# Application of Unified Smart Classification and Modified Weight Elimination Algorithms to Damage Evaluation in Composite Structures

MAHMOUD Z. ISKANDARANI
Department of Electrical Engineering
Al-Zaytoonah Private University of Jordan
P O BOX: 911597, Post Code: 11191, Amman
JORDAN
m_iskandarani@yahoo.com

*Abstract:* - Unified Smart Classification Algorithm (USCA) for the purpose of data processing and classification of data obtained from different testing techniques is designed and tested. The developed algorithm conditions data taken from damaged composite structures such as modern car bodies and Plane frame structure. It is used in conjunction with a Modified Weight Elimination Neural Networks Algorithm (MWEA) to provide predictive models for impact damage in composite structures. The developed neural models correlated between various NDT testing techniques, such that in the absence of one technique, its results are predicted by the Neural Network through interrogation of available data obtained using other testing methods. The real and predicted data showed good agreements in terms of classification and prediction.

*Key-Words:* - Neural Networks, Classification, Damage, Composites, Algorithm, Prediction, Weight Elimination, Pruning.

## 1 Introduction

Composites materials with desirable properties and low weight have become increasingly popular, with their laminated structure allowing for design improvements compared to other materials, the increase use of composites in operating structures has raised the issue of inspection reliability and condition assessment and the need for advanced techniques and algorithms for data processing and interpretation.

Modern Automotive and Aerospace applications have resulted in an increased demand for real-time, effective techniques for structural integrity monitoring and damage detection with accurate and timely assessment of component damage as a critical element of safety with short computational time. Model based techniques offer a computationally efficient approach.

The previous highlights the need for an intelligent classification system, which is flexible enough to accommodate different boundary conditions with complex non-linear behavior in a fast and accurate manner with capabilities of generalization and prediction [1-4].

There is a large amount of literature describing statistical discrimination of features for damage detection using pattern recognition algorithm. Other studies showed neural networks (NNs) as a powerful tool for damage identification where feed-forward neural networks is used to detect and locate damage. Bayesian probabilistic methods and functions also used to compute the probability of continually updated model parameters. In addition, genetic algorithm (GA) is used to identify and locate damage in a laminated composite structure.

Many studies suggested that neural network can directly map the characteristics of a material without the need for a mathematical model. Work has been carried out to build and apply neural network models to characterize, map, and predict parameters such as, temperature, effective strain, strain rate flow stress, inelastic strain, internal variables, inelastic strain rate, creep, and time. The neural network ‘‘learns’’, and if applied to another set of experimental data, may fulfill its task more accurately in shorter time periods. The present methodology may simulate to some extent the experiments after a period of learning [5-9].

Artificial Neural Networks (ANNs) are one of the adaptive systems that are widely used in signal applications because of their remarkable ability to extract patterns that are too complex to be analyzed by normal algorithms. Thus, artificial neural networks help to approximate the complex material or structural behaviors in composites, for both material and structural behaviors due to their effectiveness, robustness, and noise-tolerance.

Neural Networks can be applied to many real world problems. Multilayer neural networks are used in areas like pattern recognition, optimization, intelligent control and others. The objective of neural network design and application is to find the optimum network that can learn and generalize satisfactory taking into consideration, time, speed, reliability and possible future modifications. Hence, it is important to choose the right design with method of training and pruning to achieve optimum performance per selected application [10-16].

In this paper a combination of Modified Weight Elimination Algorithm Neural Networks and our novel Unified Smart Classification Algorithm (USCA) algorithms is used to classify and predict structural variables using different testing techniques.

## 2 Experimental Arrangements

Testing of resin injection molded (RIM) samples response to impact damage was carried out using the following techniques:

(1) Low Frequency Tapping.
(2) Low Temperature Thermal imaging.
(3) Tensile Strength.

USCA was used on each data file to produce a fingerprint. The collated data was then fed to the designed Neural Structures.

### 2.1 USCA

The initial raw data is conditioned before entering into the Neural Networks using the USCA algorithm, which utilizes matrix equations, where individual data matrices that correspond to different testing techniques are grouped into an overall matrix as shown in equation (1).

In the USCA algorithm, the converted data file is grouped into sequences $S_1$ to $S_m$ containing vectors of individual column matrices extracted from the converted source data file. The overall extracted matrix consists of discrete combination of all column sequences as in equation (2).

$$\begin{bmatrix} S_1 & S_2 & S_m \\ a_{11}.r_{11} & a_{12}.r_{12} & a_{1m}.r_{1m} \\ a_{21}.r_{21} & a_{22}.r_{22} & a_{2m}.r_{2m} \\ & & \\ & & \\ a_{n1}.r_{n1} & a_{n2}.r_{n2} & a_{nm}.r_{nm} \end{bmatrix} \quad (1)$$

Where:

$a_{ij}$ : Original matrix elements

$r_{ij}$ : Amplitude Factor

From (1) we obtain:

$$S_1 = \begin{bmatrix} A_1 \\ A_2 \\ \\ \\ A_n \end{bmatrix}, \; S_2 = \begin{bmatrix} B_1 \\ B_2 \\ \\ \\ B_n \end{bmatrix} \ldots, S_m = \begin{bmatrix} X_1 \\ X_2 \\ \\ \\ X_n \end{bmatrix} \ldots(2)$$

From (2) we obtain:

$$S_{1T} = \sum_{i=1}^{n} \frac{A_i}{\theta} \;,\; S_{2T} = \sum_{i=1}^{n} \frac{B_i}{\theta}, \ldots S_{mT} = \sum_{i=1}^{n} \frac{X_i}{\theta} \;..\; (3)$$

Where: $\theta$ is a normalizing factor.

