Engineering Journal of the University of Qatar, Vol. 11, 1998, p. 117 - 132

PREDICTION OF REINFORCED CONCRETE BEAM DEPTH USING NEURAL NETWORKS

Ahmed B. Senouci* and M. Ayman Abdul-Salam**
* Assistant Professor, Civil Engineering Department
** Professor, Civil Engineering Department
University of Qatar, Doha, Qatar

ABSTRACT

This paper discusses the development and the implementation of a neural network for the depth prediction of singly-reinforced rectangular concrete beams. The procedure of the American Concrete Institute (ACI-318 1995) was used as the basis for the development of the proposed network. A training set of 56 cases was used to train the network. The network adequately learned the training examples with an average training error of 3.0 percent. A testing set of 19 cases was used to validate the network. The network was able to predict the correct beam depth with an average error of 6.8 percent. A case study, where 878 new design cases were considered, was conducted to demonstrate the system's generalization and fault-tolerance properties. The network showed good generalization and fault-tolerance properties since it was able to predict the correct beam depths with an average error of 9.2 percent.

INTRODUCTION

Preliminary structural design is performed at an earlier stage of the project planning. Its aim is to proportion cross sections of individual beams so that: (1) the expected cost of the planned structure is estimated; (2) the proper dimensions of the architectural drawings are set; and (3) the beams' own weight and stiffness are computed.

The prediction of beam depths is an important part of the preliminary structural design. It depends mainly on experience and partially on recommended proportions in codes of practice. At the project planning stage, the actual design parameters, such as the concrete compressive strength, the reinforcing steel yield strength, and the desired reinforcement ratio, are usually not known. Beam depths can not, therefore, be determined accurately and can only be guessed or predicted by

structural design specialists. The development of a computerized tool that would improve beam depth predictions by structural design specialists is definitely useful.

Neural networks are computational models capable of mimicking the human decision-making process. They possess interesting properties such as self-organization, generalization, fault-tolerance, and massively parallel processing. The use of neural networks in the area of structural engineering has been reported by a number of researchers. Liu and Gan (1991) developed a neural network for the preliminary design of space-grid structures. Hajela and Berke (1991) developed a Hopfield network for the optimization design of trusses. Park and Adeli (1995) developed a neural dynamics model for the design optimization of steel structures.

This paper provides an overview of neural network basics and presents the development of a neural network system for the depth prediction of singly-reinforced concrete beams in ordinary housing buildings. The system was validated by data and criteria provided by the American Concrete Institute (ACI-318 1995). The generalization and fault-tolerance properties of the proposed system were demonstrated using a case study.

OVERVIEW OF NEURAL NETWORKS

A neural network consists of a network of interconnected nodes, a learning method, and an information recalling method (Adeli and Park 1995). The network consists of layers of nodes linked by weighted interconnections (Figure 1). An input node has an activation function that evaluates inputs and generates an output transmitted as an input to other nodes. An input layer receives user's input while an input layer emits computed output to the user. Between the input and output layers lie the hidden layers.

The weighted interconnections between nodes can be excitatory or inhibitory. An excitatory connection increases the input value to a connected node. An inhibitory connection decreases the input value to a connected node. The interconnections between nodes can be intra-layer (lateral connection), inter-layer, and recurrent connections (Figure 1).

The learning method adjusts interconnection weights to produce a desirable output from a given input. The learning scheme can be supervised and unsupervised. In the supervised learning, a direct comparison of the network output is performed with the desired output. Backpropagation is an example of supervised

