JKAU: Eng. Sci., **vol. 12 no. 1,** pp. 53-67 (1420 A.H. / 1999 A.D.)

Artificial Neural Networks in Structural Engineering: Concept and Applications

Abbes Berrais
Construction Department
Abha College of Technology, Abha, Saudi Arabia

Abstract. Artificial neural networks are algorithms for cognitive tasks, such as learning and optimization. They have the ability to learn and generalize from examples without knowledge of rules. Research into artificial neural networks and their application to structural engineering problems is gaining interest and is growing rapidly. The use of artificial neural networks in structural engineering has evolved as a new computing paradigm, even though still very limited.

The objective of this paper is to introduce the concept, theoretical background, advantages, and shortcomings of artificial neural networks technology. Next, some recent applications in structural engineering are briefly reviewed. Then, a conclusion is drawn.

1. Introduction

Artificial neural networks are a family of numerical learning techniques. They consist of many nonlinear computational elements that form the network nodes, linked by weighted interconnections. Research into artificial neural networks (ANNs) is trying to model the human brain neurons and their processes. The human nervous system is composed of a vast number of single, interconnected cellular units, the *neurons*[1]. Neural network models are algorithms for cognitive tasks, such as learning and optimization. They have the ability to learn and generalize from examples without knowledge of rules. The pioneering work in this field is attributed to McCulloch and Pitts[2]. Recently, ANNs are gaining interest in civil engineering. They have been used to solve many structural analysis and design problems.

This paper introduces the concept, theoretical background, benefits, and shortcomings of artificial neural networks technology. Then, the architecture and mathematic learning techniques used to train multi-layer back-propagation neural networks are briefly described. Next, some prototype applications of ANNs to structural engineering are briefly reviewed. Then, a conclusion is drawn.

2. Artificial Neural Networks

From a computational point of view, an ANN consists of a large number of interconnected processing units. Each processing unit or node, modeled as neuron, receives input from other units to which it is connected, carries out a process (or computation), and transmits the output to other processing units. Each node (processing unit) in a neural network performs a single computation, independently of the other nodes. Thus, neural networks have a parallel structure, which allows them to take advantage of parallel processing computers.

The typical architecture of a biological neuron is as shown in Fig. 1. The central part is called the cell body. From it, project several extensions, the *dendrites,* as well as a single tubular fiber, the *axon,* which in turn, ramifies into a number of small branches. The dendrites serve as receptors for signals from adjacent neurons. The role of the *axon* is the transmission of the neural activity to other nerve cells. The *synapses* are the junctions between the neurons.

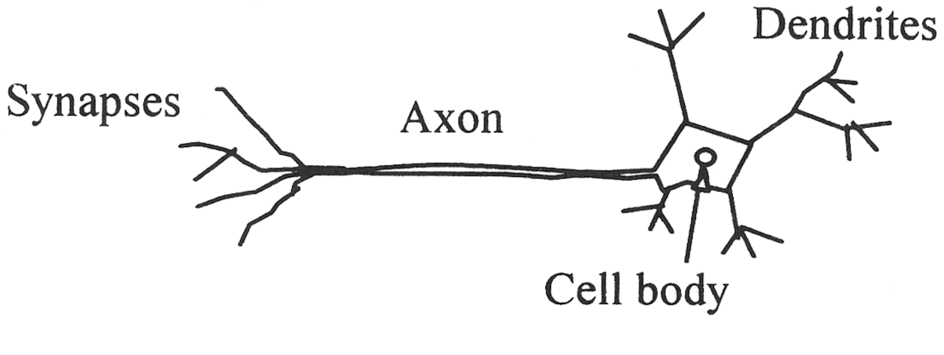


FIG. 1. Schematic view of a biological neurone.

ANNs have been applied to a number of problems with variable success[1,3,4]: speech recognition, image processing, pattern recognition, classification, optimization problems, robotics and control, and medical and commercial applications. Problems that are most suitable for ANN applications have the following characteristics[5]:

* Applications are data-intensive and depend on multiple criteria
* Problem area is rich in historical data or examples
* Data set is incomplete, noisy, and/or contains errors

* The function to determine solutions is unknown or tedious.

3. Artificial Neural Network Architecture

Different ANN topologies exist, the well known are[6,7]: Competitive learning network, the Boltzmann machine, the Hopfield network, and the Back-propagation (multi-layered) network. The latter type gets its name from the way it learns, by back-propagating the errors seen at the output nodes. The back-propagation network model is the most widely used by most neural network application developers.

