Computers & Structures Vol. 42, No. 4, pp. 649–659, 1992
Printed in Great Britain.

0045-7949/92 $5.00 + 0.00

COMPENDIUM

USE OF NEURAL NETWORKS IN DETECTION OF STRUCTURAL DAMAGE

X. WU,† J. GHABOUSSI†‡ and J. H. GARRETT, JR§

†Department of Civil Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801, U.S.A.
§Department of Civil Engineering, Carnegie-Mellon University, Pittsburgh, PA 15213, U.S.A.

(*Received* 15 *November* 1990)

Abstract—Damage to structures may be caused as a result of normal operations, accidents, deterioration or severe natural events such as earthquakes and storms. Most often the extent and the location of the damage can be determined through visual inspection. However, in some cases visual inspection may not be feasible. The study reported in this paper is the first stage of a research aimed at developing automatic monitoring methods for detection of structural damage. In this feasibility study we have explored the use of the self-organization and learning capabilities of neural networks in structural damage assessment. The basic strategy is to train a neural network to recognize the behavior of the undamaged structure as well as the behavior of the structure with various possible damage states. When the trained network is subjected to the measurements of the structural response, it should be able to detect any existing damage. We have tried this basic idea on a simple structure and the results are promising.

1. INTRODUCTION

An effective and reliable damage assessment methodology will be a valuable tool in the timely determination of damage and deterioration state of structural members. The information produced by a damage assessment process can play a vital role in the development of economical repair and retrofit programs. The most common method of damage assessment is by visual inspection. However, visual inspection of large and complex structures may be difficult and costly due to problems of accessibility and in some cases visual inspection may prove to be unreliable. Examples of such structures are offshore oil and gas exploration and production platforms and aircraft fuselages. In recent years, the indirect diagnostic methodology for structural damage assessment has attracted the attention of many researchers. The basic approach is to detect changes in the dynamic response and/or in the dynamic characteristics of the structure. These changes are subsequently related to various damage states. The objective of any damage assessment is to determine whether structural damage has occurred and if so, to determine the location and the extent of the damage.

Most of the damage assessment methods proposed in the literature follow more or less the same approach. A number of basic steps can be identified in these traditional damage assessment methods. First, a mathematical model for the structure is constructed. The mathematical model is then used to develop an understanding of the structural behavior and to establish correlations between specific member damage conditions and changes in the structural response. These mathematical models are inherently direct process models, proceeding linearly from causes to effects. However, the identification of member damage from the response of the damaged structure is an inverse process, causes must be discerned from the effects. This diagnostic step is a comprehensive search for the causes of the observed structural behavior; this process is computationally expensive and requires a well developed data tracking system.

Researchers have proposed damage assessment schemes based on analysis of measured dynamic responses of the structure before and after damage. Some of the proposed schemes only use the frequency response of the structure under applied loads or the frequency content of the ambient vibration tests [1–5]. More recent works in this area also utilize the information on the mode shapes of the structure [6]. While the existing mathematical models for complex structures may be sufficiently accurate for the purposes of analysis and design, these models are not in themselves sufficient for diagnosing which members may be causing the observed changes in the dynamic response of the structure. A robust damage assessment methodology must be capable of recognizing patterns in the observed response of the structure resulting from individual member damage, including the capability of determining the extent of member damage. This capability appears to be within the scope of the pattern matching capabilities of neural networks. The utilization of these capabilities

‡ To whom correspondence should be addressed.

of neural networks in damage assessment forms the basis of the investigation described in this paper.

In this study, neural networks are used to extract and store the knowledge of the patterns in the response of the undamaged and the damaged structure. Thus, the need for construction of the mathematical models and the comprehensive inverse search is avoided. The neural networks are computational models, loosely based on the structure and the information processing capabilities of the human brain. A neural network is an assembly of a large number of highly interconnected simple processing units (neurons). The connections between the neurons have numerical values which represent the strength of these connections called weights. Knowledge is stored in the form of a collection of connection strengths. These neural networks are capable of self-organization and knowledge acquisition (learning). This capability allows automatic determination from the connection strengths from the data containing the knowledge to be extracted.

The objective of the pilot study presented here is to explore the applicability of neural networks to the assessment of structural damage. Appropriate neural networks are trained to recognize the frequency response of an undamaged structure as well as the frequency responses of the structure in which individual members have sustained varying degrees of damage. These trained neural networks will have the capability of recognizing the location and the extent of individual member damage from the measured response of the structure.

2. NEURAL NETWORKS

As was stated in the introduction, neural networks have been inspired by the neuronal architecture and operation of the brain. Rumelhart *et al.* [7] argue that the reason the brain is capable of high performance in natural information processing tasks, such as perception, language understanding, motor control, etc., is that:

- these tasks require the simultaneous consideration of a large number of pieces of information and constraints, and
- the neuronal architecture, consisting of billions of neurons that are interconnected with a fan-in and fan-out on the order of about 1000–100,000, is ideally suited for the the relaxation-type computation necessary for the simultaneous consideration of the above mentioned multitude of information and constraints [7].

Hence, this neurally inspired computation and knowledge representation paradigm consists of a model of an individual neuron, called a processing unit, that is instantiated thousands of times, with many connections existing between these various instances. The basic model of an individual neuron was developed in 1943, by Warren McCulloch and Walter Pitts and still remains at the heart of most neural networks [8]. Given that the physiological structure of a neuron looks like that shown at the top of Fig. 1. McCulloch and Pitts put forth the following model of a neuron (see bottom of Fig. 1):

- each neuron has an activity, x_i, that is the sum of inputs that arrive via weighted pathways,
- the input from a particular pathway is an incoming signal, S_i, multiplied by the weight, w_{ij}, of that pathway,
- w_{ij} represents synaptic efficacy—the amount of post-synaptic potential transmitted to soma j if a spike occurs at soma i, and
- the outgoing signal from neuron i is computed as $S_j = f(x_j)$, where $f(x_j)$ is binary threshold function of the activity, x_j, in that neuron.