As the original matrix is simplified in (3), the data classification column matrix is represented in equation (4).

$$D = \begin{bmatrix} S_{1T} \\ S_{2T} \\ \\ \\ S_{mT} \end{bmatrix} \ldots\ldots\ldots\ldots\ldots\ldots(4)$$

The designed classification algorithm takes into account reference, undamaged sample data in its operations. Now data is ready to be exported using MWEA.

### 2.2 MWEA

Training of the Networks was carried out using Weight Elimination Algorithm (MWEA), which is a bidirectional Bottom-Up, Top-Down pruning algorithm. It starts with a simple, then complex network and drives unnecessary weights during training towards zero as follows:

1. The neural network is built constructively (Bottom-Up), where its size and complexity are modified (Top-Down) to achieve a stable network with error below a pre-defined initial value.

2. The training patterns are scaled within controllable values to prevent oscillations.

3. The network is subjected to various patterns during training with constant recording of weights and Removal of any connections that might contribute to bad classification and generalization, then redistribution of removed connection weights among the rest of the interneuron connections.

4. The overall number of actively connected neurons in the hidden layer is reduced during the process, due to weight and bias eliminations.

5. The MWEA makes use of a liability function that is based on the error function. By reducing the number of connection weights and hence the model's complexity using the weight-elimination liability term, it is expected that network's classification performance to improve. The weight-elimination overhead function is shown in (5). The liability term in weight-elimination minimizes the sum of performance error and the number of weights using standard backpropagation technique.

$$E_{Total}(W) = E_{SumSquared}(W) + E_{Liability}(W) \quad ..(5)$$

$E_{Total}(W)$ is the combined overhead function that includes the initial overhead function, $E_{Sum\,Squared}(W)$ and the weight-elimination term $E_{Liability}(W)$.

$$E_{Sum\,Squared}(W) = \frac{1}{2}\sum_{k}\left(T_k - O_k\right)^2 \quad \ldots(6)$$

Where:

$T_k$ : Target Output

$O_k$ : Actual Output

$$E_{Liability}(W) = \xi\left(\sum_{jk}\frac{\left(\dfrac{w_{jk}}{w_{epochs-n}}\right)^2}{1+\left(\dfrac{w_{jk}}{w_{epochs-n}}\right)^2}\right) \quad \ldots(7)$$

Hence;

$$\Delta W = \left(-\eta\,\frac{\partial E_{Sum\,Squared}}{\partial W}\right) - \left(\xi\,\frac{\partial E_{Liability}}{\partial W}\right) \quad \ldots(8)$$

Where

$\eta$ : Learning Rate (between 0 and 1)

W represents the weight vector, $\xi$ is the weight-reduction factor, and $w_{jk}$ represents the individual weight of the neural network model.

The role of the weight-reduction factor is to determine the relative importance of the weight-elimination term. Larger values of $\xi$ pushes small weights to further reduce their size. Small values of $\xi$ will not affect the network.

The scale parameter, $w_{epochs-n}$, is a scale parameter computed by the MWEA, and chosen to be the smallest weigh from the last epoch or set of epochs to force small weights to zero [17-19].

Figure (1) illustrates the used training neural network sufficient large no of neurons per hidden layer for better convergence, while Figure (2) shows the designed decision neural network. Training of the Neural Network model is carried using patterns similar to the one in Table (1).

## 3 Results

Using our MWEA in training the Neural Network resulted in an elimination of weak connections ($\eta = 0.1,\ \xi = 2.5$). Figure (3) show weight distribution prior to training, while Figure (4) show a fundamental change in characteristics with induced uniformity in connection weight distribution due to network convergence and stability. Weight elimination is evident in Figure (4) and in output weight characteristics in Figure (5).

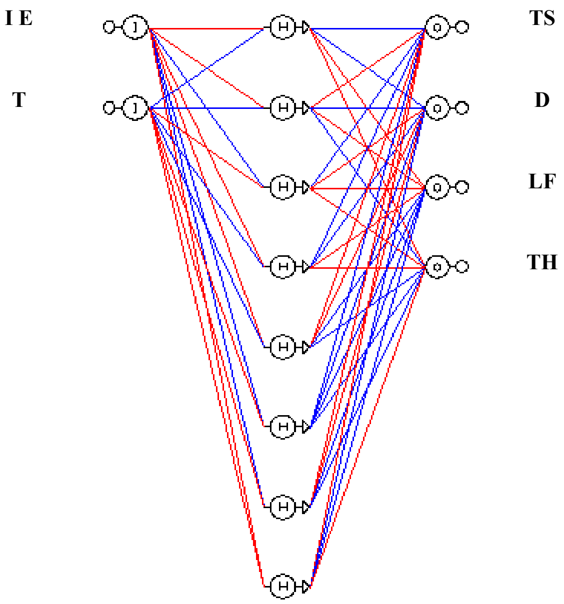


Fig.1: Training Network Neural Model.

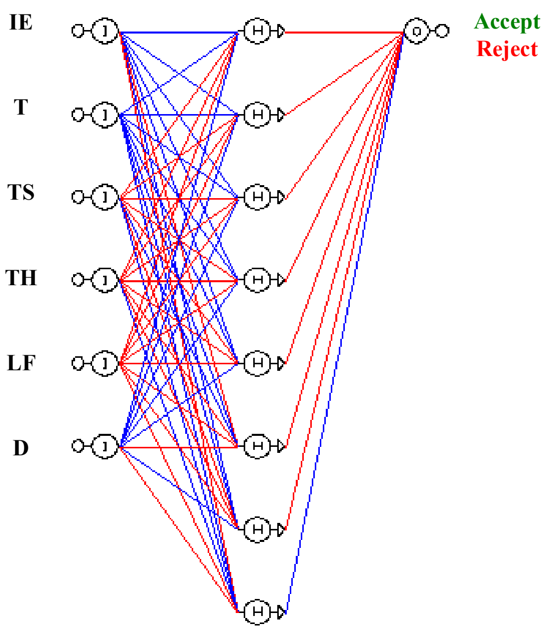


Fig.2: Damaged/Undamaged decision Neural Network.

