Sensitivity Analysis of Multilayer Perceptron to Input and Weight Perturbations

Xiaoqin Zeng and Daniel S. Yeung, *Senior Member, IEEE*

***Abstract*—An important issue in the design and implementation of a neural network is the sensitivity of its output to input and weight perturbations. In this paper, we discuss the sensitivity of the most popular and general feedforward neural networks—multilayer perceptron (MLP). The sensitivity is defined as the mathematical expectation of the output errors of the MLP due to input and weight perturbations with respect to all input and weight values in a given continuous interval. The sensitivity for a single neuron is discussed first and an analytical expression that is a function of the absolute values of input and weight perturbations is approximately derived. Then an algorithm is given to compute the sensitivity for the entire MLP. As intuitively expected, the sensitivity increases with input and weight perturbations, but the increase has an upper bound that is determined by the structural configuration of the MLP, namely the number of neurons per layer and the number of layers. There exists an optimalvalue for the number of neurons in a layer, which yields the highest sensitivity value. The effect caused by the number of layers is quite unexpected. The sensitivity of a neural network may decrease at first and then almost keeps constant while the number increases.**

***Index Terms*—Multilayer perceptron (MLP), neural networks, sensitivity analysis.**

I. Introduction

GENERALLY a neural network is capable of setting an input–output mapping by adjusting its connection weights. How can network designers predict the effect of input and weight perturbations on neural networks' output, which are unavoidable because of limited precision of digital and analog hardware? What is an appropriate measure to quantify a neural network's error-tolerance and generalization abilities? Is there any guideline for designing a more robust neural network? The sensitivity analysis of neural networks' output to input and weight perturbations is expected to provide some answers to these questions. But how to define and quantify an appropriate sensitivity for a neural network is still a problem in need of exploring. This paper investigates these issues for the most popular and general feedforward multilayer perceptron (MLP) networks.

In the past decade, a number of studies on the sensitivity of neural networks emerged [1]–[13]. For example, Stevenson *et al.* [1] made use of hypersphere as a mathematical model to theoretically analyze the sensitivity of Madalines. They defined sensitivity as the probability of erroneous output of Madalines, which was expressed as a function of the percentage error in the weights, under the assumption that input and weight perturbations were small and the number of Adalines per layer was sufficiently large. Cheng and Yeung [5] also used a geometrical technique to investigate the sensitivity of Neocognitrons by modifying the hypersphere model. Unfortunately the hypersphere model they used fails for the case of the MLP because its input and weight vectors generally do not span a hypersphere surface or volume. Piche [3] employed a statistical rather than a geometrical argument to analyze the effects of weight errors in Madalines. He made the assumption that inputs and weights as well as their errors are all independently identically distributed (i.i.d.) with mean zero. Based on such a stochastic model and under the condition that both input and weight errors were small enough, Piche derived an analytical expression for the sensitivity as the ratio of the variance of the output error to the variance of the output. Recently Yeung and Sun [14] generalized Piche's method to the neural networks with antisymmetric squashing activation functions and they removed the restriction on the magnitude of input and output errors. Their results are only applicable to an ensemble of neural networks, but not to an individual one because of too strong assumptions made by the stochastic model.

For the sensitivity studies on MLPs, there are basically two approaches. One is an analytical approach in which sensitivity is defined as a partial derivative of output to input. For example, Hashem [6] and Fu and Chen [7] computed the partial derivative one layer at a time, starting from the output layer and proceeding backward (backward chaining is used) toward the input layer. Zurada *et al.* [8] and Engelbrecht *et al.* [9], [10] also used this kind of sensitivity results to delete redundant inputs and prune the architecture of the MLP. However, this approach is only applicable to a specific MLP, i.e., weights being fixed and its input error tends to zero. The other approach is statistically based. For example, Choi and Choi [8] introduced a statistical sensitivity measure for the MLPs with differentiable activation functions. The sensitivity is defined as the ratio of the standard deviation of output errors to the standard deviation of the weight or input errors under the condition that the latter tends to zero. They obtained a formula for the sensitivity of a single-output MLP with fixed weights and a given input pattern. All in all, these two approaches are only useful for measuring the output error with respect to a given input pattern for a specific MLP. They are not able to measure the expected output error with respect to the overall input patterns, which is an important factor in considering the performance measurement of neural networks, especially for continuous neural networks like MLPs.

In this paper, we propose a new approach to theoretically and systematically analyze the sensitivity of the MLP. Instead of using a hypersphere as in [1] and [5], a hypercube is employed as

Manuscript received November 18, 2000; revised May 14, 2001. This work was supported by Hong Kong CERG 5114/97E.

The authors are with the Department of Computing, Hong Kong Polytechnic University, Hong Kong (e-mail: csxzeng@comp.polyu.edu.hk; csdaniel@comp.polyu.edu.hk).

Publisher Item Identifier S 1045-9227(01)09521-2.

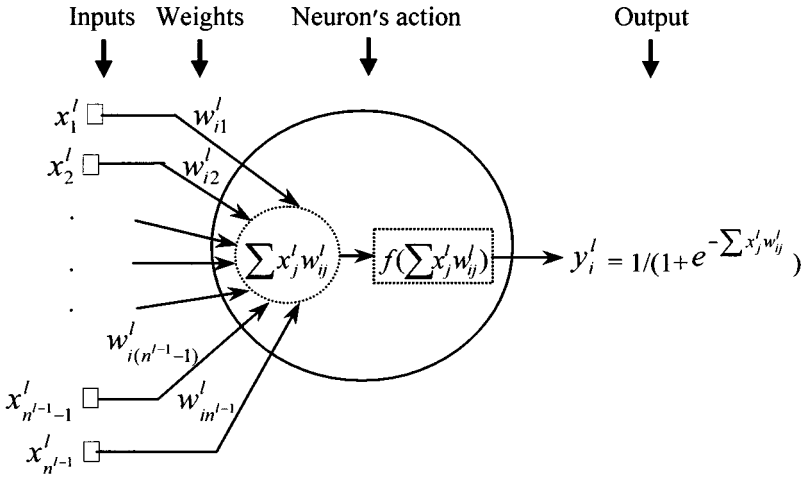

Fig. 1. The structure of a neuron (ith neuron in layer l).

the mathematical model to represent the MLPs input and weight space because they usually fall into a continuous interval. Our approach is different from the aforesaid ones and it offers certain advantages over them. For example, it does not demand the input perturbation to be very small which is required by the approach using partial derivative and the mathematical expectation used in our approach reflects the network's output error more directly and precisely than the variance does. Moreover, the sensitivity defined in this paper can be caused by input perturbation or weight perturbation or both and it is applicable not only to an ensemble of MLPs by regarding both input and weight as statistical variables, but also to a specific MLP when weights are fixed. This paper investigates the sensitivity behavior of an ensemble of MLPs.