The McCulloch–Pitts neuron can also have a bias term, Θ_j, which acts as a form of threshold. In the McCulloch–Pitts neuron model, the signals, S_i, are restricted to take on the values of 0 and 1, as is obvious from the output function $f(x_j)$.

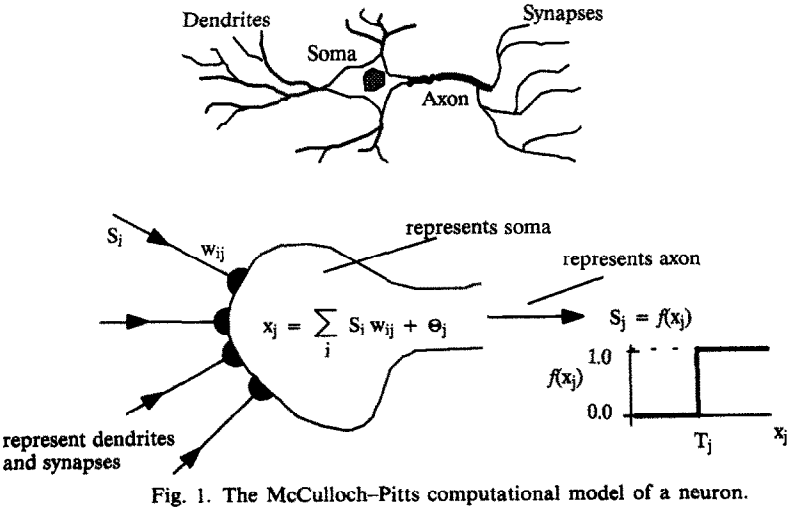

Fig. 1. The McCulloch–Pitts computational model of a neuron.

Each processor in the network thus maintains only one piece of dynamic information (its current level of activation) and is capable of only a few simple computations (adding inputs, computing a new activation level, or comparing input to a threshold value). A neural network performs 'computations' by propagating changes in activation (i.e., level of stimulation) between the processors, which may or may not involve feedback and relaxation.

Rosenblatt [9] extended the McCulloch–Pitts model with the intent of giving self-organization capabilities to networks of these units through localized, dynamic weight modification. His work, while highly criticized for dealing with only simple two-layered networks which were incapable of learning certain types of functions, did pave the way for more modern research into learning mechanisms for multi-layer networks, such as backpropagation [10–12] (discussed later in this paper). Without such capabilities, any complex computation by a McCulloch–Pitts style neural network would require the manual setting of all connection strengths, which would be impossible. Hence, a key characteristic of neural networks is their capability of self-organization or 'learning'. Unlike traditional sequential programming techniques, neural networks are trained with examples of the concepts to capture. The network then internally organizes itself to be able to reconstruct the presented examples. Several other interesting and valuable characteristics are: (1) their ability to produce correct, or nearly correct, responses when presented with partially incorrect or incomplete stimuli; and (2) their ability to perform a limited amount of generalization from the cases on which they are trained. Both of these latter characteristics stem from the fact that a neural network, through self-organization, develops an internal set of features that it uses to classify the stimuli presented to it and return the expected response.

In the early stages of learning, the immature connection strengths cause the actual response of the network to be quite different from that expected. Error is defined as a measure of the difference between the computed output pattern and the expected output pattern. Most learning mechanisms strive to reduce this error for all associations (input/output pattern pairs) to be learned through an iterative, localized weight modification strategy that serves to reduce the sum of the squares of errors for each pattern pair.

Rumelhart *et al.* [7] provide an excellent description of the basic anatomy of neural networks, which they divide into seven basic aspects: (1) a set of processing units, (2) the state of activation of a processing unit, (3) the function used to compute output of a processing unit, (4) the pattern of connectivity among the processing units, (5) the rule of propagation employed, (6) the activation function employed, and (7) the rule of learning employed. The network topology (i.e., the number of nodes and their connectivity), and the form of the rules and functions are all variables in a neural network and lead to a wide variety of network types. Some of the well-known types of neural network described in [12] are: competitive learning [13, 14], the Boltzmann machine [15], the Hopfield network [16], the Kohonen network [17], and the backpropagation network [10–12]. The backpropagation network is given its name due to the way that it learns—by backpropagating the errors seen at the output nodes. Although there are many other variations of neural networks, backpropagation networks are currently being considered by most neural network application developers.