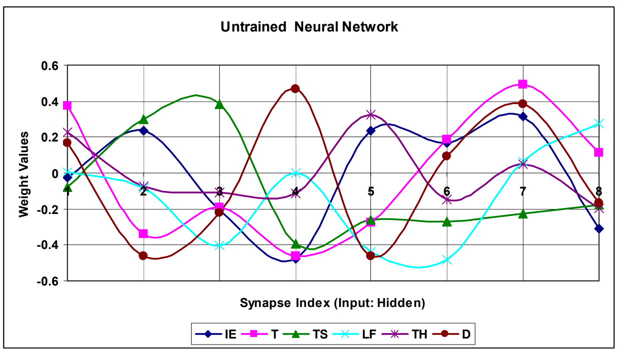


Fig.3: Untrained Network Weight Connections.

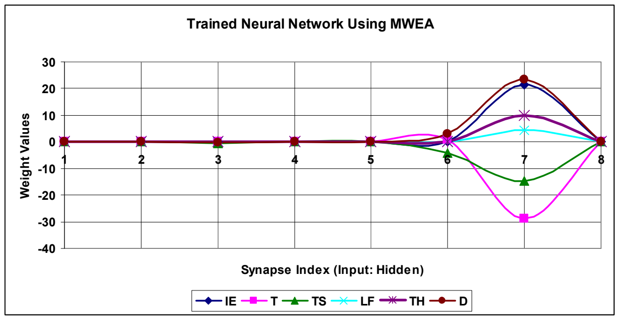


Fig.4: Trained Network Weight Connections.

| Impact Energy<br>I E<br>( J ) | Sample Thickness<br>T<br>mm | Tensile Strength<br>TS<br>N/mm | %USCA<br>Low Frequency<br>LF<br>No Units | %USCA<br>Thermo-imaging<br>TH<br>No Units | Defect Diameter<br>D<br>mm |
|---|---|---|---|---|---|
| 7.14 | 2 | 205.16 | 74.5 | 28.60 | 0.727 |
| 7.14 | 5 | 208.30 | 91.5 | 24.98 | 0.573 |
| 14.3 | 2 | 190.00 | 67.0 | 35.40 | 1.210 |
| 14.3 | 5 | 206.78 | 87.5 | 30.05 | 0.858 |
| 28.6 | 2 | 150.74 | 49.2 | 46.02 | 2.840 |
| 28.6 | 5 | 205.67 | 59.5 | 35.01 | 1.540 |
| 42.0 | 2 | 150.68 | 46.0 | 46.04 | 2.850 |
| 42.0 | 5 | 180.96 | 57.4 | 39.92 | 1.970 |
| 47.6 | 2 | 150.68 | 29.0 | 46.04 | 2.850 |
| 47.6 | 5 | 153.44 | 37.0 | 45.07 | 2.710 |

Table1: Data for training network

Matrices A to F contain weight matrices and illustrate transformation and elimination within the hidden layer with matrices G and H contain output and bias weights.

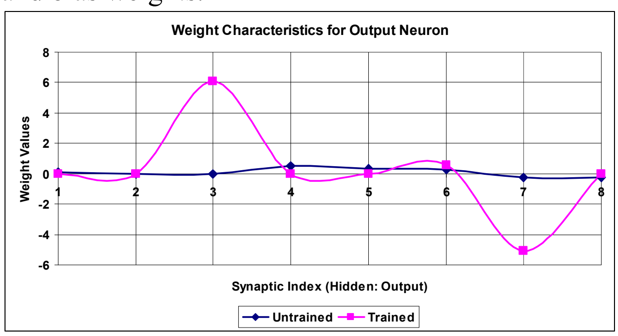


Fig.5: Weight Characteristics at Output Neuron.

$$\begin{bmatrix} W_{0\to 6}: -0.0233 \\ W_{0\to 7}: \ 0.2362 \\ W_{0\to 8}: -0.2068 \\ W_{0\to 9}: -0.4795 \\ W_{0\to 10}: \ 0.2370 \\ W_{0\to 11}: \ 0.1667 \\ W_{0\to 12}: \ 0.3155 \\ W_{0\to 13}: -0.3098 \end{bmatrix} \rightarrow \begin{bmatrix} W_{0\to 6}: \ 0.0000 \\ W_{0\to 7}: \ 0.0000 \\ W_{0\to 8}: -0.0216 \\ W_{0\to 9}: \ 0.0000 \\ W_{0\to 10}: \ 0.0000 \\ W_{0\to 11}: \ 0.2260 \\ W_{0\to 12}: 21.4772 \\ W_{0\to 13}: \ 0.0000 \end{bmatrix} \ldots(A)$$

