Article can be accessed online at http://www.publishingindia.com

A Dynamic Neural Network Model for Estimation in Software Development having Reusable Components

Jyoti Mahajan*

Abstract

In software engineering, the most critical task is to estimate the effort involved in the project due to inevitable activities involved in the software development process. The various factors such as change of requirements, intrinsic software complexity, lack of software data etc. are responsible to expect very accurate effort estimation in the software development process. Many estimation models are used to prove accuracy but none of them have able to prove that it has accuracy in all cases of software applications. The paper focuses on the estimation of the software development process having reusable components. The proposed model is based on dynamic neural network technique using back propagation algorithm which provides better accuracy for effort estimation.

Keywords: DNN, Reusability, Perceptron, Fuzzy, MRE

1. Introduction

A reusable code is basically a segment of source code that can be used again to add new functionalities with no or slight modifications. In object oriented programming, the use of reusable classes and modules reduces the implementation time because prior testing and use has eliminated bugs and localizes code modifications when a change in implementation is required. Reusability is often a required characteristic of platform software. When reusability is not required, various software developments aspects need not to be considered. A solution to a problem is based on predefined solutions to sub-problems; it is an act of synthesizing and called Reuse. There are six major steps performed at each phase in preparation for the next phase in the reuse activity. Reusability is the degree to which a component can be reused and reduces the software development cost by enabling less coding and more integration. The reusability of assets is different in different contexts. However, there are some characteristics that generally contribute to the reusability of assets. Although many of these characteristics apply to assets in general, we focus in this section, we focus on components as assets. At a higher level, we distinguish two aspects of reusability i.e. usability and usefulness.

Reusability = Usability + Usefulness

Usability is the degree to which an asset is 'easy' to use in the sense of the amount of effort that is needed to use an asset. Usability as such is independent of functionality of the component. Usefulness is the 'frequency' of suitability for use i.e. usefulness depends on the functionality, the generality and quality of a component. Selby in his research reported that developers were successful in achieving 32% reusability index from existing systems that are being reused at NASA laboratory. The experimental study conducted by Selby has identified two categories of factors for successful reuse-based software development in the large scale systems- module design factors and module implementation factors. The module design factors that characterise module reuse without revision were: low coupling and high cohesion, few input-output parameters, few reads and writes, and many comments. The main objective of software reuse is to minimise repetition of work, development time, cost and efforts, and increase reliability of the systems. It also improves the reusability and portability of the system.

* Assistant professor, Computer Engineering Department, Govt. College of Engineering & Technology, Jammu, India. Email: jmahajan1972@gmail.com

The reusability reduces the cost and maximizing profit which is considered to be an appraisal for employees of an organisation as suggested by Sandhu *et al.* (2006, 2009). In these studies it is clear that effort estimation being used on reusability is the answer for the drawbacks which the current estimation techniques are having.

2. Proposed Estimation Technique

The proposed estimation technique is having four phases – Preprocessing, Training, Estimation, and Analysis. In the first phase, the project data is analysed and a reusability matrix is derived from the project data. The software project is assumed to be divided into modules and the reusability of each module is analyzed using fuzzy rules to derive the reusability matrix. In the second phase, the training of the proposed neural network is done using the back propagation algorithm. The dynamic neural network is used in this proposed technique. The trained neural network could be used for estimation the effort involved. The results obtained in this phase could be analyzed for various purposes like analysis of finance, utilisation of resources, delivery timeline assertion and many more critical analyses. A project is said to be composed of modules. Modules could be either reusable or could be considered as new modules (m_n). The each module of a project is analyzed at an implementation stage and for characterisation a threshold is defined using a judgment model to identify the reusable component present in the module. On characterisation, the modules are further classified into 3 categories as

- Completely reusable A module is considered to be completely reusable if it is used completely without any change in the code or design and is represented as
- Reusable with fractional adaptation

A module is considered as a reusable model with partial adaption if at the implementation level the changes to be incorporated are less than the threshold and is represented as m_{fr}