2.1. *Backpropagation neural networks*

The processing units in backpropagation neural networks are arranged in layers. Each neural network has an input layer, an output layer and a number of hidden layers. As illustrated by the examples shown in Fig. 2, it is the presence of these hidden units that allows these networks to represent and compute more complicated associations between patterns. Propagation takes place in a feed-forward manner, from input layer to the output layer. The pattern of connectivity and the number of processing units in

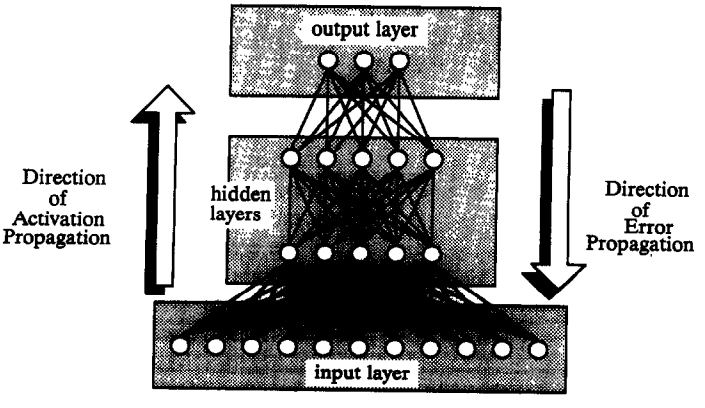

Fig. 2. A sample backpropagation neural network.

each layer may vary with some constraints. No communication is permitted between the processing units within a layer, but the processing units in each layer may send their output to the processing units in higher layers. In the specific type of neural network architecture used in this study, each unit receives input from all the units in the proceeding layer and sends its output to all the units in the next layer. A typical example of such a neural network architecture is shown in Fig. 2.

Associated with each connection is a numerical value which is the strength or the weight of that connection: w_{ij} = strength of connection between units i and j. The connection strengths are developed during training of the neural network. At the beginning of a training process, the connection strengths are assigned random value. As examples are presented during the training, application of the 'rules of learning' (to be described later) modifies the connection strengths in an iterative process. At the successful completion of the training, when the iterative process has converged, the collection of connection strengths captures and stores the knowledge and the information present in the examples used in its training. Such a trained neural network is ready to be used. When presented an input pattern, a feed-forward network computation results in an output pattern which is the result of the generalization and synthesis of what it has learned and stored in its connection strengths.

A feed-foward network computation with these backpropagation neutral networks proceeds as follows:

- the units in the input layer receive their activations in the form of an input pattern and this initiates the feed-forward process,
- the processing units in each layer receive outputs from other units and perform the following computations:
- compute their net input N_j,

$$N_j = \sum_{k=1}^{M} w_{jk} o_k,$$

where o_k = output from units impinging on unit j and M = number of units impinging on unit j.

- Compute their activation values from the net input values

$$a_j = F_j(N_j),$$

F_j is usually a sigmoid function

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

- Compute their outputs from their activation values. In the neural network type used in this study the output is the same as the activation value

$$o_j = a_j.$$

- The output values are sent to other processing units along the outgoing connections.
- This process continues until the processing units in the output layer compute their activation values. These activation values are the output of the neural computations.

2.2. *Self-organization and rules of learning*

Several mechanisms for imparting self-organization to these multi-layer networks have been developed and described in [7]. One of the main characteristics of these learning mechanisms is whether learning occurs in a *supervised* or *unsupervised* fashion. Supervised learning means that the expected output is included in what the network is to learn. Unsupervised learning means that the network is not told what it is to learn about the input with which it is presented and must, on its own, discover regularities and similarities among the input patterns. One form of supervised learning, developed by Rumelhart, Hinton, and Williams, is called the *generalized delta rule* [11] and is the learning mechanism used in backpropagation neural networks. All backpropagation networks, which use the generalized delta rule for self-organization and derive their name from the need to backpropagate error, have the same general architecture shown in Fig. 2.

The training (i.e., supervised learning) of a multilayer backpropagation neural network, via the generalized delta rule, is an iterative process. Each step involves the determination of error associated with each unit and then the modification of weights on the connections coming into that unit. Each presentation of one training case and subsequent modification of connection strengths is called a *cycle*. Each cycle actually involves three substeps: (1) for the training case to be learned, the network is presented with the input pattern and then propagates the activation through to the processing units; (2) the error at the output units is then backpropagated back to the hidden processing units; and (3) connections coming into the hidden units then modify their connection strengths using this backpropagated error.

A set of cycles, made up of one cycle for each training case, is called an *epoch*. The training process for a network may require several hundreds or thousands of epochs for all of the training cases to be 'learned' within a specified error tolerance. The generalized delta rule basically performs a gradient-descent on the error space consisting of Euclidean norms of the errors of all patterns presented during training. Convergence is not guaranteed. Convergence depends in some ways on the network architecture, its capacity and the amount of information it is to learn. This relationship is as yet not well understood and is the subject of current research.

The modification of the strengths of the connections in the generalized delta rule, described in [11], is accomplished through the gradient descent on the total error in a given training case

$$\Delta w_{ij} = \eta \delta_j o_j.$$

In this equation, η = a learning constant called the 'learning rate' and δ_j = gradient of the total error with respect to the net input at unit j. At the output units δ_j is determined from the difference between the expected activations, t_j, and the computed activations, a_j

$$\delta_j = (t_j - a_j)F'(N_j),$$

where F' is the derivative of the activation function.

At the hidden units the expected activations are not known *a priori*. The following equation gives a reasonable estimate of δ_j for the hidden units

$$\delta_j = \left(\sum_{k=1}^{M} \delta_k w_{jk} \right) F'(N_j).$$

In this equation, the error attributed to a hidden unit depends on the error of the units it influences. The amount of error from these units attributed to the hidden unit depends on the strength of connection from the hidden unit to those units; a hidden unit with a strong excitatory connection to a unit exhibiting error will be strongly 'blamed' for this error, causing this connection strength to be reduced.