**Weight Matrix: Impact Energy (IE)**

$$\begin{bmatrix} W_{1\to 6}: \ 0.3725 \\ W_{1\to 7}: -0.3389 \\ W_{1\to 8}: -0.1933 \\ W_{1\to 9}: -0.4618 \\ W_{1\to 10}: -0.2744 \\ W_{1\to 11}: \ 0.1868 \\ W_{1\to 12}: \ 0.4921 \\ W_{1\to 13}: \ 0.1139 \end{bmatrix} \rightarrow \begin{bmatrix} W_{1\to 6}: \ 0.0000 \\ W_{1\to 7}: \ 0.0000 \\ W_{1\to 8}: -0.0339 \\ W_{1\to 9}: \ 0.0000 \\ W_{1\to 10}: \ 0.0000 \\ W_{1\to 11}: \ 0.6504 \\ W_{1\to 12}: -28.9760 \\ W_{1\to 13}: \ 0.0000 \end{bmatrix} \ldots(B)$$

**Weight Matrix: Thickness (T)**

$$\begin{bmatrix} W_{2\to 6}: -0.0799 \\ W_{2\to 7}: \ 0.3017 \\ W_{2\to 8}: \ 0.3814 \\ W_{2\to 9}: -0.3912 \\ W_{2\to 10}: -0.2603 \\ W_{2\to 11}: -0.2717 \\ W_{2\to 12}: -0.2268 \\ W_{2\to 13}: -0.1751 \end{bmatrix} \rightarrow \begin{bmatrix} W_{2\to 6}: \ 0.0000 \\ W_{2\to 7}: \ 0.0000 \\ W_{2\to 8}: -0.5158 \\ W_{2\to 9}: \ 0.0000 \\ W_{2\to 10}: \ 0.0000 \\ W_{2\to 11}: -4.0326 \\ W_{2\to 12}: -14.8130 \\ W_{2\to 13}: \ 0.0000 \end{bmatrix} \ldots(C)$$

**Weight Matrix: Tensile Strength (TS)**

$$\begin{bmatrix} W_{3\to 6}: \ 0.0051 \\ W_{3\to 7}: -0.0831 \\ W_{3\to 8}: -0.4050 \\ W_{3\to 9}: -0.0010 \\ W_{3\to 10}: -0.4399 \\ W_{3\to 11}: -0.4838 \\ W_{3\to 12}: \ 0.0623 \\ W_{3\to 13}: \ 0.2765 \end{bmatrix} \rightarrow \begin{bmatrix} W_{3\to 6}: \ 0.0000 \\ W_{3\to 7}: \ 0.0000 \\ W_{3\to 8}: -0.0123 \\ W_{3\to 9}: \ 0.0000 \\ W_{3\to 10}: \ 0.0000 \\ W_{3\to 11}: \ 0.1691 \\ W_{3\to 12}: \ 4.4628 \\ W_{3\to 13}: \ 0.0000 \end{bmatrix} \ldots(D)$$

**Weight Matrix: Low Frequency (LF)**

$$\begin{bmatrix} W_{4\to 6}: \ 0.2249 \\ W_{4\to 7}: -0.0740 \\ W_{4\to 8}: -0.1089 \\ W_{4\to 9}: -0.1147 \\ W_{4\to 10}: \ 0.3268 \\ W_{4\to 11}: -0.1472 \\ W_{4\to 12}: \ 0.0494 \\ W_{4\to 13}: -0.1965 \end{bmatrix} \rightarrow \begin{bmatrix} W_{4\to 6}: \ 0.0000 \\ W_{4\to 7}: \ 0.0000 \\ W_{4\to 8}: \ 0.1496 \\ W_{4\to 9}: \ 0.0000 \\ W_{4\to 10}: \ 0.0000 \\ W_{4\to 11}: \ 0.0624 \\ W_{4\to 12}: \ 10.0940 \\ W_{4\to 13}: \ 0.0000 \end{bmatrix} \ldots(E)$$

**Weight Matrix: Thermal Imaging (TH)**

$$\begin{bmatrix} W_{5\to 6}: \ 0.1678 \\ W_{5\to 7}: -0.4605 \\ W_{5\to 8}: -0.2200 \\ W_{5\to 9}: \ 0.4695 \\ W_{5\to 10}: -0.4627 \\ W_{5\to 11}: \ 0.0926 \\ W_{5\to 12}: \ 0.3812 \\ W_{5\to 13}: -0.1687 \end{bmatrix} \rightarrow \begin{bmatrix} W_{5\to 6}: \ 0.0000 \\ W_{5\to 7}: \ 0.0000 \\ W_{5\to 8}: -0.2732 \\ W_{5\to 9}: \ 0.0000 \\ W_{5\to 10}: \ 0.0000 \\ W_{5\to 11}: \ 3.1870 \\ W_{5\to 12}: \ 23.3510 \\ W_{5\to 13}: \ 0.0000 \end{bmatrix} \ldots(F)$$

**Weight Matrix: Diameter (D)**

$$\begin{bmatrix} W_{6\to 14}: \ 0.0774 \\ W_{7\to 14}: -0.0224 \\ W_{8\to 14}: -0.0219 \\ W_{9\to 14}: \ 0.4819 \\ W_{10\to 14}: \ 0.3039 \\ W_{11\to 14}: \ 0.2679 \\ W_{12\to 14}: -0.2277 \\ W_{13\to 14}: -0.2283 \end{bmatrix} \rightarrow \begin{bmatrix} W_{6\to 14}: \ 0.0000 \\ W_{7\to 14}: \ 0.0000 \\ W_{8\to 14}: \ 6.0570 \\ W_{9\to 14}: \ 0.0000 \\ W_{10\to 14}: \ 0.0000 \\ W_{11\to 14}: \ 0.5376 \\ W_{12\to 14}: -5.0910 \\ W_{13\to 14}: \ 0.0000 \end{bmatrix} \ldots(G)$$

