CHAPTER 10
SEN SITIVITY ANALYSIS OF FUZZY AND NEURAL NETWORK MODEL

10.1 INTRODUCTION

It has been shown that soft computing techniques can be very well deployed for Software Engineering applications. Among these fuzzy and neural models are widely used to estimate lines of codes, effort, software maintainability, software understandability etc.

Each of them has their own strengths and weaknesses. Soft computing is a synergistic integration [142] of three computing paradigms viz., neural network, fuzzy logic and probabilistic reasoning. Fuzzy models have been used to estimate effort [7,113,114,115], size [181], maintainability [10,16] and understand ability [16]. Neural network models have also been used to estimate size [14], effort [212] and understand ability [6]. If both models can be used for the same application, a question arises as to which one is better? This can be found out by carrying out a sensitivity analysis of both fuzzy and neural network models considering measurement of maintainability as a case study.
10.2 SENSITIVITY ANALYSIS

Finding out its condition number carries out sensitivity Analysis of a model. Condition number [60] is defined to be the maximum values of the ratio of the relative error in the solution to the relative error in data over the problem domain. It can be expressed as:

\[ \text{Condition number } [C] = \frac{\Delta f * x}{f \Delta x} \] ............. (10.1)

Where \( f \) is a function of \( x \), \( \Delta x \) is the perturbation to each of the inputs and in this case it is taken to be 0.001. When the relative change in output is nearly equal to the change in input the condition number is close to 1. The lower the condition number, in fact the closer the condition number is to 1, the better is the model or the more stable is the model.

Sensitivity Analysis of the two models namely fuzzy and neural network models are carried out to show which one is better.
10.3 CASE STUDY TO MEASURE SOFTWARE MAINTAINABILITY

Software maintenance is defined as the process of modifying existing operational software while leaving its primary functions intact.

A model [10] has been proposed where maintainability is dependent on the following inputs: a). Comment ratio (CR), b). Average Cyclomatic Complexity [CC] c). Average number of live variables [LV]. d) Average life span of variables [LS].

The output maintainability is an integrated measure of the four inputs. This is taken as a case study and a fuzzy as well as a neural network model is built for the same and sensitivity analysis is then carried out on both.

10.3.1 FUZZY MODEL

Uncertainty is present in our every day lives, it is also present in software engineering. Real world concepts transition smoothly into one another rather than abruptly [140]. Humans reason with a fuzzy logic. Therefore fuzziness as a means of modeling linguistic uncertainty can be very well used to model real world problems and so also software engineering problems.
A fuzzy model like any other model provides mapping from input to output. The architecture of the fuzzy model is as shown in Figure 10.1. There are four major modules. The fuzzification module transforms the crisp inputs into fuzzy values. These are then processed in fuzzy domain by inference engine based on knowledge base (rule base) supplied by domain experts. Finally defuzzification transforms the processed data from fuzzy domain to crisp domain.

Two fuzzy models [10, 16] are available to measure maintainability but the first one [10] has been used here.

**Knowledge Data Base**

![Diagram of fuzzy model](image)

**Figure 10.1: Fuzzy Model for Software maintainability**

Modeling is done as follows:

(i) All the inputs and outputs were fuzzified...
(ii) All possible combination of inputs were considered which leads to $3^4$ i.e. 81 sets. The maintainability in case of all eighty-one combinations is classified as either Very Good, Good, Average, Poor or Very Poor by expert opinion. These lead to formation of 81 rules for the fuzzy model and some of them are shown below:

1. If ($CR$ is low) and ($ACC$ is low) and ($LV$ is low) and ($LS$ is low) then maintainability is very good.

2. If ($CR$ is low) and ($ACC$ is low) and ($LV$ is low) and ($LS$ is med) then maintainability is very good.

81. If ($CR$ is high) and ($ACC$ is high) and ($LV$ is high) and ($LS$ is high) then maintainability is very poor.

(iii) All eighty-one rules are inserted and a rule base is created. Depending on a particular set of inputs, a rule will be fired.
(iv) Mamdani style of inference is used.

(v) Using the rule viewer, output i.e. maintainability is observed for a particular set of inputs using the MATLAB Fuzzy toolbox.

10.3.2 NEURAL NETWORK MODEL

A four input and one output network is used. The network uses only one hidden layer. The activation functions at the hidden layer and the output layers are the tansig and pureln function. The network inputs are (a) The Comment Ratio (CR) (b) the average cyclomatic complexity (CC) (c) the average number of live variables (LV) and (d) the average life span of variables (LS) The block diagram of the network used is shown in Figure 10.2.