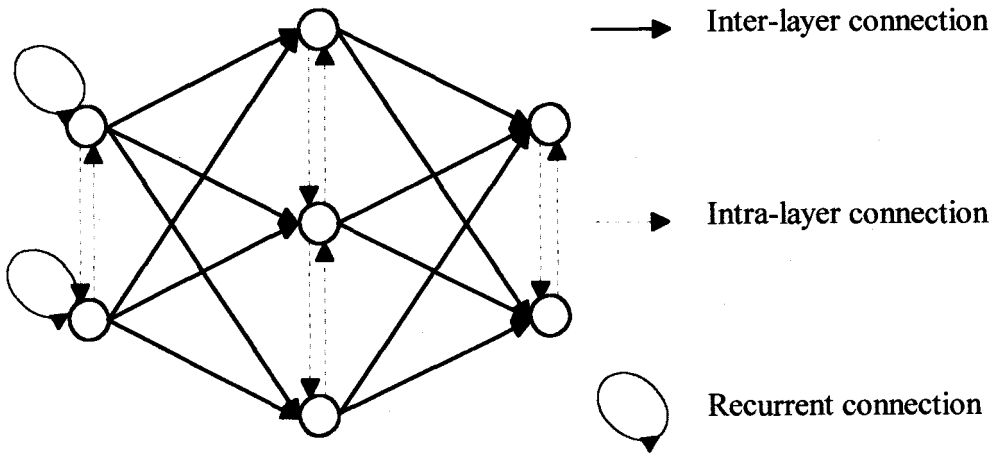

Fig. 1. Topology of a neural network and three connection types (Adeli and Park 1995)

learning algorithms (Rumelhart & McClelland & Williams 1986, Adeli & Hung 1993, Adeli & Hung 1993). In unsupervised learning, the learning goal is not defined and the network is expected to create categories from the correlation of the input data, and to produce output corresponding to the input category. Counterpropagation is an example of unsupervised learning algorithms [Hecht-Nielsen 1987, Hecht-Nielsen 1988]

The recalling methods, which define the way the network output are obtained from the given input, can be feedforward or feedback. During the feedforward recall, an input is passed through nodes and weights, and the corresponding output is produced in one pass. In the feedback recall mechanism, an input is passed through the nodes and weights, and then an output is produced which is fed back into an input or a specific layer until there is no change in the weights.

This paper employs a backpropagation neural network for the depth prediction of singly-reinforced concrete beams. A detailed discussion of neural networks can be found elsewhere (Werbos 1984, Pao 1989, and Simpson 1991). However, a brief description of some basic background of backpropagation neural networks is included here to benefit the unfamiliar reader.

BACKPROPAGATION NEURAL NETWORKS

Backpropagation neural networks have been applied to a wide variety of practical problems and have proven their ability to model nonlinear relationships such as those encountered in the design of singly-reinforced concrete beams. Figure (2) shows a typical backpropagation neural network with one hidden layer. Each connection is associated with a certain weight W_{ji} and each neuron is associated with a bias term, called threshold θ_j. Values for both W_{ji} and θ_j are determined for a backpropagation neural network during the training phase.

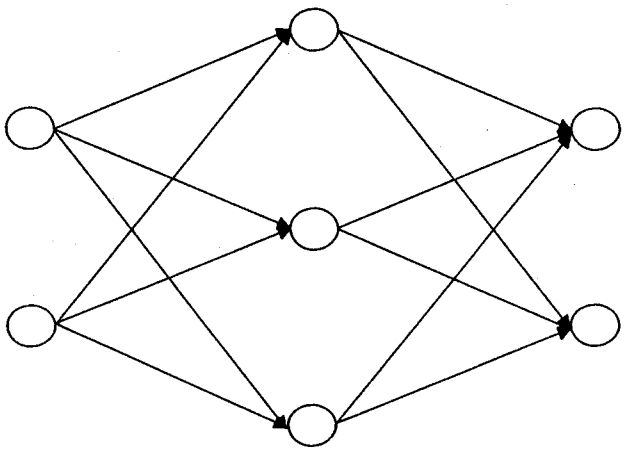

Fig. 2. Typical backpropagation neural network

The output O_j of each neuron (j) in either the hidden layer or the output layer is given by the following sigmoid transfer function (Pao 1989)

$$O_j = \frac{1}{1 + \exp(-h_j)} \tag{1}$$

where the total input h_j of neuron j is determined using the following equations:

$$h_j = \sum_{i=1}^{NIN} x_i W_{ij} - \theta_j \qquad (\forall j \in \text{hidden layer};\ i \in \text{input layer}) \tag{2}$$

$$h_j = \sum_{i=1}^{NHN} x_i W_{ij} - \theta_j \quad (\forall j \in \text{output layer}; i \in \text{hidden layer}) \qquad (3)$$

where NIN and NHN represent the number of neurons in the input and hidden layers, respectively. Thresholds (θ_j) are usually omitted from Eqs. 2 and 3 because they are always treated as connections to an input neuron that is permanently clamped at -1.