- Reusable with prominent adaptation

If the changes to be incorporated are greater than the threshold then the module is considered as a reusable module with prominent adaption represented by. Let represent the changes to be incorporated into a module for its use in the project for which estimation is to be achieved. Applying the fuzzy rules the modules could be characterized as follows

$$\left\{\begin{array}{l} m = m_{cr} \ if \ andonlyif \ \Delta = 0 \\ m = m_{cr} \ if \ andonlyif \ \Delta < \vartheta \\ m = m_{cr} \ if \ andonlyif \ \Delta \geq \vartheta \end{array}\right\}$$

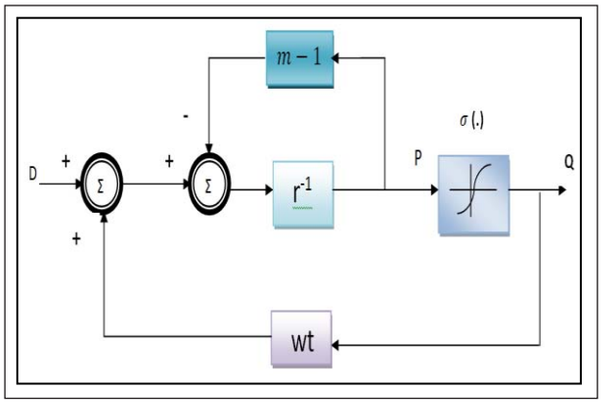
Consider R to represent the reusable matrix. The effort involved to develop the modules earlier is represented as . Let us consider that there exist number of modules and their development efforts considered be defined as. Then the reusability matrix obtained based on fuzzy logic could be represented as

$$R = \begin{bmatrix} m_{cr1} & \varepsilon_{cr1} & m_{fr1} & \varepsilon_{fr1} & m_{pr1} & \varepsilon_{pr1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ m_{crx} & \varepsilon_{crx} & m_{fry} & \varepsilon_{fry} & m_{prz} & \varepsilon_{prz} \end{bmatrix}$$

To estimate the effort involved, the proposed model uses neural networks. The static neural networks are not used for the proposed technique because they possess adaptive and learning capabilities for only static in-out relationships. In order to effectively adapt and learn the dynamic input-output of the non-linear matrices, a dynamic neural network is used here.

The dynamic Neural Network Adopted in this model is as shown as:

Figure 1: DNN Model



The training of the dynamic neural network is done using the reusable matrix obtained in the preprocessing phase. The multilayer perceptron model is used for creating a neural network, as this would be the appropriate network structure which would help in realising the problem. In a multilayer network there will be one input layer, atleast one hidden layer and one output layer. Back propagation is used as the training methodology *i.e.*, the learning rule. It is a supervised learning algorithm. It is a learning

methodology through which the network trains itself through multiple iterations over the test data. It does so by reducing an error function. The network eventually converges towards accurate values as it is trained with more and more training data. The activation function used here is the sigmoid/bipolar sigmoid function. The activation function is an abstraction of the action potential. It represents whether the cell should fire or not. The output of the dynamic neural network r (k) with respect to the input p (k) is given by

$$p(k+1) = -(m-1)\,p(k) + wtp\,(k) + d$$
$$r\,(k) = s(p(k))$$

where σ represents the sigmoid/bipolar sigmoid activation function and $(m-1)$ is the feedback where m is the learning rate constant. In the testing phase, the dynamic neural network provides the effort estimation phase wise as well as for the entire project.

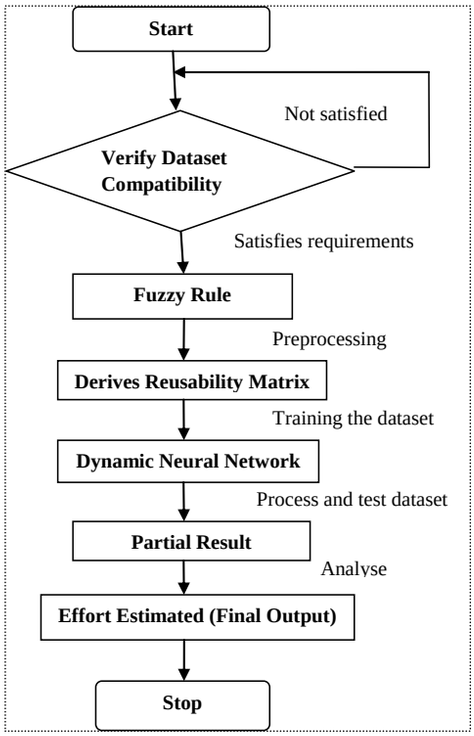
3. Experimentation and Results

For evaluation two types of datasets are used:

i. Historical Projects Dataset

ii. Commercial Projects Dataset

Figure 2: Experimental Evaluation Process



The experimental evaluation process considered is shown as a flowchart in Figure 2.

