MANTECH Publications

Journal of Software Engineering & Software Testing
Volume 7 Issue 3
ISSN: 2457-0516 (Online)

Review of Model Evaluation Factors for Software Development

Anuradha Goel[1], V. K Laxminarayan[2]

Student[1], Assistant Professor[2]

Department of CSE

Sawai Madhopur College of Engineering and Technology

Corresponding Author's E-mail:- *anuradhagoel75@gmail.com[1]*

Abstract

It takes estimation of model parameters to forecast a system's behaviour. The estimation of model parameters is done using optimization criteria. Most algorithms estimate model parameters using past data. The output generated by the model is compared to the known goal values (actual). The parameters are estimated based on the differences between the two. Different criteria are applied to compare various models created using the same data. Here, the information gathered for small-scale initiatives is utilised. Using a radial basis function network, we explore the challenge of estimating the software effort. Various current criteria for one and two predictors are used to compare accuracy. Then, we compared the outcomes of one and two predictors and suggested a new assessment criterion based on linear least squares. We have taken into account a different data set and assessed prediction accuracy with the new standard. Compared to a single statistic, the new criteria are simpler to understand. This approach may be used for any modelling and prediction, even though software effort estimation is taken into consideration.

Keywords: *Radial Basis Function, Software effort estimation, accuracy, linear least squares.*

INTRODUCTION

Understanding and forecasting a system's behaviour depends on its modelling. Each piece of software created is distinct since software is an intangible and software development does not include any

Page 97-107 © MANTECH PUBLICATIONS 2022. All Rights Reserved

manufacturing. We merely copy the programme, and this takes place quickly. There is no recognised manual for software engineering because it is not yet as developed as other technical professions. For every piece of software, there is no universal certification. The issue gets worse because there are different standards for measuring sizes throughout the world. Despite these issues, managers and software developers still need to use estimating methodologies to create a strategy. As a general rule, fundamental size measurements are made using lines of code (LOC) or function points (FP). Software work estimating techniques are offered, ranging in complexity. They can be based on an expert [1], an analogy [2], an analytical approach [3], or machine learning [4]. Neural networks, one of the machine learning techniques, are important for software development effort estimation (SDEE) [5].

A radial basis function network (RBF) may be created by altering simply the function width (spread), sometimes referred to as the impact factor [6]. RBF is commonly used to estimate the work required for software development, and it has been demonstrated to perform better [7]–[9]. The writers are inspired by this to employ RBF when estimating modest projects. When planning labour, timing, and cost for a project, the estimate is crucial. Underestimating a project might result in poor quality, a smaller scope, or even its termination. Due to managerial pressure, this might also occur when fitting the project to the budget.

On the other side, overestimating might result in workers being underutilised or an organisation losing the project during the bidding process. Both situations serve as organisational deterrents. It is necessary to calculate effort as precisely as possible. The definition of accuracy is where the actual issue is [10]. On the basis of linear least squares, a novel approach to accuracy assessment is suggested. For test data, a linear connection is constructed between the actual effort and the projected effort. For our investigations, we largely used the data from [11]. The following is how the paper is set up: The discussion of related literature is followed by a description of a radial basis function neural network in the next section. The next section offers experimental assessments made using the new methodology. References are provided after the conclusions.

RELATED STUDIES

The accuracy of SDEE or any prediction depends on the input data, algorithm, and

Page 97-107 © MANTECH PUBLICATIONS 2022. All Rights Reserved

accuracy computation criteria. Typically, training (verification) sets and testing (validation) sets are created using historical data. The model is constructed with training data. Test data is utilised to validate this model. The factor that has the greatest impact on SDEE is the size of software projects. The work required is relatively minimal for minor tasks. Based on two independent variables, new and altered (N&C) codes and reused (R) codes, Lopez-Martin [11] employed a fuzzy logic model. He contrasted the fuzzy model's performance with that of multiple regression models. According to the findings, there is no distinction between these two models.

Two fuzzy logic models Mamdani and Takai-Sugeno are studied in [12]. The evaluation of these methods with linear regression showed that Takai-Sugeno fuzzy system performs better. None of these works compares SDEE using one and two independent variables. We have used error characteristics to compare the performance of the two models as explained in [10]. We have followed the guidelines suggested in the literature. In [12], Mamdani and Takai-Sugeno fuzzy logic models are examined. The Takai-Sugeno fuzzy system outperforms various methods, according to an analysis of these techniques using linear regression. None of these studies compare SDEE using only one or a few unrelated factors. As described in [10], we utilised error characteristics to compare the effectiveness of the two models. We conducted statistical tests in accordance with the recommendations made in the literature [13].

