International Journal of Engineering and Applied Sciences (IJEAS)
Vol.2, Issue 3(2010)71-81

IS ARTIFICIAL NEURAL NETWORK SUITABLE FOR DAMAGE LEVEL DETERMINATION OF RC- STRUCTURES?

A. K. Baltacıoğlu[1], B. Öztürk[2], Ö. Civalek[3], B. Akgöz[4]
[1,3,4]Akdeniz University, Civil Engineering Department, Antalya-TURKIYE
[2]Niğde University, Civil Engineering Department, Antalya-TURKIYE

Accepted Date: 07.04.2010

Abstract

In the present study, an artificial neural network (ANN) application is introduced for estimation of damage level of reinforced concrete structures. Back-propagation learning algorithm is adopted. A typical neural network architecture is proposed and some conclusions are presented. Applicability of artificial neural network (ANN) for the assessment of earthquake related damage is investigated.

Keywords: *Damage level, artificial neural networks, earthquake.*

1. Introduction

It is obvious that extensive levels of structural damages consisting of both flexural and shear failures occur in reinforced concrete structures during earthquakes. Accordingly, there is a need for the determination of damage which needs expertise. However, due to a limited number of experts and time limitation, it is still an important problem to determine the damage levels of hundreds of thousands of buildings. The main target of this article is to answer the question whether artificial neural networks (ANN) can be used in this specific area. Consequently, it is desired to define an alternative network structure and network input vectors, if the ANN is found to be a suitable instrument for the determination of structural damage.

2. Artificial Neural Networks (ANN)

The elementary nerve cell, generally called a neuron is the fundamental structures of biological neural network. The general elements of a generic neuron are shown in Figure1. A typical cell has three major regions: the cell body, which is also called the soma, the axon and the dendrites. The axon of a typical neuron makes a thousand synapses with other adjacent neurons. Dendrites receive information (signal) from neurons through axons. The axon-dendrite contact organ is called a synapse. The synapse is where the neuron introduces its signal to the following neuron [1-10].

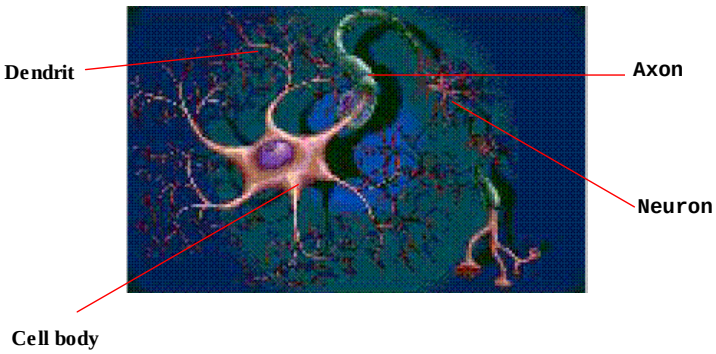

Fig. 1. A typical biological neuron and its elements

In the neural network, the unit analogous to the biological neuron is referred to as a *processing element*. A processing element (PE) has many input paths (dendrites) and combines, usually by a simple summation, the values of these input paths. The result is an internal activity level for processing element. The combined input is then modified by a transfer function. This transfer function can be a threshold function which only passes information, if the combined activity level reaches a certain level; or it can be a continuous function of the combined input. The output value of the transfer function is generally passed directly to the output path of the processing element [10-15].

