Prediction of Software Development Effort Using RBNN and GRNN

Prasad Reddy P.V.G.D 1, Sudha K. R 2, Rama Sree P 3

1 Dept. of CSSE, Andhra University, Visakhapatnam, 530003, Andhra Pradesh, INDIA
2 Dept. of EE, Andhra University, Visakhapatnam, 530003, Andhra Pradesh, INDIA
3 Dept. of CSE, Aditya Engineering College, JNTUK, Kakinada, 533003, Andhra Pradesh, INDIA.

Abstract—Software development effort prediction is one of the most key activities in software industry. Many models have been proposed to build a relationship between software size and effort; however we still have problems for effort prediction. This is because project data, available in the primary stages of project is often inadequate, unpredictable, uncertain and unclear. The need for accurate effort estimation in software industry is an ongoing challenge. Artificial Neural Network models are more apt in such situations. The present paper is concerned with developing software effort prediction models based on artificial neural networks. The models are designed to improve the performance of the network that suits to the COCOMO Model. Artificial Neural Network models are created using Radial Basis and Generalized Regression. A case study based on the NASA 93 database compares the proposed neural network models with the Intermediate COCOMO. The results were analyzed using different criterions VAF, MMRE, MARE, VARE, Mean BRE and Prediction. It is observed that the Radial Basis Neural Network provided better results.

Keywords—Intermediate COCOMO, Cost Estimation, Radial Basis Neural Networks, Generalized Regression Neural Networks.

I. INTRODUCTION

In algorithmic cost estimation [14], costs and efforts are predicted using mathematical formulae. The formulae are resultant of some historical data [2]. The algorithmic cost model best known called COCOMO (COnstructive COst MOdel) was proposed by Barry Boehm in 1981 [3]. It was developed from the analysis of sixty three (63) software projects. Boehm proposed three levels of the model called Basic COCOMO, Intermediate COCOMO and Detailed COCOMO [3], [15]. In the present paper we mainly focus on the Intermediate COCOMO.

A. Intermediate COCOMO

The Basic COCOMO model [3] is based on the relationship: Development Effort, \( DE = a \times \text{SIZE} \) where, SIZE is measured in thousand delivered source instructions. The constants \( a \) and \( b \) are dependent upon the ‘mode’ of development of projects. DE is measured in man-months. Boehm proposed 3 modes of projects [5]:

1) Organic mode: simple projects that engage small teams working in known and stable environments.
2) Semi-detached mode: projects that engage teams with a mixture of experience. It is in between organic and embedded modes.
3) Embedded mode: complex projects that are developed under tight constraints with changing requirements.

The accuracy of Basic COCOMO is limited because it does not consider the factors like hardware, personnel, use of modern tools and other attributes that affect the project cost. Further, Boehm proposed the Intermediate COCOMO [3], that adds accuracy to the Basic COCOMO by multiplying ‘Cost Drivers’ into the equation with a new variable: EAF (Effort Adjustment Factor) shown in Table 1.

<table>
<thead>
<tr>
<th>Development Mode</th>
<th>Intermediate Effort Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organic</td>
<td>( DE = \text{EAF} \times 3.2 \times \text{SIZE}^{1.05} )</td>
</tr>
<tr>
<td>Semi-detached</td>
<td>( DE = \text{EAF} \times 3.0 \times \text{SIZE}^{1.12} )</td>
</tr>
<tr>
<td>Embedded</td>
<td>( DE = \text{EAF} \times 2.8 \times \text{SIZE}^{1.2} )</td>
</tr>
</tbody>
</table>

The EAF term is the product of 15 Cost Drivers [15] that are listed in Table 2. The multipliers of the cost drivers are Very Low, Low, Nominal, High, Very High and Extra High. For example, for a project, if RELY is Low, DATA is High, CPLX is extra high, TIME is Very High, STOR is High and rest parameters are Nominal then EAF = 0.75 * 1.08 * 1.65 * 1.30 * 1.06 * 1.0. If the category values of all the 15 cost drivers are “Nominal”, then EAF is equal to 1.
### TABLE III

