer and reaching the output layer, are fixed in this first interaction. The example in Figure 3 shows how this interaction occurs in an MLP neural network (4-3-2).

In retro propagation, all the synaptic weights are adjusted from the output layer to the input layer, through the generation of what is called the error signal, which is based on the difference between the output generated from the network and the desired output. This signal is propagated back through the network from

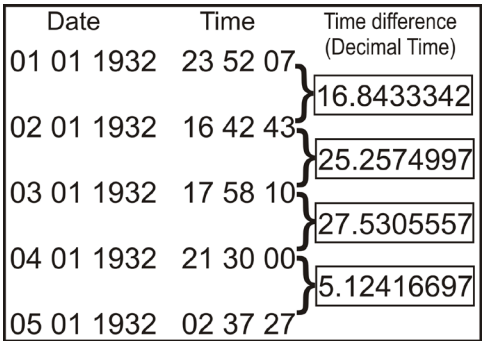



Figure 2. Scheme of time differences.

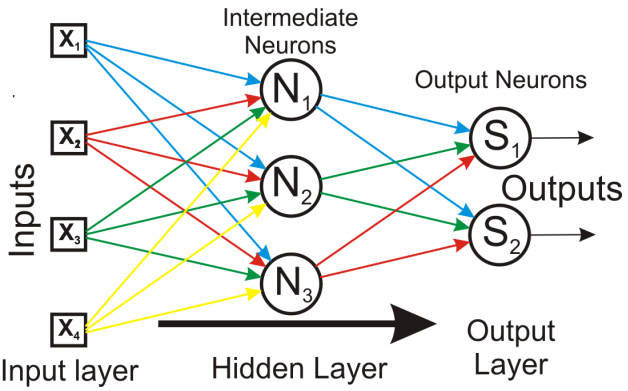



Figure 3. Illustration of a multilayer neural network structure. $X_i$ are input data, $N_i$ are intermediate layer neurons, and $S_i$ are neurons of the output layer. The connections between the layers are made by the weights.

the output layer to the hidden layer and the respective weights that interconnect these layers are adjusted so that the response generated by the network approximates the desired response. Figure 4 shows how backpropagation is performed.

## 5. Learning Algorithm

The learning type of the network is supervised and its input and output values can be binary or continuous (limited by computer precision). Its propagation rule in each neuron is shown in Equation (1)

$$net_j = \sum_{i=1}^{k} \omega_{ij} \cdot y_i + \theta_j \tag{1}$$

The learning of backpropagation is based on the updating of the synaptic weights of the network by minimizing the mean squared error using the Descending Gradient method [21]. Thus, the updating of the weight $\omega_{ij}$ with respect to the input $i$ of the neuron $j$ is shown in Equation (2).

$$\Delta\omega_{ij} = -\eta \cdot \frac{\partial E}{\partial \omega_{ij}} \tag{2}$$

where $\Delta\omega_{ij}$ is the weight variation of the input $i$ in the neuron $j$, $\eta$ is the learning rate and $E$ is the sum of the mean square error which is defined in Equation (3) as being the sum of the mean square error of all patterns inserted into the neural network [22].

$$E = \frac{1}{2}\sum_{p}\sum_{i=1}^{k}\left(d_i^p - y_i^p\right)^2 \tag{3}$$

where, $p$ is the number of patterns introduced into the network, $k$ is the number of neurons that are in the network output, $d$ is the desired output of the network and $y_i$ is the output obtained by the network for a certain standard introduced to the neural network. For each standard, the average quadratic error can be minimized, also generally leading to the minimization of the total mean quadratic error. Thus, the error can be defined by Equation (4).

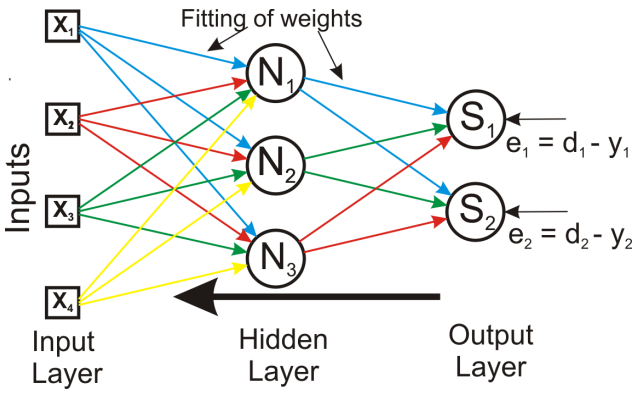



Figure 4. Illustration of the backpropagation of the error signals in the neural network. $e_i$ represents the error signal obtained between the desired output and the output of the network and which will be propagated in the network to adjust the weights.

$$E = \frac{1}{2}\sum_{i=1}^{k}\left(d_i - y_i\right)^2 \tag{4}$$

In minimizing the mean square error, we determine the error gradient in relation to the weight $\left(\frac{\partial E}{\partial \omega_{ij}}\right)$.

To continue the calculation of the gradient of the mean square error, we have two possibilities: the calculation of the error in the output layer and the indirect calculation of the error in the hidden layer based on errors of the output layer.