**Weight Matrix: Output**

$$\begin{bmatrix} Bias_{Neuron6}: -0.2828 \\ Bias_{Neuron7}: -0.0148 \\ Bias_{Neuron8}: \ 0.3650 \\ Bias_{Neuron9}: -0.2289 \\ Bias_{Neuron10}: \ 0.2919 \\ Bias_{Neuron11}: \ 0.2330 \\ Bias_{Neuron12}: \ 0.2932 \\ Bias_{Neuron13}: \ 0.4604 \end{bmatrix} \rightarrow \begin{bmatrix} Bias_{Neuron6}: \ 0.0000 \\ Bias_{Neuron7}: \ 0.0000 \\ Bias_{Neuron8}: \ 0.0391 \\ Bias_{Neuron9}: \ 0.0000 \\ Bias_{Neuron10}: \ 0.0000 \\ Bias_{Neuron11}: \ 0.0557 \\ Bias_{Neuron12}: -7.5470 \\ Bias_{Neuron13}: \ 0.0000 \end{bmatrix} \ldots(H)$$

**Bias Matrix**

## 4 Discussion and Conclusion

Table (2) shows the predicted data with classification using the networks shown in Figures (1) and (2). From the table, it is observed that a very good predication is achieved with excellent classification using MWEA. Table (3) contains the decision implementation of the MWEA whether to accept or reject a component by using all or some of the input techniques and data used earlier, which accounts for missing, incomplete and corrupted data files. This provides a fast process of testing and production.

Table (4) compares decisions for initial, trained with complete data, and trained with incomplete and corrupted data. As shown the decision values start at a certain number before converging and being mapped to different values depending on the input testing pattern. Figures (6) and (7) show the decision boundaries for Tables (2) and (3). It is clear that the initial state of the network form a decision ring with a uniform and patterned specific spread of decision points either side of it as mapped by the MWEA in response to the applied testing patterns. The used decision criterion is:

Decision >0.5 ➔ Accept
Decision <0.5 ➔ Reject

Even thought the correct classification is obtained, comparing figure (6) with figure(7) demonstrate that the classification pattern around starting decision values is distinct for the data files with missing or corrupted data compared to the one with normal data. This in turn affects the classification and decision values where it drops just above the threshold level for the corrupted data file. Some classification values remain at a high level due to the missing part of data being not so significant in contributing towards a final decision.

The decision points (1.00 and 0.00) common to both normal and corrupted data; support the importance of tensile strength in judging damage effect on a composite structure, which is affected by the impact energy that affects components differently depending on component thickness. The resulting defect diameter of the applied impact load is also a function of impact energy, tensile strength and component thickness.

Hence, at the start of network learning and looking at Table (4), all initial decision values starts as acceptance values, even for damaged structures, which is normal as training and weight elimination will map the training patterns and optimize the network. Starting with positive valued decision surface is a good idea to avoid sinking into minima.

The decision vector ($d_j$) changes length in accordance with the classification decision as it maps values from initial to final. The change in length of the vector depends on the quality of testing data, error value, and number of neurons per hidden layer.

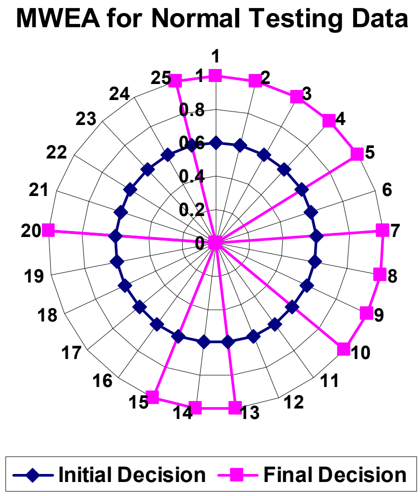


Fig.6: Decision Boundary for Normal Data

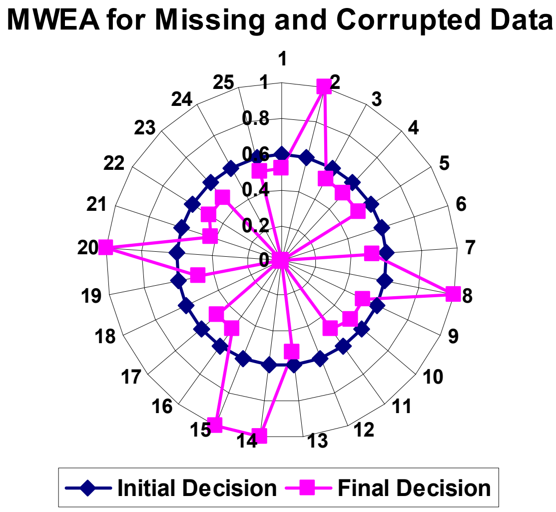


Fig.7: Decision Boundary for Missing or Corrupted Data

Table (5) shows effect of weight reduction factor $\xi$ on current training error for a fixed number of epochs, learning rate, and momentum factors. The value of $\xi = 2.5$ is chosen for training as it corresponds to the smallest error.

