ABSTRACT

As a result of their black-box nature, neural networks resist traditional methods of certification and therefore cannot be used in safety critical applications. This situation is undesirable as neural networks can provide an effective solution to many engineering problems. The object of the current paper is to explore the possibility of quantifying and qualifying the reliability of neural networks by a means outside the traditional framework. The approach used here will follow Ben-Haim’s information-gap theory of uncertainty. This is a non-probabilistic approach which may lend itself well to certification of black-box systems. The approach is demonstrated here on a neural network regression model of the process of pre-sliding friction between solids.

INTRODUCTION

Over the past few years there has been an enormous increase in the number of proposed application areas for Artificial Neural Networks (ANNs) within the sphere of engineering; particularly in control systems, fault diagnostics and “smart structures”. Recent examples include applications to aircraft wing damage detection [1, 2], damage assessment in steel frame structures [3], and the generic use of such systems for changes in structural parameters [4,5]. However, despite intensive activity within both academia and industrial R&D environments, the uptake of ANN technology by industry has been minimal, particularly within Europe. One of the main reasons for this resistance is the apparent “black-box” nature of ANNs which makes them resistant to traditional methods of certification and therefore severely restricts their application to safety-critical systems. For example, within aerospace applications, the decisions made by a neural network would determine if the aircraft should be withdrawn from service for detailed inspection or repair. The consequences of errors by the network are potentially costly and have obvious life-safety implications. There is clearly some need for an evaluation of the network dependability. ANNs often perform well within an environment where the training data sets and test data sets are well matched to each other. However in “real-world” applications it is precisely the unpredictable nature of variations (whether environmental or instrumentation dominated) between what the network has been trained on, and the current data presentation, that causes most concern. A typical practical neural network consists of a non-linear mapping between input and output vectors. Although a huge number of different approaches are possible [6,7], the Multi-Layer Perceptron (MLP) remains probably the most widely used network architecture. An MLP network can model any continuous function, to arbitrary accuracy, provided there is a sufficient number of hidden nodes, and that the network weights and biases are appropriate [8]. It is in the potential complexity of the transfer function, due to the distribution of connection weights combined with the inherent non-linearity of an MLP structure, that the problem of network characterization arises. No matter how extensive the training and testing, there may always be a suspicion that a particular combination of input conditions will arise (as yet not presented to the network) that would lead to unusual and unforeseen outputs. Most statistical methods of reliability assessment rely on the availability of probability distributions and the concept of probability of failure or Mean Time To Failure (MTTF) [9]. However, these probability distributions are usually estimated from the low order moments of the data, typically mean and standard deviation, and do not necessarily represent the tails of the distributions with any accuracy. This represents a problem as the events of interest in reliability are extreme events associated with outliers in the probability distribution.

The approach in the current work is to adopt a non-probabilistic approach, based on the theory of convex models and information-gap uncertainty as pioneered by Ben-Haim [10, 11, 12]. The response of a simple MLP network
trained on input/output data representing the force/displacement characteristics of a pair of surfaces in the regime of pre-sliding friction is investigated. After conventional training and testing of the network, the propagation of intervalised [13] vectors through the network structure is studied. These interval inputs represent the set of all input conditions that can arise (within certain prescribed limits) for presentation to the network. The intervalised network response therefore provides a conservative estimate of all possible responses to the network and therefore must contain the worst case error scenario. Additionally, if a particular amount of uncertainty can be tolerated, it is possible to reduce the best case error; this represents the opportunity of the system in Ben-Haim’s terminology.

NETWORK STRUCTURE

The current approach is one founded in system identification, i.e. to identify a model that generates a valid representation between given input and output data. We use an ANN with an MLP structure to provide a non-linear black-box approach to finding the transfer function between a vector of inputs \( x_i \) and a vector of outputs \( y_i \).

The input/output data was modelled by fitting to a NARX model (Non-linear, AutoRegressive with eXogenous inputs), i.e. a particular instantaneous output value \( \hat{y}_i \) is determined partly by the previous output values \( y_i \) and partly by previous input values \( x_i \) [14].

\[
\hat{y}_i = F [y_{i-1}, \ldots, y_{i-n_y}, x_{i-1}, \ldots, x_{i-n_x}]
\]

*Eqn. (1)*

For a linear system:

\[
\hat{y}_i = a_1 y_{i-1} + a_2 y_{i-2} + \ldots + a_n y_{i-n_y} + b_1 x_{i-1} + b_2 x_{i-2} + \ldots + b_n x_{i-n_x}
\]

*Eqn. (2)*

The requirement is to find the coefficients \( a \) and \( b \) that will satisfy the above equations (1) and (2).