The output path of a processing element can be connected to input paths of other processing elements through connection weights which correspond to the synaptic strength of neural connections. Since each connection has a corresponding weight, the signals on the input lines to a processing element are modified by these weights prior to being summed. Thus, the summation function is a weighted summation. In itself, this simplified model of a neuron is not very interesting; the interesting effects result from the ways neurons are interconnected [16-21]. In an artificial neural network model, the input to a node is calculated by the weighted sum of outputs from the nodes in the previous layer. A node has an activation function that evaluates inputs and generates an output as an input to other nodes. McCullough and Pitts [22] proposed a simple model of a biological neuron as a binary threshold unit as shown below (Fig.2). Specially, the model neuron computes a weighted sum of its inputs from other units, and outputs a one or a zero given by[30],

$$y_j = F\left(\sum_{i=1}^{N} x_i w_{ji} - \theta_j\right) \tag{1}$$

where F is an activation function that will be explained in the next section, y_j is the output of the jth neuron, x_i is the input for this node, w_{ji} is the weighting coefficient, and θ_j is the bias term.

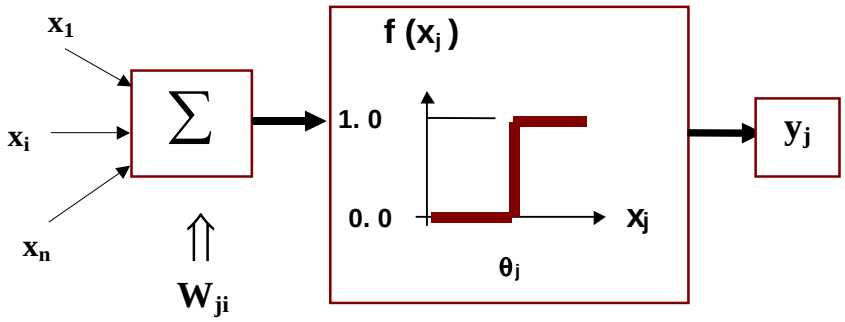

Fig. 2. McCullough - Pitts model of a biological neuron

A neural network is defined by its node characteristics, the learning rules, the network topology, and the activation function. The learning rules control the improvement of network performance through appropriate adaptive changes of the weights of the links [15-20].

All learning methods can be classified into three categories: supervised learning, unsupervised learning and reinforcement learning. Supervised learning is a process that incorporates an external teacher and/or global information. A teacher specifies the desired output of the network, and the training data consist of input-output pairs. In reinforcement learning, the input is not a precise teaching input, but is rather only a *good* or *bad* performance rating. In other words, reinforcement is like supervised learning, except that in supervised learning, the feedback provided to the network is instructive, whereas in reinforcement learning, it is evaluative. In unsupervised learning, the network attempts to develop internal models to capture patterns of regularity in the input signals [23-25].

Among the various neural network paradigms available, back-propagation networks are by far the most widely utilized for their relatively simple mathematical proofs and good generalized capabilities. The paradigm can be applied to a wide range of applications involving classification, estimation, prediction, and functions synthesis etc[26-28].

3. Back-Propagation Neural Networks

The back-propagation algorithm is one of the most important and well known learning algorithms for neural networks especially in engineering applications. It provides important results since multi-layer networks can be adapted through supervised teaching with a training set of vectors. Among the various neural network paradigms available, back-propagation networks are by far the most widely utilized for their relatively simple mathematical proofs and efficient generalization capabilities. Back-propagation networks have two major characteristics: they are multi-layered in structure and they generally incorporate a nonlinear sigmoid transfer function for their processing elements [29-32].

The back-propagation neural network is a multi-layered network consisting of fully interconnected layers comprising many simple and identical units. The architecture of back-propagation network with one hidden layer is shown in Figure 3.

In Figure 3, x_n , y_m, and z_k represent the input, hidden and output layers. Weights on connections between the input and hidden layers are denoted by W_{ij} , while weights on

connections between the hidden and output layers are denoted by W_{jk}. Each unit in one layer is connected in the forward direction to every unit in the next layer. Activation flow occurs from the input layer through the hidden layer, then on to the output layer. As usual, the knowledge of the network is encoded in the weights on connections between units.

The back-propagation training algorithm is an iterative gradient algorithm designed to minimize the mean square error between the actual output of a multi-layer feed-forward perceptron and the desired output. Although back-propagation is a trained algorithm, it is generally used and well known as a type of artificial neural networks.