**Intermediate COCOMO Cost Drivers with Multipliers**

<table>
<thead>
<tr>
<th>S. No</th>
<th>Cost Driver Symbol</th>
<th>Very low</th>
<th>Low</th>
<th>Nominal</th>
<th>High</th>
<th>Very high</th>
<th>Extra high</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RELY</td>
<td>0.75</td>
<td>0.88</td>
<td>1.00</td>
<td>1.15</td>
<td>1.40</td>
<td>—</td>
</tr>
<tr>
<td>2</td>
<td>DATA</td>
<td>—</td>
<td>0.94</td>
<td>1.00</td>
<td>1.08</td>
<td>1.16</td>
<td>—</td>
</tr>
<tr>
<td>3</td>
<td>CPLX</td>
<td>0.70</td>
<td>0.85</td>
<td>1.00</td>
<td>1.15</td>
<td>1.30</td>
<td>1.65</td>
</tr>
<tr>
<td>4</td>
<td>TIME</td>
<td>—</td>
<td>—</td>
<td>1.00</td>
<td>1.11</td>
<td>1.30</td>
<td>1.66</td>
</tr>
<tr>
<td>5</td>
<td>STOR</td>
<td>—</td>
<td>—</td>
<td>1.00</td>
<td>1.06</td>
<td>1.21</td>
<td>1.56</td>
</tr>
<tr>
<td>6</td>
<td>VIRT</td>
<td>—</td>
<td>—</td>
<td>1.00</td>
<td>1.07</td>
<td>1.15</td>
<td>—</td>
</tr>
<tr>
<td>7</td>
<td>TURN</td>
<td>—</td>
<td>0.87</td>
<td>1.00</td>
<td>1.07</td>
<td>1.15</td>
<td>—</td>
</tr>
<tr>
<td>8</td>
<td>ACAP</td>
<td>—</td>
<td>0.87</td>
<td>1.00</td>
<td>1.07</td>
<td>1.15</td>
<td>—</td>
</tr>
<tr>
<td>9</td>
<td>AEXP</td>
<td>1.29</td>
<td>1.13</td>
<td>1.00</td>
<td>0.91</td>
<td>0.82</td>
<td>—</td>
</tr>
<tr>
<td>10</td>
<td>PCAP</td>
<td>1.42</td>
<td>1.17</td>
<td>1.00</td>
<td>0.86</td>
<td>0.70</td>
<td>—</td>
</tr>
<tr>
<td>11</td>
<td>VEXP</td>
<td>1.21</td>
<td>1.10</td>
<td>1.00</td>
<td>0.90</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>12</td>
<td>LEXP</td>
<td>1.14</td>
<td>1.07</td>
<td>1.00</td>
<td>0.95</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>13</td>
<td>MODP</td>
<td>1.24</td>
<td>1.10</td>
<td>1.00</td>
<td>0.91</td>
<td>0.82</td>
<td>—</td>
</tr>
<tr>
<td>14</td>
<td>TOOL</td>
<td>1.24</td>
<td>1.10</td>
<td>1.00</td>
<td>0.91</td>
<td>0.83</td>
<td>—</td>
</tr>
<tr>
<td>15</td>
<td>SCED</td>
<td>1.23</td>
<td>1.08</td>
<td>1.00</td>
<td>1.04</td>
<td>1.10</td>
<td>—</td>
</tr>
</tbody>
</table>

The 15 cost drivers are broadly classified into 4 categories [3], [15].