### 5.1. Calculation of the Error in the Output Layer

Figure 5 demonstrates the output neuron *j*, fed by the activations of the neurons of the previous layer. The inner activation of neuron *j* is given according to Equation (5).

$$net_j = \sum_{i=1}^{v} \omega_{ij} \cdot y_i \tag{5}$$

where $v$ is the total number of inputs applied to the neuron *j*, its respective activation being given by Equation (6)

$$y_j = f\left(net_j\right) \tag{6}$$

In addition, we define, according to Equation (7), the error $e_j$, the difference being of values between the desired output and the output generated by the network,

$$e_j = d_j - y_j \tag{7}$$

Through Equation (7), Equation (4) is represented in the following way

$$E = \frac{1}{2}\sum_{i=1}^{k}\left(e_j\right)^2 \tag{8}$$

Therefore, using Equation (8), we calculate the gradient of the mean square error with respect to weight:

$$\frac{\partial E}{\partial \omega_{ij}} = \frac{\partial E}{\partial y_j} \cdot \frac{\partial y_j}{\partial net_j} \cdot \frac{\partial net_j}{\partial \omega_{ij}} \tag{9}$$

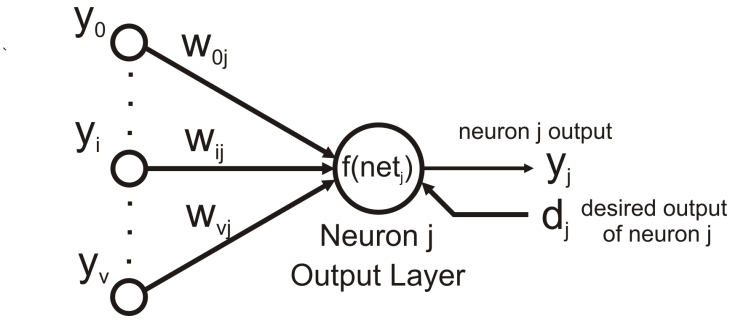



Figure 5. Output neuron *j*. The inputs are the activations of the neurons of the previous layer connected to the output. $y_j$ represents the output generated by the network and $d_j$ represents the desired output of neuron *j*.

The term $\left(\partial E/\partial y_j\right)$ will have a differential value in the calculation of the error gradient. When considering the output layer, this calculation will be very simple, because the desired outputs for the neural network are known. For the hidden layer neuron, this term will be calculated indirectly through the back-propagation of the output error across the network, which we will cover in the next section.

Thus, for the output neuron, Equation (9) is represented as follows:

$$\frac{\partial E}{\partial \omega_{ij}} = \frac{\partial E}{\partial e_j} \cdot \frac{\partial e_j}{\partial y_j} \cdot \frac{\partial y_j}{\partial net_j} \cdot \frac{\partial net_j}{\partial \omega_{ij}} \tag{10}$$

By calculating the derivatives of Equation (10) we get:

$$\frac{\partial E}{\partial \omega_{ij}} = -e_j \cdot f'\left(net_j\right) \cdot y_i \tag{11}$$

Equation (11) represents the derivative of the mean square error with respect to the synaptic weight $\omega_{ij}$ of neuron *j* of the output layer. In this way, we define the local gradient $\delta_j$ according to Equation (12)

$$\delta_j = -\frac{\partial E}{\partial e_j} \cdot \frac{\partial e_j}{\partial y_j} \cdot \frac{\partial y_j}{\partial net_j} = e_j \cdot f'\left(net_j\right) \tag{12}$$

Thus, the update of the weights of the neurons of the output layer are given by Equation (13)

$$\omega_{ij} = \omega_{ij} + \Delta\omega_{ij} = \omega_{ij} + \left(-\eta \cdot \frac{\partial E}{\partial \omega_{ij}}\right) = \omega_{ij} + \eta \cdot \left(d_j - y_j\right) \cdot f'\left(net_j\right) \cdot y_i \tag{13}$$

where $\eta$ represents the learning rate of the neural network.

### 5.2. Calculation of the Error in the Hidden Layer

When we consider the neuron *j* as a neuron of the hidden layer, there is no desired output for this neuron and its respective error signal must be calculated based on all the error signals of all the neurons connected to this unhidden neuron. Figure 6 shows the hidden layer neuron *j*, fed by the neurons of the output layer.

From Equation (12), we can redefine the local gradient $\left(\delta_j\right)$ for the hidden layer neuron *j*

$$\delta_j = -\frac{\partial E}{\partial y_i} \cdot \frac{\partial y_j}{\partial net_j} = -\frac{\partial E}{\partial y_j} \cdot f'\left(net_j\right) \tag{14}$$

For the neuron *k* shown in Figure 6 of the output layer,

$$E = \frac{1}{2} \sum_{k \in saida} \left(e_k\right)^2 \tag{15}$$

from Equation (15), we get the value of $\left(\frac{\partial E}{\partial y_i}\right)$

$$\frac{\partial E}{\partial y_i} = \sum_k e_k \cdot \frac{\partial e_k}{\partial y_i} \tag{16}$$

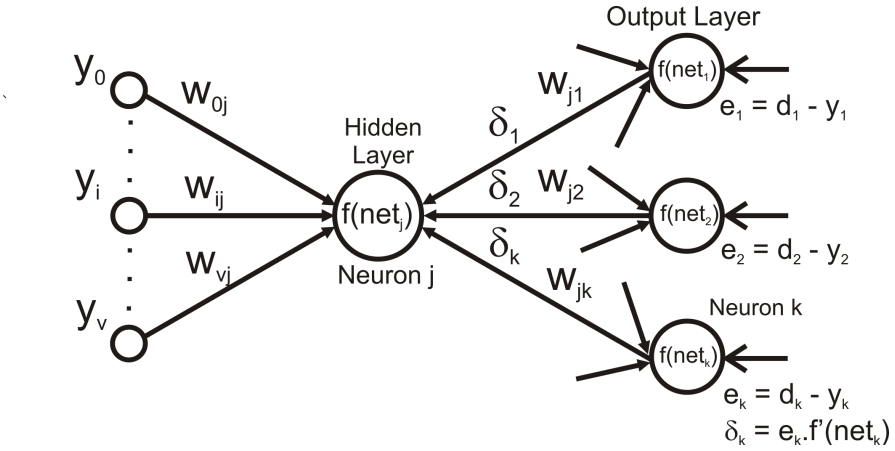



**Figure 6.** Hidden neuron *j*. The inputs are the activations of the neurons of the previous layer connected to the hidden neuron, being this neuron connected to the output neurons. $\delta_k$ represents the local gradient of the output neurons. When we have only one hidden layer, the inputs of these neurons are the data input.

