SOFTWARE EFFORT ESTIMATION USING RADIAL BASIS FUNCTION NEURAL NETWORKS

Ana Maria Bautista, Angel Castellanos, Tomas San Feliu

Abstract: *One of the biggest challenges that software developers face is to make an accurate estimate of the project effort. Radial basis function neural networks have been used to software effort estimation in this work using NASA dataset. This paper evaluates and compares radial basis function versus a regression model. The results show that radial basis function neural network have obtained less Mean Square Error than the regression method.*

Keywords: *software effort estimation, software repositories, radial basis function and artificial neural networks.*

ACM Classification Keywords: *I.2.6 Artificial Intelligence – Connectionism and neural nets, H.2.7 Database Administration – Data Ware house and repository.*

Introduction

The software projects are complex products of engineering which include many resources and the value of any must be accurate to make project not be defeated.

The reason for an emphasis on software effort estimation is that it provides essential part of the foundation for project management. Without a reasonably effort estimation capability the software projects often experience a lot of problems. An incorrect assumption of the software projects resources may lead the software projects to undesirable results. For any software organization, accurate estimation of effort is crucial for successful management and control of software project. In other words, in any software effort estimation, making an estimate of the person-months and the duration required to complete the project, is very important [Malhotra, 2011].

The goal of effort estimation is the management of software projects and achieving a comprehensive view of the costs of producing software. It is clear that the software projects effort estimation is a basic and key part of the software engineering. So, the software engineering uses the effort estimation and tries to give method for software projects to the project manager. The software project manager must define the success factors making the programming and controlling processes of the project be developed to avoid project be defeated and must utilize the needed limitations for developing the software projects. An important task in software project management is to understand and control critical variables that influence software effort.

In software engineering, effort is used to define the total time that takes members of a development team to perform a project. Software effort estimation is one of the most important processes in software projects development. SEE must be done before coding the software projects. The purpose of SEE to determine the scope of the project, estimate the amount of work required and the program is scheduled to run software projects.

The effort estimation of the software is input information in order to organize software development teams and allocate the project resources. So, the project manager executes techniques to meet the needs taking into

consideration the estimation effort. One of the main factors of software projects management is the accurate information about the time, effort and costs needed for the project execution. Also using the project records is very effective in success of the software projects and the estimation could be done more reliably.

A software engineering (SE) data repository is defined as a set of well-defined, useful, and pertinent real-world data related to software projects, called datasets, which include quantitative and descriptive information about resources, products, processes, techniques, management, etc. Such data are being collected for various purposes by recognized organizations, as well as by individual software organizations and researchers. In most scientific and engineering disciplines, these data are useful for conducting benchmarking, experimental, and empirical studies. While highly varied and widely available in mature disciplines, data repositories are much less frequently found in emerging disciplines, including software engineering, as illustrated by the Guide to the Software Engineering Body of Knowledge [SWEBOK, 2004].

In this paper, we have done empirical study and comparison of some of the models software effort estimation. The models, which we are dealing with, are developed using statistical and neural networks methods in order to verify which model performs the best. Linear Regression and Radial Basis Function Neural Network have been used in this work. These methods have seen an explosion of interest over years and hence it is important to analyze their performance. They have been analyzed using NASA93 dataset of PROMISE repository that collected information about 93 projects.

This paper will be organized as follows: Previous research works section will describe software effort estimation methods and radial basis function neural networks. Next section will explain the proposed model. Later, the results of the proposed will be described and finally, the conclusion and the future works will be presented.

Previous research works

1. Software Effort Estimation Methods

There are numerous Software Effort Estimation Methods such as Algorithmic effort estimation, machine learning, empirical techniques, regression techniques and theory based techniques. For Software estimation methods, there are several models developed, which can be grouped in two major categories:

- Parametric models, which are derived from the statistical or numerical analysis of historical project data;
- Non-parametric models, which are based on a set of artificial intelligence techniques as neural networks, regression trees, genetic algorithms and rule base induction.

1.1. Parametric Models

Different algorithm models for work estimation, scheduling and costs of the software projects are suggested. Boehm defined one of the most known models for estimation of costs in 1981. COCOMO I was presented in 1981 and, COCOMO II was presented in 2000. It is used for getting estimation of the time and costs activities. The successful management of the software projects depends on the accurate estimation of the projects. Project manager must predict the probable problems and give comprehensive solution for them. Also the project manager must estimate the time and the resources needed for the activities in a way that the work force used in an optimized manner.

Different models of Effort Estimation are presented of which we have taken into consideration the following models.

One of the most identified algorithmic models for SEE is the COCOMO model [Boehm, 2000]. The COCOMO model is used for effort estimation of different software projects. The base COCOMO model is identified as equation (1) for SEE:

$$E = a * (Size)^b \quad (1)$$

The main factor in SEE is the effort rate needed for completing the project. In equation (1), the parameters 'a' and 'b' are the inaccurate estimation of the complexity of the software and 'Size' is the number of the lines of the program in KLOC which is the important factor affecting the accuracy and the efficiency of the estimation [Boehm, 1981].