A typical example of a back-propagating network architecture is shown in Fig. 2, where the processing units in this network are arranged in layers. Each neural network has an input layer, an output layer, and a number of hidden layers. The latter compute complicated associations between patterns, and the propagation takes place in a feed-forward manner, from the input layer to the output layer.

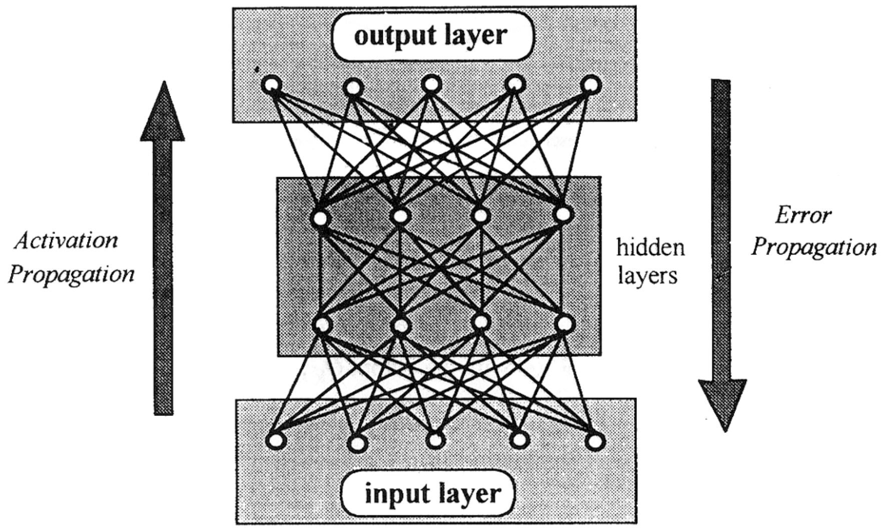


FIG. 2. Architecture of a back-propagation neural network.

Associated with each connection between two units is a numerical value W_{ij}, which represents the weight of that connection[8]: W_{ij} = weight of connection between units i and j. These weights are modified during the training of the neural network in an iterative process. When the iterative process has converged, the collection of connection weights captures and stores the information present in the example used in its training.

A feed-forward network computation in a back-propagation neural network proceeds as follows:

1 – The units in the input layer receive their activations in the form of an input pattern. This initiates the feed-forward process.

2 – The processing units in each layer receive outputs from other units and perform the following computations (as shown in Fig. 3):

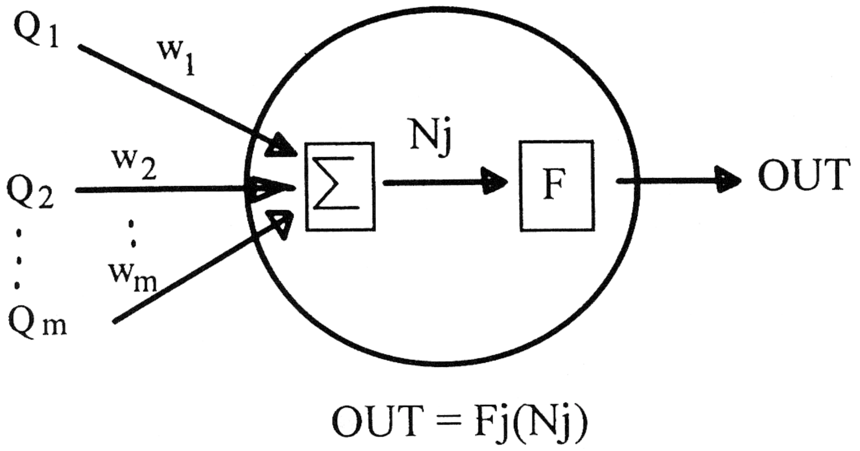


FIG. 3. Schematic representation of processing within an artificial neuron.

a) Compute their net input N_j

$$K_j = \sum_{k=1}^{M} w_{jk} Q_k \tag{1}$$

in which Q_k = output from units branching on unit j,

w_{jk} = weight associated with the output Q_k ; and

M = number of units branching on unit j.

b) Compute their activation values from their net input values

$$a_j = F_j(N_j) \tag{2}$$

F_j is usually a sigmoid function (activation function). Both the function and its derivatives are continuous everywhere.

$$F_j = \frac{1}{1 + e^{-(N_j - \theta_j)}} \tag{3}$$

in which θ_j is threshold value of the unit j or the solution bias. θ_j can improve the network accuracy and can be assigned a small random value and change it during the training of the network.

c) Compute their outputs from their activation values.

