MLP NEURAL NETWORKS BASED APPROACH FOR THE ASSESSMENT OF SOFTWARE PERFORMANCE TO IMPROVE THE SOFTWARE RELIABILITY

BONTHU KOTAIAH, R.A. KHAN

Abstract: Software Reliability is defined as the probability of the Failure-free operation over a specified period of time in a specified environment. We need to maintain the Software Reliability at an acceptable stage at all the stages of the Software Development Life Cycle(SDLC) Process. Hence, we should develop secure and efficient modeling techniques for software quality prediction, which in turn improves the Software Reliability. For this purpose Software Reliability Growth Models(SRGMS) will be used at different stages of the SDLC process to predict the Software Reliability effectively. Several different models have been proposed to predict the software reliability growth model (SRGM); But none of them has proven to perform suitable for all kinds of the Software Projects and they do not have the proper ability to find out number of errors in the software during development and testing processes. In this paper we are using multi-layer perceptron neural network model as an approach for assessing the software performance to improve the software reliability and the final results observed show that the neural network model adopted has good predictive capability.

Keywords: Artificial Neural Network, Software Error, Software Reliability, Software Development Life Cycle.

Introduction: Now a days, the assessment and the validation of the software reliability at different stages of the Software Development Life Cycle(SDLC) is very essential to find out the errors at every module of the developed software. The Software Reliability is the most and very important attribute of the software quality and at the end of the software development stage we should measure the software reliability to get the better quality software product. Previously so many software reliability growth models exist, but these models used only for solving the urgent needs of the software architects, software designers, software analysts, software engineers to convert the software parameters used for assessing the software reliability and to perform the better prediction of the software quality. The management of different software organizations used the information provided by these models to solve the issues regarding the software reliability effectively and efficiently.

The software reliability models traditionally are of two types.

1. Analytical software reliability growth models (SRGM) and;
2. Data-driven models.

The analytical SRGMs use stochastic models to specify the software failure process under several assumptions to provide mathematical tractability. The main disadvantages of these models is that they are restricted to assumptions. But, the data-driven models adapted the method s of time-series analysis, including the traditional autoregressive methods and modern artificial neural network (ANN) techniques , Case based reasoning, and optimal set reduction and genetic algorithms and most of the Machine Learning Techniques. Always these data-driven models depend upon the past error history data of the software to estimate the current error rate. The main goal of the data-driven models is that they will provide the guidance to the software developers to find out which modules are error prone and in which areas of the software development needs more concentration in the maintenance cycle of the Software Development Life Cycle(SDLC) process.

The connectionist neural networks are used widely as an intelligent tools for forecasting purposes. They will learn from the beginning by adjusting the interconnections between the layers of the neural network. It will use so many machine learning algorithms and gives very good learning capability on the basis of the statistics learning theory. The main aim of our paper is to establish a method for tracing out the multi-layer perceptron neural-network model capability for modeling the software reliability prediction and there by increasing the software quality of the developed software. The multi-layer perceptron neural-network model we discussed in this paper uses the back-propagation algorithm for learning and training purposes.

The paper organization can be understood from the following discussion.

Section 2- Gives clear idea about the overview of the multi-layer perceptron neural-network model and its learning methodology.

Section 3- The data set used for discussing the results elaborated briefly.

Section 4- We perform some experiments and discussed on the performance evaluation of the proposed technique for assessing the software reliability.

Section 5 – The detailed results of the experiments conducted in section 4 are discussed.

Section 6- Provides the conclusion of the paper.

Connectionist Predicting Models

For the past years, so many software reliability

growth models are identified and came as a result out from so many researchers to assess the software reliability of the software systems there by increasing the software quality to deliver the effective and efficient software product. But the main challenge for all these kinds of models is that is there any model to estimate the software reliability before it is delivered from the software development company to the end – user. The answer is absolutely "NO".

In the current research of the software reliability assessment of the software product, there are 2 methods are adapted widely in the software market. They are

1. The use of traditional analytical Software Reliability Growth models(SRGM)s for assessing the software reliability.
2. The use of data-driven connectionist models which depend heavily on the past failure history data for assessing the software reliability of the current software project being developed.

The connectionist neural networks models perform the learning process by adjusting the interconnections (in the form of attached weights) between the layers of the neural network. If we train the neural networks effectively and efficiently, then the neural networks be able to generalize the corresponding output/result for the given set of the input training data. Neural Networks will perform the learning process by the training algorithms like back-propagation algorithm etc., The neural networks training algorithm used to facilitate the neural networks to perform the learning process on the given set of input data, in a loop wise manner adjusts the interconnection weights((synapses). The used neural networks learning algorithm in our paper is back-propagation algorithm.