The organization of this paper is as follows. The notation and structure of the MLP are briefly described in Section II. The definition of sensitivity, the derived analytical expression for the sensitivity of a single neuron and an algorithm for the computation of the sensitivity of an MLP are given in Section III. In Section IV, we analyze the sensitivity to input and weight perturbations for both a single neuron and an entire MLP. Experimental results that support the theoretical results are also presented in this section. Section V concludes the paper.

II. The MLP Model

A. Notation

Generally, an MLP can be assumed to have $L\,(L \geq 1)$ layers and each layer $l\,(1 \leq l \leq L)$ has $n^l\,(n^l \geq 1)$ neurons. The form $n^0 - n^1 - \cdots - n^L$ is used to represent an MLP with certain structural configuration, in which each $n^l\ (0 \leq l \leq L)$ not only indicates the number of neurons but also represents a layer from left to right including the input layer. Layer 0 is the input layer and layer L is the output layer. For a neuron $i\ (1 \leq i \leq n^l)$ in layer l, its input vector is $X^l = (x_1^l, \ldots, x_{n^{l-1}}^l)^T$, its weight vector is $W_i^l = (w_{i1}, \ldots, w_{in^{l-1}}^l)^T$ and its output is $y_i^l = f(X^{l*}W_i^l)$ where $f(\cdot)$ is a differentiable activation function. For a layer l, all neurons have the same input vector X^l and different neurons have different weight vectors that form the layer's weight set $W^l = \{W_1^l, \ldots, W_{n^l}^l\}$. The output vector is $Y^l = (y_1^l, \ldots, y_{n^l}^l)^T$. For the entire network, its input is X^1 or Y^0, its weight is $W = W^1 \cup \cdots \cup W^L$ and its output is Y^L. With regard to perturbations, let $\Delta X^l = (\Delta x_1^l, \ldots, \Delta x_{n^{l-1}}^l)^T$, $\Delta W_i^l = (\Delta w_{i1}^l, \ldots, \Delta w_{in^{l-1}}^l)^T$, $\Delta W^l = \{\Delta W_1^l, \ldots, \Delta W_{n^l}^l\}$, $\Delta W = \Delta W^1 \cup \cdots \cup \Delta W^L$ and $\Delta Y^l = (\Delta y_1^l, \ldots, \Delta y_{n^l}^l)^T$ be the corresponding errors for input weight and output, respectively.

B. Structure

The structure of a single neuron is shown in Fig. 1. With neurons as cells, the structure of an MLP is sketched in Fig. 2. Neurons in the MLP are organized into layers by linking them together. Links only exist between neurons of two adjacent layers and there is no link between neurons in the same layer and in any two nonadjacent layers. All neurons in a layer are linked to every neuron in the immediately preceding layer and are also linked to every neuron in the immediately succeeding layer. At each layer except the input layer, the inputs of each neuron are the outputs of the neurons in the previous layer, namely $X^l = Y^{l-1}\,(l \geq 1)$, with $X^l \in [0,1]^{n^{l-1}}$ because the activation function chosen for this study is the most commonly used sigmoidal function, that is

$$f(x) = \frac{1}{1+e^{-x}}. \tag{1}$$

III. Sensitivity

When considering a neural network for an application, perturbation, and imprecision are unavoidable. Sensitivity is actually one of the measures being used to evaluate the degree of the response of a network to the perturbation and imprecision of its parameters. For the case of a single neuron, the most direct and natural way to express it is the difference of its perturbed and unperturbed output

$$\begin{aligned}\Delta y_i^l &= g\left(X^l, \Delta X^l, W_i^l, \Delta W_i^l\right)\\ &= f\left(\left(X^l + \Delta X^l\right)^*\left(W_i^l + \Delta W_i^l\right)\right)\\ &\quad - f\left(X^{l*}W_i^l\right).\end{aligned} \tag{2}$$

By (2), Δy_i^l can be readily computed if the values of X^l, ΔX^l, W_i^l and ΔW_i^l are all known. But in real-life situations,

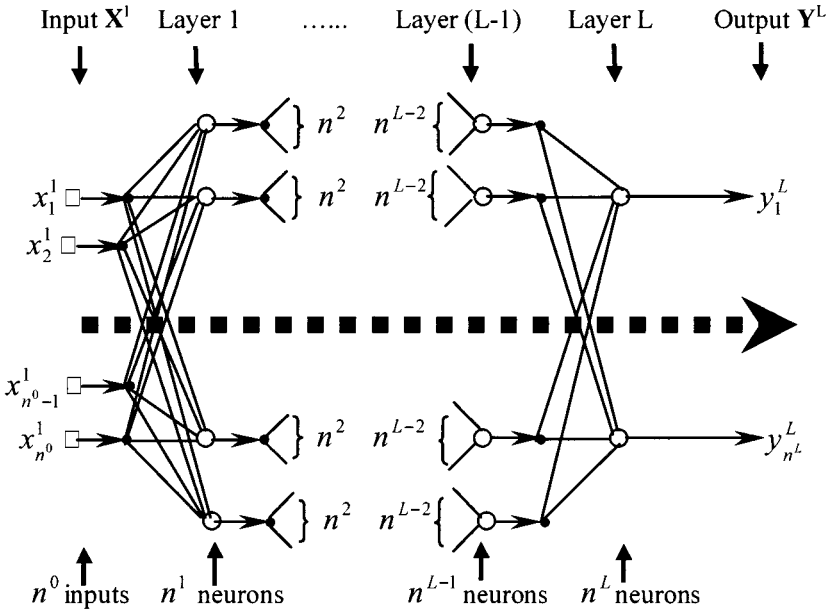

Fig. 2. The structure of an MLP.

some of these values are not always available. For example, before an MLP is trained, not much information is known about its weights except the interval from which the weights' values may be drawn. Even for those known values, it may be unnecessary to compute every Δy_i^l. For example in evaluating a network's performance, it is unnecessary to compute Δy_i^l with an individual input. Rather it would be more desirable to find the relationship between the output error and the input or weight error or both in an ensemble sense by taking all possible inputs and/or weights into consideration.

It is obvious that (2) gives a relationship between Δy_i^l and ΔX^l and ΔW_i^l. The problem is to define the sensitivity that keeps this relationship but, in the meanwhile, without reference to X^l and W_i^l. Our approach is to consider the mathematical expectation of Δy_i^l with respect to all possible inputs X^l and weights W_i^l in a given interval. We use a bottom-up way to define the sensitivity of different granules in the MLP. First is the sensitivity of a single neuron, then the sensitivity of a layer and finally the sensitivity of the entire MLP.