3 – The output values are sent to other processing units along the outgoing connections.

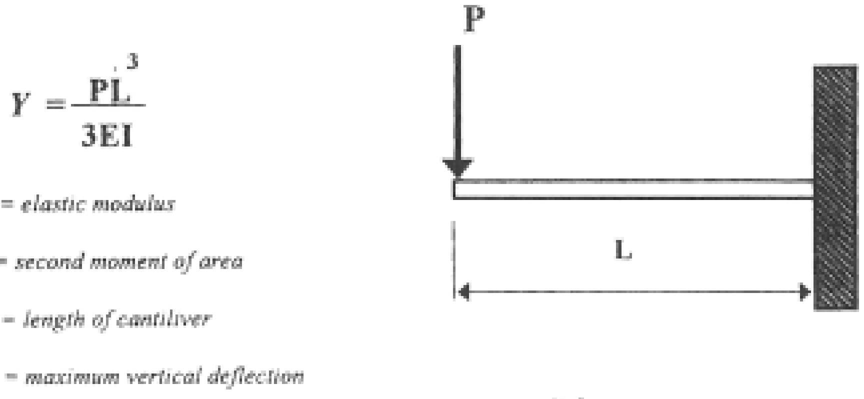
4 – This process continues until the processing units in the output layer compute their derivation values. These activation values are the output of the neural computations.

The training of a back-propagation neural network can be envisaged as fellows: during the forward propagation step, the input patterns generate a forward flow of signal from the input layer to the output layer. The error of each output node is then computed from the difference between the computed output and the desired output. The second stage involves adjusting the weights in the hidden and output layers in order to minimize the difference between the actual and desired output. This training is classified as a supervised training algorithm[6]. Thus, the network learns how to respond to patterns of data presented to it. Different learning algorithms exist for training neural networks. The most commonly used one is the Delta rule[9] (or back-error propagation algorithm), which can be applied to adjust connections weights so that the network can predict unknown examples correctly.

The behavior of an ANN depends on its topology, the weighting system used, and the activation function[9]. The choice of the activation function is based on the types on input and outputs desired and on the learning algorithm to be used. In addition , there is no direct way of determining the most appropriate number of nodes (processing units) to include in each hidden layer. The choice of the number of layers, the size of each layer, and the way each layer is to be connected is left to the developer to experiment with. Increasing the number of hidden layer nodes makes the network more powerful, but the training time and operation of the network will be increased. A general rule is to start with a simple network with one hidden layer, and then increasing the hidden layers and observe the performance of the network until the best architecture of the network is reached.

ANN could be developed using special development tools. For example, several software programs are spreadsheet-based. Other software tools are designed to work with expert systems as hybrid development tools. Also, there are some neural network shells which exist for commercial use. The advantage of these shells is that programming experience is not required and programming time is reduced. For example NeuroShell[10] is an ANN development environment based on back propagation and is best for stand-alone systems. References[10 and 11] list some of the used ANN development systems. ANN can also be implemented directly as semiconductor circuits, or using electro-optical technology[4].

Goh[12] has presented a simple example of the application of ANN in structural engineering. The example is included here for clarification purpose. This example involves the computation of the deflection of a cantilever beam subjected to a point load as shown in Fig. 4. The neural network is composed of an input layer with 3 input nodes (*P, 1/EI,* and *L*). The input values were normalized to be in the range 0-1. The output layer contains one single node which represent the normalized deflection. The hidden layer contains 3 nodes. During the training session, 45 sets of input data was used[12], each set comprised randomly selected values of *P, 1/EI,* and *L*. The normalized deflection was used as the desired output value. Sample training patterns, testing data, and the comparison of the theoretical and predicted deflections can be found in reference[12].



4. Structural Engineering Applications

Research on the application of ANNs to civil engineering problems is growing rapidly. The use of ANNs in structural engineering has evolved as a new computing paradigm, even though still very limited. It has been applied for different areas such as: structural design, material behavior modeling, damage assessment, structural analysis, and FE analysis. In the following section some prototype applications in structural engineering are briefly described.