Backpropagation learning algorithm is based on the error reduction between the actual output of a processing element and its desired output by modifying incoming connection weights. This is done through an iterative process during which the network modifies the initial random values of W_{ji} to converge the results to the desired (known) output. The algorithm can be thought of as a minimization problem in which the network error, E, is minimized. The network error, E, is defined as the average of output square errors over all the training examples (input-output pairs), and is computed using the following formula:

$$E = \frac{1}{2P} \sum_{j=1}^{P} \sum_{i=1}^{NON} (D_{ij} - O_{ij})^2 \qquad (4)$$

where NON is the number of neurons in the output layer, P is the number of training examples, D_{ij} is the desired output value of the ith processing element in the output layer, pertaining to the jth training example; and O_{ij} is the computed output value of the ith processing element in the output layer, pertaining to the jth training example.

NEURAL NETWORK FOR BEAM DEPTH PREDICTION

The neural network presented here was developed using the design procedure established by the American Concrete Institute (ACI-318 1995). In this procedure, the effective depth, d, of singly-reinforced concrete beams is computed using the following equation (Fintel 1985):

$$d = \sqrt{\frac{M_u}{0.9 \rho b F_y (1 - 0.59 \rho \frac{F_y}{f_c'})}} \qquad (5)$$

where f'_c = concrete compressive strength (MPa), F_y = reinforcing steel yield strength (MPa), ρ = desired reinforcing steel ratio, b = beam width (mm), and M_u = maximum factored bending moment (kN.m).

The desired reinforcing steel ratio ρ must be within the following limits (Fintel 1985):

$$\rho_{min} \leq \rho \leq \rho_{max} \tag{6}$$

where:

$$\rho_{max} = 325 \frac{f'_c}{F_y(600 + F_y)}$$
$$\rho_{min} = \frac{1.4}{F_y} \tag{7}$$

Equation 5 shows that an accurate computation of the beam effective depth can only be performed when all of the design parameters are known. When a beam depth is sought, the actual design is not yet known. Therefore, beam depth prediction can only be an expert's estimate. The accuracy of this estimation is highly dependent upon the expert's experience and technical knowledge.

The experience and the knowledge of structural experts can be simulated by a neural network trained to learn the many possibilities of relating a beam effective depth to the design variables previously mentioned (f'_c, F_y, b, and M_u). In other words, the neural network is trained to learn the different design alternatives used locally by structural engineers.

The proposed neural network system is to be used by both structural and non-structural engineers, who are able to provide the following descriptive information:

1. Is the beam span length very low, low, medium, high, or very high (Table 1) ?

2. Is the beam load intensity very low, low, medium, high, or very high (Table 1) ?

3. Is the beam equal to 200, 250, or 300 mm ? Beam widths are usually prescribed by the width of the walls.

Table 1. Value Range of User-Provided Information

Design Variable	Description	Value Range
Span	Very Short	Span <= 3 m
Span	Short	3 < Span <= 4
Span	Medium	4 < Span <= 5
Span	Long	5 < Span <= 6
Span	Very Long	Span >= 6
Load	Very low	Load <= 20 kN/m
Load	Low	20 < Load <= 35
Load	Medium	35 < Load <= 50
Load	High	50 < Load <= 65
Load	Very High	Load > 65 kN/m
Beam Width	Width = 200 mm	
Beam Width	Width = 250 mm	
Beam Width	Width = 300 mm	

Therefore, the user-provided information for the network should include the beam span, load, and width. Even when the correct values of concrete strength f'_c and the reinforcing steel strength F_y are missing from the input, the proposed neural network can still make adequate beam depth predictions (Case Study). This is due to the fact that neural networks have good generalization and fault-tolerance properties.

Network Architecture

In developing a neural network for the depth prediction of singly-reinforced concrete beams, a backpropagation neural network with one hidden layer was used. The neural network, which has been chosen for the present research, has 13 input neurons and one output neuron. Table (2) summarizes the input and output components of the neural network. Figure (3) shows the selected backpropagation neural network.

The input components take binary values (i.e., either 0 or 1). The set of components 1-5 represents whether the beam span length is very low, low, medium, high, or very high; the set of components 6-10 represents whether the beam load intensity is very low, low, medium, high, or very high; the set of components 11-13 represents

whether the beam width is equal to 200, 250, or 300 mm. The components in each set are not mutually independent. In other words, if a component in a set takes the value of 1.0, all the other components in the set will take the value of 0. As an alternative scheme, a continuously valued component could be used for representing each set of factors. Though this scheme reduces the amount of computation, it is not preferred because the separability or distance between different inputs is reduced. In contrast, the output component, which represent beam depth, takes continuous values between 0 and 1.

