Software Reliability Analysis Using Parametric and Non-Parametric Methods

Sultan Aljahdali
School of Information Tech
George Mason University
Fairfax, VA, 22030, USA
saljahda@gmu.edu

Alaa Sheta
Computer and Systems Dept.
Electronics Research Institute
Cairo, Egypt
asheta1@eri.sci.eg

Muhammad Habib
School of Information Tech
George Mason University
Fairfax, VA, 22030, USA
mhabib@gmu.edu

Abstract

In this paper we discuss an experimental evaluation of software reliability analysis using parametric and non-parametric methods. The experimental data set for different, small and large projects were used. We used the normalized root mean square error (NRMSE) as evaluation criteria. The experiments show that the non-parametric models are superior when compared to the parametric models in their ability to provide an accurate estimate when historical data is missing. A comparison among the power, exponential, S-Shape parametric, regression models and the neural network and fuzzy logic models are provided.

1. Introduction

In recent years, researchers have proposed several software reliability models to estimate the number of faults during the software testing process. These parametric models have a set of unknown parameters. These parameters must be estimated using observed historic failure data. Statistical estimation methods have been heavily used to solve this problem. For example, least square, maximum-likelihood, instrumental variable method were used to solve the parameter estimation problem.

In the past few years, a number of different software reliability models have been introduced [4] to solve the above-described problem. These software reliability models have been developed in response to the urgent need for software engineers, system engineers and mangers to quantify the concept of software reliability prediction. These models were useful in cases like:

- Managing reliability.
- Managing program changes.
- Monitoring test progress.
- Making software engineering and trade offs with schedules and cost.

In this paper, we provide a detailed comparison between various models that have been provided in literature for predicting faults in the software testing process. They are commonly known as software reliability models.

2. Software reliability model selections

Selection of a particular model is a challenging problem for software reliability prediction. There are two reasons for that. They are the selection of the release time and the value of resource allocation decision. These factors can affect the accuracy of the prediction. In the past, several solutions have been proposed to address the solution for the above-described problems. They are [6]:

- Use several software reliability models and select the one that gives the highest confidence.
- Use the recalibration method to compensate for the bias of a model.
- Use an adaptive model as an alternate approach

3. Evaluation Criteria

The model performance was measured in terms of the *NRMSE*.

$$NRMSE = \frac{1}{n-1}\sqrt{\frac{\sum_{k=1}^{n}(y(k)-\hat{y}(k))^2}{\sum_{k=1}^{n}(y(k))^2}}$$

Where $y(k)$ is the actual accumulated faults and $\hat{y}(k)$ is the estimated accumulated fault using various software reliability models.

4. Comparison among the models

In this section we cover the analysis part of the experiments. We have used three different data sets as a Benchmark, collected from different projects [1]. The data type is stochastic. Data sets range from military application projects, real time and control applications and finally, operating systems applications. This data are intensively used in literature to evaluate software reliability model [5].

4. 1. Power model

The Power model seems to be a poor estimator in many cases. Its *NRMSE* values are: 11.5220, 6.5423 and 4.4730, for the following projects: Military application # 40, Real time control #1, and Operating system # ss1c. The Power model consistently performs poorly in most cases. The observations show that the Power model is the worst predictor and the highest value of *NRMSE* among the parametric models in most cases.

4.2. Exponential model

The Exponential model's behavior is somewhat similar to that of S-shaped model, except for project operating system # ss1c, where it has the lowest value of 2.3925. Its *NRMSE* values are 9.0910 for military applications # 40, 2.8991 for the real time control #1, and 2.3925 for operating system # ss1c, which is the lowest predictor when it is compared to other predictions by other parametric models. This finding suggests that the Exponential model has the best predictability compared to the Power model and S-shaped model.