A. Definition of Sensitivity

We first define the sensitivity for a single neuron as follow. For an input vector $X^l \in [0,1]^{n^{l-1}}$ and a weight vector $W_i^l \in [0,1]^{n^{l-1}}$ and given their corresponding absolute values of errors $|\Delta X^l| = (|\Delta x_1^l|, \ldots, |\Delta x_{n^{l-1}}^l|)^T$ and $|\Delta W_i^l| = (|\Delta w_{i1}^l|, \ldots, |\Delta w_{in^{l-1}}^l|)^T$, the sensitivity of neuron i in layer l is defined as the mathematical expectation of the absolute value of Δy_i^l with respect to X^l and W_i^l, which is expressed as

$$\begin{aligned} s_i^l =& E\left(\left|g\left(X^l, W_i^l\right)\right|\right) \\ =& E\left(\left|f\left(\left(X^l + \left|\Delta X^l\right|\right)^* \left(W_i^l + \left|\Delta W_i^l\right|\right)\right)\right.\right. \\ & \left.\left. - f\left(X^{l*} W_i^l\right)\right|\right). \end{aligned} \tag{3}$$

Obviously, s_i^l can be regarded as a function of $|\Delta X^l|$ and $|\Delta W_i^l|$ only, independent of any specific X^l and W_i^l. The definition in (3) is a general expression and the most commonly encountered situations are: neural networks with input or weight perturbations or both, neural networks without fixed weights before implementation or with fixed weights after training and their combinations. It may also take $|\Delta X^l|$ or $|\Delta W_i^l|$ as statistical variables if there is such a requirement for a particular situation.

Next, the definition of the sensitivity for a layer is given. For an input vector $X^l \in [0,1]^{n^{l-1}}$ and a weight vector set $W^l = \{W_i^l | W_i^l \in [0,1]^{n^{l-1}}, 1 \le i \le n^l\}$ and given their corresponding absolute values of error $|\Delta X^l| = (|\Delta x_1^l|, \ldots, |\Delta x_{n^{l-1}}^l|)^T$ and $|\Delta W^l| = \{|\Delta W_1^l|, \ldots, |\Delta W_{n^l}^l|\}$, the sensitivity of layer l is defined as

$$S^l = \left(s_1^l, \ldots, s_{n^l}^l\right)^T. \tag{4}$$

Here, S^l is a vector which is composed of the sensitivity values of all neurons in the layer l. The error ΔX^l $(l > 1)$ is in fact the sensitivity of the previous layer, that is $\Delta X^l = S^{l-1}$ $(l > 1)$. Since sensitivity is generally recognized as scalar rather than a vector, one may choose, for instance, $\text{norm}(S^l)$, $\max(S^l)$, $\text{avg}(S^l)$, or $\min(S^l)$ as the representation of a layer's sensitivity depending on the nature of the application involved.

Finally, the sensitivity for the entire network is defined as the sensitivity of its output layer. That is

$$S = S^L = \left(s_1^L, \ldots, s_{n^L}^L\right)^T. \tag{5}$$

For the purpose of exploring the behavior of the MLPs sensitivity, we assume, without loss of generality, input and weight to be in a small interval so as to simplify the formula derivations and the computer simulations. It is noticed that, in the MLP with the activation function $f(x) = 1/(1+e^{-x})$, the input interval

of all layers except the input layer is $[0,1]^{n^{l-1}}$ $(l \geq 2)$ and it is possible to limit the input of the first layer to be in $[0,1]^{n^0}$ because any continuous interval $[a,b]$ can be transformed to [0,1] by the one to one mapping: $(x_j^1 - a)/(b-a)$. So the assumption $X^l \in [0,1]^{n^{l-1}}$ $(l \geq 1)$ in the sensitivity definition is practically acceptable. According to the derivation of the sensitivity formula presented in the Appendix, it is easy to extend the interval of weight to any positive interval by directly substituting new lower and upper integral bounds for the integrals in the derivation. In extension to a negative interval, the sigmoid function's feature: $f(x) = 1 - f(-x)$ can be used to replace the integrand $f(x)$ by $(1 - f(-x))$ in the derivation.

B. Sensitivity for a Single Neuron

This section presents analytical expressions involved in the calculation of the sensitivity of a single neuron. A brief explanation of the derivation of the expressions is given here and the details of the derivation are given in the Appendix.

Now the sensitivity defined in (3) can be expressed as

$$\begin{aligned} s_i^l =& E\left(\left|g\left(X^l, W_i^l\right)\right|\right) \\ =& E\left(\left|f\left(\left(X^l + \left|\Delta X^l\right|\right)^* \left(W_i^l + \left|\Delta W_i^l\right|\right)\right)\right.\right. \\ & \left.\left. - f\left(X^{l*} W_i^l\right)\right|\right) \\ =& \int_{V_{X^l}} \int_{V_{W_i^l}} \varphi\left(X^l, W_i^l\right) \left| f\left(\left(X^l + \left|\Delta X^l\right|\right)\right.\right. \\ & \left.\left. {}^* \left(W_i^l + \left|\Delta W_i^l\right|\right)\right) \right. \\ & \left. - f\left(X^{l*} W_i^l\right)\right| dV_{X^l} dV_{W_i^l} \end{aligned} \tag{6}$$

where V_{X^l} and $V_{W_i^l}$ are the areas spanned by input and weight vectors. The integral area of the expectation is actually a hypercube $[0,1]^{2n^{l-1}}$ in $2n^{l-1}$-dimensional hyperspace by taking each element of X^l and W_i^l as a coordinate axis. For simplicity, we assume the input X^l and weight W_i^l are i.i.d. in the hypercube, so the density function is

$$\varphi\left(X^l, W_i^l\right) = 1. \tag{7}$$

In the derivation, we use the following approximations and integral formula to simplify the calculations:

$$\frac{1}{1+x} \approx \sum_{k=0}^{p} (-1)^k x^k \quad \text{for } |x| < 1 \; |x| > 1$$
$$\text{where } 1 \leq p < +\infty \tag{8}$$

$$e^x \approx \sum_{h=0}^{q} \frac{x^h}{h!}, \quad \text{where } 1 \leq q < +\infty \tag{9}$$

and

$$\int x^n e^{ax} dx = \sum_{r=0}^{n} \left(\frac{(-1)^r n!\, x^{n-r} e^{ax}}{(n-r)!\, a^{r+1}} \right) + C. \tag{10}$$

The derived result is

$$s_i^l \approx \sum_{k=1}^{p} (-1)^k \left(\prod_{j=1}^{n^{l-1}} \left(\sum_{h=1}^{q} \left(\frac{k^{h-1}}{h} \right) A \right) - \left(\sum_{h=1}^{q} \left(\frac{k^{h-1}}{h} \right) B \right)^{n^{l-1}} \right) \tag{11}$$

where we have (12), shown at the bottom of the page, and

$$B = \sum_{r=0}^{h-2} \frac{-e^{-k}}{(h-1-r)!\, k^{r+1}} + \frac{1 - e^{-k}}{k^h}. \tag{13}$$

C. Sensitivity for an Entire MLP

This section presents an algorithm for the computation of the sensitivity of an entire MLP.