By extending Equation (16) a little more, we obtain Equation (17)

$$\frac{\partial E}{\partial y_i} = \sum_k e_k \cdot \frac{\partial e_k}{\partial net_k} \cdot \frac{\partial net_k}{\partial y_i} \tag{17}$$

The output neuron error *k* and its respective derivative are given by Equation (18) and Equation (19).

$$e_k = d_k - f\left(net_k\right) \tag{18}$$

$$\frac{\partial e_k}{\partial net_k} = -f'\left(net_k\right) \tag{19}$$

In addition, as seen in **Figure 6**, we can verify that the internal activation level of the neuron *k* and its respective derivative are given by Equation (20) and Equation (21).

$$net_k = \sum_j \omega_{jk} y_j \tag{20}$$

$$\frac{\partial net_k}{\partial y_i} = \omega_{jk} \tag{21}$$

Thus, from the results of the derivatives in Equation (17), we get

$$\frac{\partial E}{\partial y_i} = -\sum_k e_k \cdot f'\left(net_k\right) \cdot \omega_{jk} = -\sum_k \delta_k \cdot \omega_{jk} \tag{22}$$

The term $e_k \cdot f'\left(net_k\right)$ in Equation (22) was defined as $\delta_k$ as well as in Equation (12), by only changing the index *j* to index *k*.

Thus, by replacing Equation (22) in Equation (14), we obtain the expression of the local gradient $\delta_j$ for the hidden layer neuron *j*:

$$\delta_j = f'\left(net_j\right) \cdot \sum_k \delta_k \cdot \omega_{jk} \tag{23}$$

With this, the update of the weights of the neurons of the hidden layer, are given by Equation (24)

$$\omega_{ij} = \omega_{ij} + \left( -\eta \cdot \frac{\partial E}{\partial \omega_{ij}} \right) = \omega_{ij} + \eta \cdot y_i \cdot \delta_j = \omega_{ij} + \eta \cdot y_i \cdot f'\left(net_j\right) \cdot \sum_k \delta_k \cdot \omega_{jk} \quad (24)$$

### 5.3. Learning Parameters

The learning algorithm is performed by minimizing the mean square error as a function of the synaptic weights which generates a movement for an overall minimal error throughout the interactions. The main parameters that have direct intervention in the learning process of the network are the learning rate and the momentum term.

The learning rate ($\eta$) is a constant parameter that varies at interval [0, 1] and influences the convergence of the learning process, orienting the change of the synaptic weights. A small learning rate generates a very slight change in weights, however, it requires a very long training time with the added possibility of the error dropping to a local minimum preventing it from leaving this point [22].

If the learning rate is very large, for example, near the maximum value that is 1, there are larger changes in the weights, allowing for instabilities around the global minimum. A learning rate value that does not generate problems for error minimization should be large enough so as not to cause oscillations in minimization and should only result in faster learning [22].

An alternative that can be used to increase the learning rate without creating oscillations around the global minimum is found when we modify Equation (13) or Equation (24) and include the term momentum, which brings information of the past changes of the weights in the direction of update of the new weights. Equation (25) shows how the updating of the weights with the inclusion of the term momentum is modified.

$$\Delta\omega_{ij}^{t+1} = \eta \cdot \delta_j \cdot y_i + \alpha \cdot \Delta\omega_{ij}^{t} \quad (25)$$

where $\Delta\omega_{ij}^{t+1}$ and $\Delta\omega_{ij}^{t}$ correspond to the variation of the network weights of an input or neuron $i$ bound to the neuron $j$ at time $t+1$ and $t$ respectively, $\eta$ is the learning rate, and $\alpha$ is the momentum term [22].

## 6. Results

### 6.1. Varying the Network Inputs

In this analysis, in order to verify the relationship between the number of inputs of the network with the distribution of the data tested, we consider the input data as the time differences between all the seismological events occurring sequentially, without filtering of the measured magnitude. Thus, through the network structure of **Figure 7(a)**, the number of network inputs were varied, adjusting the 100 subsequent data and testing the next 100 data, as shown in **Figure 7(b)**. In this manner, we made the variations for 10 inputs, 40 inputs, 60 inputs and 100 inputs.