Commonly used accuracy evaluation criteria are Mean Magnitude of Relative Error (MMRE), PRED which are defined as below [10], [14].

MRE= abs (actual - predicted)/actual (1)

Magnitude of relative error is calculated for each project. This is added for each project and average is calculated.

MMRE = sum (MREi)/n (2)

PRED(l) = k/n (3)

where k represents the quantity of projects with relative errors MRE lower than l. MRE is 90% if the actual value is 100 and the projected value is 10. If the projected value is 100 and the actual value is 10, on the other hand, MRE is 90%. MRE prefers the lower estimate even if the error is 90 in both scenarios. The Mean Magnitude of Error Relative (MMER), which substitutes

Page 97-107 © MANTECH PUBLICATIONS 2022. All Rights Reserved

projected values for actual values in the denominator, is added to prevent this.

MER=abs(actual-predicted)/predicted (4)

MMER = sum (MREi)/n (5)

This statistic favors over estimation. Another reason to support (4) is that the error (actual-predicted) is correlated with actual. To avoid the above two problems it is suggested to use balanced relative error

BRE = abs (actual - predicted)/min(actual, predicted) (6)

Extreme values also have an impact on the standard deviation, or mean, of the errors. All of these provide a challenge because we're seeking for a summary statistic. As an alternative, we suggest fitting a linear least squares curve between the actual and anticipated values. This equation ought to have an intercept of 0 and a slope of 1. Since we are comparing specific numbers rather than MMRE/lowest MMER's or PRED's maximum, this has a number of advantages.

MEASUREMENTS

For our experiment, we used the data provided in [11] and [15]. For short projects in a classroom context, LOPEZ1 data consists of Actual Effort (AE), N&C code (N&C), and reused code (R) [11]. The dependent variable or response is effort in minutes, while N&C code and R code are the two independent variables or predictors. 163 projects are used for training, while 68 projects are utilised for testing. Both training (N & C, R, AE) and test data are summarised in Table I. (N & CT, RT, AET). Table II provides the Pearson correlation coefficients for various variables. As can be seen, the linear association between R code and real work is weaker than the correlation between N&C code. [11] has further information on the data.

The LOPEZ2 data consists of a dependent variable, Development Time (DT), in minutes, and three independent variables, McCabe Complexity (MC), Dhama Coupling (DC), and Lines of Code (LOC) [14]. There are 41 observations in all. Eight observations have been chosen at random for testing, while the remaining ones are for training. We have presented summary statistics for the whole dataset in Table III due to the small sample size. Table IV lists the correlation coefficients for various factors. All of the relationships are substantial, as can be observed.

Page 97-107 © MANTECH PUBLICATIONS 2022. All Rights Reserved

Table. I Characteristics of Lopez1 Data

Variable	Mean	Stddev	Minimum	Median	Maximum
N&C	35.56	26.60	10.00	27.00	137.00
R	41.82	30.86	4.00	34.00	149.00
AE	77.07	37.81	19.00	67.00	195.00
N&CT	44.93	21.28	12.00	41.00	104.00
RT	35.43	23.71	1.00	30.00	100.00
AET	79.16	26.47	11.00	78.00	144.00

Table .II Pearson Correlation Coefficients for Lopez1 Data

N&C,R	N&C, AE	R, AE	N&C, RT	N&C, AET	RT,AET
0.114	0.747	-.032	-0.175	0.307	0.190

Table. III Characteristics of Lopez2 Data

Variable	Mean	Stddev	Minimum	Median	Maximum
MC	2.707	1.006	1.000	3.000	5.000
DC	0.169	0.058	0.077	0.167	0.333
LOC	13.610	5.563	4.000	13.000	31.000
DT	16.634	3.673	9.000	16.000	25.000