In contrast to the parallel relaxation method used by Hopfield network, back-propagation networks perform a simpler computation. Because, activation flow is in only one direction, and there is no need for an iterative relaxation process. The activation levels of the units in the output layer determine the output of network. Back-propagation is currently the most important and most widely used algorithm for connectionist learning. Its rapid rise in popularity has been a major factor in the resurgence of neural networks.

As is the case with most neural networks, the aim is to train the net to achieve a balance between the ability to respond correctly to the input patterns that are used for training and the ability to give reasonable responses to input that is similar, but not identical, to that used in training. The training of a network by back-propagation involves three stages: the feed-forward of the input training pattern, the calculation and back-propagation of the associated error, and the adjustment of the weights.

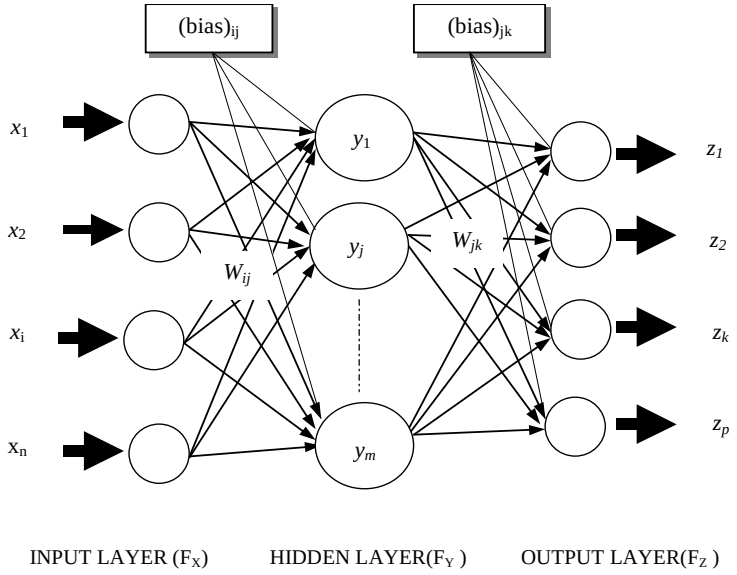

Fig. 3. A typical multi-layer neural network

Back-propagation networks have two main characteristics: they are multi-layered in structure and they generally incorporate a nonlinear sigmoid transfer function for their processing element. An activation function for a back-propagation network should have several important characteristics as it should be continuous, differentiable, and monotonically non-decreasing. Furthermore, for computational efficiency, it is desirable that its derivative be

easy to compute [27]. For the most commonly used activation functions, the value of the derivative can be expressed in terms of the value of the function. One of the most typical functions is the binary sigmoid function, which has a range of (0,1) and is defined as

$$f(x) = \frac{1}{1+\exp(-net_j)} \tag{2}$$

where x_j is the weighted summation of the total input. Back-propagation networks are typically trained using the generalized delta rule, application of which involves the calculation of the network output, a comparison of this output with the desired output, the calculation of an *error*, and a backward propagation of this error in order to *correct* future outputs. In this process, each neuron updates the weights of its input connections in such a way that the error associated with its own output activation is decreased. Like a perceptron, a back-propagation network typically starts out with a random set of weights.

The Back-propagation algorithm is summarized below [30-35] ;

Weight Initialization : Set all weights and node thresholds to small random numbers. Note that the node threshold is the negative of the weight forming? the bias unit.

Calculation of Activation :

The activation level of an input unit is determined by the instance presented to the network.

The activation level O_j of a hidden and output layer is determined by

$$O_j = F\left(\sum W_{ji} O_i - \theta_j\right)$$

where W_{ji} is the weight from an input O_i , θ_j is the node threshold, F is a sigmoid function that was defined above equation (Eq.2).