From the results of the 100 trained data and 100 data tested (**Figure 8(a)**), we performed the distribution of the difference between the real values and the

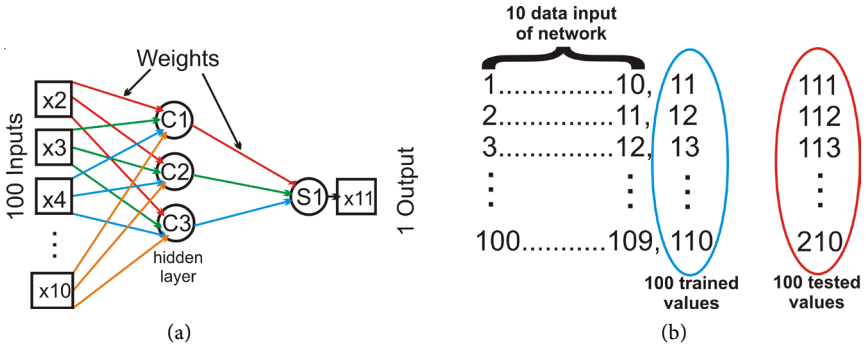



**Figure 7.** Schematic of neural network used and how training and actual data were performed. (a) Schematic of neural network used; (b) Training scheme and training of actual data.

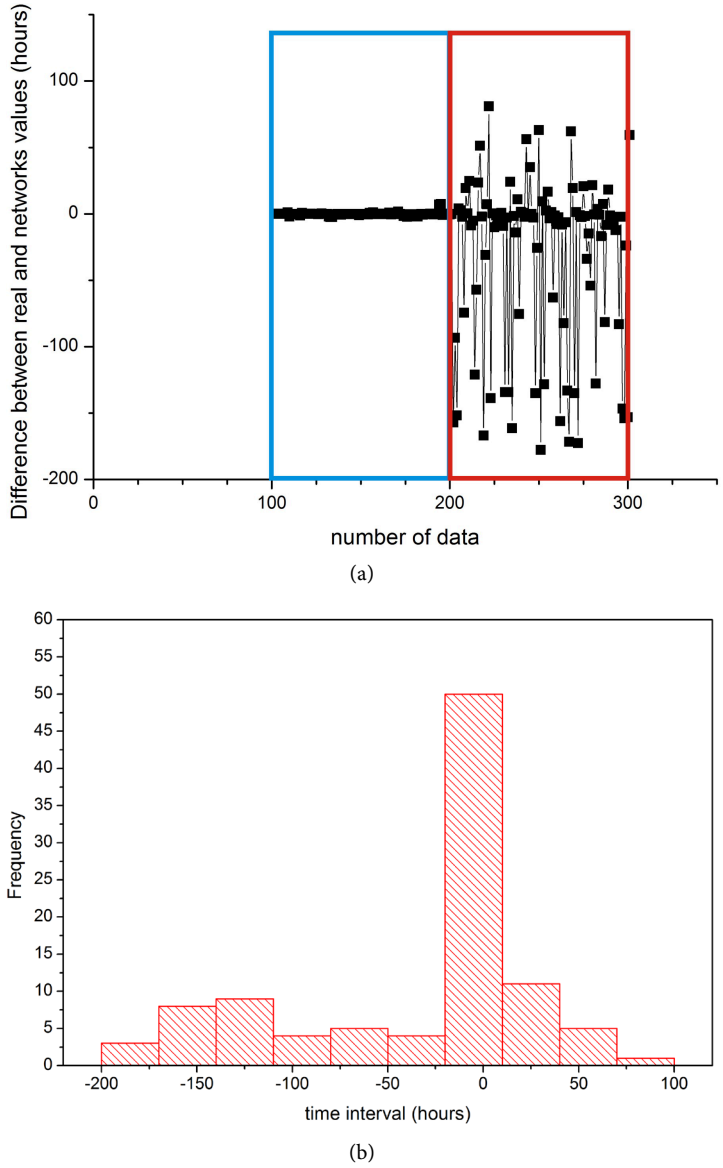



**Figure 8.** Scheme of the data trained and tested with histogram of the data tested. (a) Schematic of the trained data and tested; (b) Histogram of the difference between the data tested and the actual and estimated values.

values generated by the network and thus obtained better results for the configuration of the network with 100 inputs, shown in Figure 8(b). As seen in this graph, the distribution of these data is around 50%, thus indicating a good generalization of the data. The training interval for these data was from 12/04/1932 to 08/07/1932, and the test interval for these data was from 10/07/1932 to 10/09/1932.

### 6.2. Values Constant Input of the Network

The structure represented by Figure 9(a), shows how the network training was performed with 100 input values to train the subsequent 100 entries represented in Figure 8(b). The difference of the previous analysis is that we move the 100 data that would enter the network of 5, 10, 15, 20, 25 and 30 values, always training the subsequent 100 and testing the next values to a deadline. Therefore, with each interaction, we had ever smaller values tested after the training of the data. The objective of this verification was a better observation of the distribution of the data tested when shifting the data, as the number of data tested would be smaller.

Figure 10(a) shows how the data was shifted for application of data input, training and data testing. The distribution of the difference data, the real values, the values generated by the network and the shifting of the input data from the network of 5 (Figure 10(b)), was the best of our results showing a frequency around 50 compared to the other displacements of data %, which may indicate a good generalization of the network training.