3. NEURAL NETWORK-BASED APPROACH TO DAMAGE ASSESSMENT

The basic strategy for developing a neural network-based approach to damage assessment of a structural system is to train a backpropagation neural network to recognize individual member damage from the measured response of structures. This strategy is based on the fact that the measured response of any structure contains information regarding its natural frequencies and vibration mode shapes. Structural damage affects these dynamic characteristics of the structure. Thus, in theory, the measured response of a damaged structure, when compared with the response of the undamaged structure, contains information regarding the location and the extent of damage. However, this information may be extremely difficult to extract by the traditional methods; the rules governing the cause and effect relationships must be established explicitly and methodology for using these relationships must be developed *a priori*. In the proposed approach, the training process is used to extract the cause and effect relationships which are then stored within the connection strengths of an appropriate backpropagation neural network. In this way, the self-organization and learning capability of these neural networks are utilized to eliminate the need for explicitly extracting the cause and effect relationships. These relationships exist in a distributed way within the connection strengths of the neural networks but are not obvious or externally available to the user.

The training of a neural network with appropriate data containing the information regarding the cause and effect relationships is at the heart of the proposed method of damage assessment. Thus, the first step is to generate a battery of data that can be used in training an appropriate neural network to recognize damage patterns from the response of the structure. Ideally, this data set should contain the response of the undamaged structure as well as the responses of the structure in various damaged states. This data can be generated through measurement of structural response, model test results or through numerical simulation, or a combination of all three types of data. In this first study, we have used numerical simulations.

In any typical application of backpropagation neural network, at first, an appropriate network architecture must be determined and a training algorithm selected. Then, this particular neural network is trained with the training data set. The trained network is then tested to verify how well it has learned the training cases. The last step is usually to test the gneralization capability of the neural network. In this step, the trained network is tested with data which was not present in the training data set. This last step is actually the test of the performance capability of the network. How well a trained network is able to generalize, very much depends on the adequacy of the selected network architecture and the richness of the training data set on the information needed for damage assessment. Often, changes in the network architecture and/or additions to training data set are needed. Such changes are followed by the repetition of the whole training and testing process. This quasi-iterative fine tuning is repeated until satisfactory performance is obtained.

In this study we have used the computed acceleration time histories as measured response of the structure. These acceleration time histories are then passed through a FFT process and the resulting Fourier spectra of the acceleration time histories are used as input to the neural network. The Fourier spectrum between 0 and 20 Hz is discretized at intervals of 0.1 Hz and the spectral values are input at 200 nodes in the input layer, as shown in Fig. 4. The output layer contains one node per member and the activation values at these nodes represent the damage state in that member; an activation value of one means no damage and an activation value of zero mean complete damage. We have tried various number of hidden layers. Generally, the capacity of a neural network is a function of the number of hidden layers, the number of processing units in each layer, and the pattern of connectivity between layers [18]. In this study a simple pattern of connectivity is used:

each processing unit has outgoing connections to all the processing units in the next layer. The capacity of the neural network is somehow related to the amount of the information in the training data and the complexity of the knowledge contained in that data. Currently there are no reliable rules for determining the capacity of a feed-forward multi-layer neural network, as this question is not yet well understood.

4. CASE STUDY

In this case study the proposed method has been applied to a simple three story frame, as shown in Fig. 3. The frame is modeled as a 'shear building', with girders assumed to be infinitely rigid and the columns being flexible. The structure has three degrees of freedom (floor translations) and the columns in each story represent a member. The structure was subjected to earthquake base acceleration and the transient response was computed in time domain with Newmark method [19]. The Fourier spectra of the computed relative acceleration time histories of the top floor were used in training the neural network.

Member damage was defined as a reduction in the member stiffness. The dynamic responses of the structure at the third or top floor for various member damage conditions were computed. For this investigation, only one member was 'damaged' at a time; damage was represented by a percentage reduction in stiffness for the structural member, and the damage state of the element is defined as the percentage retention in stiffness in the neural network output representation, that is, if the structure is intact, the damage state is designated as 100%. A Fast Fourier Transformation (FFT) was performed on the transient relative acceleration response to convert it from time domain to frequency domain, and the discretized Fourier spectra of the relative accelerations were used as input data in the training data sets. The selected neural network was trained on a number of the Fourier spectra of relative accelerations recorded at the top floor to identify the structural member damage, and the approach is schematically shown in Fig. 3. The following sections describe the scheme for input and output representation, network architecture determination, and the training and testing of the network in more detail.

4.1. *Input and output to the neural network*

The input to the neural network was taken as the Fourier spectra of the relative accelerations recorded at the third or top floor. Note, that for more complicated structures, more accelerometers would have to be used in order to be able to discern the damage state corresponding to one structural member from another. The purpose of this investigation was to illustrate the neural network-based methodology and to show that this approach could be developed to identify member damage. However, more complicated structures will have to be investigated in future research to determine how this method 'scales up'.