Figure 3: Training Graph of Training Dataset

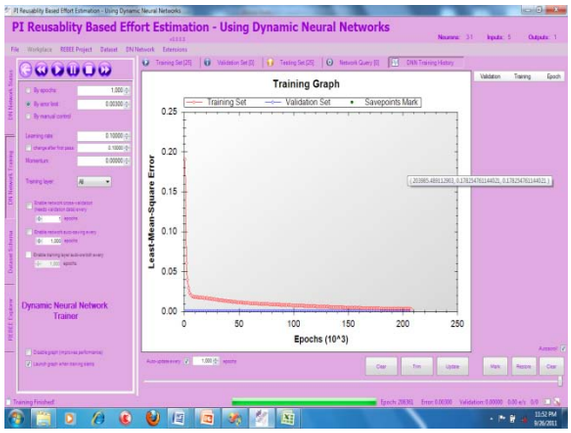


Figure 4: MRE Performance of Projects (Test Data) for Various Models

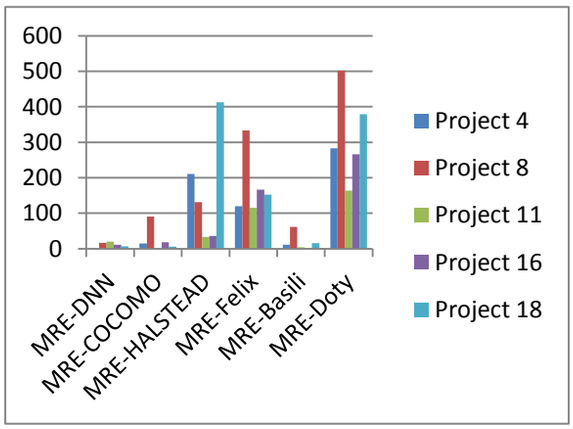
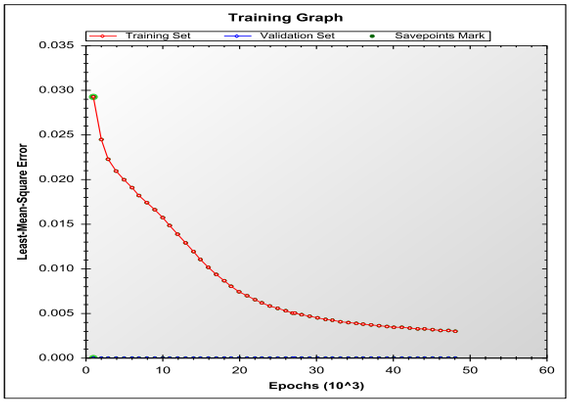


Figure 5: Training of Real Project Dataset



The dataset of NASA is used for the comparison of different models and experiments have been conducted to explore strength of the developed DNN based model.

Figure 6: Comparison of Actual Effort and Network Effort

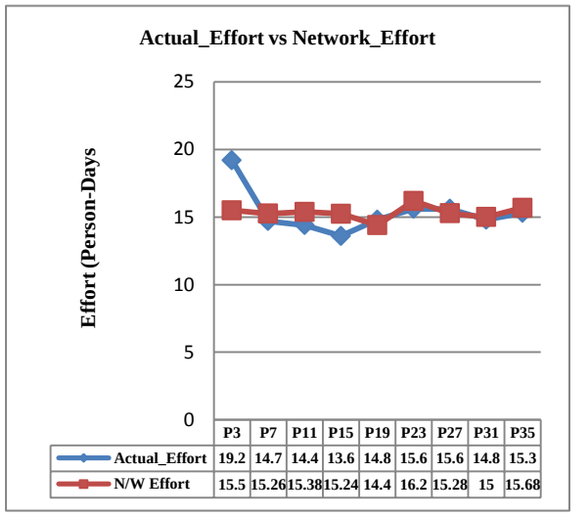
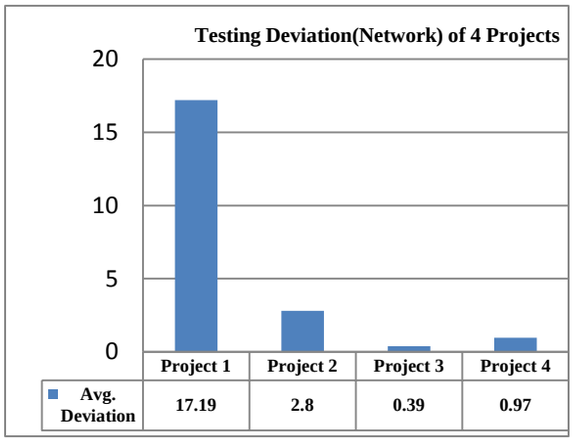