.

| I E J | T mm | TS N/mm | %USCA LF No Units | %USCA TH No Units | D mm | Decision Accept (A) Reject (R) |
|---|---|---|---|---|---|---|
| 7.14 | 2 | 205.16 | 74.5 | 28.60 | 0.727 | 1.00 (A) |
| 7.14 | 3 | 207.44 | 61.2 | 26.47 | 0.785 | 1.00 (A) |
| 7.14 | 4 | 208.21 | 77.6 | 25.05 | 0.643 | 1.00 (A) |
| 7.14 | 5 | 208.30 | 91.5 | 24.98 | 0.573 | 1.00 (A) |
| 7.14 | 6 | 208.14 | 96.8 | 25.48 | 0.525 | 1.00 (A) |
| 14.3 | 2 | 190.00 | 67.0 | 35.40 | 1.210 | 0.00 (R) |
| 14.3 | 3 | 205.37 | 34.0 | 32.40 | 1.157 | 1.00 (A) |
| 14.3 | 4 | 205.94 | 36.9 | 33.13 | 1.333 | 1.00 (A) |
| 14.3 | 5 | 206.78 | 87.5 | 30.05 | 0.858 | 1.00 (A) |
| 14.3 | 6 | 207.80 | 96.1 | 26.43 | 0.615 | 1.00 (A) |
| 28.6 | 2 | 150.74 | 49.2 | 46.02 | 2.840 | 0.00 (R) |
| 28.6 | 3 | 152.18 | 79.1 | 45.75 | 2.686 | 0.00 (R) |
| 28.6 | 4 | 205.36 | 88.5 | 32.60 | 0.847 | 1.00 (A) |
| 28.6 | 5 | 205.67 | 59.5 | 35.01 | 1.540 | 1.00 (A) |
| 28.6 | 6 | 205.73 | 57.6 | 35.12 | 1.648 | 1.00 (A) |
| 42.0 | 2 | 150.68 | 46.0 | 46.04 | 2.850 | 0.00 (R) |
| 42.0 | 3 | 151.54 | 57.3 | 45.52 | 2.761 | 0.00 (R) |
| 42.0 | 4 | 153.07 | 63.6 | 44.76 | 2.610 | 0.00 (R) |
| 42.0 | 5 | 180.96 | 57.4 | 39.92 | 1.970 | 0.00 (R) |
| 42.0 | 6 | 205.76 | 42.4 | 34.57 | 1.488 | 1.00 (A) |
| 47.6 | 2 | 150.68 | 29.0 | 46.04 | 2.850 | 0.00 (R) |
| 47.6 | 3 | 151.25 | 34.2 | 45.70 | 2.789 | 0.00 (R) |
| 47.6 | 4 | 152.87 | 42.5 | 44.78 | 2.582 | 0.00 (R) |
| 47.6 | 5 | 153.44 | 37.0 | 45.07 | 2.710 | 0.00 (R) |
| 47.6 | 6 | 205.41 | 38.6 | 33.62 | 1.200 | 1.00 (A) |

Table2: Predicted data with classification

| I E J | T mm | TS N/mm | %USCA LF No Units | %USCA TH No Units | D mm | Decision Accept (A) Reject (R) |
|---|---|---|---|---|---|---|
| 7.14 | 2 | 0 | 74.5 | 28.60 | 0.727 | 0.52 (A) |
| 7.14 | 3 | 207.44 | 0 | 26.47 | 0.785 | 1.00 (A) |
| 7.14 | 4 | 208.21 | 77.6 | 0 | 0.643 | 0.52 (A) |
| 7.14 | 5 | 208.30 | 91.5 | 24.98 | 0 | 0.52 (A) |
| 7.14 | 6 | 208.14 | 96.8 | 0 | 0.525 | 0.52 (A) |
| 14.3 | 2 | 190.00 | 0 | 35.40 | 1.210 | 0.00 (R) |
| 14.3 | 3 | 0 | 34.0 | 32.40 | 1.157 | 0.52 (A) |
| 14.3 | 4 | 205.94 | 0 | 33.13 | 1.333 | 1.00 (A) |
| 14.3 | 5 | 206.78 | 87.5 | 0 | 0.858 | 0.52 (A) |
| 14.3 | 6 | 207.80 | 96.1 | 26.43 | 0 | 0.52 (A) |
| 28.6 | 2 | 150.74 | 49.2 | 0 | 2.840 | 0.48 (R) |
| 28.6 | 3 | 152.18 | 0 | 45.75 | 2.686 | 0.01 (R) |
| 28.6 | 4 | 0 | 88.5 | 32.60 | 0.847 | 0.52 (A) |
| 28.6 | 5 | 205.67 | 0 | 35.01 | 1.540 | 1.00 (A) |
| 28.6 | 6 | 205.73 | 57.6 | 0 | 1.648 | 1.00 (A) |
| 42.0 | 2 | 150.68 | 46.0 | 46.04 | 0 | 0.48 (R) |
| 42.0 | 3 | 151.54 | 57.3 | 0 | 2.761 | 0.48 (R) |
| 42.0 | 4 | 153.07 | 0 | 44.76 | 2.610 | 0.01 (R) |
| 42.0 | 5 | 0 | 57.4 | 39.92 | 1.970 | 0.48 (R) |
| 42.0 | 6 | 205.76 | 0 | 34.57 | 1.488 | 1.00 (A) |
| 47.6 | 2 | 150.68 | 29.0 | 0 | 2.850 | 0.42 (R) |
| 47.6 | 3 | 151.25 | 34.2 | 45.70 | 0 | 0.48 (R) |
| 47.6 | 4 | 152.87 | 42.5 | 0 | 2.582 | 0.48 (R) |
| 47.6 | 5 | 153.44 | 0 | 45.07 | 2.710 | 0.00 (R) |
| 47.6 | 6 | 0 | 38.6 | 33.62 | 1.200 | 0.52 (A) |