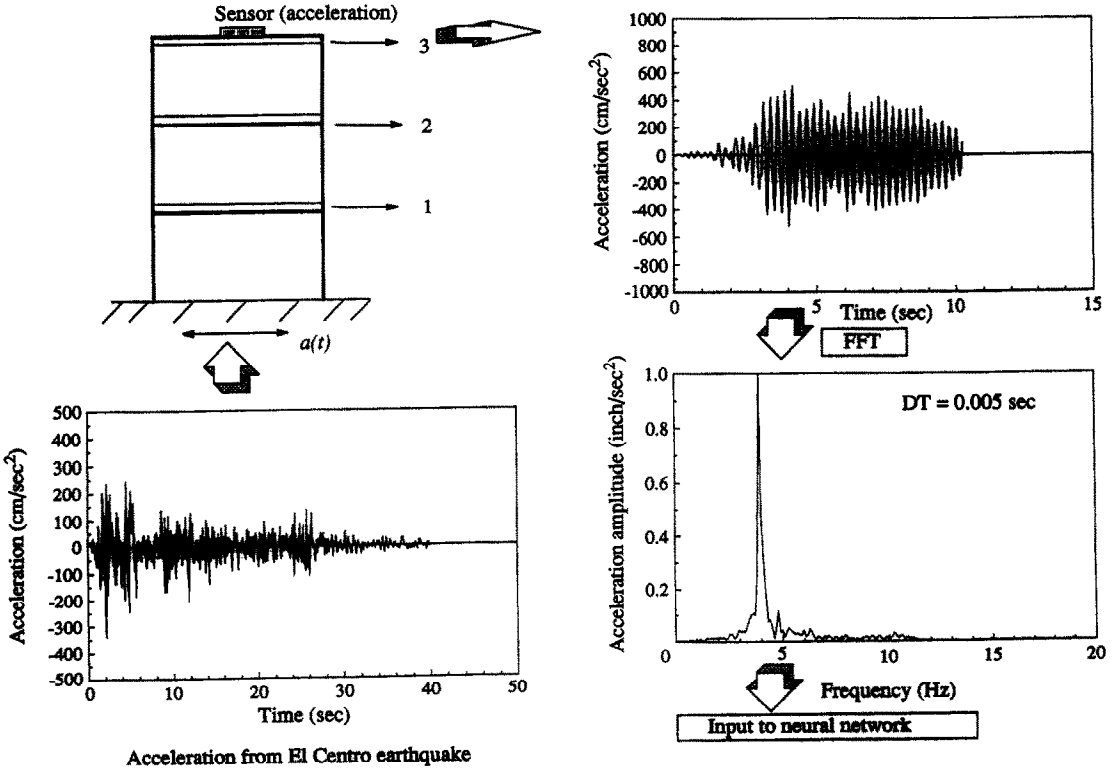

Fig. 3. The damage assessment problem.

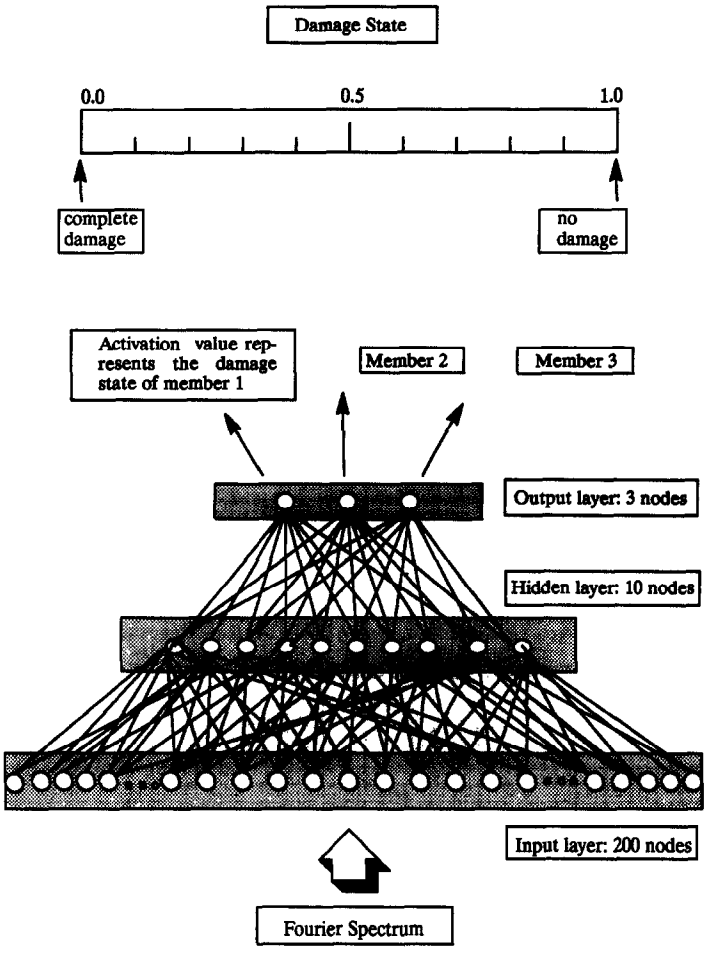

Fig. 4. Architecture of the neural network and the damage state of structural elements.

The Fourier spectrum recorded at the third floor was represented by a set of 200 processing units, where each unit represented a portion of the frequency spectrum. Hence, the input to the neural network consists of 200 units with varying levels of activation value representing the amplitudes of the Fourier spectrum. The output of the neural network is an identification of the extent of the damage for each member in the structure; units with low activation represent severe damage of a structural member and the activation value of 1.0 represents no damaged structural members in the structure. Hence, the output layer of the neural network in this investigation consisted of three nodes, each representing the damage condition for each member in the structure.

4.2. *Architecture of the neural network*

As stated previously, a backpropagation neural network architecture was used in this investigation. In the previous section, the input and output layers were described. It was determined that one hidden layer with 10 nodes be used. The final architecture of the neural network with an illustration of the representation scheme for the damage state of structural elements are shown in Fig. 4. It was discovered through experiments that if the hidden layer was too small, the network would not converge; if the network was too large, it would not converge either. With a hidden layer of 10 nodes, the network converged and modification of the architecture ceased at that point.