Structural Design

Mukherjee and Deshpande[13] have used ANNs for modeling an initial design process. They developed a multi-layer feedforward network model for the initial design of reinforced concrete rectangular single-span beams. They used two hidden layers and 8 nodes in the input layer. One node each was selected for span, dead load, and live load. For type of steel, one node each was assigned for the three grades of steel. Similarly, for the two possible grades of concrete, M-15 and M-20, two nodes were provided. The output layer consists of a node each for area of tensile steel, depth, width, cost of beam per meter, and the moment capacity. The network has been tested to predict the probable values of different beam parameters for new examples.

Abdallah and Stavroulakis[14] developed a back-propagation neural network model that uses experimental steel connections data for an estimation of the mechanical behavior of semi-rigid steel connections. Two types of steel structure connections have been considered: single-web-angle beam-to-column and single-plate beam-to-column connections. The design variables used for the training and testing of the neural network are: number of bolts, web angle, thickness of the plate, and length. These design variables were taken as input data, and the measured moment-rotation curve is considered as the output data for the neural network.

Chen *et al.*[15] developed a back-propagation neural controller for active control of structures under dynamic loading. Their model consists of two parts: Neural Emulator Network to represent the structure to be controlled; and Neural Action Network to determine the control action on the structure. The neural emulator network is constructed with a four-node input layer, a four-node hidden layer, and a one-node output layer. The input data to the network are displacements on the roof and ground floor, the output of the network is the displacement on the roof at the next time step.

StructNet[16] a neural network model to select the most effective structural member materials given a building project's attributes. StructNet was developed using the THINK C compiler. The input for StructNet include construction project parameters, which include information about the building design and the project constraints. The output for the network is the percentage of times a particular type of beam, column, or slab was used for similar training project. The neural network contains 15 input nodes, 1 hidden layer with 15 nodes, and 14 output nodes.

Kang and Yoon[17] developed a two-layer neural network (single-layer perceptron) for truss design. The type of truss used in their model is a simple one and the problem considered is the selection of economical member areas that will satisfy stresses requirements. The input data to the network are horizontal and vertical loads at the truss joints. The output of the network is the design areas of all members of the truss.

NEUROFLOOR[18] is a neural network system to assist building designers and engineers in the design of steel bar joist flooring systems. The input parameters include joist span length; joist spacing; live load. The output parameter for the network are the steel bar joist type and size. NEUROFLOOR was developed using the software package NeuroShell 4.1[18], which is a back-propagation, single hidden-layer, feed-forward network.

Material Behavior Modeling

Ghaboussi *et al.*[8] developed a back-propagation neural network for modeling the behavior of concrete in a state of plane stress under monotonic biaxial loading and compressive uniaxial cyclic loading. The processing units in the input and output layers of the neural network represent stresses, strains and their increments. The neural network has six units in the input layer and two units in the output layer, and the six input units are two stresses, two strains, and two stress increments; and the output units are two strain increments.

Structural Damage Assessment

Elkordy *et al.*[19, 20] developed a neural network system for structural damage monitoring. The network was to detect the damage and to determine its class. The network was composed of eight input nodes, nine hidden nodes, and two output nodes. The neural network model developed can diagnose damage based on simultaneous consideration of different data sets representing different signatures of the structures.

Kirkegaard and Rytter[21] developed a Multilayer Perceptron (MLP) network for damage assessment of cracked straight steel beams based on vibration measurements. The MLP network was trained using the back-propagation algorithm, and the 5 lowest modes of the bending natural frequencies were used as training tests. A four layers neural network was used with 5 input nodes, 7 nodes in each of the two hidden layers and 2 output nodes. The output nodes give the crack location and size.

Stephens and VanLuchene[22] explored the use of ANNs for the damage assessment in determining the safety condition of a structure following a seismic event. The input to the neural network consists of three types of damage indices: maximum displacement, cumulative dissipated energy, and stiffness degradation. The network contained single middle layer with seven nodes per layer. The neural network model was developed using the computer program NNICE[23].

Begum *et al.*[24] developed an ANN system for the testing of concrete surfaces. FEM has been used to simulate defective components subject to impact-echo systems. The output of the FEM results are interpreted using the neural network which predicts the depth range of the crack in the concrete surface. The input to the network is the amplitudes at equidistant sampling points on the amplitude-frequency spectrum. The output of the network is the overall probability of the defect interface occurring within a given depth range.

Watson *et al.*[25] used ANN for pile integrity testing. The network is composed of an input layer of 102 processing units, one hidden layer with 15 processing units. The neural network detects a pile shape directly from its spectrum. The output from the network was the pile position of fault and length. The velocity response of the pile head to a hammer blow was generated by the FEM using the LUSAS software[25].