Start at the output units and work backward to the hidden layers recursively. Adjust weights by

$$W_{ji}(t+1) = W_{ji}(t) + \Delta W_{ji}$$

where $W_{ji}(t)$ is the weight from unit *i* to unit *j* at time *t* (or *t*th iteration) and ΔW_{ji} is the weight adjustment.

The weight change is computed by

$$\Delta W_{ji} = \alpha \delta_j O_i$$

where α is a trial-independent learning rate ($0< \alpha <1$) and δ_j is the error gradient at unit *j*. Convergence is sometimes faster by adding a momentum term [29];

$$W_{ji}(t+1) = W_{ji}(t) + \alpha \delta_j O_i + \beta[W_{ji}(t) - W_{ji}(t-1)]$$

where β is known as the momentum term. If unit j is an output unit, then its δ_j is calculated by

$$\delta_j = (T_j - O_j) F'(net_j)$$

where

$$net_j = \sum_i W_{ji} O_i$$

If unit j is a hidden unit, then its δ_j is given by

$$\delta_j = F'_j(net_j) \sum_k \delta_k W_{kj}$$

where F_j' is the derivative of the activation function . Thus this derivative is

$$F'(net_j) = \frac{\exp(-net_j)}{\left[1+\exp(-net_j)\right]^2} = F(net_j)\left[1 - F(net_j)\right]$$

The error gradient is given by :

a) For the output units ;

$$\delta_j = O_j(1-O_j)(T_j - O_j)$$

where T_j is desired (target) output activation and O_j is the actual output activation at output unit *j*.

b) For the hidden units ;

$$\delta_j = O_j(1-O_j)\sum_k \delta_k W_{kj}$$

where δ_k is the error gradient at unit k to which a connection points from hidden units *j*. Repeat iterations until convergence in terms of the selected error criterion. The well known error criterion [9,38] is defined

$$E = \frac{1}{2}\sum_{j}(T_j - O_j)^2$$

$$E = \frac{1}{2P}\sum_{p}\sum_{j}(T_j - O_j)^2$$

The second error statement is usually known as the whole system average error, and P is the total number of instances.

4. Architecture of the Network

In general, there are many different types of neural networks architectures and usually there is not one connections topology that is suitable for all types of problems. The main types of artificial neural network are the Hopfield network [17], the Kohonen network [19,20,21], ART (Adaptive Resonance Theory), Boltzmann machine, perceptron, back-propagation networks and nowadays statistical neural network [17,19,21]. In the present study, back-propagation type neural network will be used on the estimation of damage level of structures after an earthquake.

The network must be trained so as to produce an acceptable output for a given input pattern. The network learning approach used in the present work can be described as belonging to the category of supervised learning techniques. In such an approach, the output response is known for a range of variation of input parameters. These input-output training pairs are presented to the network with some initial estimates of weights and thresholds, and are resolved by adjustment of those constants.

The architecture of the neural network used in this study is shown in Figure 4. The network is feed-forward, multi-layer neural network which have ten input nodes in the input layer, and three output nodes in the hidden layer. One hidden layer, however, has been used in the developed network. This layer included five nodes. Six nodes (or neurons) are used in the input layer. Such a network is described as having a (6: 5:1) architecture feed-forward multi-layer network.

In Figure 4, the components of the developed multi-layer network are provided. These are mainly index for crack values of structural members, index for geometrical and structural defects, index for material and workmanship quality, index for soil parameter and liquefaction, earthquake field properties and index for rigidity of structural members. These components which are found to be relevant for the developed network are explained below.

1. Index for crack values of structural members which provide information regarding the level of crack in beam, column and structural walls of a building,
2. Index for geometrical and structural defects which consider condition and level of structural shortcomings such as short column, inappropriate girder allocation,
3. Index for material and workmanship quality grading the level of quality of these two parameters,
4. Index for soil parameter and liquefaction considering the soil type, field conditions and the level of risk for liquefaction,
5. Seismic zone for the area where the structure is located,

6. Index for rigidity of structural members which consider the distribution of structural elements and their allocation. This index refers to items such as the existence of shearwall, soft story effects, existence of adequate level of shearwall in both directions and the relevance of structural system of the building.