For a non-linear MLP with a single hidden layer with a non-linear \( \tanh \) activation function, and an output layer with a linear activation function, the output is given by:

\[
\hat{y}_i = b_0 + \sum_{j=1}^{n_h} w_{j} \cdot \tanh \left[ \sum_{k=1}^{n_i} v_{jk} y_{i-k} + \sum_{m=0}^{n_i-1} u_{jm} x_{i-m} + b_j \right]
\]

*Eqn. (3)*

The first stage of network operation for system identification is to establish the appropriate values for the connection weights and biases \( u_{jm}, v_{jk}, w_{j}, b_{j} \). This is the training or learning phase and is accomplished using an iterative non-linear optimisation procedure to find the minimum in the error function between actual network output and the current target vector. The MLP structures investigated were generated using algorithms programmed in MATLAB™ using the NETLAB [8] toolbox. Optimisation was performed using weight decay regularisation [6,7,8] and scaled conjugate gradient descent [8].

The type of error function used for NETLAB optimisation depends on the characteristics of the activation function of the output layer. In this case since a linear activation function was utilised; the mean square error function (MSE) was given by:

\[
MSE = \frac{100}{n \cdot \sigma^2} \sum_{i=1}^{n} (t_i - \hat{y}_i)^2
\]

*Eqn. (4)*

where \( \hat{y}_i \) are the network outputs and \( t_i \) are the desired target values.
This definition of MSE has the useful property that if the mean of the output signal is used as the model, then the MSE = 100. Experience shows that an MSE of less than 5.0 indicates good agreement, whilst one of less than 1.0 indicates an excellent fit [14].

It is important to distinguish between two different methods for evaluating the errors between the predicted network output values and the target output values. These are the One Step Ahead (OSA) and the Model Predicted Output (MPO). The OSA prediction utilises all the previous \( y_i \) data to generate the network predictions of \( \hat{y}_i \) values, whilst the MPO prediction uses the previously calculated values of \( \hat{y}_i \) to feed back into the network. In practice it is the MPO error that provides a more stringent test of network performance. To illustrate the difference consider a network with 2 y-lags and 3 x-lags.

For the OSA prediction:

\[
\hat{y}_3 = F[y_2, y_1, x_3, x_2, x_1] \\
\hat{y}_4 = F[y_3, y_2, x_4, x_3, x_2] \\
\hat{y}_5 = F[y_4, y_3, x_5, x_4, x_3]
\]

Eqn. (5) a,b,c

For the MPO prediction:

(N.B. \( y_2 \) and \( y_1 \) are the initial values; all subsequent \( y \)-values are model predictions \( \hat{y}_i \))

\[
\hat{y}_3 = F[y_2, y_1, x_3, x_2, x_1] \\
\hat{y}_4 = F[\hat{y}_3, y_2, x_4, x_3, x_2] \\
\hat{y}_5 = F[\hat{y}_4, \hat{y}_3, x_5, x_4, x_3]
\]

Eqn. (6) a,b,c

After training was completed for all presentations of the training data set, the network weights and biases were fixed and the validation data was forward propagated through the network. The MPO error of the validation data set was used to judge the “best network” on criteria to be discussed later. Network propagation tests were then conducted on an independent data set, the so-called test data.

INTERVAL ARITHMETIC

After verifying the network performance to crisp (single-valued) input data, the next step in the analysis was to reformulate the test set input data as a series of interval number inputs. Interval numbers [13] occupy a range of the number line, and can be defined as an ordered pair of real numbers \([a, b]\) with \(a < b\) such that:

\[
[a, b] = \{x \mid a \leq x \leq b\}
\]

Eqn. (7)

Interval numbers have specific rules for the standard arithmetic operations of addition, subtraction, multiplication and division [13].