Figure 10.2: A neural network model to compute maintainability
10.4 EXPERIMENTAL METHODOLOGY TO CONDUCT

SENSITIVITY ANALYSIS

1. A fuzzy model was created as explained in the earlier section. Two hundred random sets of normalized (between -1 and +1) inputs (CR, CC, LV, LS) were given and the output value of maintainability was obtained from the model. These were used as training cases for the neural network model.

2. One thousand random sets of normalized inputs were provided to the model to obtain output values of maintainability. These were used as test cases for the neural network model.

3. By perturbing each of the one thousand and two hundred (two hundred training sets and one thousand test sets) exemplars by 0.001, condition number for CR, CC, LV and LS was noted down for the fuzzy model. This is shown in Table 10.2.

4. The neural network was trained using training data of two hundred exemplars obtained in step one from the fuzzy model. Training function used was trainbr [Bayesian regularization] and the effort function used for msereg. The tansig function was used in the first layer, & purelin function was used in the second layer. The numbers of neurons were kept as ten in the hidden layer. The learning function used was learnngdm.
5. The neural network was tested with one thousand exemplars obtained in step two.

6. To find out how well the neural network models the fuzzy models, error measurements were noted down for training as well as test data. They are shown in Table 10.1

7. By perturbing each of the training as well as test data by 0.001, condition numbers for CR, CC, LV and LS for training as well as test data was found out. The maximum of the pair was considered to be the condition number for the neural network.

The condition number for each of the four inputs is expressed as:

\[
\text{Condition number } [C]_{CR} = \frac{\Delta f}{f} \times \frac{CR}{\Delta CR} \quad \ldots (10.2)
\]

\[
\text{Condition number } [C]_{CC} = \frac{\Delta f}{f} \times \frac{CC}{\Delta CC} \quad \ldots (10.3)
\]

\[
\text{Condition number } [C]_{LV} = \frac{\Delta f}{f} \times \frac{LV}{\Delta LV} \quad \ldots (10.4)
\]

\[
\text{Condition number } [C]_{LS} = \frac{\Delta f}{f} \times \frac{LS}{\Delta LS} \quad \ldots (10.5)
\]
Where the perturbation $\Delta CR = \Delta CC = \Delta LS = \Delta LV = 0.001$

The results are shown in Table 10.2

### 10.5 RESULTS AND DISCUSSIONS

The error measurements in the case of a neural network model are shown in Table 10.1.

<table>
<thead>
<tr>
<th></th>
<th>For Training set</th>
<th>For Test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>ME</td>
<td>0.001</td>
<td>-0.030</td>
</tr>
<tr>
<td>MAE</td>
<td>0.306</td>
<td>0.485</td>
</tr>
<tr>
<td>MPE</td>
<td>-0.005</td>
<td>-0.012</td>
</tr>
<tr>
<td>MAPE</td>
<td>0.056</td>
<td>0.090</td>
</tr>
<tr>
<td>MSE</td>
<td>0.166</td>
<td>0.393</td>
</tr>
<tr>
<td>STD</td>
<td>0.408</td>
<td>0.626</td>
</tr>
<tr>
<td>MEDIAN</td>
<td>0.023</td>
<td>-0.016</td>
</tr>
</tbody>
</table>

**Table 10.1: Error Measurements**

The condition number for the two models are shown in Table 10.2
Table 10.2: Comparison of condition numbers for the two models

<table>
<thead>
<tr>
<th>Condition number</th>
<th>For Fuzzy Model</th>
<th>For Neural Network Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_{CR}$</td>
<td>28.79</td>
<td>0.91</td>
</tr>
<tr>
<td>$C_{CC}$</td>
<td>3.09</td>
<td>1.94</td>
</tr>
<tr>
<td>$C_{LV}$</td>
<td>8.03</td>
<td>1.39</td>
</tr>
<tr>
<td>$C_{LS}$</td>
<td>3.41</td>
<td>1.59</td>
</tr>
</tbody>
</table>

It is thus seen that the training data leads to MAPE as 5.6 % whereas the test data leads to MAPE as 9.06 %. This shows how well the neural network models the fuzzy model.

Table 10.2 compares the condition numbers for both the models. The neural network model shows reduced condition numbers as $C_{CR} = 0.91$, $C_{CC} = 1.94$, $C_{LV} = 1.39$ and $C_{LS} = 1.59$.

10.5 CONCLUSION

Sensitivity analysis proves that the neural network using Bayesian Regularization training algorithm produces reduced condition numbers as compared to the fuzzy model having membership functions whose derivatives show discontinuities at some points. In fact it is at these points, that the condition number increases. Therefore it is concluded from this work that the neural network model using Bayesian Regularization training algorithm is a more stable
model than the fuzzy model having membership functions whose derivatives show discontinuities at some points for all applications and specifically to software engineering applications.