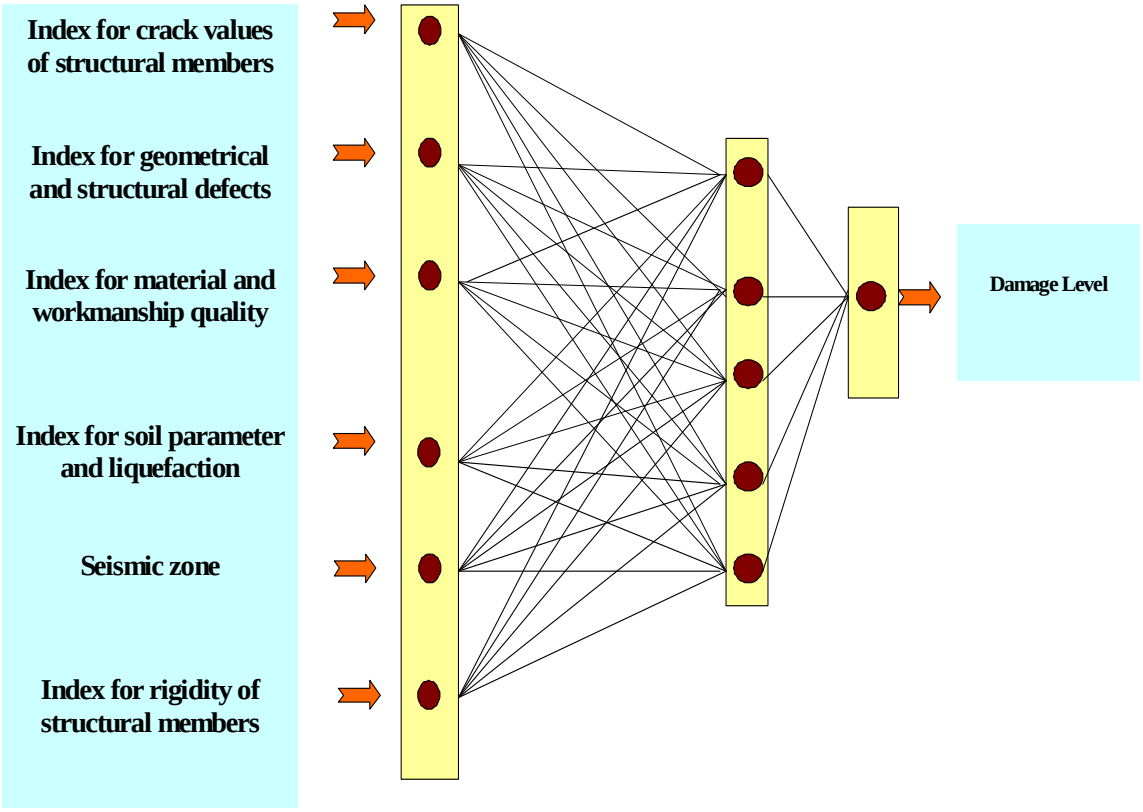

Fig. 4. Architecture of the developed multi-layer network

The developed artificial neural network well converged with one hidden layer of 5 nodes. The selection of number of nodes in the hidden layer is an important factor in the architecture of the network. The number of neurons of the hidden layer is recommended to be at least greater than square root of the sum of the number of neurons in the input and output layers [8], or usually selected as the mean value of the number of the input and output nodes plus the input nodes. More complicated networks use dynamic node growing in the hidden layer. The training set contains input and output (target) vectors. The input and output data have been normalized so that the maximum value is 1. Normalization of the data can often be as simple as either dividing the values by the maximum value or by subtracting the minimum value and then dividing the values by the range, which is the maximum value minus the minimum value.