Takahashi and Yoshioka[26] developed a neural network system for fault identification in a beam structure. The network used has 4 layers with 5 input nodes, 5 nodes and 6 nodes in each hidden layers and 2 output nodes. The network was trained with numerical values of the relative changes of the lowest five natural frequencies of the beams.

Szewczyk and Hajela[27] developed a counter-propagation ANN for structural damage detection. The network was used to model the inverse mapping between a vector of the stiffness of individual structural elements and the vector of the global static displacements under a testing load. The network has one input layer, one hidden layer, and one output layer. The input to the network is the static displacement vector and the output is the Young's moduli.

Structural Analysis

Flood and Kartam[28, 29] introduced the concepts, theoretical limitations, and efficiency of ANN with reference to a simple structural analysis example. Issues such as: number of hidden layers and hidden nodes, distribution and format of training patterns, validation of network, and processing speed have been investigated. They concluded that the success of a neural network implementation is dependent on the followings: quality of the data used for training, type and structure of the network, method of training, and the way in which both input and output data are structured and interpreted.

Gunaratnam and Gero[30] studied the effect of the representations used to describe the patterns that the network learns to map on the performance of multilayer feedforward networks in structural engineering applications. In their model they have used dimensional analysis technique to define the space in which learning is to take place. Load position-bending moment patterns in beams has been used as an example for their model. They have noticed that an improvement in the performance of the network compared with other researchers has been made, and they recommended that their approach can be applied to other domain where dimensional analysis is applicable.

Anderson *et al.*[31] developed an ANN to predict design moment resistance and secant stiffness for minor axis steel connections. The input variables to the network included: depth of section, flange and web thickness for column, flange breadth and depth of section for beam, number of bolts and plate thickness for the connection plate. The outputs are the ultimate moment and the joint rotation.

The network is composed of 7 nodes input layer, 7 and 4 nodes in the first and second hidden layers respectively, and 2 nodes output layer.

Adeli and Park[32] compared the performances of back-propagation and counter-propagation (CPN) learning algorithms for structural engineering applications. The CPN is a combination of supervised and unsupervised (self-organizing) mapping neural network, whereas the back-propagation is a supervised learning algorithm. They concluded that the CPN can improve the convergence speed of neural network learning algorithms. Two examples have been used to test their CPN model: a simple reinforced concrete beam, and a simply supported rectangular plate, both subjected to a unit concentrated load. The CPN model was developed in FORTRAN on a Cray YMP8/864 machine using CF77 compiler.

Rogers[33] proposed guidelines for designing and training a neural network to simulate a structural analysis program and to reduce the amount of time it takes an optimization process to converge to an optimum design. These guidelines include the selection of training pairs and determining the number of nodes on the hidden layers. He investigated the problem of optimizing the shape of a beam to minimize the weight while satisfying stress constraints. The network was created with 5 nodes input layer and 41 nodes on the output layer. The network was developed using a tool called NETS/PROSS[33].

Jenkins[34] investigated the application of ANN to structural analysis with reference to their organization, input and output processing, topology, and training. He applied a neural network model to analyze a multi-storey structural frame subjected to gravity and wind loads. A neural network with 4 input nodes and bias node, and a variable number of nodes in the hidden layer and 3 nodes in the output layer was used. The input data to the network comprises: outer bay span, storey height, and beam and column second moments of area. The output nodes represent the structure displacements and stress resultants.

Finite Element Analysis

Khan *et al.*[35] implemented a neural network system for finite element mesh generation to determine the number of mesh elements generated in a subdomain. A parallelisation method using a multiple instruction multiple data memory architecture has been used to speed up the back-propagation algorithm. The input data to the network are data regarding the geometry of individual element and nodal mesh parameters. The output for the network is the number of triangle finite elements and nodal meshes generated. The neural network was developed using the program NETS 2.01[35].

Goh *et al.*[36] used back-propagation neural network for the multivariate modeling of FEM data. They applied their model for the initial estimate of the maximum wall displacements for braced excavation in soft to medium clays. The data to train the network was derived from extensive finite element parametric studies. The neural network consisted of 7 input variables, 3 nodes hidden layer, and a single node output layer. The input to the network are: excavation width and height, depth to bedrock, soil strength and stiffness. The output from the network is the maximum lateral wall deflection.