### 6.3. Filtering the Magnitudes

Since the previous data were made with all magnitude values and with the tests we also noticed that indicators of promising results were obtained due to the peak of the distribution being found around zero. We performed the same type of analysis, filtering the time differences of the data greater than 3.0, 3.5, 4.0 and 4.5, also observing in the data below 4.0 that it was necessary to withdraw from the data the events which are "quarry blast" and "sonic boom" (sonic blasts). For

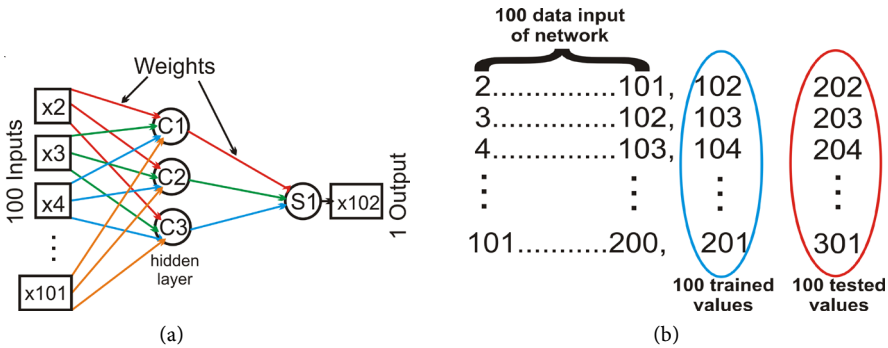



Figure 9. Scheme of the neural network used and how training and actual data were performed. (a) Schematic of the neural network used; (b) Training scheme in the training of the real data.

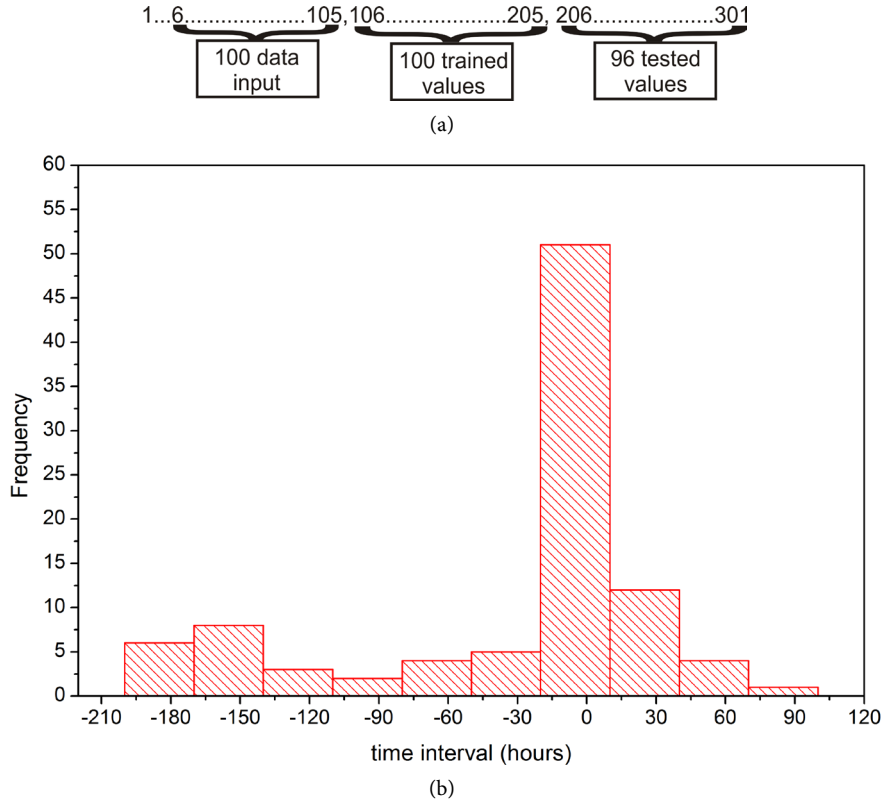



**Figure 10.** Data Shift Scheme and Histogram of the values tested after the training. (a) Scheme of the displacement of 5; (b) Histogram of the difference between real and network values for a 5 data shift.

data of magnitudes greater than 3.0 we found a total of 19,984 data, for magnitudes greater than 3.5, 6809 given, for magnitudes greater than 4,0, 2240 and for magnitudes larger than 4.5, 714 given.

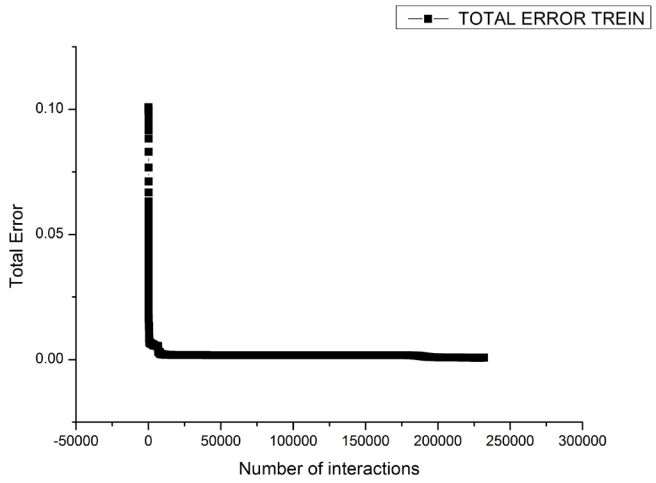
In order to improve the results, we performed the criterion of stopping data training based on the convergence of the training error and the minimum error of prediction of the data. We had good results for magnitudes greater than 4.0 and 4.5. The graph in **Figure 11(b)** shows the minimum prediction error of the data for magnitudes greater than 4.0 and due to this error we stop the training of the network that already had its error minimized (**Figure 11(a)**).

The graph in **Figure 12(a)** shows the difference of the real data and the network for 100 data that were trained and 100 data that were predicted, already a **Figure 12(b)** presents the data distribution.