The number of hidden neurons is chosen to be 10 on a trial-and-error basis. It is desirable to have as few hidden neurons as possible, but too few hidden neurons, the network will not converge during the training process. We started with 5 hidden neurons and increased by 1 each time the network did not converge to a desired level.

Table 2. Neural Network Input Output Components

Component Type	Component Number	Attribute	Description	Value
Input	1	Span	Very Short	Binary (0 or 1)
	2	Span	Short	Binary (0 or 1)
	3	Span	Medium	Binary (0 or 1)
	4	Span	Long	Binary (0 or 1)
	5	Span	Very Long	Binary (0 or 1)
	6	Load	Very low	Binary (0 or 1)
	7	Load	Low	Binary (0 or 1)
	8	Load	Medium	Binary (0 or 1)
	9	Load	High	Binary (0 or 1)
	10	Load	Very High	Binary (0 or 1)
	11	Beam Width	Width = 200 mm	Binary (0 or 1)
	12	Beam Width	Width = 250 mm	Binary (0 or 1)
	13	Beam Width	Width = 300 mm	Binary (0 or 1)
Output	1	Beam Height		Continuous (0-1)

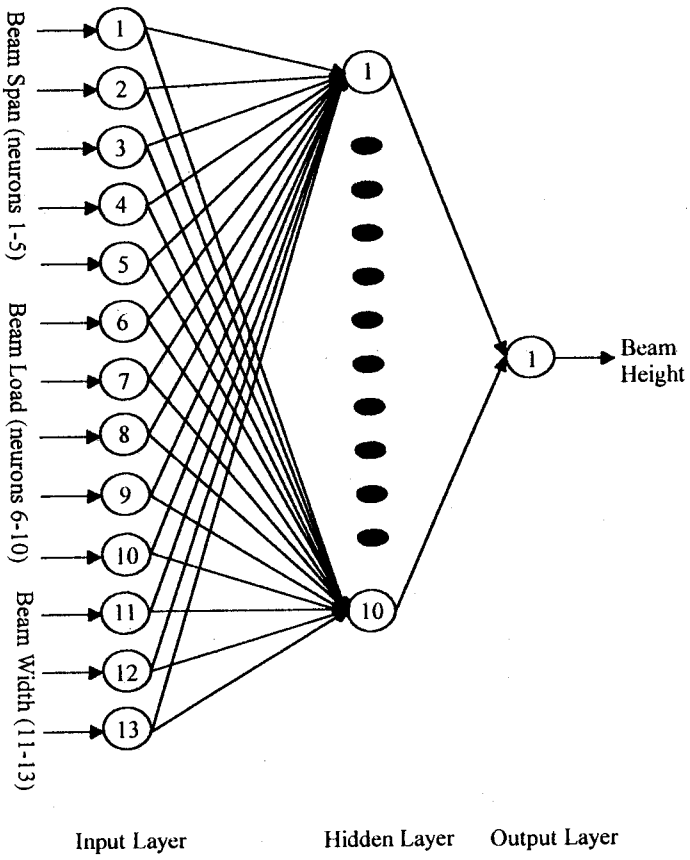

Fig. 3. Selected backpropagation neural network

Network Input and Output Data

The next step in the development of a neural network is to obtain good training and testing examples. A program was written in FORTRAN to generate the training and testing examples. The program algorithm, whose flowchart is shown in Figure (4), can be summarized as follows.

Step 1

The program starts by setting the concrete strength (f'_c) equal to 20 MPa and the reinforcing steel strength F_y equal to 300 MPa.