From a global point of view, the sensitivity of an MLP is its ouput error of the output layer, which can be regarded as a function of the first layer's input error ΔX^1 and the total weight error ΔW. As the outputs of all neurons in a layer are inputs to each neuron in the next succeeding layer, the sensitivity of the output layer can be computed neuron by neuron from the first layer to the last layer. That is S^1 can be computed with $|\Delta X^1|$ and $|\Delta W^1|$ by (11)–(13), then S^2 with $|\Delta X^2| = S^1$ and $|\Delta W^2|$, etc., finally S^L with $|\Delta X^L| = S^{L-1}$ and $|\Delta W^L|$.

The algorithm MLP_SEN_ALG is given to compute the sensitivity. The input error $|\Delta X^1|$ or S^0 and the computed sensitivity S^l $(1 \leq l \leq L)$ can be arranged into a two-dimensional array. The total weight error $|\Delta W|$ can be arranged into a three-dimensional array. By controlling the correspondences between the iterative time and the arrays' subscript, it is easy to compute the sensitivity for each neuron and then for each layer and the entire MLP.

Algorithm MLP_SEN_ALG ($|\Delta X^1|$, $|\Delta W|$)

1) Set up input error $|\Delta X^1|$ and weight error $|\Delta W|$ as arrays;
2) Compute the sensitivity neuron by neuron from the first layer to the last layer.

```
For l = 1 step 1 to L
    For i = 1 step 1 to n^l
        Compute s_i^l by using (11)-(13)
and save result.
```

IV. Analysis of the Sensitivity

In the above sections, an analytical expression to calculate the sensitivity for a single neuron and an algorithm to compute the sensitivity for an entire MLP have been derived. Although it is not provide an exact mapping from input and weight errors to output error for a specific MLP, it can, however, approximately

$$A = \sum_{r=0}^{h-1} \frac{\left|\Delta w_{ij}^l\right|^{h-1-r} e^{-k(1+|\Delta x_j^l|)|\Delta w_{ij}^l|} - \left(1 + \left|\Delta w_{ij}^l\right|\right)^{h-1-r} e^{-k(1+|\Delta x_j^l|)(1+|\Delta w_{ij}^l|}}{(h-1-r)!\, (k(1+|\Delta x_j^l|))^{r+1}} \tag{12}$$

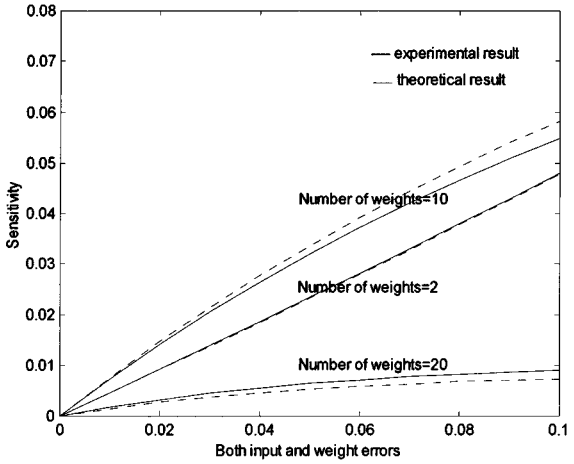

Fig. 3. Sensitivity of a neuron to input and weight errors.

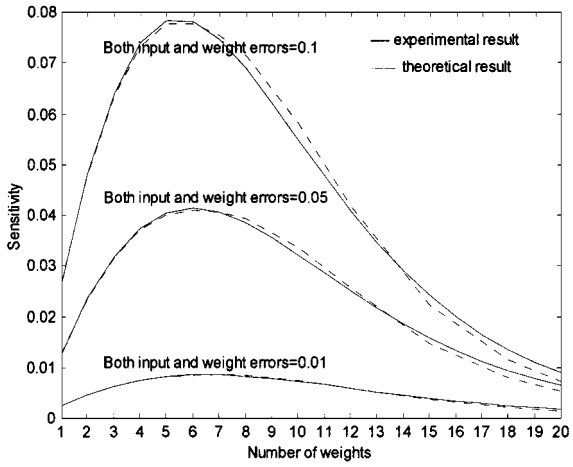

Fig. 4. Sensitivity of a neuron to the number of weights.

indicate the trends of such mappings for an ensemble of MLPs. These analytical expressions provide us insights into some significant behaviors of the MLPs. In this section the derived analytical expression and the results obtained by using the algorithm MLP_SEN_ALG will be analyzed in order to discover certain behaviors of both the neurons and the MLPs.

A. Effects of Parameters on the Sensitivity of a Single Neuron

Our analysis is essentially based on (11)–(13). It is rather complex so some simplification and computer aided computation are desirable to make the analytical and computational efforts more clear and precise.

In (11), let $p = 1$ and $q = 1$, a simplified formula can be obtained as

$$s_i^l \approx e^{-n^{l-1}} - e^{-\sum_{j=1}^{n^{l-1}} (1+|\Delta x_j^l|)(1+|\Delta w_{ij}^l|)}. \qquad (14)$$

1) Input and Weight Errors: Formula (14) obviously shows that the sensitivity increases with input and weight errors and in the extreme cases, the sensitivity is zero when both input and weight errors are zero or the maximum value is reached, i.e., $e^{-n^{l-1}}$, when either input or weight error is infinite. This result is coincident with what we intuitively expect. The following example is used to illustrate the results computed by (11)–(13). Theoretically the larger the p and q in (11) are, the more precise the results will be. But practically they are restricted by the precision limits of the computer used. In the example, we choose $p = 100\,000$ and $q = 15$, which are so selected that if either of them is made any larger it will cause an overflow error. The solid lines in Fig. 3 show the theoretical relationship between the sensitivity and input and weight errors, in which the horizontal axis represents both input and weight errors (for simplicity and without losing generality, we assume that the elements of input error $|\Delta X^l|$ and weight error $|\Delta W_i^l|$ are all identical and thus can be represented by the same point on the horizontal axis). Thus, either from the analysis of (14) or from the computation of (11)–(13), it can be concluded that the sensitivity of a single neuron increases with its input and weight errors, but the increase has an upper bound that is determined by the number of weights (or inputs).

2) Number of Weights: According to (14), the sensitivity of neuron i in layer l can be represented in a general form: $s_i^l = C_1^{n^{l-1}} - C_2^{n^{l-1}}$, where n^{l-1} is the number of weights, C_1 is determined by the ranges of X^l and W_i^l, C_2 is determined by the ranges of X^l and W_i^l, $|\Delta X^l|$ and $|\Delta W_i^l|$ and the inequality: $1 > C_1 \geq C_2 > 0$ is satisfied. By analyzing the feature of the function: $n(x) = C_1^x - C_2^x$, we find that $x' = \ln((\ln C_2)/(\ln C_1))/(\ln C_1 - \ln C_2)$ is the function's only maximum value point and x' varies in the interval $(0, +\infty)$ with the variation of C_1 and C_2. Hence, theoretically, it can be concluded that the sensitivity increases with n^{l-1} increasing when $n^{l-1} < x'$ is true and decreases with n^{l-1} increasing when $n^{l-1} \geq n^{l-1}x'$. But practically, as n^{l-1} should be an integer and small in magnitude and as the decrease of input and weight errors force C_2 to increase and this in turn forces x' to increase, we have the following conclusions about the relationship among the sensitivity, the input and weight errors and the number of weights.