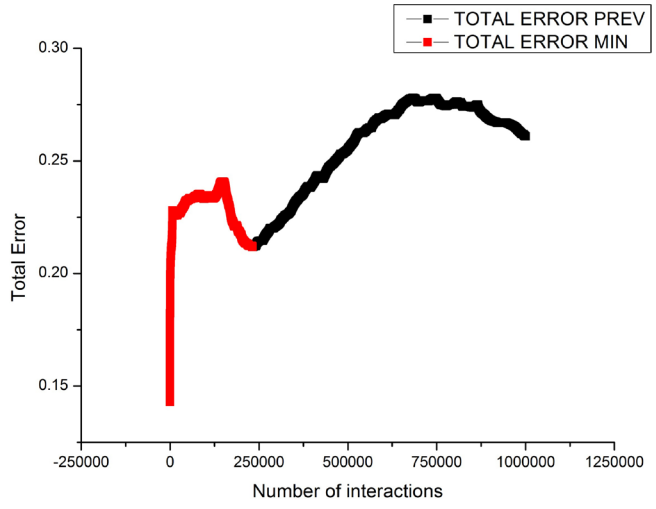
The graph in **Figure 12(b)** shows a 750 hour forecast interval of around 65%, this time the interval corresponding to a value almost twice as high as the average of 400 hours between the intervals tremors.

The graph in **Figure 13(b)** shows the minimum prediction error of the data for magnitudes greater than 4.5. With this error we stop the training of the network that already had its error minimized (**Figure 13(a)**).

The graph in **Figure 14(a)** shows the difference of the real data and the network for 100 data that were trained, and 100 data that were already predicted

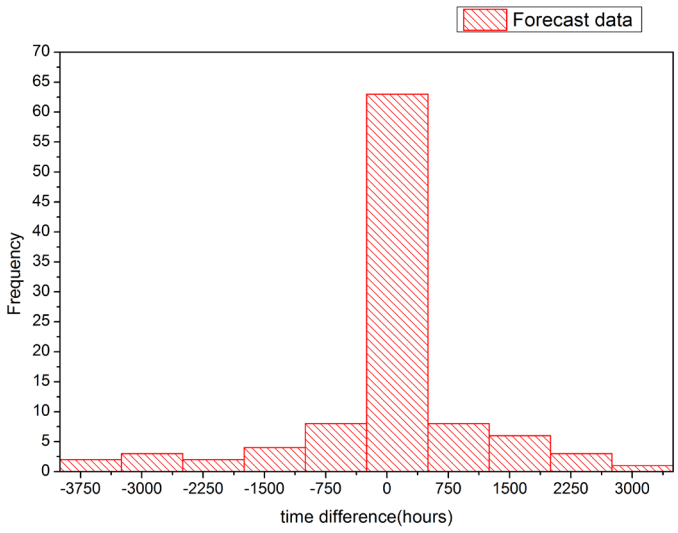



(a)

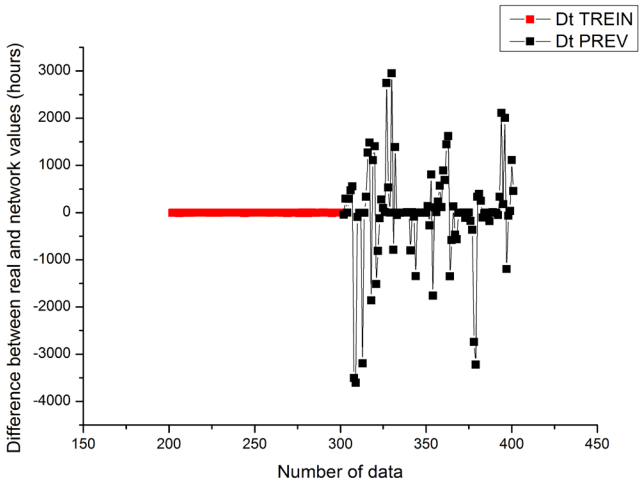



(b)

**Figure 11.** Total network training error interrupted in the minimum error of the data forecast. (a) Scheme of the displacement of 5; (b) Total error of the prediction of the data.

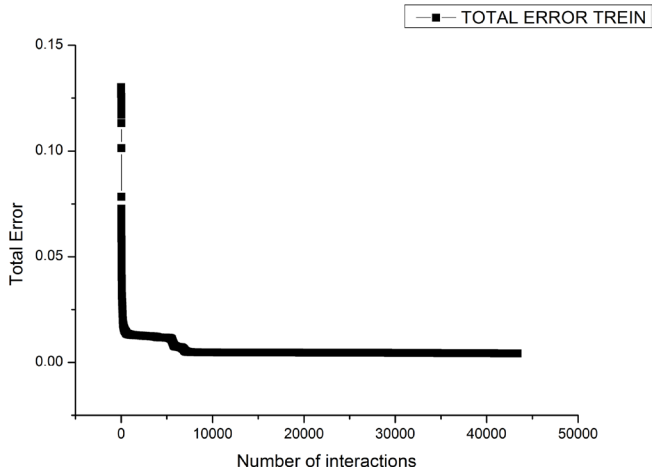



(a)

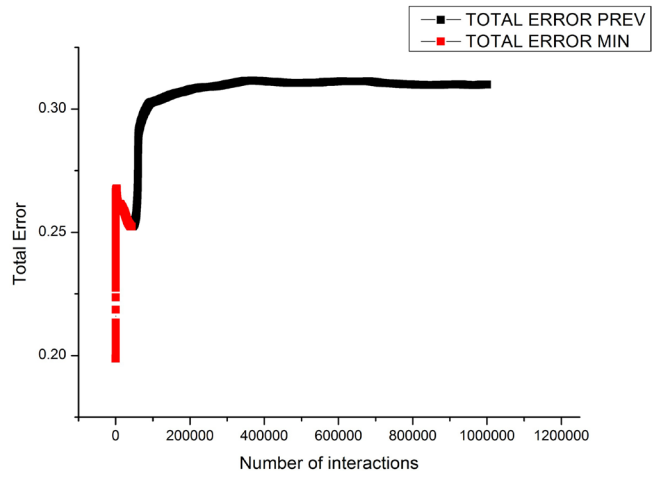



(b)

**Figure 12.** Graph of training and prediction of data and distribution of forecast data. (a) Data distribution Data forecast; (b) Network training graph and.