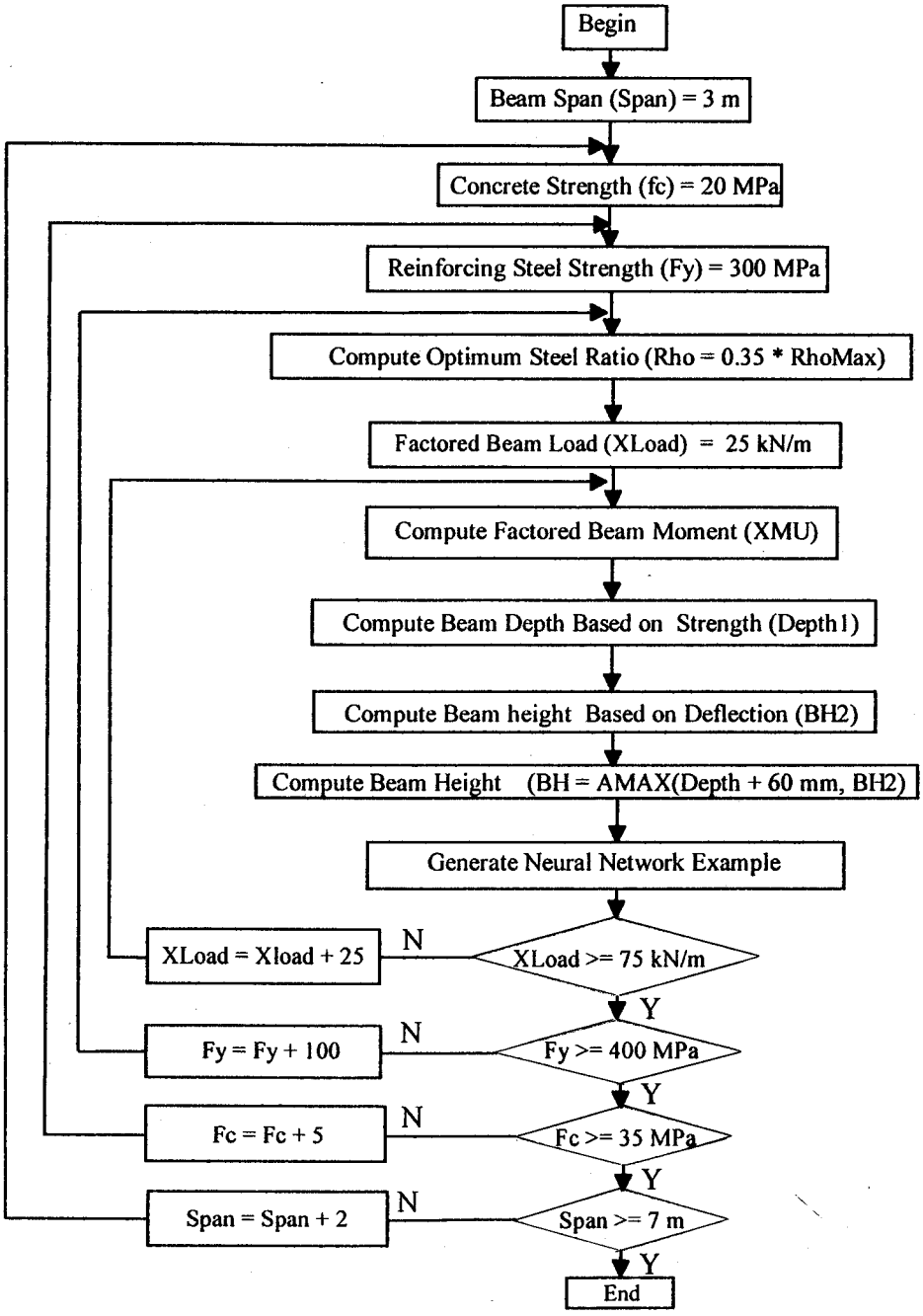

Fig. 4. Flowchart of neural network input generation program

Step 2
The desired reinforcing steel ratio ρ is computed using the following equation (McCormac 1993):

$$\rho = 0.5 * \rho_{max} \tag{8}$$

In view of both economy and deflection control, the use of reinforcing steel ratios in the range of $0.5\ \rho_{max}$ (or $0.18\frac{f_c'}{F_y}$) will yield reasonable results (McCormac 1993).

Step 3
The first beam span length is set equal to 3 meters.

Step 4
The first beam factored load is set equal to 20 kN/m.

Step 5
The factored beam moment XMU is computed using the following equation:

$$XMU = \frac{XLoad * Span * Span}{8} \tag{9}$$

Eq. 9 defines the maximum positive moment for a simply-supported beam (conservative case).

Step 6
The beam effective depth is computed using Eq. 8. The beam depth BH is determined by adding the concrete cover (60 mm) to the beam effective depth.