Each input value \(x_i\) of the test set was intervalised by a parameter \(\alpha\) such that:

\[
[x_{ia}, x_{ib}] = [(x_i - \alpha), (x_i + \alpha)]
\]

Eqn. (8)

The netlab routines to perform forward propagation through an MLP structure were modified to conform to the rules of interval arithmetic. The multiplication of an input matrix to a given layer of the network by the weight matrix for that layer, required careful consideration. Following the intervalised matrix product of input and weight matrix, the bias was added and the activation function applied to the upper and lower bounds separately (possible due to the monotonic nature of \(\tanh(x)\)). The next step was to calculate the network errors associated with the interval propagation. As with the crisp data inputs both the OSA and MPO errors were evaluated. There was a critical difference from the crisp case in that the routine checked which network output interval bound lay farthest from the current target. Within the terminology of information-gap analysis [10,11] the farthest bound (whether
higher or lower) to the target was assigned to the worst case error whilst the closest bound (whether higher or lower) to the target was assigned to the opportunity. If the interval actually contained the current target value then the closest bound was assigned an error value of zero.

**TRAINING DATA AND NETWORK TESTING**

The data set investigated was a model of pre-sliding friction data comprising 24000 pairs of points of input/output data the first 1000 of which are shown in figure 1. The x-data was a vector of microscopic displacement values ($\delta x$) between two plates in the regime of pre-sliding friction opposed by friction forces ($\delta F$) comprising the y-data. The data was split into three equal parts of 8000 points each to form the training set, the validation set and the test set.

![Figure 1: First 1000 points of the friction data set](image)

<table>
<thead>
<tr>
<th>y-lag</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>x-lag</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>102.86</td>
<td>101.08</td>
<td>190.95</td>
<td>187.76</td>
<td>173.81</td>
<td>132.50</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>28.18</td>
<td>187.26</td>
<td>137.61</td>
<td>147.42</td>
<td>170.12</td>
<td>87.76</td>
<td>143.76</td>
</tr>
<tr>
<td>2</td>
<td>22.16</td>
<td>5.81</td>
<td>5.22</td>
<td>4.77</td>
<td>4.49</td>
<td>6.45</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>17.51</td>
<td>3.33</td>
<td>5.30</td>
<td>7.13</td>
<td>5.40</td>
<td>5.75</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>16.10</td>
<td>3.21</td>
<td>7.73</td>
<td>5.34</td>
<td>5.82</td>
<td>4.57</td>
<td>7.03</td>
</tr>
<tr>
<td>5</td>
<td>15.34</td>
<td>3.22</td>
<td>10.25</td>
<td>4.85</td>
<td>9.03</td>
<td>5.26</td>
<td>8.52</td>
</tr>
<tr>
<td>6</td>
<td>14.06</td>
<td>3.26</td>
<td>9.63</td>
<td>3.67</td>
<td>5.97</td>
<td>4.32</td>
<td>4.50</td>
</tr>
</tbody>
</table>

*Table 1: Mean square MPO errors on validation data set, n_hidd=1.*

Training was performed exclusively on the training data set. The validation data was then used to investigate network performance on previously unseen data. An "optimal" network structure was then selected by looking for the lowest MPO mean square error from all the network structures. Finally a test data set was used to verify that this selected network operated in a satisfactory fashion (i.e. continued to return an acceptably low MPO error). The training set was normalised to lie in the range -1 to +1, and the validation and test data sets were scaled and translated by the same amounts. A series of MLP networks were trained such that the number of x-lags was in the range 0-6, the number of y-lags was in the range 0-6, and the number of hidden nodes was in the range 1-15. Each network was trained 10 times, therefore a total of 7350 different MLP networks were investigated. The OSA and MPO errors for both the training data and the validation data for all networks were calculated. Table 1 illustrates the lowest MPO errors from the validation data set for each of the 10 MLP networks.
The network with x-lag=2, y-lag=1, n-hidd=1 produced the lowest MPO error (the corresponding error on the training set was 3.18 thus illustrating good network generalisation).

This network structure was then investigated by training 1000 MLP networks on the training data, and analyzing the resulting network MPO errors on the test data set. The results are illustrated in figure 2 and summarized in table 2. This network was capable of providing consistent performance to the test data set.