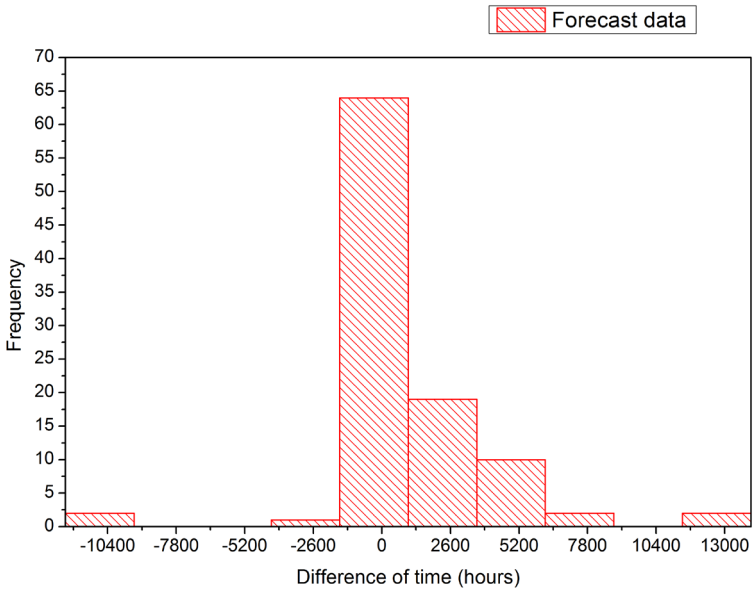



(a)

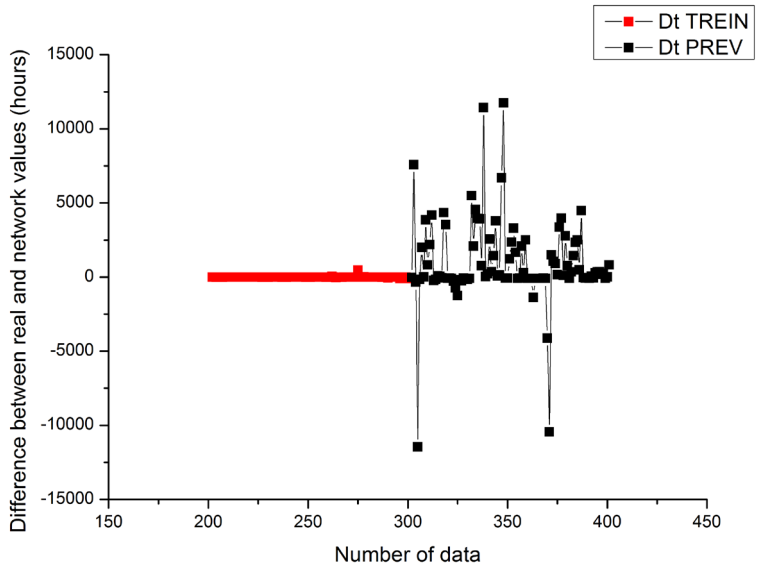



(b)

**Figure 13.** Total network training error interrupted at minimum data prediction error. (a) Total network training error; (b) Total data prediction error.




(a)




(b)

**Figure 14.** Graph of training and forecasting of data and distribution of forecast data. (a) Distribution of data and data forecast. Forecasting the network; (b) Graph of the network training.

**Figure 14(b)** presents the distribution of the predicted data.

The graph in **Figure 14(b)** shows a range of 2600 forecast hours around 65%. This time interval between tremors corresponds to a value almost two times greater than the average of 1505 hours.

## 7. Conclusion

The peaks around zero in the distribution using all values of magnitudes in our network, proved to be a good indicator of seismological prediction. When ap-

plying this same procedure to data with magnitudes greater than 3.0, we had magnitudes greater than the range of 4.0 and 4.5. These are promising results when we introduce the training stop criterion at the moment in which the minimum error of forecast of the data reaches the global minimum. Therefore, we can verify that our model has a response to the data forecast. The prediction estimate was calculated roughly by the width of the bins of the histograms. A better idea of the relationship between prediction interval and the mean time between events will be obtained by adjusting the data to some statistical distribution that allows quantitative calculation of the half-life of the distribution. This study is in course and will be published elsewhere.

## Acknowledgements

V. H. A. D. thanks CAPES (Brazilian Education Founding Agency) for a fellowship. A. R. R. P. thanks CNPq (Brazilian Research Founding Agency) for a fellowship.

## References

[1] Grossberg, S. and Stone, G. (1987). Neural Dynamics of Word Recognition and Recall: Attentional Priming, Learning, and Resonance. *Advances in Psychology*, **43**, 403-455. https://doi.org/10.1016/S0166-4115(08)61768-9

[2] Katkov, O.N. and Pimenov, V.A. (2004) Properties of Neural Networks and Their Use for Voice Signals Recognition. *Journal Telecommunications and Radio Engineering*, **62**, 467-476.