Step 7
A neural network example is generated. The example consists of 13 input components and one output component. The beam depth has to be scaled to values in the range 0-1. This is necessary because the sigmoid transfer function modulates the output values between 0 and 1. In this study, the following expression was used to normalize the beam depth:

$$\text{Normalized Value} = \frac{\text{Actual Value} - \text{Minimum Value}}{\text{Maximum Value} - \text{Minimum Value}} \tag{10}$$

Step 8
Steps 3 through 7 are repeated for each value of beam width, load, and span.

Network Training

The network training and testing were performed using the neural network simulator, BrainMaker (BrainMaker 1993). A set of 56 training examples were used to train the neural network. The training process was continued until the mean training error reached the value of 3.0 %. Table (3) summarizes the mean and the standard deviation of the training error. The results shows that the network adequately learned the training examples. The training time was about 15 minutes on a Pentium 133.

Table 3. Training, Testing, and Case Study Error Analysis

Phase	Mean Error (%)	Standard Deviation (%)	Maximum Error (%)
Training	4.1	2.7	12.9
Testing	5.7	5.0	18.5
Case Study	8.1	5.0	18.0

Network Testing

Another set of 19 examples was used to validate the developed neural network. These examples were unknown to the network since they were not used during the training phase. Table (3) summarizes the mean and the standard deviation of the testing error. These results show that the network beam depth predictions were acceptable.

PERFORMANCE EVALUATION OF NETWORK

A trained network should be capable of generalizing the governing rules to accurately determine an output from new (not previously introduced) inputs. In order to verify the generalization and fault-tolerance properties of the developed system and to evaluate its performance, a case study was conducted. The case study involved 878 new cases of beam depth predictions. The beam depths obtained using the ACI design method were compared with those predicted by the neural network.

Figure (5) shows the error histogram between the beam depths obtained using the neural network and those obtained using the ACI design method. The results show that the neural network predictions were adequate. The mean error between the beam depths provided by the ACI method and those provided by the neural network was found to be equal to 9.2 % while the maximum error was about 21%..

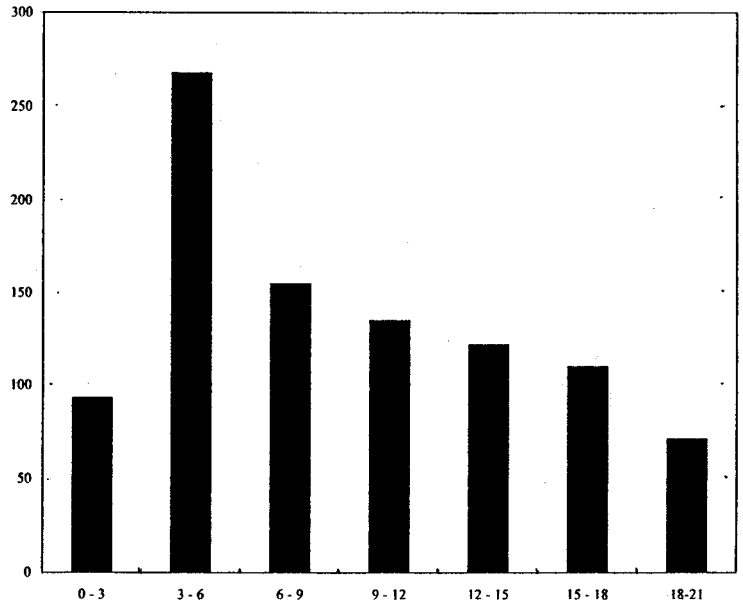

Fig. 5. Case study error histogram

Table (4) shows a portion of the case study results. The table shows the concrete strength, reinforcing steel strength, beam width, beam span, and beam loading. The table also shows the beam depths obtained using both the ACI design method and the neural network. The last column of the table summarizes the percentage difference or error between the beam depths obtained using the ACI design method and those obtained using the proposed neural network.

Even though the values of beam spans, concrete strengths, and steel strengths used in the Case Study were different from those used during the training and testing phases, the network was still able to make good beam depth predictions. This shows that the proposed neural network possess good generalization and fault-tolerance properties.