1) The sensitivity may first increase with small n^{l-1} and then decreases with n^{l-1} increasing when errors are large.
2) The sensitivity may moderately increase with n^{l-1} when the errors are small.

The solid lines in Fig. 4, also computed by (11)–(13), again show the theoretical relationship between the sensitivity and the number of weights.

B. Effects of Parameters on the Sensitivity of an Entire MLP

In order to study the effects of parameters on the sensitivity of the MLP, algorithm MLP_SEN_ALG is implemented and run for some representative MLPs.

1) Input and Weight Errors: To study the effect of input and weight errors, three MLPs are constructed. They are 2-2-1, 2-3-1 and 2-2-2-1. The sensitivities computed by the algorithm MLP_SEN_ALG for these three MLPs are shown in Fig. 5, like those in Fig. 3, with the horizontal axis representing both input and weight errors. The result clearly shows that the sensitivity of the MLP also increases with its input and weight errors.

From Fig. 5, it can also be found that the sensitivity of the MLP varies with its structural configuration. This phenomenon tells us that the structural configuration of the MLP, such as the

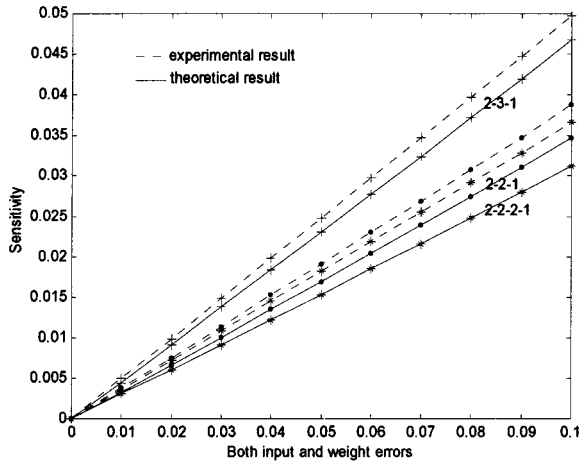

Fig. 5. Sensitivity of MLPs to input and weight errors.

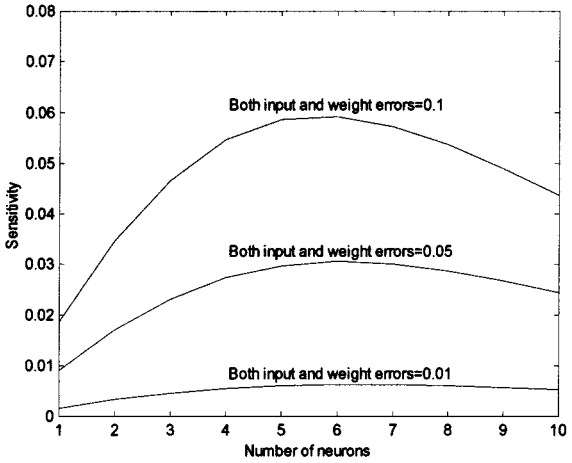

Fig. 6. Sensitivity of MLPs to the number of neurons in a layer.

number of neurons per layer and the number of layers, has a significant effect on the MLPs sensitivity.

2) Number of Neurons Per Layer: To study the effect of the number of neurons in a layer, ten MLPs: 2-1-1, 2-2-1,… and 2-10-1 are constructed. They can also be represented as a set: $\{2-n-1|\ 1 \le n \le 10\}$. The difference among them is the number of neurons in layer 1. The sensitivities computed by the algorithm MLP_SEN_ALG for these ten MLPs are shown in Fig. 6. The result is similar to those presented in Section IV-A-II for the number of weights of a single neuron except a little bit smaller in magnitude.

Since an MLP may have several layers, the number of neurons in different layers may have different effects. To study the effect of the number of neurons in different layers, we construct three sets of MLPs. They are $\{n-2-2-1|1 \le n \le 10|\}$, $\{2-n-2-1|1 \le n \le 10|\}$ and $\{2-2-n-1|1 \le n \le 10\}$. The first set is used to study the effect of the number of inputs in input layer, the second one is on the number of neurons in layer 1 and the third is on the number of neurons in layer 2. The results are shown in Figs. 7–9, respectively. Comparing these three figures, we find that the effect of the number of neurons in a layer is different for different layers. The nearer a layer to the output layer is, the more effect the number of neurons in the layer has. This, from another angle, shows us that the number of layers in the MLP has an effect on the sensitivity.

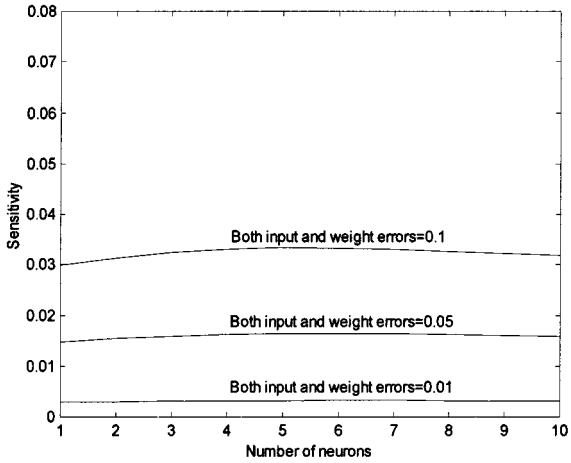

Fig. 7. Sensitivity of MLPs to the number of inputs.

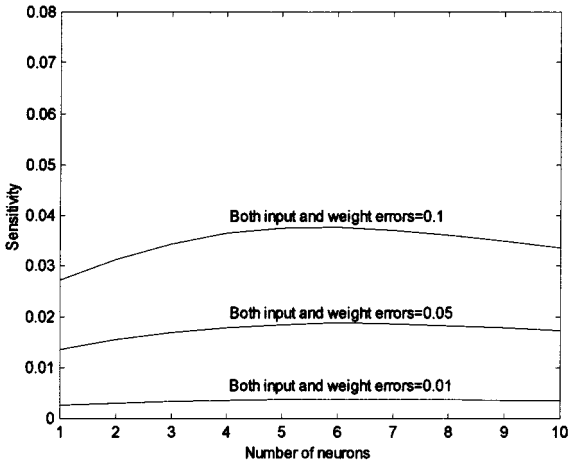

Fig. 8. Sensitivity of MLPs to the number of neurons in the first layer.