Topping and Bahreininejad[37] examined the use of parallel unsupervised recurrent ANN to the partitioning of unstructured adaptive meshes for domain decomposition of FE meshes for explicit time-stepping dynamic analysis. In their approach they partitioned a coarse mesh on the basis of the predicted number of triangular elements which will be generated within each triangle of the coarse mesh after the remeshing. The neural network was implemented using T800 transputers.

5. Summary and Conclusion

In this paper the concept and theoretical background of ANNs have been introduced. ANNs form a family of numerical learning techniques. They can be used to model any nonlinear mapping between variables and can be considered as complementary to the existing conventional programming tools. The novelty of ANNs lies in the way in which the novel computing techniques are able to contribute new and effective solutions to problems that have challenged the power of more conventional methods.

The investigation described in this paper has provided a valuable insight and identified a number of advantages and shortcomings of using ANNs in structural engineering. Some of the advantages of ANNs are:

* The ability to extract information from incomplete and noisy data.
* Acquire experience and knowledge through self training and organization of the knowledge.
* The potential for very fast optimization.
* Their suitability for problems in which algorithmic solutions are difficult to develop or do not exist.
* Suitability of rapid application development

The shortcomings of ANNs can be summarized in the following points:

* ANN are 'black boxes'. They provide solutions and results without being able to explain how they arrive at their solutions.
* ANN are not very good at performing symbolic computations.
* There are no formal techniques for developing ANN applications. Therefore suitable experiments have to be conducted to determine the best ANN architecture and design.

* Accuracy of ANN's performance is dependent to a large extent upon the quality of the training examples, and no guarantee of success in finding an acceptable solution.

* ANN can only provide solutions to the set of problems with which they were trained.

Different ANN applications in structural engineering have been reviewed. Most of the developed ANNs are based on the back-propagation algorithm, primarily due to its simplicity. In addition, many current ANN applications have been developed as stand-alone systems with little integration with other systems, and have not reached the full deployment in the construction industry.

ANNs can be made more powerful systems if they are integrated and used as front-end to expert systems. In particular, this approach is relevant in structural engineering where all successful applications of expert systems provide powerful means for the user to manage and understand the performance of these tools. In the near future, ANNs will have an increased impact and more prototypes will be developed for other civil engineering applications. Despite promising results in various engineering applications, the development of ANNs still requires extensive experimentation, imagination, and human judgment.

References

[1] **Muller, B. J.** and **Reinhardt, J.,** *Neural Networks: An Introduction,* Springer-Verlag, Berlin (1990).

[2] **McCulloch, W. S.** and **Pitts, W.,** A Logical Calculus of the Ideas Immanent in Nervous Activity *Bull. Math. Biophys.* Vol. **5:** 115-133 (1943).

[3] **Werbos, P.,** Backpropagation: past and future, *Proceedings IEEE Int. Conference on Neural Networks,* Vol. **1,** Inst. of Electrical and Electronics Engrs. (IEEE), New York, 343-353 (1988).

[4] **Lisboa, P. G. J.,** *Neural Networks: Current Applications,* Chapman & Hall (1992).

[5] **Dutta, S.,** *Knowledge Processing & Applied Artificial Intelligence,* Butterworth-Heinemann Ltd. (1993).

[6] **Rumelhart, D. E. and McClelland, J. L.,** Parallel distributed processing, Vol. **1,** *Foundations,* MIT Press, Cambridge, Mass (1986).

[7] **Ebhart, R. C.** and **Dobbins, R. W.,** *Neural Network PC Tools,* Academic Press, San Diego, Calif. (1990).

[8] **Ghaboussi, J., Garrett, J. H.** and **Wu, X.,** Knowledge-based modeling of material behavior with neural networks, *J. of Engineering Mechanics,* Vol. **117,** No. **1,** January, 132-152 (1991).

[9] **Hogood, A. A.,** *Knowledge-Based Systems for Engineers and Scientists,* CRC Press (1993).

[10] **Medsker, L.** and **Liebowwitz, J.,** *Design and Development of Expert Systems and Neural Networks,* Macmillan College Publishing Company, New York, USA (1994).

[11] **Reid, K.** and **Zeichick, A.,** Neural Network Resource Guide, *AI Expert,* Vol. **7,** June (1992).

[12] **Goh, A. T. C.,** Experiments with Neural Networks as a Design-Support Tool for Complex Engineering Systems, *Civil Engineering Systems,* Vol. **12,** pp. 327-342 (1995).