![Figure 2: Distribution of test data set errors for [x_lag=2 y_lag=1, n_hidd=1]](image)

<table>
<thead>
<tr>
<th>Mean (%)</th>
<th>Variance (%)</th>
<th>Maximum (%)</th>
<th>Minimum (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.6347</td>
<td>0.30549</td>
<td>12.909</td>
<td>3.1369</td>
</tr>
</tbody>
</table>

**Table 2**: Statistics of MPO error distribution for [x_lag=2 y_lag=1, n_hidd=1]

**INTERVAL PROPAGATION AND VALIDATION THROUGH MONTE-CARLO SIMULATION**

After establishing consistent MLP network performance, the network structure response to intervalised input data was investigated. Considerable insight into the performance of the interval routines can be obtained by the observation of the time domain output data from the interval analysis and a number of time domain MPO outputs will be illustrated.

The performance of the interval calculations is checked by using Monte-Carlo simulations to add random noise to the input data. There is a very important issue here which is at what stage the random noise is added to the signal. In this system identification problem, we are utilising a NARX model which uses past values of input and output data to predict the future model output values. In generating the data matrix presented to the input of the MLP network, if the number of x-lags is greater than 1, then if the noise addition is performed before re-ordering, all the x; component values will be identical. That is to say the x3 value in equation 6(a) is identical to the x3 value in equation 6(b) and 6(c). This effect tends to reduce any dependency effects propagating through the network, since less variation in input space to the MLP network is being covered. This will be referred to herein as original x data noise addition. However if noise is added to the x; components after re-ordering then all of the x-lag inputs can vary randomly over the range of the current interval independently of each other i.e. the x3 values in equations 6(a), 6(b) and 6(c) are different from each other. This approach acts to increase dependency effects propagating through the network. Since the interval analysis technique considers variation across the largest possible range of values, we would expect better agreement with the second noise addition mechanism. This second approach will herein be referred to as independent x data noise addition. The first case considered consists of a relatively large range of interval values to exaggerate the effects.
Since the data set is normalized to ±1, an interval of ± 0.1 corresponds to a size of ± 10% which is a significant variation in the input data. Figure 3 illustrates the worst case and opportunity MPO errors. It is apparent that even the smallest interval investigated here (1%) produced a very large worst case MPO error of nearly 200%. This is of course completely unacceptable for a practical application, but will illustrate the characteristics of the time domain data more clearly than smaller intervals.

Figure 3: MPO Interval Output Errors, [x_lag=2 y_lag=1, n_hidd=1]

Figure 4: Interval outputs from MLP network interval size 0.02 [x_lag=2 y_lag=1, n_hidd=1]

A small Monte-Carlo simulation run using independent x data noise addition and 100 runs demonstrates that the interval bounds for the MPO network output (figure 5) contain all the crosses of the Monte-Carlo simulations. By using original x-data only noise addition in the Monte-Carlo simulation (figure 6) it is clear that the spread in outputs from the MLP network is significantly reduced.

Figure 5: Monte-Carlo simulation for interval size 0.02, 100 runs, independent x data noise addition

Figure 6: Monte-Carlo simulation for interval size 0.02, 100 runs, original x data noise addition
This demonstrates an important property of the interval analysis. The interval output bounds are conservative, i.e. it is possible to guarantee that the output bounds will contain the target, but the bounds may be larger than is required by purely physical constraints on the system. In the present example, there is no implication of error in the calculated interval bounds, merely that the physical spread of the data across the input mapping space of the MLP network is significantly lower than the maximum available spread.

A further illustration of this effect can be observed by introducing a large variation to the input test data matrix. Rather than adding random noise to the x-lag samples, the maximum positive interval size was added to the first column of x-lag, and the maximum negative interval size to the second column of x-lag. Then the process is reversed, i.e. add the maximum negative interval size to the first column of x-lag, and maximum positive interval size to the second column of x-lag

So the first data set is:

\[
\hat{y}_2 = F[y_1, (x_2 - \text{int}), (x_1 + \text{int})]
\]

\[
\hat{y}_3 = F[\hat{y}_2, (x_3 - \text{int}), (x_2 + \text{int})]
\]

\[
\hat{y}_4 = F[\hat{y}_3, (x_4 - \text{int}), (x_3 + \text{int})]
\]

Eqn. (9)

and the second data set is:

\[
\hat{y}_2 = F[y_1, (x_2 + \text{int}), (x_1 - \text{int})]
\]

\[
\hat{y}_3 = F[\hat{y}_2, (x_3 + \text{int}), (x_2 - \text{int})]
\]

\[
\hat{y}_4 = F[\hat{y}_3, (x_4 + \text{int}), (x_3 - \text{int})]
\]

Eqn. (10)

This simulates excitation over the greatest possible range of input values, independently, which is what the conservative interval analysis represents. The results in figure 7 show that the interval bounds are identical to the network MPO outputs under these extremum inputs.