Figure 7: Testing Deviation Result of the Real Projects



4. Conclusion

Careful analysis of the results obtained using proposed dynamic neural network based model provides the information that the proposed estimation has a deviation of about 8% over the traditional method that has been chosen. This deviation is not much considering the fact that the effort estimated by the traditional method has also not being found to be accurate when applied to real time projects. The model has been evaluated on the historic dataset of different kinds of SEL projects. The development languages for these projects also varied from project to project. The reusability level of the projects varied from about 0% to a high of 96%.

The efficiency of the proposed model can be judged where the average estimation error for the historic dataset of SEL projects is coming about as low as 1.25%.Further the proposed model has been applied on real time data from four projects that have been specified above and interpretation of the results obtained and mentioned in the graph indicate that the estimated effort has been found to be always on the higher side of actual effort. The deviation found here is found to be on the positive side. When the results were analyzed with the real time data the proposed model has been found to be more accurate as the deviation varies from 0.3 % to 17%.

The DNN model trained using experimental data was found to have good generalisation capabilities and is able to successfully predict the effort closely matching the actual effort.

Future Scope

The future scope for the proposed model is based in the direction that the model developed needs to be applied to large number of test cases *i.e.*, real time projects as the proposed model has a unique feature of learning through usage. The model converges towards more accurate values as it used over time. The model developed can be evolved even further in the view that more number of parameters which have a minor effect on the effort estimation be also considered for effort estimation and the model can be evolved.

References

Abran, A., & Robillard, P. N. (1996). *Function Points Analysis: An Empirical Study of its Measurement Processes*. IEEE Transactions on Software Engineering, 22(12), 895-910.

Addison, T., & Vallabh, S. (2002). Controlling Software Project Risks - An Empirical Study of Methods Used by Experienced Project Managers. SAICSIT 2002, Port Elizabeth, South Africa.

Basavaraj, M. J., & Shet, K. C. (2008). Empirical validation of software development effort multipliers of intermediate COCOMO model. *Journal of Software*, May, 3(5), 65.

Jones, C. (2007). Estimating software costs: Bringing realism to estimating (2nd ed.). New York, NY: McGraw-Hill.

Jorgensen, M., & Sjoberg, D. I. K. (2001). Impact of effort estimates on software project work. *Information and Software Technology*, 43(15), 939-948.

Jørgensen, M. (2005). *Practical Guidelines for Expert-Judgment- Based Software Effort Estimation*. IEEE Software, May-June, 22(3), 57-63.

Peischl, B., Nica, M., & Zanker, M. (2009). *Recommending Effort Estimation Methods for Software Project Management*. In Proceedings of IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology, 3, 77-80.

Sandhu, P. S., & Singh, H. (2006). Automatic reusability appraisal of software components using neuro-fuzzy approach. *International Journal of Information Technology*, 3(3), 209-214.

Sandhu, P. S., Kaur, H., & Singh, A. (2009). Modeling of reusability of object oriented software system. *Journal of World Academy of Science, Engineering and Technology*, August, 56(1), 162.

Selby, R. W. (1988). *Empirically analyzing Software Reuse in a Production Environment in Software Reuse*: *Emerging Technology*. IEEE Computer Society Press.

Selby, R. W. (2005). *Enabling Reuse-based Software Development of Large Scale Systems*. IEEE Transactions on Software Engineering, June, 31(6), 495-510.