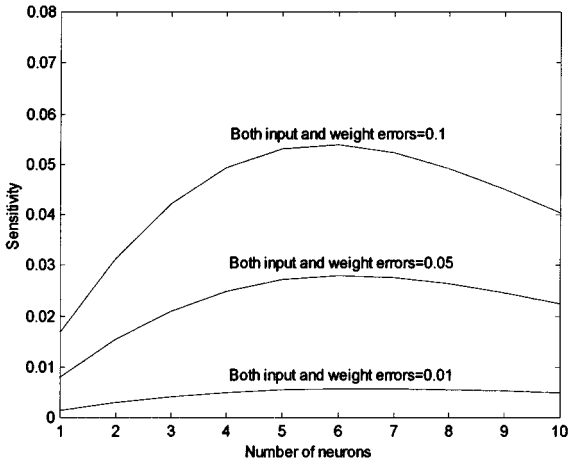

Fig. 9. Sensitivity of MLPs to the number of neurons in the second layer.

3) Number of Layers: To study the effect of the number of layers, ten MLPs are constructed. They are 2-1, 2-2-1, 2-2-2-1,… and 2-2-2-2-2-2-2-2-2-2-1. From the second to the last, we each time add a layer with two neurons into the previous MLP and keep the output layer with only one neuron. The

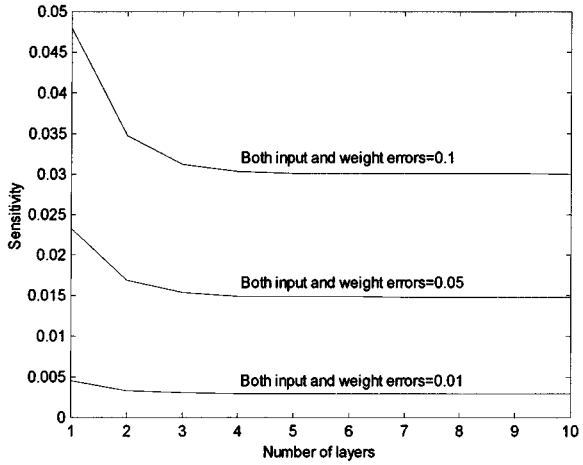

Fig. 10. Sensitivity of MLPs to the number of layers.

sensitivities computed by the algorithm MLP_SEN_ALG for these MLPs are shown in Fig. 10. It is found that the sensitivity decreases with the number increases when the number is small and the decrease levels off when the number becomes large.

C. Experimental Verification

In order to verify the above theoretical results, a computer simulation was run to obtain experimental results. As the sensitivity of a single neuron is fundamental to the sensitivity of an entire MLP, the sensitivity for a single neuron is considered first. In the simulation, under a given n^{l-1} (the number of weights), more than 100 000 000 samples are randomly selected with different inputs and weights distributed uniformly in the hypercube $[0,1]^{2n^{l-1}}$. For a given input error and weight error (to be consistent with the above theoretical results, we assign them identical values), the average output error for a neuron over all the selected samples are computed. The dotted lines in Figs. 3 and 4 show the experimental results. The simulation for an entire MLP was done by using the algorithm MLP_SEN_ALG for 3 MLPs: 2-2-1, 2-3-1 and 2-2-2-1. The dotted lines in Fig. 5 show the experimental results. The agreement between the theoretical results and experimental results is quite good.

V. Conclusion

In this paper, we propose a bottom-up approach to study the sensitivity of the MLP to input and weight perturbations. It starts from the sensitivity of a single neuron. The derived analytical expressions (11)–(13), especially its simplified form (14), clearly expresses the relationship between a neuron's output error and its input and weight errors. It is then used to compute the sensitivity of an entire MLP. The given algorithm MLP_SEN_ALG makes it feasible to quantify the sensitivity of an MLP and to analyze the relationship between the MLPs output error and its input and weight errors as well as its structural configuration.

The theoretical results obtained are in reasonable agreement with our experimental results. The sensitivity expression is general in the sense that it covers both input and weight errors and treats both input and weight as statistical variables. As for the usefulness of this sensitivity measure, the derived sensitivity formula, although being too general to be applied to a specific MLP, does disclose significant characteristics for an ensemble of MLPs so that certain guidelines such as the appropriate number of layers in a network and the appropriate number of neurons in a layer and other *a priori* information could be obtained prior to the network design and implementation. Further, with more specific parameter information such as fixed weight set, etc., this sensitivity measure can certainly be applicable to a specific MLP to measure and then to improve its performance. For example, the sensitivity measure can be used as a relative criterion for the selection of more robust weight set with lower sensitivity during the training of an MLP.

Appendix
The Derivation of the Formula for the Sensitivity of a Neuron

This Appendix gives the detailed derivation of the analytical expression for the sensitivity of a neuron

$$\begin{aligned}
s_i^l &= E\left(\left|g\left(X^l, W_i^l\right)\right|\right) \\
&= E\left(\left|f\left(\left(X^l + \left|\Delta X^l\right|\right)\right.\right.\right. \\
&\qquad \left.\left.\left. * \left(W_i^l + \left|\Delta W_i^l\right|\right)\right) - f\left(X^{l*} W_i^l\right)\right|\right) \\
&= \int_{V_{X^l}} \int_{V_{W_i^l}} \varphi\left(X^l, W_i^l\right) \left|f\left(\left(X^l + \left|\Delta X^l\right|\right)\right.\right. \\
&\qquad \left.\left. * \left(W_i^l + \left|\Delta W_i^l\right|\right)\right) \right. \\
&\qquad \left. - f\left(X^{l*} W_i^l\right)\right| dV_{X^l} dV_{W_i^l} \\
&= \int_{V_{X^l}} \int_{V_{W_i^l}} \left|f\left(\left(X^l + \left|\Delta X^l\right|\right)\right.\right. \\
&\qquad \left.\left. * \left(W_i^l + \left|\Delta W_i^l\right|\right)\right) \right. \\
&\qquad \left. - f\left(X^{l*} W_i^l\right)\right| dV_{X^l} dV_{W_i^l} \\
&= \int_{V_{X^l}} \int_{V_{W_i^l}} \left(f\left(\left(X^l + \left|\Delta X^l\right|\right)\right.\right. \\
&\qquad \left.\left. * \left(W_i^l + \left|\Delta W_i^l\right|\right)\right) \right. \\
&\qquad \left. - f\left(X^{l*} W_i^l\right)\right) dV_{X^l} dV_{W_i^l} \qquad \text{by (7)}
\end{aligned}$$

by $f(x)$ being monotone increase, as follows:

$$= \int_{V_{X^l}} \int_{V_{W_i^l}} \left(\frac{1}{1 + e^{-\sum_{j=1}^{n^{l-1}} (x_j^l + |\Delta x_j^l|)(w_{ij}^l + |\Delta w_{ij}^l|)}} - \frac{1}{1 + e^{-\sum_{j=1}^{n^{l-1}} x_j^l w_{ij}^l}} \right) dV_{X^l} dV_{W_i^l} \qquad \text{by (1)}$$