[13] **Mukherjee, A. and Deshpande, M.,** Modeling Initial Design Process Using Artificial Neural Networks, *Journal of Computing in Civil Engineering,* Vol. **9,** No. **3,** July, 194-200 (1995).

[14] **Abdalla, K. M.** and **Stavroulakis, G. E.,** A Backpropagation neural network model for semi-rigid steel connections, *Microcomputers in Civil Engineering.* Vol. **10,** No. **2,** 77-87 (1995).

[15] **Chen, H. M., Tsai, K. H., Qi, G. Z., Yang, J. C. S.** and **Amini, F.,** Neural Network for Structure Control, *Journal of Computing in Civil Engineering,* Vol. **9,** No. **2,** April, 168-176 (1995).

[16] **Messner, J. I., Sanvido, V. E.** and **Kumara, S. R. T.,** StructNet: A Neural Network for Structural System Selection, *Microcomputers in Civil Engineering,* Vol. **9,** No. **2,** 109-118 (1994).

[17] **Kang, H. T.** and **Yoon, C. J.,** Neural Network Approaches to Aid Simple Truss Design Problems, *Microcomputers in Civil Engineering,* Vol. **9,** No. **3,** 211-218 (1994).

[18] **Issa, R. R. A.** and **Fletcher, D.,** NEUROFLOOR: A Flooring System Selection Neural Network, *Proceeding of the Fifth Int. Conference on Computing in Civil and Build. Engrg.,* 1125-1132 (1993).

[19] **Elkordy, M. F., Chang, K. C.** and **Lee, G. C.,** A Structural Damage Neural Network Monitoring System, *Microcomputers in Civil Engineering,* Vol. **9,** No. **2,** 83-96 (1994).

[20] **Elkordy, M. F., Chang, K. C.** and **Lee, G. C.,** Neural networks trained by analytically simulated damage states, *Journal of Computing in Civil Engineering,* Vol. 7, No. **2,** April, 130-145 (1993).

[21] **Kirkegaard, P. H.** and **Rytter, A.,** The Use of Neural Networks for Damage Detection and Location in a Steel Member, *CIVIL-COMP93, The Third International Conference in the Application of Artificial Intelligence to Civil and Structural Engineering,* 17th-19th August , B. H. Topping & A. I. Khan (eds), Heriot-Watt University, Edinburgh, UK, 1-9 (1993).

[22] **Stephens, J. E.** and **VanLuchene, R. D.,** Integrated Assessment of Seismic Damage in Structures, *Microcomputers in Civil Engineering,* Vol. **9,2,** 119-128 (1994).

[23] **VanLuchene, R. D.** and **Sun, R.,** Neural Networks in Structural Engineering, *Microcomputers in Civil Engineering,* Vol. **5,** No. **3,** 207-215 (1990).

[24] **Begum, R, Chamberlain, B.** and **Hirson,** A., Integrity Testing of Concrete Surfaces Using Artificial Neural Networks, *CIVIL-COMP95 Developments in Neural Networks and Evolutionary Computing for Civil and Structural Engineering,* **Topping, B. H. V.** (ed.), CIVIL-COMP Press, Edinburgh, UK, 1-6 (1995).

[25] **Watson, J. N., Wan, F. C.** and **Sibbald,** A., The Use of Artificial Neural Networks in Pile Integrity Testing, *CIVIL-COMP95 Developments in Neural Networks and Evolutionary Computing for Civil and Structural Engineering,* **Topping, B. H. V.** (ed.), CIVIL-COMP Press, Edinburgh, UK, 7-13 (1995).

[26] **Takahashi, I.** and **Yoshioka, T.,** Use of Neural Networks for Fault Identification in a Beam Structure, *CIVIL-COMP95 Developments in Neural Networks and Evolutionary Computing for Civil and Structural Engineering,* **Topping, B. H. V.** (ed.), CIVIL-COMP Press, Edinburgh, UK, 15-23 (1995).

[27] **Szewczyk, Z. P.** and **Hajela, P.,** Damage Detection in Structures Based on Features-Sensitive Neural Networks, *J. of Computing in Civil Engineering,* Vol. **8,** No. **2,** 163-178 (1994).

[28] **Flood, I.** and **Kartam, N.,** Neural Networks in Civil Engineering. I: Principles and Understanding, *J. of Computing in Civil Engineering,* Vol. **8,** No. **2,** 131-148 (1994).