5. Concluding Remarks

Compared to conventional digital or numerical computing techniques based on some mathematical technique, neural networks are advantageous because they can learn from example and generalize solutions to new renderings of a problem, can adapt to fine changes in the nature of the problem, are tolerant to errors in the input data, can process information

rapidly, and are readily transportable between computing systems. Certain conventional methods of computing can provide some of these advantages, but never more than a few. Neural networks do, however, suffer from a number of shortcomings, notably, a lack of precision, limited theory to assist in their design, lack of guarantee of success in finding an acceptable solution, and a limited ability to rationalize the solutions provided.

Hajela and Berke [14] stated that: "Despite their limitations, neural networks offer a powerful means of solving poorly defined problems that have eluded solution by conventional digital computing techniques. Problems of this type are in common place in civil engineering and already, within just a few years of the explosion in interest of this technology, successful solutions to seemingly intractable problems have been developed. The scope and pace of this accomplishment is likely to increase rapidly within civil engineering over the coming years. A trained (learned) neural network gives some distinct advantages over the numerical analysis/computing or conventional mathematical programs. It provides a rapid mapping of a given input into the desired output quantities, thereby enhancing the efficiency of the reanalysis. This major advantage of a trained neural network over the conventional procedure, under the provision that the predicted results fall within acceptable tolerances, leads to results that can be produced in a few clock cycles representing orders of magnitude of less computational effort than the conventional computational process".

The primary purpose of this study is to illustrate the use of a back-propagation neural network to predict the damage level of structures after an earthquake. The designed neural network is a three layer feed-forward neural network consisting of 6 nodes in input, 5 nodes in hidden and 1 node in output layers and was trained by generalized delta rule for back-propagation error. Neural networks are capable of describing, input-output functional relations, even when a mathematically explicit formula is unavailable. To create such mappings, it suffices to present a neural network with a set of known input-output pairs. During the training process, the interconnection weights of all neurons in the network are computed according to the applied learning rule. It should be noted that once the network is trained, the time required results for a given set of input is nearly instantaneous for a PC. It has been demonstrated that the artificial neural network approach applied in this study is highly successful for the purposes of damage level determination of reinforced concrete structures. In addition to these, developed network can be used on the architecture area or project office for preliminary design of structures under the earthquake effect to give an idea to the designer.

References

[1] Adeli, H., Hung, SL., Machine learning- neural networks, genetic algorithms and fuzzy systems, John Wiley & Sons, Inc., 1995.
[2] Adeli, H., Yeh, C. Perceptron learning in engineering design, Microcomputer in Civil Eng.,1989; 4: 247-56.
[3] Aleksander, I., Morton, I., An introduction to neural computing, International Thomson Computer Press., 1995.
[4] Civalek, Ö., The design of structures under earthquake effects by using neuro-fuzzy method., Fourth National Earthquake Engineering Conferences, 17-19 September, Ankara, 1997:431-38.
[5] Civalek, Ö., linear and nonlinear static-dynamic analysis of plates and shells by neuro-fuzzy technique, Ms Thesis, University of Fırat, (in Turkish), Elazığ, 1998.