$$\approx \int_{V_{X^l}} \int_{V_{W_i^l}} \left(\sum_{k=0}^{p} (-1)^k e^{-k \sum_{j=1}^{n^{l-1}} (x_j^l + |\Delta x_j^l|)(w_{ij}^l + |\Delta w_{ij}^l|)} - \sum_{k=0}^{p} (-1)^k e^{-k \sum_{j=1}^{n^{l-1}} x_j^l w_{ij}^l} \right) dV_{X^l} dV_{W_i^l} \quad \text{by (8)}$$

$$= \int_{V_{X^l}} \int_{V_{W_i^l}} \left(\sum_{k=1}^{p} (-1)^k \left(e^{-k \sum_{j=1}^{n^{l-1}} (x_j^l + |\Delta x_j^l|)(w_{ij}^l + |\Delta w_{ij}^l|)} - e^{-k \sum_{j=1}^{n^{l-1}} x_j^l w_{ij}^l} \right) \right) dV_{X^l} dV_{W_i^l}$$

$$= \int_{V_{X^l}} \int_{V_{W_i^l}} \left(\sum_{k=1}^{p} (-1)^k \left(\prod_{j=1}^{n^{l-1}} e^{-k(x_j^l + |\Delta x_j^l|)(w_{ij}^l + |\Delta w_{ij}^l|)} - \prod_{j=1}^{n^{l-1}} e^{-k x_j^l w_{ij}^l} \right) \right) dV_{X^l} dV_{W_i^l}$$

$$= \sum_{k=1}^{p} (-1)^k \left(\int_{V_{X^l}} \int_{V_{W_i^l}} \left(\prod_{j=1}^{n^{l-1}} e^{-k(x_j^l + |\Delta x_j^l|)(w_{ij}^l + |\Delta w_{ij}^l|)} - \prod_{j=1}^{n^{l-1}} e^{-k x_j^l w_{ij}^l} \right) dV_{X^l} dV_{W_i^l} \right)$$

$$= \sum_{k=1}^{p} (-1)^k \left(\int_0^1 \cdots \int_0^1 \left(\prod_{j=1}^{n^{l-1}} e^{-k(x_j^l + |\Delta x_j^l|)(w_{ij}^l + |\Delta w_{ij}^l|)} - \prod_{j=1}^{n^{l-1}} e^{-k x_j^l w_{ij}^l} \right) dx_1^l \ldots dx_{n^{l-1}}^l dw_{i1}^l \ldots dw_{in^{l-1}}^l \right)$$

$$= \sum_{k=1}^{p} (-1)^k \left(\prod_{j=1}^{n^{l-1}} \left(\int_0^1 \int_0^1 e^{-k(x_j^l + |\Delta x_j^l|)(w_{ij}^l + |\Delta w_{ij}^l|)} dx_j^l dw_{ij}^l \right) - \prod_{j=1}^{n^{l-1}} \left(\int_0^1 \int_0^1 e^{-k x_j^l w_{ij}^l} dx_j^l dw_{ij}^l \right) \right)$$

$$= \sum_{k=1}^{p} (-1)^k \left(\prod_{j=1}^{n^{l-1}} \left(\int_0^1 \left[\frac{-e^{-k(x_j^l + |\Delta x_j^l|)(w_{ij}^l + |\Delta w_{ij}^l|)}}{k\left(w_{ij}^l + \left|\Delta w_{ij}^l\right|\right)} \right]_0^1 dw_{ij}^l \right) \right.$$
$$\left. - \prod_{j=1}^{n^{l-1}} \left(\int_0^1 \left[\frac{-e^{-k x_j^l w_{ij}^l}}{k w_{ij}^l} \right]_0^1 dw_{ij}^l \right) \right)$$

$$= \prod_{k=1}^{p} (-1)^k \left(\prod_{j=1}^{n^{l-1}} \left(\int_0^1 \frac{e^{-k|\Delta x_j^l|(w_{ij}^l + |\Delta w_{ij}^l|)} - e^{-k(1+|\Delta x_j^l|)(w_{ij}^l + |\Delta w_{ij}^l|)}}{k\left(w_{ij}^l + \left|\Delta w_{ij}^l\right|\right)} dw_{ij}^l \right) \right.$$
$$\left. - \prod_{j=1}^{n^{l-1}} \left(\int_0^1 \frac{1 - e^{-k w_{ij}^l}}{k w_{ij}^l} dw_{ij}^l \right) \right)$$

$$= \prod_{k=1}^{p} (-1)^k \left(\prod_{j=1}^{n^{l-1}} \left(\int_0^1 \frac{e^{-k(1+|\Delta x_j^l|)(w_{ij}^l + |\Delta w_{ij}^l|)} \left(e^{k(w_{ij}^l + |\Delta w_{ij}^l|)} - 1 \right)}{k\left(w_{ij}^l + \left|\Delta w_{ij}^l\right|\right)} dw_{ij}^l \right) \right.$$
$$\left. - \prod_{j=1}^{n^{l-1}} \left(\int_0^1 \frac{e^{k w_{ij}^l} \left(e^{k w_{ij}^l} - 1 \right)}{\left(k w_{ij}^l \right)} dw_{ij}^l \right) \right)$$

$$\approx \sum_{k=1}^{p} (-1)^k \left(\prod_{j=1}^{n^{l-1}} \left(\int_0^1 e^{-k(1+|\Delta x_j^l|)(w_{ij}^l + |\Delta w_{ij}^l|)} \left(\sum_{h=1}^{q} \frac{\left(k\left(w_{ij}^l + \left|\Delta w_{ij}^l\right|\right)\right)^{h-1}}{h!} \right) dw_{ij}^l \right) \right.$$
$$\left. - \prod_{j=1}^{n^{l-1}} \left(\int_0^1 e^{k w_{ij}^l} \left(\sum_{h=1}^{q} \frac{\left(k w_{ij}^l\right)^{h-1}}{h!} \right) dw_{ij}^l \right) \right) \quad \text{by (9)}$$

$$= \sum_{k=1}^{p} (-1)^k \left(\prod_{j=1}^{n^{l-1}} \left(\sum_{h=1}^{q} \left(\int_0^1 \frac{e^{-k(1+|\Delta x_j^l|)(w_{ij}^l + |\Delta w_{ij}^l|)} \left(k\left(w_{ij}^l + \left|\Delta w_{ij}^l\right|\right)\right)^{h-1}}{h!} dw_{ij}^l \right) \right) \right.$$
$$\left. - \prod_{j=1}^{n^{l-1}} \left(\sum_{h=1}^{q} \left(\int_0^1 \frac{e^{k w_{ij}^l} \left(k w_{ij}^l\right)^{h-1}}{h!} dw_{ij}^l \right) \right) \right) \right)$$