Table IV. Pearson Correlation Coefficients for Lopez2 Data

MC,DC	MC, LOC	DC, LOC	DT, MC	DT, DC	DT, LOC
-0.386	0.765	-0.435	0.708	-0.705	0.583

Radial Basis Function Neural Network

In situations where the precise link between input and output cannot be determined or the relationship is nonlinear, neural networks are frequently used. Feed forward neural networks are iterative in nature and call for a large number of parameters to be defined. But when the output layer is linear, RBF networks don't require iteration and its output may be calculated easily [6]. According to Reference [14], compared to a back propagation neural network, the RBF network is best suited to effort prediction for the software business. The Architecture of RBF is shown in Fig. 1 which consists of input layer, hidden layer and output layer. Hidden layer has h neurons and uses radial basis function

$$(x) = \exp\left[\frac{|x-c|^2}{\sigma_i}\right] \quad (7)$$

cj is the center and j is the radial distance spread. The output is given by

$$F(x) = \sum_{j=1}^{h} \beta_j \varphi(x) \quad (8)$$

βj is the output layer weights.

The output layer weights are determined using generalized inverse. In our study we have used MATLABR2010a® Neural Network toolbox function.

Page 97-107 © MANTECH PUBLICATIONS 2022. All Rights Reserved

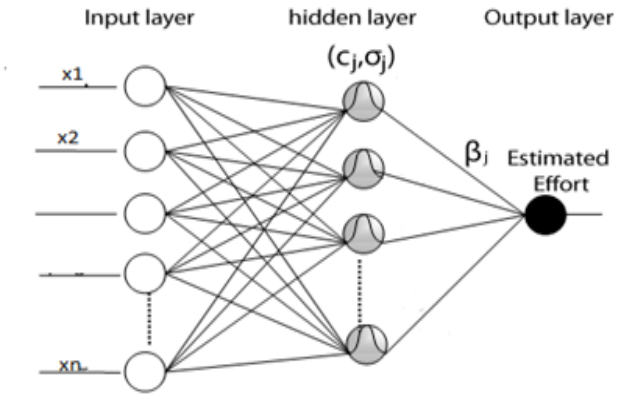


Fig. 1 Radial basis function neural network

EXPERIMENTAL RESULTS

A. LOPEZ1Data

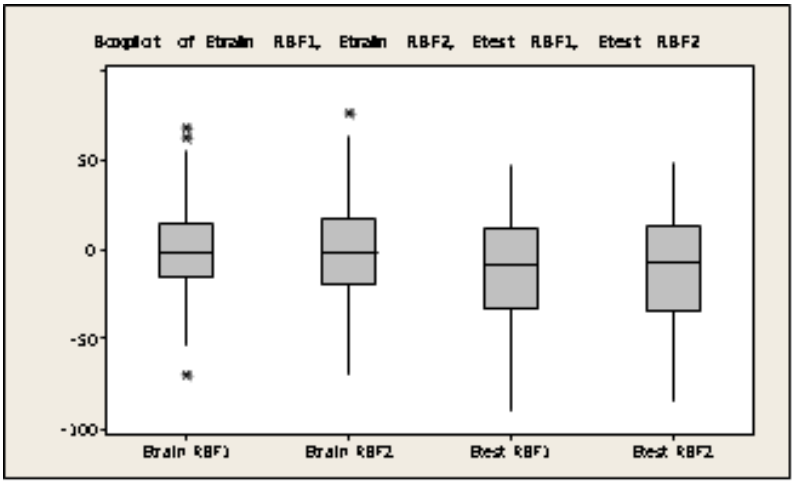


Fig. 2 Box plot of training and test errors for one and two variables for LOPEZ1 data

Studies were conducted using the 163 observation LOPEZ1 training data. The RBF (7) in an RBF neural network contains two constants, cj and j. The user can only define the spread; the centre, cj, is chosen from the input. The spread varies for the two input&C, R, and one output effort from 0.1 to 10. When spread is set to 1.0 and the number of hidden neurons is seven, RBF performance (measured as mean square error, or MSE), which is 0.0194, is at its lowest. When the spread is 1.0, the minimal MSE of 0.0190 is realised for the single input N&C. For 68 projects, the trained network is utilised to assess the RBF network's predictive power. Figure 2 presents the box plot of the training errors and the test (prediction) errors for the single-variable (RBF1) and two-variable (RBF2) examples. Table V provides the

Page 97-107 © MANTECH PUBLICATIONS 2022. All Rights Reserved

mean, median, and interquartile range (IQR) for the error (actual-predicted) data. One can see that there isn't much of a difference between one and two variables. We wish to use statistical tests to verify this finding. Table VI provides the obtained p-values for the nonparametric Mann-Whitney and t-tests.