4.3. *Training of the neural network*

A total of 42 training cases (consisting of 42 frequency spectra of relative acceleration recorded at the top floor along with the information that indicates which member is damaged and to what extent it is damaged) were presented to the neural network as follows:

- no structural member is damaged, i.e., all of the output units have activation levels near 1.0, for the six different earthquakes used as base accelerations for the structure (six cases);
- the structural member 1 for the first floor is assumed damaged to certain extent, i.e., its stiffness is reduced by 75% and 50% (represented by activation levels of 0.25 and 0.50, respectively), for six different earthquakes (12 cases);

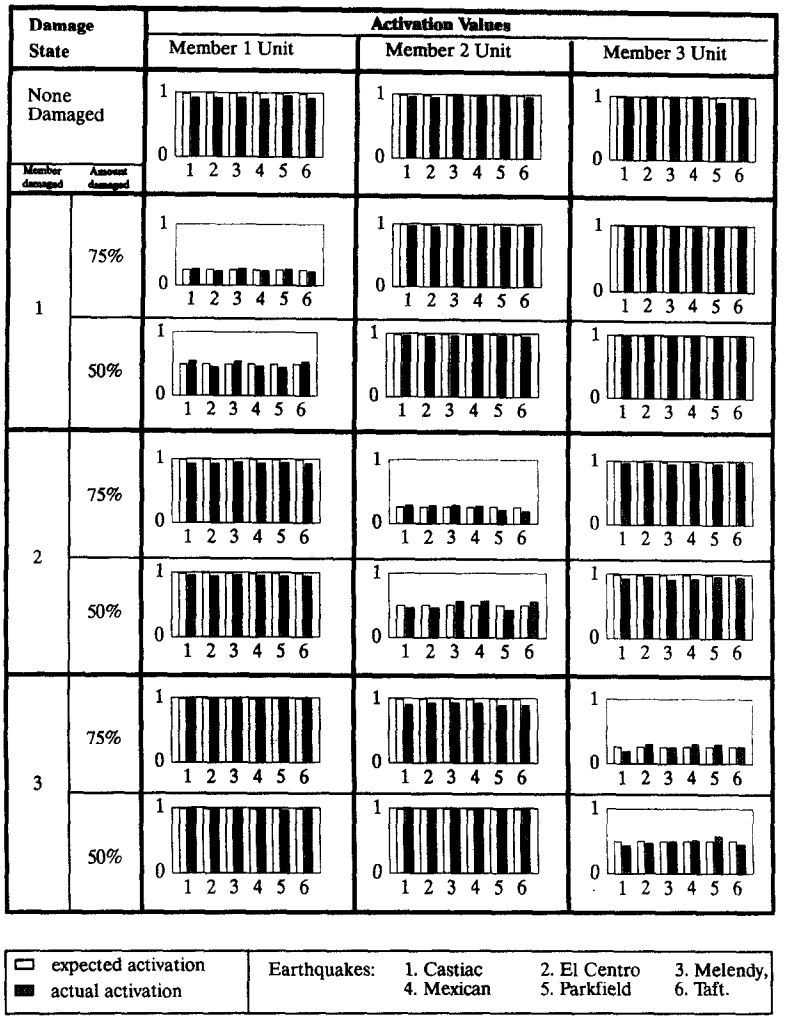

Fig. 5. Damage assessment training results for one-layer network.

- the structural member 2 for the second floor is damaged to certain extent, i.e., its stiffness is reduced by 75% and 50% for six different earthquakes (12 cases);
- the structural member 3 for the third floor is damaged to certain extent, i.e., its stiffness was reduced by 75% and 50% for six different earthquakes (12 cases).

It was discovered in our research on the material modeling with neural networks [20, 21] that an incremental training was more efficient than simply dumping the whole training set to the network. The training patterns were presented to the neural network in stages, with frequency response spectra corresponding to no damage and the 50% damage for all earthquakes records being presented and learned first, followed by those corresponding to 75% damage cases. For the network to 'learn' the first 24 cases, it took 1280 epochs to convergence (an epoch is defined as all the operations needed for feed-forward and error backpropagation of all the cases), with a learning rate of $\eta = 0.2$ and a maximum permissible error of 10%. For the second set of 18 cases, it took the network an additional 1437 epochs. As can be seen by the training results shown in Fig. 5, the network was able to learn the relationship between patterns of frequency response spectra and the corresponding damage state of the structure to identify all 42 damage cases when trained on the Fourier spectra of relative acceleration recorded at the top floor.

4.4. *Testing of the neural network*

After the network was trained on the 42 training cases, it was tested to see how well it would recognize other states of damage, such as those corresponding to a 60% structural member damage. The results of these tests are shown in Fig. 6 and, as can be seen from this figure, the network was able to adequately recognize the damage of structural member 3. However, the trained network was not able to adequately identify the damaged member and to recognize the extent that damage had occurred for structural members 1 and 2. Nevertheless, the network was able to detect that damage had occurred in all three cases, but it was confused as to whether member 1 or member 2 was damaged. While this error may be due

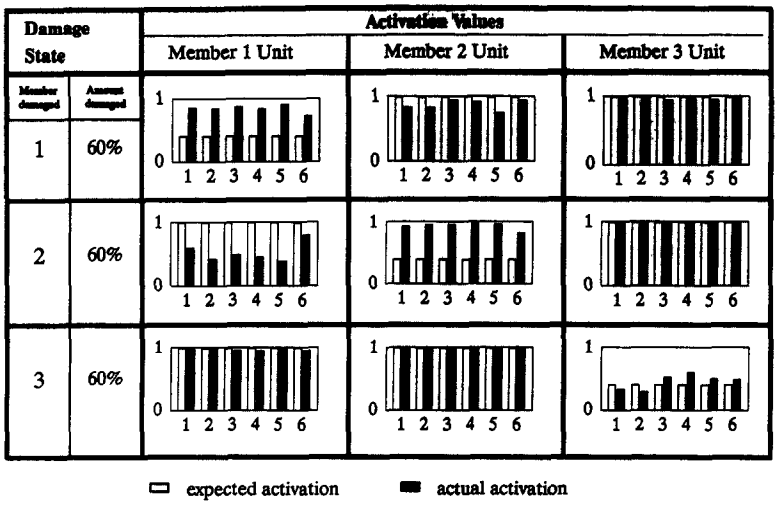

Fig. 6. Damage assessment test results for one-layer network.

to a variety of different causes, it was felt that this inability was most likely due to the network not being provided with enough information to distinguish between the structural member damage states. This has been pointed out by Norton [22] that the selection of suitable location for the measurement transducers or accelerometers is of primary importance in correctly capturing the vibration characteristics of the structure.