1) **Product:**
   - RELY - Required software reliability
   - DATA - Database size
   - CPLX - Product complexity

2) **Platform:**
   - TIME - Execution time
   - STOR - Main storage constraint
   - VIRT - Virtual machine volatility
   - TURN - Computer turnaround time

3) **Personnel:**
   - ACAP - Analyst capability
   - AEXP - Applications experience
   - PCAP - Programmer capability
   - VEXP - Virtual machine experience
   - LEXP - Language experience

4) **Project:**
   - MODP - Modern programming
   - TOOL - Use of software tools
   - SCED - Required development schedule

Depending on the projects, multipliers of the cost drivers will vary and thereby the EAF may be greater than or less than 1, thus affecting the Effort [12], [13], [15].

### II. PROPOSED NEURAL NETWORK MODELS

A neural network [9] is a massive parallel distributed processor made up of simple processing units, which has a natural propensity for storing experimental knowledge and making it available for use. It resembles the brain in two respects [8], [4]:

1) Knowledge is acquired by the network from its environment through a learning process [10].
2) Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge.

In this section we are going to present the two network models [1] used for the case study i.e. Radial Basis Neural Network (RBNN) and Generalized Regression Neural Network (GRNN).

#### A. Radial Basis Neural Network

Radial Basis Neural Network (RBNN) consists of two layers: a hidden radial basis layer of $S_1$ neurons, and an output linear layer of $S_2$ neurons [1]. A Radial Basis neuron model with $R$ inputs is shown in Fig. 1. Radial Basis Neuron uses the radbas transfer function. The net input to the radbas transfer function is the vector distance between its weight vector $w$ and the input vector $p$, multiplied by the bias $b$. (The \| dist \| box in this figure accepts the input vector $p$ and the single row input weight matrix, and produces the dot product of the two.) The transfer function for a radial basis neuron is given in “Eq. (1)”.

$$radbas(n) = e^{-n^2}$$  \hspace{1cm} (1)

![radbas transfer function](image)

Fig. 1. Radial Basis neuron model

A plot of the radbas transfer function is shown in Fig. 2.

![radbas transfer function](image)

Fig. 2. radbas( ) transfer function

The radial basis function has a maximum of 1 when its input is 0. As the distance between $w$ and $p$ decreases, the output increases. Thus, a radial basis neuron acts as a detector that produces 1 whenever the input $p$ is identical to its weight vector $w$. The bias $b$ allows the sensitivity of the radbas neuron to be adjusted. RBNN architecture is shown in Fig. 3.
The output of the first layer of this network can be obtained using “Eq.(2)”.

\[ a[1] = \text{radbas}(\text{netprod}(\text{dist}(\text{net}, IW[1,1], p, \text{net}, b[1]))) \] (2)

If you present an input vector to this network, each neuron in the radial basis layer will output a value according to how close the input vector is to each neuron's weight vector. If a neuron has an output of 1, its output weights in the second layer pass their values to the linear neurons in the second layer. The second-layer weights LW 2,1 (or in code, LW{2,1}) and biases b2 (or in code, b{2}) are found by simulating the first-layer outputs a1 (A{1}), and then solving the linear expression “Eq.(3)”.

\[ [w{2,1}b{2}] * [A{1}; ones(1,O)] = T \] (3)

We know the inputs to the second layer (A{1}) and the target (T), and the layer is linear. We can use “Eq.(4)” to calculate the weights and biases of the second layer to minimize the sum-squared error.3

\[ Wb = T / [P; ones(1,O)] \] (4)

Here Wb contains both weights and biases, with the biases in the last column. There is another factor called SPREAD used in the network. The user chooses SPREAD that is the distance an input vector must be from a neuron's weight. A larger SPREAD leads to a large area around the input vector where layer 1 neurons will respond with significant outputs. Therefore if SPREAD is small, the radial basis function is very steep, so that the neuron with the weight vector closest to the input will have a much larger output than other neurons. The network tends to respond with the target vector associated with the nearest design input vector.

### B. Generalized Regression Neural Networks

A generalized regression neural network (GRNN) is often used for function approximation [11]. It has a radial basis layer and special linear layer. The architecture for the GRNN is shown in Fig. 4. It is similar to the radial basis network, but has a slightly different second layer.