Table3: Classification of Missing or Corrupted data

| Decision (Initial) (Rounded) | Decision (Trained) Normal ( Rounded) | Decision (Trained) Corrupt (Rounded) |
|---|---|---|
| 0.60 | 1.00 | 0.52 |
| 0.60 | 1.00 | 1.00 |
| 0.60 | 1.00 | 0.52 |
| 0.60 | 1.00 | 0.52 |
| 0.60 | 1.00 | 0.52 |
| 0.60 | 0.00 | 0.00 |
| 0.60 | 1.00 | 0.52 |
| 0.60 | 1.00 | 1.00 |
| 0.60 | 1.00 | 0.52 |
| 0.60 | 1.00 | 0.52 |
| 0.60 | 0.00 | 0.48 |
| 0.60 | 0.00 | 0.01 |
| 0.60 | 1.00 | 0.52 |
| 0.60 | 1.00 | 1.00 |
| 0.60 | 1.00 | 1.00 |
| 0.60 | 0.00 | 0.48 |
| 0.60 | 0.00 | 0.48 |
| 0.60 | 0.00 | 0.01 |
| 0.60 | 0.00 | 0.48 |
| 0.60 | 1.00 | 1.00 |
| 0.60 | 0.00 | 0.42 |
| 0.60 | 0.00 | 0.48 |
| 0.60 | 0.00 | 0.48 |
| 0.60 | 0.00 | 0.00 |
| 0.60 | 1.00 | 0.52 |

Table4: Decision Mapping

| Reduction Factor ξ | Percentage error |
|---|---|
| 0.5 | 5.8 |
| 1.5 | 0.32 |
| 2.5 | 0.3 |
| 3.5 | 0.36 |
| 4.5 | 0.43 |
| 5.5 | 0.33 |

Table5: Reduction Factor versus Error

Different reduction factor values are chosen and the network response is observed. The value is increased if the weights are not diminishing and decreased if all the weights are forced to zero. The results shown in Table (5) indicates that the value of 2.5 is the one that provides minimum error with network stability for the used network model, learning rate, momentum, number of used neurons and type of training and testing data.

Table (7) shows a comparison between MWEA and Backpropagation (BP). Starting from same decision values prior to training, the network model in Figure (2) trained using BP and MEWA algorithms for the same number of epochs. The results shown in Table (7) illustrates that for the used training data and network model, MWEA managed to correctly classify all corrupted data records in comparison to BP which failed to correctly classify 5 out of 25 data records.

Figure (8) show weights characteristics for BP trained network. By comparison to Figures (2) and (3), it is clear that MWEA produced a more uniform weight characteristic via pruning and weight elimination with better classification and stability.

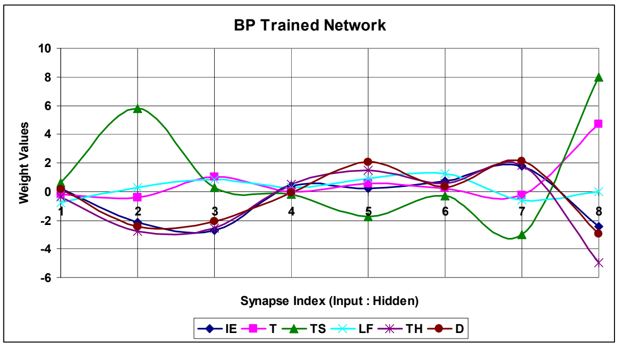


Fig.8: Trained Network Weight Connections Using BP.

| Decision (Initial) BP (Rounded) | Decision (Initial) MWEA ( Rounded) | Decision (Trained) Corrupted (BP) (Rounded) | Decision (Trained) Corrupted (MWEA) (Rounded) |
|---|---|---|---|
| 0.60 | 0.60 | 0.21 | 0.52 |
| 0.60 | 0.60 | 0.93 | 1.00 |
| 0.60 | 0.60 | 0.99 | 0.52 |
| 0.60 | 0.60 | 0.99 | 0.52 |
| 0.60 | 0.60 | 0.99 | 0.52 |
| 0.60 | 0.60 | 0.10 | 0.00 |
| 0.60 | 0.60 | 0.01 | 0.52 |
| 0.60 | 0.60 | 0.92 | 1.00 |
| 0.60 | 0.60 | 0.99 | 0.52 |
| 0.60 | 0.60 | 0.99 | 0.52 |
| 0.60 | 0.60 | 0.30 | 0.48 |
| 0.60 | 0.60 | 0.10 | 0.01 |
| 0.60 | 0.60 | 0.21 | 0.52 |
| 0.60 | 0.60 | 0.91 | 1.00 |
| 0.60 | 0.60 | 0.99 | 1.00 |
| 0.60 | 0.60 | 0.21 | 0.48 |
| 0.60 | 0.60 | 0.33 | 0.48 |
| 0.60 | 0.60 | 0.10 | 0.01 |
| 0.60 | 0.60 | 0.01 | 0.48 |
| 0.60 | 0.60 | 0.91 | 1.00 |
| 0.60 | 0.60 | 0.27 | 0.42 |
| 0.60 | 0.60 | 0.20 | 0.48 |
| 0.60 | 0.60 | 0.69 | 0.48 |
| 0.60 | 0.60 | 0.1 | 0.00 |
| 0.60 | 0.60 | 0.01 | 0.52 |

Figure (9) presents a comparison between the MWEA and BP output weight characteristics with clear indication of more stable network under MWEA training.

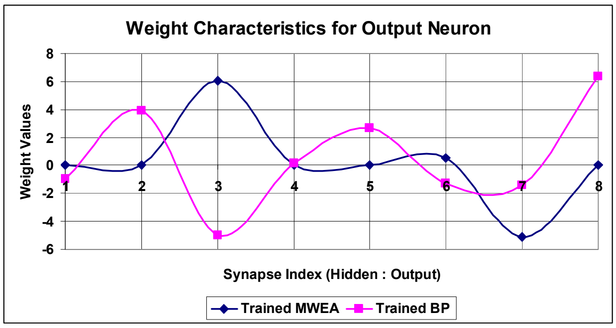


Fig.9: Comparison of Weight Characteristics at Output Neuron.