To increase the information on which the damaged member identification is based, it was decided to: (1) provide the network with Fourier spectra recorded at two floor levels (the second and the third floors); (2) modify the network architecture to consist of two 200 node input sets (one obtained from the second floor and another from the third floor) and two hidden layers; and (3) re-train the network. Two hidden layers were thought to be necessary so as to be able to look for combinations (i.e., features) of the frequencies in the Fourier spectra recorded at the two floor levels. The size of the hidden layers for which the network was able to converge turned out to be eight units in each layer. The training results for this new network were almost identical to those shown in Fig. 5, and thus are not shown here in the interest of space. However, the testing results of the new network did change from those of Fig. 6 and are shown in Fig. 7. Note that the network, when provided with more information about the dynamic behavior of the structure, was able to better distinguish between the various states of damage and was able to identify the structural member experiencing 60% damage. The network was able to correctly identify damage to both the first and third member, but it was still unable to identify that the second structural member was damaged by 60%.

5. CONCLUDING REMARKS

On the basis of the case studies presented, it is felt that the use of neural networks for structural damage assessment is a promising line of research. These early

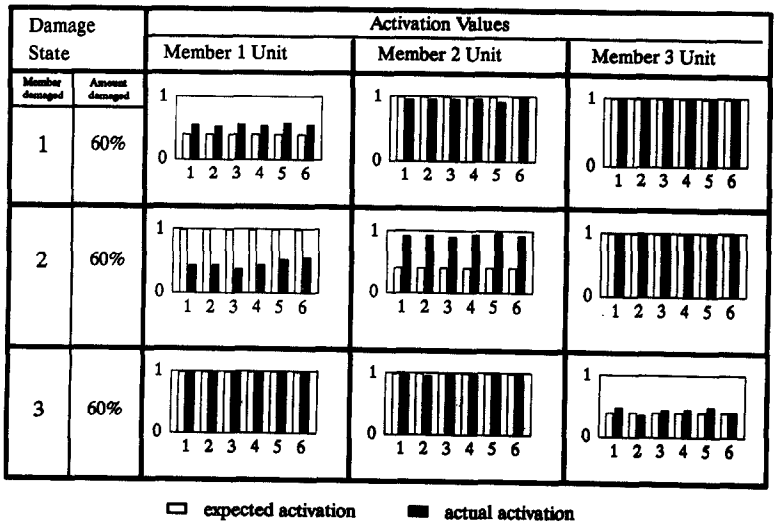

Fig. 7. Damage assessment test results for two-layer network.

results indicate that neural networks are capable of learning about the behavior of undamaged and damaged structures and to identify the damaged member and the extent of the damage from patterns in the frequency response of the structure. However, some issues remain to be resolved before this approach becomes a truly viable method of structural damage assessment. At first, for real world complex structures, the neural network after proper training should be able to distinguish between the frequency response patterns recorded at several locations of the structure so that the damaged member and its extent of damage can be identified. Secondly, the amount of information needed for training a neural network, such that the trained network not only learned the training cases but also can accurately generalize to other cases, must be quantified; that is, how many accelerometer recordings, and at what locations, must be obtained in order to reliably identify the damaged structural members. For more complicated structures, it is anticipated that many more than two accelerometer recordings (as in this case study) will have to be used as input to the network. Finally, how many damage states for each structural members must be presented to the network before it gains adequate performance for all states of single member damage? It is anticipated based on the generalization capability of neural networks that only several variations of damage state for selected key structural member may be sufficient in order to adequately train the network.

Further research in this area should not only address the above mentioned issues, but should also investigate the role of noise in actual measurements, application to large and complex structures, use of measurements from applied force tests or ambient vibration tests.