4.3. S-shaped model

The S-shaped model seems to perform relatively well. Though it is a best predictor in most cases, it has projected the remaining faults most accurately in two out of three projects. The *NRMSE* of the S-shaped model measures of 8.0388, 5.4161, and 2.4177 are the lowest for the project military application # 40, operating system # ss1c, and real time control #1. This finding suggests that the S-shaped model has good predictability. The results show that the Exponential model is superior to the other parametric models, and the S-shaped is close to the Exponential models.

4.4. A Parsimonious auto-regression model

The Auto-regression models magnitude of order 4 seems to perform poorly among the developed models. For example, the *NRMSE* in the military application project is 3.1413, real time control project # 1 is 1.7086, and the operating system # ss1c is 1.0659, which has the highest *NRMSE* value among non-parametric models compared to the fuzzy logic and neural network.

4.5. Neural Networks model

From the observation in Table 1, The Neural Networks (NN) seems to perform relatively well. Although the neural networks are the best predictors in most cases, they have projected the lowest *NRMSE* in two of three projects. For example, in the military application projects they are 0.5644, which is the lowest *NRMSE* among the entire project under study. In real time and control project # 1, the *NRMSE* is 1.0755, and for the operating system # ss1c, the *NRMSE* is 0.7714, mean while neural networks perform 66% better than fuzzy logic model.

4.6. Fuzzy logic model

The Fuzzy logic model performs relatively well through out all the experiments. Also, fuzzy logic has the best predictive capability in all the projects. For example, in the following, project military application # 40, and operating system # ss1c, the *NRMSE* are 1.1081, 1.2901, but in the real time control project # 1, the *NRMSE* is 0.9358, which is the lowest when compared between the two projects. As we have seen from this observation of the results, neural networks model gives the best or close to the best predictions in most of the cases. Also, the neural networks model is superior to the parametric models, regression model, and fuzzy logic.

Model	Military # 40	Real-time Control # 1	Operating Sys SS1C
Power	11.5220	6.5423	4.4730
Exponential	9.0910	2.9991	2.3925
S-Shaped	8.0388	2.4177	5.4161
Auto Regression (4)	3.1434	1.7086	1.659
Fuzzy Logic	1.1081	0.9358	1.2901
Neural Network	0.5644	1.0755	0.7714

Table 1: The performance of these models in term of normalized root mean square error *NRMSE*

5. One-Way Analysis of Variance (ANOVA)

The purpose of one-way ANOVA is to find out whether data from several groups have a Common mean. That is to determine whether the groups are actually different in the Measured Characteristic.

From the observation in Table 1, we found out that the real time and control project NRMSE is the lowest compared to other models. This is why we decided to investigate this model by implementing an ANOVA test with a significance level, $\alpha = 0.1$.
In Table 2, we show the result of one-way ANOVA test, the F value is $F_{(2,405)} = 2.83$ where 0.1. Is the significant level, and 2 and 405 represent the degrees of freedom. The observed significance level is p-value is equal to 0.06. We can assert that there is a significant difference among groups since $p < 0.1$.

Source	SS	DF	MS	F	Prob>F
Columns	0.0005	2	0.00025	2.83	0.0603
Error	0.03579	405	0.00009		
Total	0.03629	407			