This artificial multi-layer perceptron neural network methodology helps the researchers to deliver the very good and useful and efficient non-linear systems which are accepting the large numbers of the input data, produces the correct or relevant output. It also maps the inputs to the outputs effectively in its design itself. It is always based on the relationship between the input and output.

For a training input set T consistingof n argument value pairs and given a d –dimensionalargument x and the respective target value t will begenerated by the neural network output. The functionfor this input-output mapping could be represented as g : $X \rightarrow T$ (1)

But, in most of the applications the input training data set T is in noisy mode. Our aim is to perform some smoothing functions in the form of activation functions to generate a network function that makes the generalization of the input training data set into the new function output values.

We should choose the weights properly to perform the learning process of the neural networks on the input training set effectively.

The representation of the closure on the input training set T is typically generalized with the addition of the error function of the of the form

$$E=\sum_{i=1}^{n} (y_i-t_i)^2$$

where yi is the trained neural network output.

Multi-Layer Perceptron : Among the available neural networks models, the multilayer perceptron(MLP) model is the best one. It is the widely used feed-forward neural network of the current generation. The best training algorithm for the these kind of feed-forward neural networks is the back-propagation algorithm. Especially the back-propagation algorithm that we used for training the multilayer perceptron neural networks is having the shape of the gradient descent. These multilayer perceptron neural networks are subject to local minima in corresponding the assigned weights of the nodes of the network. There exists a number of methods for diverting these back-propagation training algorithm based networks not to be subjected to local minima in the nodes weight space of the multilayer perceptron feed-forward neural network.

Among the adapted methods for minimizing the local minima, the best approach is to use the addition of momentum term to the back-propagation formula, that adds an additional input vector to the present node weight value. Like that some other methods like the Newton's method, or conjugate gradient algorithms exist [8]. As shown in Figure 1, the feed--forward neural network architecture contains a number of non-linear elements neurons. These neurons are grouped together to form a layer like inpu/hidden/output layers. Each neuron has a number of inputs and a single output. Each input has an assigned factor or parameter or value called the weight.

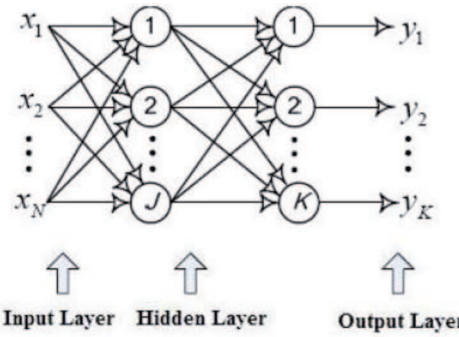

Figure 1, the multi-layer perceptron neural networks structure

The training algorithm used for training the neural networks, the back-propagation training algorithm is an loop wise gradient descent algorithm(when we see it in the mathematical notation),which is developed to minimize the Root Mean Squared Error E between the desired output of the multilayer perceptron feed-forward networks and the actual output of the multilayer perceptron feed-forward networks. Then it will modifies the weights in the direction of the gradient descent manner.

The process can be shown with the help of the formula as shown below in the form of expression:

$\Delta w = -\eta \nabla E$

where the parameter $\eta > 0$ is the learning rate that controls the learning speed of multilayer perceptron feed-forward networks. The gradient descent back-propagation algorithm is very fast, but it doesn't provide the globally accepted outputs in all the cases.

Finally, we will conclude that the error function to be minimized is the very high non-linear dimensional space, because the dimensions generally will be given by the weights attached with the neurons in the network and the number of weights in the neural network and to minimizing the local minima.

Data Set : The AT & T Bell Labs, Real-Time Command Control software package supplied an authenticated and useful source of data in the form of software information, which will supply the software for the software community. John Musa of Bell Telephone Laboratories compiled [2] a software reliability database. His objective was to collect failure interval data to assist software managers in monitoring test status, predicting schedules and to assist software researchers in validating software reliability models[18]. These models are applied in the discipline of Software Reliability Engineering. The dataset consists of software failure data on 16 projects. Careful controls were employed during data collection to ensure that the data would be of high quality. The data was collected throughout the mid 1970s. It represents projects from a variety of applications including real time command and control, word processing, commercial, and military applications. In our paper, we used data from one project, the .Real-Time Control project.

Experiment Setup

Test/Debug data for Real-Time Control : Observation of data for test/debug of a program for realtime control was used. The size of the program is 870 kilo-steps of COBOL and a middle level language. Since the test data is recorded day by day, the test operations performed in a day are regarded to be a test instance. The trainings accomplish for different models by dividing the data set into two sections, training and test sets, comprising of 70% and 30% of the total data set respectively. For real-time and control case study, we took the first 96 data points for training and the next 40 points for prediction/validation of the developed model.