REFERENCES

1. B. G. Burke, C. Sundararajan and F. M. Safaie, Characterization of ambient vibration data by response shape vectors. In *Proceedings of the Offshore Technology Conference*, OTC paper 3861, Houston, TX, pp. 53–61 (1980).
2. R. N. Coppolino and K. J. Rubin, Detectability of structural failures in offshore platforms by ambient vibration monitoring. In *Proceedings of the Offshore Technology Conference*, OTC paper 3865, Houston, TX, pp. 101–110 (1980).
3. S. Rubin, Ambient vibration survey of offshore platform. *J. Engng Mech., ASCE* **106**, 425–441 (1980).
4. F. Shahrivar and J. G. Bouwkamp, Damage detection in offshore platforms using vibration information. *Proceedings of the Third International Offshore Mechanics and Arctic Engineering Symp.*, ASME, II, pp. 174–185 (1984).
5. J. K. Vandiver, Detection of structural failure on fixed platform by measurement of dynamic response. *Proceedings of the Offshore Technology Conference*, OTC paper 2267, Houston, TX, pp. 243–252 (1975).
6. F. Shahrivar and J. G. Bouwkamp, Signal separation method for tower mode shape measurement. *J. Struct. Engng, ASCE* **115**, 707–723 (1989).
7. D. E. Rumelhart, J. L. McClelland and the PDP Research Group, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*. MIT Press, Cambridge, MA (1986).
8. W. S. McCulloch and W. Pitts, A logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophys.* **5**, 115–133 (1943).
9. F. Rosenblatt, *Principles of Neurodynamics*. Spartan, New York (1962).
10. D. B. Parker, Learning logic. Invention report S81-64, File 1, Office of Technology Licensing, Stanford University (1982).
11. D. E. Rumelhart, G. E. Hinton and R. J. Williams, Learning internal representations by error propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations* (Edited by D. E. Rumelhart and J. L. McClelland). MIT Press, Cambridge, MA (1986).
12. P. Warbos, Beyond regression: New tools for prediction and analysis in behavioral sciences. Ph.D. dissertation, Harvard University (1974).
13. S. Grossberg, Adaptive pattern classification and universal recoding: Part 1. Parallel development and coding of natural features detectors. *Biological Cybernetics* **23**, 121–134 (1976).
14. D. E. Rumelhart and D. Zipser, Feature discovery by competitive learning. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations* (Edited by D. E. Rumelhart and J. L. McClelland). MIT Press, Cambridge, MA (1986).
15. G. E. Hinton, T. J. Sejnowski and D. H. Ackley, Boltzmann machines: Constraint satisfaction networks that learn. Technical report No. CMU-CS-84-119, Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA (1984).
16. J. J. Hopfiled, Neural networks and physical systems with emergent collective computational abilities. *Proc. natl Acad. Sci., U.S.A.* **79**, 2554–2558 (1982).
17. T. Kohonen, *Self Organization and Associative Memory*. Springer, Berlin (1984).
18. K. Hornik, M. Stinchcombe and H. White, Multilayer feedforward networks are universal approximators. *Neural Networks* **2**, 356–366 (1989).
19. N. M. Newmark, A method of computation for structural dynamics. *J. Engng Mech., ASCE* **3**, 67–94 (1959).
20. J. H. Garrett, Jr, J. Ghaboussi and X. Wu, Neural networks. In *Expert Systems in Civil Engineering—Knowledge Representations* (Edited by R. H. Allen). The Expert Systems Committee of the American Society of Civil Engineers (1990).
21. J. Ghaboussi, J. H. Garrett, Jr and X. Wu, Knowledge-based modeling of material behavior with neural networks. *J. Engng Mech.* **117**, Jan. (1991).
22. M. P. Norton, *Fundamentals of Noise and Vibration Analysis for Engineers*. Cambridge University Press, Cambridge (1989).
23. IEEE, *Proceedings of the First IEEE International Conference on Neural Networks*. San Diego, CA, June 21–24 (1987).
24. IEEE, *Proceedings of the 1988 IEEE International Conference on Neural Networks*. San Diego, CA, July 24–27 (1988).
25. IJCNN, *Proceedings of the International Joint Conference on Neural Networks*. Co-sponsored by IEEE and the International Neural Network Society, Washington DC, June 18–22 (1989).
26. IJCNN, *Proceedings of the International Joint Conference on Neural Networks*. Co-sponsored by IEEE and the International Neural Network Society, Washington DC, January (1990).
27. IJCNN, *Proceedings of the International Joint Conference on Neural Networks*. Co-sponsored by IEEE and

the International Neural Network Society, San Diego, CA, June 17–21 (1990).
28. I. Aleksander, (Ed.), *Neural Computing Architecture*. MIT Press, Cambridge, MA (1989).
29. T. Ash, Dynamic node creation in backpropagation networks. In *Proceedings of the International Joint Conference on Neural Networks*, Washington DC, June 18–22 (1989).
30. A. G. Barto, Connectionist learning for control: An overview. COINS technical report 89–89, Department of Computer and Information Science, University of Massachusetts, Amherst, MA (1989).
31. G. Carpenter and S. Grossberg, Adaptive resonance theory: Stable self-organization of neural recognition codes in response to arbitrary lists of input patterns. In *Eighth Annual Conference of the Cognitive Science Society*. Lawrence Erlbaum Associates (1986).
32. R. W. Clough and J. Penzien, *Dynamics of Structures*. McGraw-Hill, New York (1975).
33. S. E. Fahlman and C. Lebiere, The cascade–correlation learning architecture. Technical report CMU-CS-90-100, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA (1990).
34. D. O. Hebb, *The Organization of Behavior: A Neuropsychological Theory*. John Wiley, New York (1949).
35. M. Minsky and S. Papert, *Perceptrons*. MIT Press, Cambridge, MA (1969).
36. D. R. Rehak, C. R. Thewalt and L. B. Doo, Neural network approaches in structural mechanics computations. *Computer Utilization in Structural Engineering—The Proceedings of the Sessions Related to Computer Utilization at Structures Congress '89*. ASCE, pp. 168–176 (1989).
37. M. F. Tenorio and W.-T. Lee, Self-organizing neural network for optimum supervised learning. Technical report TR-EE-89-30, School of Electrical Engineering, Purdue University, West Lafayette, IN (1989).
38. M. D. Trifunac, Comparisons between ambient and forced vibration experiments. *Earthquake Engng Struct. Dynamics* **1**, 133–150 (1972).
39. B. Widrow and M. E. Hoff, Adaptive switching circuits. *1960 IRE WESCON Convention Record*, New York, IRE, pp. 96–104 (1960).