Table 4. Case Study Results

Span (m)	Concrete Strength (MPa)	Steel Strength (MPa)	Beam Load (kN/m)	Beam Width (mm)	Beam Height		Percent Error (%)
					ACI Method (mm)	Neural Network (mm)	
6.2	25	300	40	300	480	479	0
5.2	25	300	85	250	623	641	3
5.4	20	300	55	200	648	617	5
4.2	25	300	85	300	475	484	2
4.4	25	400	55	250	461	416	10
4.6	30	300	25	200	336	310	8
3.2	25	400	55	300	326	315	4
3.4	20	300	25	250	283	274	3
7.2	25	400	70	200	888	878	1
6.4	25	300	75	250	711	696	2
6.6	25	400	40	200	634	592	7
5.8	25	300	70	300	580	606	4
5.6	25	300	45	250	501	444	11
4.8	25	400	85	200	668	633	5
5.0	30	300	40	300	369	331	10
3.6	30	300	55	200	380	346	9
7.6	20	300	85	300	899	930	3

CONCLUSIONS

In this study, a neural network system for the depth prediction of singly-reinforced reinforced concrete beams was developed. The study showed that that the neural network system performed well in predicting the depth of singly-reinforced concrete beams. The following summarizes the findings of the study.

1. The presented neural network offered a systematic procedure that predicted beam depths closer to those obtained using the ACI design method.

2. The network was able to adequately learn from the training examples with an average absolute error equal to 3.0 %.

3. The network showed a good generalization and fault-tolerance capabilities, and was able to predict beam depths with an average absolute error of 9.2% for 878 design cases.

4. Similar neural networks can be developed for different sets of data related to other design environment

REFERENCES

ACI-318, 1995. Building Code Requirements for Reinforced Concrete, American Concrete Institute, Detroit, USA.

Adeli, H. and Hung S. L., 1993. A Fuzzy Neural Network Learning Model for Image Recognition. Integrated Computer Aided Engineering, Vol.1, No.1, pp. 43-55.

Adeli, H. and Hung S. L., 1993. A Concurrent Adaptive Conjugate Gradient Learning Algorithm on MIMD Share-Memory Machines. Journal of Supercomputer Application, Vol. 7, No. 2, pp. 155-166.

Adeli, H. and Park S. L., 1995. A Neural Dynamics Model for Structural Optimization - Theory. Computers and Structures, Vol. 57, No. 3, pp. 383-390.

BrainMaker, 1993. Neural Network Simulator, California Scientific Software, Nevada City, CA 95959, USA.

Fintel, M., 1985. Handbook of Concrete Engineering, 2nd Edition, Van Nostrand Reinhold Company, New York, New York 10020, USA.

Hajela, P. and Berke, L., 1992. Neural Networks in Structural Analysis and Design: An overview. Computing Systems in Engineering, 3 (1-4), pp. 525-538.

Hetch-Nielsen, R., 1987. Counterpropagation Networks. Application Optics, Vol. 26, pp. 4979-4985.

Hetch-Nielsen, R., 1988. Application of Counterpropagation Networks. Neural Networks, Vol. 1, pp. 131-139.

Liu, X. and Gan, M., 1991. A Preliminary Design Expert System (SPRED-1) Based on Neural Networks. Artificial Intelligence in Design, Butterfield-Heinman Ltd., Oxford, England, pp. 785-800.

McCormac J. C., 1993. Design of Reinforced Concrete. Harper-Collins College Publishers, New York, NY 10022, USA.

Pao, Y. H., 1989. Adaptive Pattern Recognition and Neural Networks. Addison-Wesley Publishing Company, Reading, Massachusetts, USA.

Park, H. S. and Adeli, H., 1995. A Neural Dynamics Model for Structural Optimization-Application to Plastic Design of Structures. Computers and Structures, Vol. 57, No. 3, pp. 391-399.

Rumelhart, D.; Hinton, G.; and Williams, R., 1986. Learning Internal Representation by Error Propagation. in Parallel Distributed Processing, Vol. 1, pp. 318-362 (Edited by Rumelhart, D.; McClelland, J.; and the PDP Research) MIT Press, Cambridge, Massachusetts.

Simpson, P.K., 1991. Artificial Neural Systems, Foundations, Paradigms, Applications, and Implementations. Pergamon Press.

Werbos, P., 1984. Beyond Regression: New Tools for Prediction and Analysis in Behavioral Sciences. Ph.D. Thesis, Harvard University, USA.