![Figure 7: Comparison between interval bounds and network output for extremum input values.](image)

**Interval size 0 - 0.005 step size 0.0005**

Since the data set was normalized to ±1, an interval of ± 0.005 corresponded to a size of ± 0.5 % which is a small variation in the input data. This smaller interval size provided a better output fit (in the MPO error sense) as illustrated in figure 8.

The more usual information gap plot of worst case error and opportunity is illustrated in figure 9. Setting a maximum tolerable worst case error of 10 %, then the corresponding interval size is 0.12 % with an opportunity of 1.32 %.
From figures 8 and 9 it is possible to conclude that the MLP network is tolerant of only small variations on the input data if the MPO mean square noise figure is to be kept within an acceptable limit. However, it has already been demonstrated that since the interval bounds are conservative, there may be some scope for improvement depending on the physical situation. Consider now a Monte-Carlo simulation for an interval size of 0.0012 adding noise to the original x-data only. Figure 10 shows that the Monte-Carlo results fall well inside the conservative interval bounds. It is very likely therefore that the information-gap graph of figure 9 is overly pessimistic. Therefore the next step was to conduct a Monte-Carlo investigation across a range of interval sizes from 0 to 0.05 with a step size of 0.005 and perform the information gap analysis on the Monte-Carlo output data. The results are presented in figure 11.

Setting a maximum tolerable worst case error of 10 %, then the corresponding Monte-Carlo interval size is 3.53 % with an opportunity of 0.80 %. For the worst case error this represents almost a factor 30 improvement on error tolerance over the conservative interval bound case. Of course there is a danger in the Monte-Carlo approach that the absolute worst case combination of inputs that produces the greatest change in output value has not been observed. An infinite number of runs would be required to guarantee this condition. In contrast the true interval bound outputs are guaranteed to contain the absolute worst case scenario. In the present example, the propagation of dependency effects through the MLP network makes it highly likely that the true worst case output
bounds are closer to the Monte-Carlo simulations. However to be certain, it will be necessary to reduce the dependency propagation effects, and therefore tighten the limits of the true conservative interval bound outputs.

CONCLUSIONS

This paper has presented results to illustrate a technique for evaluating the tolerance of a Multi-Layer-Perceptron neural network to noise on its input data. Using input/output data representing the displacement/force characteristics for a pair of surfaces in the regime of pre-sliding friction, a large number of MLP networks with different numbers of data point lags and internal network hidden nodes were trained. Having identified the MLP with the best performance (quantified from the mean square error of the model predicted output from the network), the network was then tested for stability. Input data was converted into interval form to represent all possible input conditions for a given level of noise. The intervalised input data was then propagated through the MLP network, and the output results compared with Monte-Carlo simulations of noise propagating through the MLP. It was found that the output interval bounds were conservative, i.e. they always contained both the crisp (single-valued) output and the results of the Monte-Carlo simulations. However due to dependency effects, the output interval bounds were always significantly larger than the observed spread from the Monte-Carlo simulations. Using the information-gap approach, we conclude that to maintain a worst case network output MPO error below 10 % required a tight tolerance (0.12 %) on the input data when using the true interval bounds. Performing the same analysis on the Monte-Carlo simulation allowed the tolerance to be relaxed to 3.53 % on the input data to give a worst case MPO error of 10 %. Given the effects of dependency on the interval analysis propagation through the network, it is likely that the true tolerance of the network lies closer to the Monte-Carlo simulation values. Future work will focus on methods to try to reduce the dependency effects and therefore lower the spread of the true conservative interval bound outputs.

ACKNOWLEDGEMENTS

This work was supported by EPSRC grant number RA 013700. Data for the project was supplied by Farid Al-Bender of The Catholic University of Leuven, and funded by Volkswagen Stiftung. The authors would also like to express their thanks to Ian Nabney of Aston University (UK) for making freely available his NETLAB software.

REFERENCES