We have also given effect size as suggested in the literature [13]. It is clear that statistically there is no significant difference between usages of one or two variables. We want to fit a linear least squares equation between actual and predicted effort. actual effort= a * predicted effort + b (9)If the actual effort and predicted effort are equal, the intercept (b) should be zero and slope (a) should be unity. The coefficients obtained for LOPEZ1 data are shown in Table.

This finding suggests that the intercept-based test data prediction is biassed in some way. For training data, RBF estimates perform well. The intercept and slope of the test data with one variable are somewhat lower. This demonstrates that for the LOPEZ1 data set, a single input performs better for prediction than two inputs.

Table .V Training and Test Errors for One and Two Variables Lopez1 Data

Variable	**One Variable Training**	**(N&C) Test**	**Two Variables Training**	**(N&C,R) Test**
Mean	0.000	-10.620	0.000	-10.36
Median	*-2.378*	-9.244	*.2.223*	-6.669
InterQuartile				
range	28.902	45.589	36.446	46.755

Table. VI Statistical Tests for One and Two Variables Lopez1 Data

	Training	Test
t-test, p values	1.000	0.961
Mann-Whitney test, p values	0.961	0.901
Effect Size	0.507	0.494

Page 97-107 © MANTECH PUBLICATIONS 2022. All Rights Reserved

LOPEZ2 Data

33 observations of the LOPEZ2 training data were used in the studies. The model may forecast the result by changing the spread parameter for the three inputs McCabe Complexity, Dhama Coupling, and LOC, as well as the single output Design Time, from 0.1 to 1.0. The spread of 0.40 and the number of hidden neurons of five result in the lowest RBF performance mean square error of 0.01329, respectively. For eight projects, the trained network is utilised to assess its capacity for prediction. Fig. 3 presents the box plot of training errors and test (prediction) errors. Table VIII provides the means, medians, and interquartile ranges (IQRs) for the errors.

Table. VII Coefficients for One and Two Variables Lopez1 Data

Variable	One Variable Training	(N&C) Test	Two Variables Training	(N&C,R) test
Intercept (b)	0.000	48.438	0.000	52.376
Slope (a)	1.000	0.342	1.000	0.299

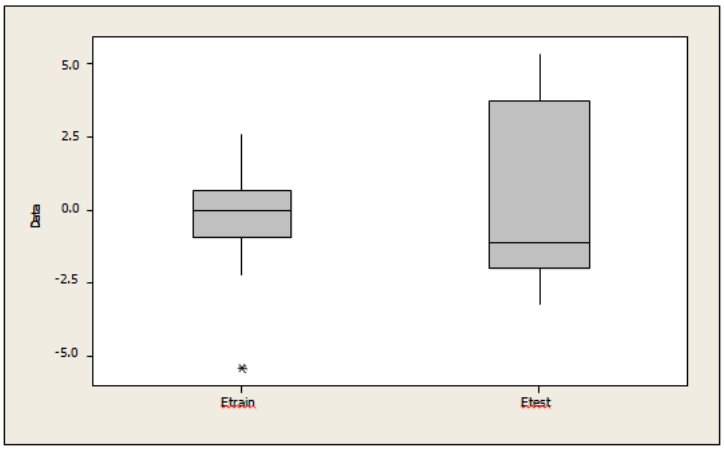


Fig. 3 Box plot of training and test errors for LOPEZ2 data.

Table VIII .Characteristics of Training and Test Errors Lopez2 Data

	Training	Test
Mean	0.000	0.27
Median	-0.017	-1.10
Inter Quartile Range	1.592	5.67

Page 97-107 © MANTECH PUBLICATIONS 2022. All Rights Reserved

Table IX. Coefficients for Training and Test Lopez2 Data

Coefficients	Training	Test
Intercept (b)	0.000	-3.0942
Slope (a)	1.000	1.2059

We want to fit a linear least squares equation between actual and predicted effort. If the actual effort and predicted effort are equal, the intercept (b) should be zero and slope (a) should be unity. The coefficients obtained for LOPEZ2 data are shown in Table IX. This indicates that the prediction is not as good as training.