$$= \sum_{k=1}^{p} (-1)^k \left(\prod_{j=1}^{n^{l-1}} \left(\sum_{h=1}^{q} \left(\frac{k^{h-1}}{h} \right) A \right) - \prod_{j=1}^{n^{l-1}} \left(\sum_{h=1}^{q} \left(\frac{k^{h-1}}{h} \right) B \right) \right)$$

$$= \sum_{k=1}^{p} (-1)^k \left(\prod_{j=1}^{n^{l-1}} \left(\sum_{h=1}^{q} \left(\frac{k^{h-1}}{h} \right) A \right) - \left(\sum_{h=1}^{q} \left(\frac{k^{h-1}}{h} \right) B \right)^{n^{l-1}} \right)$$

where we have the following:

$$
\begin{aligned}
A &= \int_0^1 \frac{\left(w_{ij}^l + \left|\Delta w_{ij}^l\right|\right)^{h-1} e^{-k(1+|\Delta x_j^l|)(w_{ij}^l+|\Delta w_{ij}^l|)}}{(h-1)!}\, dw_{ij}^l \\
&= \left[\sum_{r=0}^{h-1} \frac{(-1)^r \left(w_{ij}^l + \left|\Delta w_{ij}^l\right|\right)^{h-1-r} e^{-k(1+|\Delta x_j^l|)(w_{ij}^l+|\Delta w_{ij}^l|)}}{(h-1-r)!\left(-k\left(1+\left|\Delta x_j^l\right|\right)\right)^{r+1}}\right]_0^1 \qquad \text{by (10)} \\
&= \sum_{r=0}^{h-1} \frac{\left|\Delta w_{ij}^l\right|^{h-1-r} e^{-k(1+|\Delta x_j^l|)|\Delta w_{ij}^l|} - \left(1+\left|\Delta w_{ij}^l\right|\right)^{h-1-r} e^{-k(1+|\Delta x_j^l|)(1+|\Delta w_{ij}^l|)}}{(h-1-r)!\left(k\left(1+\left|\Delta x_j^l\right|\right)\right)^{r+1}} \\
B &= \int_0^1 \frac{\left(w_{ij}^l\right)^{h-1} e^{-kw_{ij}^l}}{(h-1)!}\, dw_{ij}^l \\
&= \left[\sum_{r=0}^{h-1} \frac{(-1)^r \left(w_{ij}^l\right)^{h-1-r} e^{-kw_{ij}^l}}{(h-1-r)!(-k)^{r+1}}\right]_0^1 \qquad \text{by (10)} \\
&= \sum_{r=0}^{h-2} \frac{-e^{-k}}{(h-1-r)!k^{r+1}} + \frac{1-e^{-k}}{k^h}.
\end{aligned}
$$

REFERENCES

[1] M. Stevenson, R. Winter, and B. Widrow, "Sensitivity of feedforward neural networks to weight errors," *IEEE Trans. Neural Networks*, vol. 1, pp. 71–80, Mar. 1990.

[2] C. Alippi, V. Piuri, and M. Sami, "Sensitivity to errors in artificial neural networks: A behavioral approach," *IEEE Trans. Circuits Syst. I*, vol. 42, pp. 358–361, June 1995.

[3] S. W. Piche, "The selection of weight accuracies for Madalines," *IEEE Trans. Neural Networks*, vol. 6, pp. 432–445, Mar. 1995.

[4] L. Yang, D. Hu, Y. Luo, and X. Zhang, "Robustness analysis of feedforward neural networks composed of threshold neurons," in *Proc. IEEE Int. Conf. Intell. Processing Syst.*, vol. 1, Oct. 1997, pp. 502–506.

[5] A. Y. Cheng and D. S. Yeung, "Sensitivity analysis of neocognitron," *IEEE Trans. Syst., Man, Cybern. C*, vol. 29, pp. 238–249, May 1999.

[6] S. Hashem, "Sensitivity analysis for feedforward artificial neural networks with differentiable activation functions," in *Proc. IJCNN'92*, vol. 1, Baltimore, MD, 1992, pp. 419–424.

[7] L. Fu and T. Chen, "Sensitivity analysis for input vector in multilayer feedforward neural networks," in *Proc. IEEE Int. Conf. Neural Networks*, vol. 1, San Francisco, CA, 1993, pp. 215–218.

[8] J. M. Zurada, A. Malinowski, and S. Usui, "Perturbation method for deleting redundant inputs of perceptron networks," *Neurocomput.*, vol. 14, pp. 177–193, 1997.

[9] A. P. Engelbrecht and I. Cloete, "A sensitivity analysis algorithm for pruning feedforward neural networks," in *Proc. IEEE Int. Conf. Neural Networks*, vol. 2, Washington, DC, 1996, pp. 1274–1277.

[10] A. P. Engelbrecht, L. Fletcher, and I. Cloete, "Variance analysis of sensitivity information for pruning feedforward neural networks," in *Proc. IEEE Int. Conf. Neural Networks*, Washington, DC, 1999, pp. 1829–1833.

[11] J. Y. Choi and C.-H. Choi, "Sensitivity analysis of multilayer perceptron with differentiable activation functions," *IEEE Trans. Neural Networks*, vol. 3, pp. 101–107, Jan. 1992.

[12] M.-Y. Chow and S. O. Tee, "A measure of relative robustness for feedforward neural networks subject to small input perturbations," *Int. J. Neural Syst.*, vol. 3, no. 3, pp. 291–299, 1992.

[13] S.-H. Oh and Y. Lee, "Sensitivity analysis of a single hidden-layer neural networks with threshold function," *IEEE Trans. Neural Networks*, vol. 6, pp. 1005–1007, July 1995.

[14] D. S. Yeung and X. Sun, "Using function approximation to analyze the sensitivity of the MLP with antisymmetric squashing activation function," IEEE Trans. Neural Networks, to be published.

Xiaoqin Zeng received the B.S. degree from Nanjing University, China, in 1982 and the M.S. degree from Southeast University, China, in 1989, both in computer science. He is currently pursuing the Ph.D. degree in the Department of Computing at the Hong Kong Polytechnic University

He is currently an Associate Professor in Hohai University, China. His main research interests include neural networks, object technologies, and databases.

Daniel S. Yeung (M'89–SM'99) received the Ph.D. degree in applied mathematics from Case Western Reserve University, Cleveland, OH, in 1974.

In the past, he has worked as an Assistant Professor of Mathematics and Computer Science at Rochester Institute of Technology, Rochester, NY, as a Research Scientist in the General Electric Corporate Research Center, and a System Integration Engineer at TRW, Inc. Currently, he is a Chair Professor of Computing, Hong Kong Polytechnic University, Hong Kong. His current research interests include neural-network sensitivity analysis, expert neural-network hybrid systems, off-line handwritten Chinese character recognition, and fuzzy expert systems.

Dr. Yeung was the President of the IEEE Computer Chapter of Hong Kong for 1991 and 1992. He now chairs the technical subcommittee on Expert and Knowledge Based Systems of the IEEE Systems, Man, and Cybernetics Society.