[3] Aso, N., Ohta, K. and Ide, S. (2013) Tectonic, Volcanic, and Semi-Volcanic Deep Low-Frequency Earthquakes in Western Japan. *Tectonophysics*, **600**, 27-40. https://doi.org/10.1016/j.tecto.2012.12.015

[4] Pu, H.-C., Lin, C.-H., Chang, L.-C., Kan, C.-W., Lin, C.-M., Li, Y.-H., Lai, Y.-C. and Shih, M.-H. (2017) Geological Implications of the 212 Earthquake in 2014 at the Tatun Volcano Group of Taiwan: Synergistic Effects of Volcanic and Faulting Activities. *Journal of Asian Earth Sciences*, **149**, 93-102.

[5] Chaudhary, B., Hazarika, H., Ishibashi, I. and Abdullah, A. (2017) Sliding and Overturning Stability of Breakwater under Combined Effect of Earthquake and Tsunami. *Ocean Engineering*, **136**, 106-116. https://doi.org/10.1016/j.oceaneng.2017.03.021

[6] Ruiz-Pinilla, J.G., Adam, J.M., Pérez-Cárcel, R., Yuste, J. and Moragues, J.J. (2016) Learning from RC Building Structures Damaged by the Earthquake in Lorca, Spain, in 2011. *Engineering Failure Analysis*, **68**, 76-86. https://doi.org/10.1016/j.engfailanal.2016.05.013

[7] Pakiser, L., Eaton, J., Healy, J. and Raleigh, C. (1969) Earthquake Prediction and Control. *Science*, **166**, 1467-1474. https://doi.org/10.1126/science.166.3912.1467

[8] Scholz, C., Sykes, L. and Aggarwal, Y. (1973) Earthquake Prediction: A Physical Basis. *Science*, **181**, 803-810. https://doi.org/10.1126/science.181.4102.803

[9] Abe, S. and Suzuki, N. (2005) Scale-Free Statistics of Time Interval between Successive Earthquakes. *Physica A*: *Statistical Mechanics and Its Applications*, **350**, 588-596. https://doi.org/10.1016/j.physa.2004.10.040

[10] Ferreira, D.S.R., Papa, A.R.R. and Menezes, R. (2014) The Small World of Seismic

Events. *Studies in Computational Intelligence*, **549**, 97-105. https://doi.org/10.1007/978-3-319-05401-8_10

[11] Ferreira, D.S.R., Papa, A.R.R. and Menezes, R. (2015) On the Agreement between Small-World-Like OFC Model and Real Earthquakes. *Physics Letters*, *Section A*: *General*, *Atomic and Solid State Physics*, **379**, 669-675.

[12] Adeli, H. and Panakkat, A. (2009) A Probabilistic Neural Network for Earthquake Magnitude Prediction. *Neural Networks*, **22**, 1018-1024. https://doi.org/10.1016/j.neunet.2009.05.003

[13] Panakkat, A. and Adeli, H. (2009) Recurrent Neural Network for Approximate Earthquake Time and Location Prediction Using Multiple Seismicity Indicators. *Computer-Aided Civil and Infrastructure Engineering*, **24**, 280-292. https://doi.org/10.1111/j.1467-8667.2009.00595.x

[14] Alarifi, A.S.N., Alarifi, N.S.N. and Al-Humidan, S. (2012) Earthquakes Magnitude Predication Using Artificial Neural Network in Northern Red Sea Area. *Journal of King Saud University—Science*, **24**, 301-313.

[15] Reyes, J., Morales-Esteban, A. and Martínez-Álvarez, F. (2013) Neural Networks to Predict Earthquakes in Chile. *Applied Soft Computing Journal*, **13**, 1314-1328. https://doi.org/10.1016/j.asoc.2012.10.014

[16] Koirala, M.P. and Hayashi, D. (2010) Fault Type Analysis along the San Andreas Fault Zone: A Numerical Approach. *Journal of Mountain Science*, **7**, 36-44.

[17] Salmela, P., Lehtokangas, M. and Saarinen, J. (1999) Neural Network Based Digit Recognition System for Voice Dialing in Noisy Environments. *Information Sciences*, **121**, 171-199. https://doi.org/10.1016/S0020-0255(99)00077-8

[18] Sibai, F.N., Hosani, H.I., Naqbi, R.M., Dhanhani, S. and Shehhi, S. (2011) Iris Recognition Using Artificial Neural Networks. *Expert Systems with Applications*, **38**, 5940-5946. https://doi.org/10.1016/j.eswa.2010.11.029

[19] Karunasinghe, D.S. and Liong, S.-Y. (2006) Chaotic Time Series Prediction with a Global Model: Artificial Neural Network. *Journal of Hydrology*, **323**, 92-105. https://doi.org/10.1016/j.jhydrol.2005.07.048

[20] Zhao, P., Xing, L. and Yu, J. (2009) Chaotic Time Series Prediction: From One to Another. *Physics Letters A*, **373**, 2174-2177. https://doi.org/10.1016/j.physleta.2009.04.033

[21] Rumelhart, D.E., Hinton, G.E. and Williams, R.J. (1986) Learning Representations by Back-Propagating Errors. *Nature*, **323**, 533-536. https://doi.org/10.1038/323533a0

[22] Haykin, S. (2001) Neural Networks—A Comprehensive Foundation, Canada. Pearson Prentice Hall, Upper Saddle River.