CONCLUSION

Based on this study, it is possible to forecast the effort of small programmes using a single variable, N&C, as statistical tests do not reveal a significant difference between the two scenarios of one and two variables for LOPEZ1 data. The revised evaluation standards favour one variable for effort estimate while verifying for intercept and slope and fitting a linear least squares curve. RBF training is effective for LOPEZ2 data, however prediction accuracy is not.

Future research should strive to lower the intercept and achieve a slope of the linear least squares fit between projected and actual effort that is close to one. We have not attempted to compare various models; instead, the purpose of the study is to show the use of the new assessment criterion. But we have utilised two distinct data sets. The two main findings of the present study are as follows: I the use of linear correlation as a preprocessing step facilitate the selection of independent variables for effort estimate; and (ii) the use of linear regression to evaluate the predictive power of a model.

Although the effort estimating problem was used to illustrate the new criterion, this technique may be used to the assessment of any model. This approach compares various summary statistics looking for minimum or maximum with the predicted value of slope one and intercept zero of a straight line.

REFERENCES

1. M. Jørgensen, “A Review of Studies on Expert Estimation of Software Development Effort,” The Journal of Systems and Software, vol. 70, pp. 37–60, 2004.

2. M.J. Shepperd, C. Schofield, “

Page 97-107 © MANTECH PUBLICATIONS 2022. All Rights Reserved

Estimating Software Project Effort Using Analogies," IEEE Trans. Software Eng., vol. 23, pp. 736- 743,1997 A.W. Brown, S. Chulani, C. Abts, Software Cost Estimation with COCOMO II, Prentice Hall, 2000.

3. V.S. Dave, K. Dutta, "Comparison of Regression model, Feed-forward Neural Network and Radial Basis Neural Network for Software Development Effort estimation," ACM SIGSOFT Software Engineering Notes, vol. 36, pp. 1-5, 2011.

4. P.V.G.D. Prasad Reddy, K.R. Sudha, P. Rama Sree, S.N.S.V.C. Ramesh, "Software Effort Estimation using Radial Basis and Generalized Regression Neural Networks," Journal of Computing, vol. 2, pp. 87-92, 2010.

5. Simon Haykin, Neural Networks and Learning Machines, PHI learning Private Limited, New Delhi, 3rd edition, 2010.

6. A. Adri, A. Zakrani, "Design of Radial Basis Function Neural Networks for Software Effort Estimation," International Journal of Computer Science Issues, vol. 4, pp. 11-17, 2010.

7. V. S Dave, K. Dutta," Neural Network based Models for Software Effort Estimation: A Review," Artificial Intelligence Review, Springer, online 06 May 2012.to reduce the intercept and make the slope of linear least

8. B.A. Kitchenham, L.M. Pickard, S.G. Mac Donell and M.J. Shepperd, "What accuracy statistics really measure," IEE Proc. Software, vol. 148, pp. 81-85, 2001.

9. E. Praynlin, P. Latha, "Software Effort Estimation Models Using Radial Basis Function," International Journal of Computer, Information, Systems and Control Engineering, vol. 8, no. 1, World Academy of Science, Engineering and Technology, pp. 248-253, 2014.

10. J. Wen, S. Li, Z. Lin, Y. Hu, C. Huang, "Systematic literature review of machine learning based

Page 97-107 © MANTECH PUBLICATIONS 2022. All Rights Reserved

software development effort estimation models," Information & Software Technology, vol. 54, pp. 41-59, 2012.

11. C. Lopez-Martin, "A Fuzzy Logic Model for predicting the Development effort of Short Scale Programs based upon Two Independent Variables," Applied Soft Computing, vol.11, pp.724-732,

12. N. Garcia-Diaz, C. Lopez-Martin, A. Chavoya, "A Comparative study of Honolulu, USA, May 21-28, 2011, pp. 1-10.

13. C. Lopez-Martin, J. Leboeuf Pasquier, Cornelio Yanez. M, Augustin Gutierrez, T, "Software Development Effort Estimation Using Fuzzy Logic: A case study," Proceedings of the sixth International Conference on Computer Science, (ENC'05), IEEE Computer Society, 26-30 Sep. 2005, pp.113-120.

Page 97-107 © MANTECH PUBLICATIONS 2022. All Rights Reserved