[6] Civalek, Ö., The analysis of the rectangular plates without torsion via hybrid artificial intelligent technique, Proceedings of the Second International Symposium on Mathematical & Computational Applications, September 1-3, Azerbaijan, 1999:95-101
[7] Civalek, Ö., The analysis of rectangular plates via neuro-fuzzy technique, III. National Computational Mechanic Conferences, 16-18 November, Istanbul, 1998:517-25.
[8] Eberhart, R. C., and Dobbins, R. W., Neural network PC tools , Academic Press, San Diego, California,1990.
[9] Fausett, L., Fundamentals of neural networks, architectures, algorithms, and applications., Prentice-Hall, Inc., New-Jersey, 1994.
[10] Fu, LM, Neural Networks in Computer Intelligence., McGraw-Hill, Inc. New York.,1994.
[11] Ghaboussi, J., Garrett, Jr., Wu, X., Knowledge- based modeling of material behavior with neural networks, Journal of Structural Engineering, ASCE, 1991; 117: 1, 132-53.
[12] Ghaboussi, J., Lin, CC., New method of generating spectrum-compatible accelerograms using neural networks, Earthquake Eng. And Structural Dynamics, 1998; 27: 377-96.
[13] Goldberg, DE., Genetic algorithms in search optimization and machine learning, Addison-Wesley, MA, 1989.
[14] Hajela, P., Berke, L., Neurobiological computational models in structural analysis and design, Computers and Structures, 1991; 41(4): 657-67.
[15] Hani, KB., Ghaboussi, J., Neural networks for structural control of a benchmark problem, active tendon system, Earthquake Eng. And Structural Dynamics, 1998; 27:1225-45.
[16] Hertz, J., Krogh, A., Palmer, R. G., Introduction to Theory of Neural Computing, Addison – Wesley Publishing, 1991.
[17] Hopfield, JJ., Neural networks and physical systems with emergent collective computational abilities., In Proceedings of National Academy of Sciences, 1982;79: 2554-58.
[18] Kang, HT., Yoon, C J., Neural networks approaches to aid simple truss design problems, Microcomputers in Civil Eng., 1994; 9:211-18.
[19] Kohonen, T., Content addressable memories, Springer-Verlag, New-York, 1980.
[20] Kohonen, T., Associative memory: a system-theoretical approach, Spring-Verlag, New York., 1977.
[21]Kohonen, T., Self-organization and associative memory., Spring-Verlag, New York., 1988.
[22] McCullogh, WS., and Pitts, W., A logical calculus of ideas imminent in nervous activity., Bull. Math. Biophysics, 1943;5: 115-33.
[23] Civalek, Ö., Flexural And Axial Vibration Analysis Of Beams With Different Support Conditions Using Artificial Neural Networks, International Journal of Structural Engineering and Mechanics, 18(3), 303-314,2004.
[24] Michalewicz, Z., Genetic algorithms + data structure = evolution programs, Springer, Germany, 1992.
[25] Park, HS., Adeli, H., Distributed neural dynamics algorithms for optimization of large steel structures, Journal of Structural Engineering, ASCE, 1997; 123(7):880-88.
[26] Civalek, Ö., Yapay Zeka-Söyleşi, Türkiye İnşaat Mühendisleri Odası-TMH, Mühendislik Haberleri, Sayı 423, 40-50, 2003.
[27] Rojas, R., Neural networks, A systematic introduction., Springer, Germany,1996.
[28] Ross, TJ., Fuzzy logic with engineering applications, McGraw-Hill, Inc.,1995.
[29] Rumelhart, DE., Hinton, GE., and Williams, R J., Learning internal representation by error propagation. in parallel distributed processing : Explorations in the microstructures of cognition, MIT Press, Cambridge, MA., 1986.
[30] Szewezyk, ZP., Hajela, P., Damage detection in structures based on feature- sensitive neural networks, J Computing Civil Eng., ASCE, 1994; 8(2):163-78.

[31] Civalek, Ö., The analysis of circular plates via neuro-fuzzy technique, Journal of Eng. Science of Dokuzeylül University,1999;1(2):13-31.
[32] Thomsan, WT., Dahleh, MD., Theory of vibration with applications, Prentice Hall, New Jersey, 1998.
[33]Vanluchene, RD., and Roufei, S., Neural networks in structural engineering, Microcomputers in Civil Eng., 1990; 5:207-215.
[34] Wu, X., Ghaboussi, J., Garrett, JH., Use of neural networks in detection of structural damage, Computers & Structures, 1992 ; 42(4): 649-59.
[35] Zurada, J M., Introduction to artificial neural networks, West Publishing Com.,1992.