Fig. 4. Generalized Regression Neural Network Architecture

The first layer is just like that for Radial Basis networks. The second layer also has as many neurons as input/target vectors, but here LW{2,1} is set to T. Here the nprod box shown above produces S2 elements in vector n2. Each element is the dot product of a row of LW2,1 and the input vector a1. The user chooses SPREAD, the distance an input vector must be from a neuron's weight.

### C. Advantages of Radial Basis Networks

1) Radial basis networks can be designed in a fraction of the time that it takes to train standard feed forward networks. They work best when many training vectors are available.

2) Radial Basis Networks are created with zero error on training vectors.

### III. VARIOUS CRITERIONS FOR ASSESSMENT OF ESTIMATION MODELS

1. Mean Absolute Relative Error (MARE)

\[ \text{MARE} \% = \frac{\sum f(R_E)}{\sum f} \times 100 \] (5)

2. Variance Absolute Relative Error (VARE)

\[ \text{VARE} \% = \frac{\sum f(R_E^2 - \text{mean}R_E^2)}{\sum f} \times 100 \] (6)

3. Prediction (n)

Prediction at level n is defined as the % of projects that have absolute relative error less than n.

4. Balance Relative Error (BRE)

\[ \text{BRE} = \frac{|E - \hat{E}|}{\text{min}(E, \hat{E})} \] (7)

5. Mean Magnitude of Relative Error (MMRE)

\[ \text{MMRE} \% = \frac{1}{N} \sum_{i=1}^{N} MRE_i \times 100 \] (8)

Where

\[ MRE = \left| \frac{\hat{E} - E}{E} \right| \] (9)
N = No. of Projects, E = estimated effort, \( \hat{E} \) = actual effort

\[
\text{Absolute Relative Error (RE)} = \left| \frac{\hat{E} - E}{E} \right|
\]

6) Variance Accounted For (VAF)

\[
VAF \% = \left( 1 - \frac{\text{var}(E - \hat{E})}{\text{var} E} \right) \times 100
\]

A model which gives lower MARE “Eq. (5)” [12], [13] is better than that which gives higher MARE. A model which gives lower VARE is better than that which gives higher VARE “Eq. (6)”. A model which gives lower BRE “Eq. (7)” is better than that which gives higher BRE. A model which gives higher Pred (n) is better than that which gives lower Pred (n). A model which gives lower MMRE “Eq. (8)” [12], [13] is better than that which gives higher MMRE. A model which gives higher VAF “Eq. (11)” [6] is better.

IV. EXPERIMENTAL STUDY

In carrying out our experiments, we have chosen the NASA 93 dataset taken from different NASA Centre’s. Out of 93 projects, randomly selected 83 projects are used as training data. A Radial Basis Network and Generalized Regression Network are created. The two networks are tested using the entire 93 dataset. For creating radial basis network, newrbe ( ) is used and for creating generalized regression network, newgrnn ( ) is used. The estimated efforts using Intermediate COCOMO, RBNN and GRNN are shown for some sample projects in Table 3. The Effort is calculated in man-months. Table 4 and Fig.5, Fig.6, Fig.7, Fig.8, Fig.9, Fig.10 & Fig.11 shows the comparisons of various models [7] basing on different criterions.