[29] **Flood, I.** and **Kartam, N.,** Neural Networks in Civil Engineering. II: Systems and Application, *J. of Computing in Civil Engineering,* Vol. **8,** No. **2,** 149-162 (1994).

[30] **Gunaratnam, D. J.** and **Gero, J. S.,** Effect of representation on the performance of neural networks in structural engineering applications, *Microcomputers in Civil Engineering,* Vol. **9,2,** 97-108 (1994).

[31] **Anderson, D., Hines, E. L., Arthur, S. J.** and **Eiap, E. L.,** Application of Artificial Neural Networks to Prediction of Minor Axis Steel Connections, *The Third International Conference in the Application of Artificial Intelligence to Civil and Structural Engineering,* 17th-19th August, **B. H. Topping & A. I. Khan** (eds), Heriot-Watt University, Edinburgh, UK, 31-37 (1993).

[32] **Adeli, H.** and **Park, H. S.,** Counterpropagation Neural Networks in Structural Engineering, *Journal of Structural Engineering,* Vol. **121,** No. **8,** August, 1205-1212 (1995).

[33] **Rogers, J. L.,** Simulating Structural Analysis With Neural Network, *J. of Computing in Civil Engineering,* Vol. **8,** No. **2,** 252-265 (1994).

[34] **Jenkins, W. M.,** Neural Network-Based Approximations for Structural Analysis, *CIVIL-COMP95 Developments in Neural Networks and Evolutionary Computing for Civil and Structural Engineering,* **Topping, B. H. V.** (ed.), CIVIL-COMP Press, Edinburgh, UK, 25-35 (1995).

[35] **Khan, A. I., Topping, B. H. V.** and **Bahreininejad,** A., Parallel Training of Neural Networks for Finite Element Mesh Generation, *CIVIL-COMP93, The Third International Conference in the Application of Artificial Intelligence to Civil and Structural Engineering,* 17th-19th August, **Topping, B. H.** and **Khan, A. I.** (eds), Heriot-Watt University, Edinburgh, UK, 81-94 (1993).

[36] **Goh, A. T. C., Wong, K. S.** and **Broms, B. B.,** Multivariate Modelling of FEM Data Using Neural Networks, *CIVIL-COMP95 Developments in Neural Networks and Evolutionary Computing for Civil and Structural Engineering,* **Topping, B. H. V.** (ed.), CIVIL-COMP Press, Edinburgh, UK, 59-64 (1995).

[37] **Topping, B. H. V.** and **Bahreininejad, A.,** Subdomain in Generation using Parallel Q-State Potts Neural Networks, *CIVIL-COMP95 Developments in Neural Networks and Evolutionary Computing for Civil and Structural Engineering,* **Topping, B. H. V.** (ed.), CIVIL-COMP Press, Edinburgh, UK, 65-78 (1995).

تطبيقـات الشبـكات العصبيـة الاصطناعيـة في الهندسـة الإنشائيـة : المفهـوم والحـالة الراهنـة

عبـاس برايس
قسم هندسة التشييد ، الكلية التقنية بأبها ،
أبهـــــا - المملكة العربية السعودية

المستخلص . تعتبر الشبكات العصبية الاصطناعية ANN أحد فروع الذكاء الاصطناعي وهي تهتم بإنشاء خوارزميات لعمليات الإدراك مثل التعلم وإيجاد الحل الأمثل لمسائل معينة . وتتصف الشبكات العصبية الاصطناعية بخاصية التعلم والتعميم إذا دربت باستعمال أمثلة معينة في مجال معين بحيث يمكن أن تتوصل إلى حل بدون استعمال قواعد معينة . وقد أصبح الآن الاهتمام في البحث في تطوير وتطبيق الشبكات العصبية الاصطناعية في مجال الهندسة الإنشائية ، وهناك عدة نماذج من الشبكات العصبية الاصطناعية التي طورت لحل بعض المسائل في الهندسة الإنشائية ولكن مازالت في بداية الطريق ولم تتوصل بعد إلى مرحلة متقدمة مقارنة مع تقنيات أخرى مثل النظم الخبيرة . ويقدم هذا البحث تعريفا بمبدأ وخصائص وفوائد وتقييدات الشبكات العصبية الاصطناعية ، ثم يعرض بعض تطبيقات الشبكات العصبية الاصطناعية في مجال الهندسة الإنشائية ، وينهى البحث باستنتاج في أهمية ودور هذه التقنية في الهندسة الإنشائية .