Table 2: The result of the ANOVA test for the project Real-time and control #1

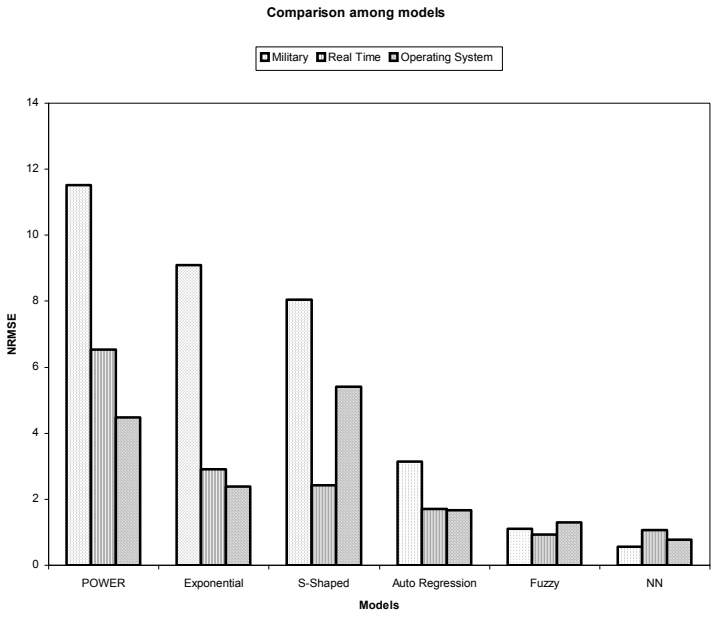

Figure 1: A comparison between various software reliability models with respect to NRMSE

Data sets for both large and small projects from diverse sources have been analyzed. Results presented here indicate that some perform better than others in most cases. This research shows evidence that a non-parametric approach provides the lowest normalized root mean square error and accurate result for the range of values in the experiments in most cases. The presented results show also that a fuzzy model and neural networks can be used to build prediction models for software reliability.

The results of the fuzzy logic and neural networks models were very promising. The error difference between the actual and estimated response was small. This finding gives a good indication of prediction capabilities of the developed fuzzy model and neural networks.

6. Conclusions and Future work

In this paper, we presented a comparison between various software reliability models. They include the power, exponential, S-Shape parametric, regression, neural network and fuzzy logic models. The experiments show that the non-parametric models are superior to the parametric models in their ability to provide an accurate estimate of reliability when historical data is missing. Currently we are investigating the use of evolutionary computations to solve the software reliability-modeling problem.

Bibliography

[1] J. Musa, "Data analysis center for software: An information analysis center," *Western Michigan University library, Kalamazoo, MI* 1980.

[2] K.M Atsumoto and K. Inoue, "Experimental evaluation of software reliability growth models," *in Proceeding of the IEEE of FTCS18*, pp.148-153, 1988.

[3] S. Brocklehurst, P.Y. Chan, B. Littlewood, and J. Snell, "Recalibrating software reliability models" *IEEE Trans. Software Engineering*, vol.16,pp. 458-470, 1990.

[4] J.D. Musa and K.Okumoto, "A logarithmic Poisson execution time model for software reliability measurement," in *proceeding of the 7th inter. Conf. Software Engineering*, pp. 475-478, 1984.

[5] Y. K. Malaya, N. Karunanithi, and P. Verman, " predictability measures for software reliability models" *in proceeding of the 14 the IEEE inter. Conf. Computer Software Applications*, pp. 7-12, 1993.

[6] N. Karunanithi, D. Whitely, and M.K., " Prediction of software reliability using connectionist models," *IEEE Trans. on software Engineering*, Vol. 18, no 7, pp. 563-574, 1992.

[7] R. Sitte, "Comparison of software reliability growth prediction: Neural Networks VS Parametric recalibration," *IEEE Trans. on Reliability*, vol. 48, no.3, pp. 285-291, 1999.

[8] Aljahdali, S., Sheta, A., "Modeling of Nonlinear Systems Using Genetic Algorithms", Proceedings of the IASTED International Conference on Artificial Intelligence and Application, 4-7, September 2001, pp.220-224.

[9] Aljahdali, S., Sheta, A., and Rine, D., "Predicting Accumulated Faults in Software Using Radial Basis Function Network", Proceedings of the ISCA 17th International Conference on Computers and their Application, 4-6, April 2002, pp. 26-29.

[10] Aljahdali, S., Sheta, A., and Rine, D., "Prediction of Software Reliability: A Comparison between Regression and Neural network Non-parametric Models", Proceeding of the IEEE/ACS Conference, 25-29, June 2001,pp.470-471.