Figure (10) show BP decision characteristic in reference to the initial decision boundary ring. When compared to Figure (7), a clear pattern difference appears in the decision distribution around the initial boundary values with MWEA producing a much more uniform decision values.

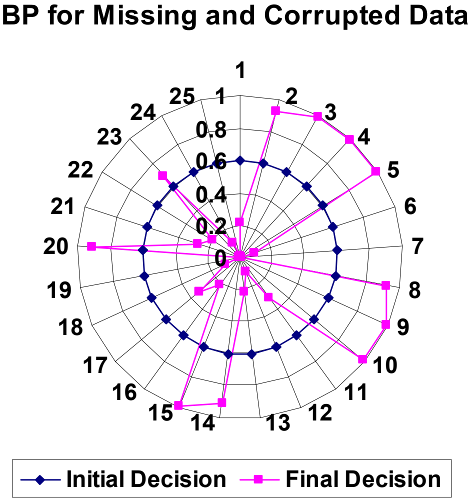


Fig.10: Decision Boundary for Missing or Corrupted Data

In conclusion, the overall system is devised to make use of intelligent algorithms in classifying damage in composites whereby the absence of one testing data file is complemented by the experience gained within the network. This system makes full use of the associative and predictive properties of Neural Networks for the application discussed in this paper where Neural Networks served to save time and effort in determining the extent of damage in composite structures, which is invaluable if the received data is corrupted or part of it is missing.

*References:*
[1] J. Davim, J. Rubio, A. Abrão, A novel approach based on digital image to evaluate the delamination factor after drilling compositen laminates, Composites and Science and Technology, Vol.67, 2007, 1939-1945.
[2] J. Lee, S. Kim, Structural damage detection in the frequency domain using neural networks, Journal of Intelligent Materials Systems and Structures, Vol. 18, 2007, pp. 785-792.
[3] Z. Li, X.Yang, Damage identification for beams using ANN based on statistical property of structural responses, Computers and Structures, Vol. 86, 2008, pp. 64-71.
[4] Y. Liu, A. Li, Q. Fei, Y. Ding, Feature extraction and damage alarming using time series analysis, Journal of Southeast University, Vol. 23, No.1, 2007, pp. 86-91.
[5] K. Nair, A. Kiremidjian, Time series based structural damage detection algorithm using Gaussian mixtures modeling, Journal of Dynamic Systems, Measurement, and Control, Vol. 129, 2007, pp. 285-293.
[6] Q. Liu, C. Xun, N. Gindy, Fuzzy pattern recognition of AE signals for grinding burn, International Journal of Machine Tools & Manufacture, Vol.45, 2005, pp.811-818.
[7] Rami Haj-Ali, Hoan-Kee Kim, Nonlinear constitutive models for FRP composites using artificial neural networks, Journal of Mechanics of Materials, Vol.39, 2007, pp.1035-1042.
[8] D. Chakraborty, Artificial neural network based delamination prediction in laminated composites, Material Design, Vol.26, No.1, 2005, pp.1-7.
[9] P.F. Pai, System Reliability Forecasting by Support Vector Machines with Genetic Algorithms, Mathematical and Computer Modeling, Vol.43, 2006, pp.262-274.
[10] Jie Yang , Hongjian Qu , Liang Zhou, Research on the Evaluation Methods of Bid of Construction Project Based on Improved BP Neural Network, WSEAS Transactions on Computers, Vol. 1, No. 9, 2010, pp. 93-102.
[11] Dominic Palmer-Brown, Sin Wee Lee, Chrisina Draganova, Miao Kang, Modal Learning Neural Networks, WSEAS Transactions on Computers, VOL.2, No.8, 2009, pp. 222-236.
[12] Li Xiaojuan Chen Cunshe, An improved BP Neural Network for Wastewater Bacteria Recognition Based on Microscopic Image Analysis, WSEAS Transactions on Computers, Vol.2, No.8, 2009, pp.237-247.
[13] José De Jesús Rubio Avila, Andrés Ferreyra Ramírez And Carlos Avilés-Cruz, Nonlinear system identification with a feedforward neural network and an optimal bounded ellipsoid algorithm, Key WSEAS Transactions on Computers, Vol.5, No.7, 2008, pp.542-551.
[14] Long Yu, Jian Xiao, Song Wang, Reduced-Set Vector-Based Interval Type-2 Fuzzy Neural Network, WSEAS Transactions on Computers, Vol.10, No.7, 2008, pp.1574-1583.
[15] Hazem M. El-Bakry, Nikos Mastorakis, A New Technique for Detecting Dental Diseases by using High Speed Artificial Neural Networks, WSEAS Transactions on Computers, Vol.12, No.7, 2008, pp.1977-1987.
[16] O. Wang, J. Lai a, D. Sun, Artificial neural network models for predicting flow stress and microstructure evolution of a hydrogenized titanium alloy, Key Engineering Materials, Vol.353, 2007, pp.541-544.
[17] F. Han, Q. Ling, D. Huang, Modified constrained learning algorithms incorporating additional functional constraints into neural networks, Information Sciences, Vol.178, 2008, pp.907–919.
[18] K. Suzuki, Determining the receptive field of a neural filter, J. Neural Eng. Vol.1, 2004, pp.228–237.
[19] J. Lee, H. Lee, J. Kim, D. Nam, C. Park, Self-Organizing neural Networks by Construction and Pruning, IEICE Trans. Inf.& Syst., Vol. E87, No. 11, 2004, pp.2489-2498.