Connectionist Models Structures

In our paper, we tried to implement the connectionist models using modeling software like MATLAB where the architecture of the MLP neural networks used for software reliability prediction consists of an input layer, one hidden layer, and an output layer. The input layer contains a number of neurons equal to the number of delayed measurements allowed to build neural networks model. In our case, there are four inputs to the network, they are y(k -1), y(k - 2), y(k - 3), and y(k - 4). Where y(k -1) the observed faults one-day is before the current day, the hidden layer consists of 4 nodes. The output layer consists of one output neuron producing the estimated value of the fault. The hidden layer and output layer nodes have tanh-sigmoidal activation function and learning rate is 0.002.

Evaluation Criteria : For assessing the performance of the developed model, we used the mathematical evaluation criterion for assessing the performance of the developed model. The criterion of evaluation (i.e. performance) to measure the performance of the developed connectionist model was defined as the sum of the square of the error(Root /Mean Squared Error):

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

Where y(k) is the observed fault and $\hat{y}$ (k) is the predicted fault for the given model structure and 'N' represents the number of measurements used for estimating the model.

Experimental Results : This section demonstrated the results of the above connectionist learning algorithm runs, which explore various aspects of specification connectionist modeling methodology For the selected algorithms run, the best model is evaluated over the training and test set using prediction performance measurements. The training data from real time control and their predicted results from different model are shown in Figure 2. The forecasted and actually measured values where compared to verify the generated models and their predictability and generalization capability in Table 1. As shown in Table 1, The MLPmodel have less training error and high testing error as compared to the other models for the effective assessment of the software reliability.

Table 1- the values of the errors for MLP based training/test data set

	RMSE	
	Training Data	Test Data
MLP	0.6061	0.6677

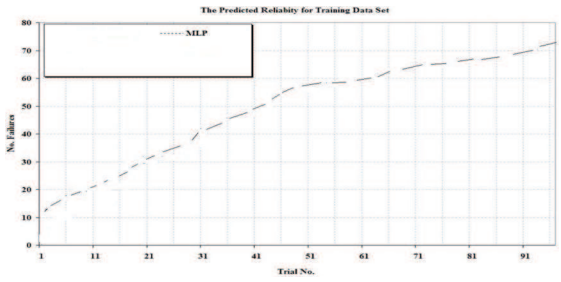

Fig 2: Actual and estimated faults for real time and control application for different models on training data set.

Conclusions : In this paper, we demonstrate the results of utilizing the connectionist modeling and learning algorithms for prediction the software reliability. The demonstration includes a comparison of the four connectionist models with different leaning algorithms or structures. The Elman recurrent neural networks is a robust technique for function prediction due capturing the dynamic behavior of the data set. The preliminary computational results in the MATLAB environment seem quite promising and give insight into the generalization capability of these models. In the future work, we can explore other soft computing techniques and other different data set.

References:

1. Aljahdali, S. "Prediction of Software Reliability Using Neural Network and Fuzzy logic",Ph.D. Dissertation presented to the faculty of College of Graduate Studies., Dept. of the Software Engineering and Info. System, George Mason University, Fairfax, Virginia, U.S.A, May 2003.
2. Aljahdali S., and El-Telbany M., "Genetic Algorithms for Optimizing Ensemble of Models in Software Reliability P
3. rediction", In the International Journal on Artificial Intelligence and Machine Learning (AIML), V8, ICGST, 2008.
4. Aljahdali, S., Sheta, A., and Habib, M. "Software Reliability Analysis Using Parametric and Non-Parametric Methods", Proceedings of the ISCA 18th International Conference on Computers and their Application, March 26-28, 2003, pp. 63-66.
5. 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.
6. 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, pp.470-471, 2000.
7. Demuth H., and Beale M., MATLAB, Neural Network Toolbox, version 4.0, 2004.
8. Ganesan K., Khoshgoftaar T.M., and Allen E.B," Case based Software quality prediction", International Journal of Software Engineering and Knowledge Engineering, 9(6), 1999.
9. Haykin S., Neural networks: A Comprehensive Foundation, Prentice Hall, 1999.
10. Hu Q., Xie M., and Ng S., Software Reliability Predictions using Artificial Neural Networks, Computational Intelligence in Reliability Engineering (SCI) 40, 197-222, 2007.
11. Jang, J-S.R and Mizutani, C-T., Neuro-Fuzzy and Soft Computing: A Computational Approach toLearning and Machine Intelligence. Eaglewood Cliffs, NJ: Prentice Hall, 1997.

Research Scholar, Babasaheb Bhimrao Ambedkar University, Lucknow, India
Associative Professor, Babasaheb Bhimrao Ambedkar University, Lucknow, India
kotaiah_bonthuklce@yahoo.com, [2]khanraees@yahoo.com