### TABLE IIIII

<table>
<thead>
<tr>
<th>Project ID</th>
<th>Actual Effort</th>
<th>Estimated Effort using COCOMO</th>
<th>RBNN</th>
<th>GRNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>8.40</td>
<td>8.40</td>
<td>9.96</td>
<td>6.37</td>
</tr>
<tr>
<td>6</td>
<td>10.80</td>
<td>10.80</td>
<td>14.92</td>
<td>10.72</td>
</tr>
<tr>
<td>13</td>
<td>36.00</td>
<td>27.86</td>
<td>42.25</td>
<td>27.82</td>
</tr>
<tr>
<td>18</td>
<td>98.80</td>
<td>98.80</td>
<td>78.23</td>
<td>63.66</td>
</tr>
<tr>
<td>21</td>
<td>60.00</td>
<td>60.00</td>
<td>60.59</td>
<td>50.64</td>
</tr>
<tr>
<td>22</td>
<td>48.00</td>
<td>48.00</td>
<td>65.95</td>
<td>29.36</td>
</tr>
<tr>
<td>23</td>
<td>90.00</td>
<td>90.00</td>
<td>72.96</td>
<td>62.72</td>
</tr>
<tr>
<td>29</td>
<td>72.00</td>
<td>72.00</td>
<td>100.25</td>
<td>32.98</td>
</tr>
<tr>
<td>45</td>
<td>352.80</td>
<td>326.40</td>
<td>326.42</td>
<td>290.51</td>
</tr>
<tr>
<td>49</td>
<td>400.00</td>
<td>400.00</td>
<td>404.10</td>
<td>279.83</td>
</tr>
<tr>
<td>56</td>
<td>750.00</td>
<td>703.06</td>
<td>738.81</td>
<td>602.70</td>
</tr>
<tr>
<td>57</td>
<td>600.00</td>
<td>600.00</td>
<td>600.00</td>
<td>436.04</td>
</tr>
<tr>
<td>65</td>
<td>1248.00</td>
<td>1248.00</td>
<td>1248.00</td>
<td>1509.59</td>
</tr>
<tr>
<td>66</td>
<td>420.00</td>
<td>420.00</td>
<td>420.00</td>
<td>1236.66</td>
</tr>
<tr>
<td>67</td>
<td>2120.00</td>
<td>2120.00</td>
<td>2120.00</td>
<td>1139.56</td>
</tr>
<tr>
<td>79</td>
<td>480.00</td>
<td>480.00</td>
<td>465.14</td>
<td>278.48</td>
</tr>
<tr>
<td>82</td>
<td>599.00</td>
<td>599.00</td>
<td>596.62</td>
<td>387.30</td>
</tr>
<tr>
<td>84</td>
<td>480.00</td>
<td>480.00</td>
<td>1174.01</td>
<td>471.71</td>
</tr>
<tr>
<td>91</td>
<td>4178.20</td>
<td>4178.20</td>
<td>3375.24</td>
<td>2393.96</td>
</tr>
</tbody>
</table>

### TABLE IV

<table>
<thead>
<tr>
<th>Model</th>
<th>VAF</th>
<th>MARE</th>
<th>VARE</th>
<th>Mean BRE</th>
<th>MMRE</th>
<th>Pred(30) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBNN</td>
<td>98.477</td>
<td>7.26</td>
<td>5.18</td>
<td>0.293</td>
<td>28.5399</td>
<td>91</td>
</tr>
<tr>
<td>GRNN</td>
<td>98.476</td>
<td>18.60</td>
<td>11.24</td>
<td>0.339</td>
<td>30.8303</td>
<td>80</td>
</tr>
<tr>
<td>COCOMO</td>
<td>33.645</td>
<td>47.22</td>
<td>46.89</td>
<td>0.776</td>
<td>59.5027</td>
<td>53</td>
</tr>
</tbody>
</table>

Fig. 5. Actual Effort versus RBNN Effort

Fig. 6. Estimated Effort of various models versus Actual Effort
V. CONCLUSIONS

Referring to Table 4, we see that Radial Basis Neural Networks yields better results for maximum criterions when compared with the other models. Thus, basing on VAF, MARE, VARE, Mean BRE, MMRE & Pred(30) we come to a conclusion that RBNN is better than GRNN or Intermediate COCOMO. Therefore we proved that it’s better to create a Radial Basis Neural Network for software effort prediction using some training data and use it for effort estimation for all the other projects.

REFERENCES