Also, parameter 'E' the amount of effort based on units is Man-Months and this value is directly dependent on the size and complexity of the project. Whatever size and complexity of the project, the more would be the effort on the project. In COCOMO model parameters 'a' and 'b' depends on the size of the project. The different models of COCOMO for effort estimation using different values of 'a' and 'b' are showed in Table (1).

Table 1. COCOMO basic models for effort estimation

$E = 2.4 * (KLOC)^{1.05}$ Organic
$E = 3 * (KLOC)^{1.12}$ Semidetached
$E = 3.6 * (KLOC)^{1.20}$ Embedded

COCOMO basic model is a project estimation model that identifies the effort and software projects management using the models of Table (1). Using the COCOMO model it is possible to identify the effort estimation and identify the needed activities for reaching the goals of the project. The main goal of COCOMO model is that all elements of the project get the same view of the goals, stages, organization and the technical and management procedures of the project and the effort of these elements are in direction of the software projects goals.

Some of various models of SEE are presented in Table (2). These models, which are used by the software teams, are the tools for contributing the effort estimation and controlling the software projects. The main goal of the various models of SEE is to be sure of the final results and the costs of the project. The models of Table (2) compare the software projects from financial, technical and human points to the various models of effort estimation and make the techniques and tools be used by the project manager in execution of the project.

Model Name Model Equation

Table 2. Various models of effort estimation

$E = 5.2 * (KLOC)^{1.50}$ Halstead [Halstead, 1977]
$E = 5.5 + 0.73 * (KLOC)^{1.16}$ Bailey-Basili [Bailey, 1981]
$E = 5.288 * (KLOC)^{1.047}$ Doty [Laird, 2006]

1.2. Non parametric models

To extract information from the Software Repositories different techniques are used. Mohanty et al. classify intelligent techniques in the following [Mohanty, 2010]:

1. Different neural network (NN) architecture including multilayer perception (MLP) and cascade correlation NN;
2. Fuzzy logic;
3. Genetic algorithm (GA);
4. Decision tree;
5. Case-based reasoning (CBR);
6. Soft computing (hybrid intelligent systems).

The other techniques:

1. Analogy based;
2. Support vector machine;
3. Self organizing maps (SOM).

Specifically, this work will focus on studying the application of neural networks in existing repositories.

Neural networks are used broadly in the studies we have selected.

There are several neural network methods that hare used in software estimation. The most common neural networks are the following:

- MultiLayer Perceptron;
- Radial basis function (RBF) network;
- Neuron fuzzy networks.

2. Radial Basis Function Networks

In late 80's Radial basis function emerged as a new artificial neural network. Radial basis neural networks (RBF) are a powerful alternative to approximate and classify a pattern set some times better than multilayer perceptron (MLP) neural networks [Minku, 2013].

RBFs differ from MLPs in that the overall input-output map is constructed from local contributions of Gaussian axons, require fewer training samples and train faster than MLP. The most widely used method to estimate centers and widths consist on using an unsupervised technique called the k-nearest neighbor rule (see figure 1). The centers of the clusters give the centers of the RBFs and the distance between the clusters provides the width of the Gaussians. Computation of the centers, used in the kernels function of the RBF neural network, is being the main focus to study in order to achieve more efficient algorithms in the learning process of the pattern set. The choice of adequate centers implies a high performance, concerning the learning times, convergence and generalization. The activation function for RBFs network is given by $\phi_i = \phi\left(\frac{\|X(n) - C_i\|}{d_i}\right) \; for\, i = 1,2...,m$

where $C_i = (c_{i1},...,c_{ip})$ are the center of the function radial, d_i is standard deviation. The Gaussian function $\phi(r) = e^{\left(\frac{-x^2}{2}\right)}$ is the most useful in these cases [Moody, 1989].

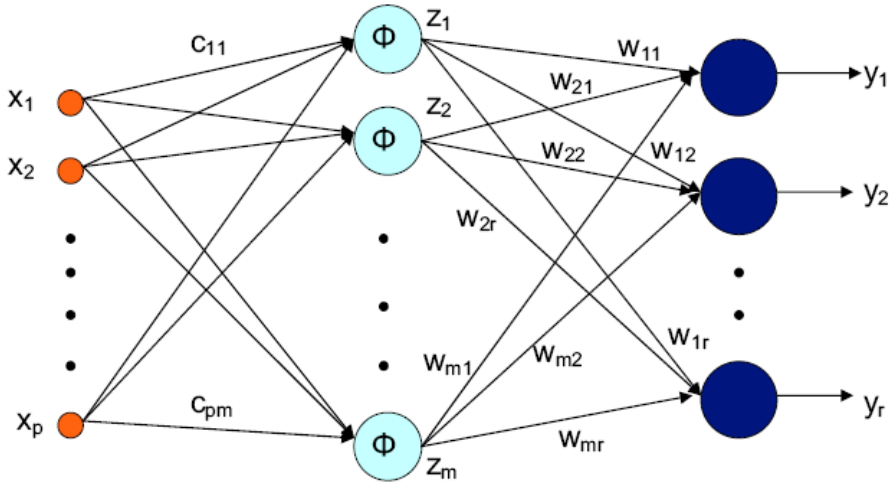

Figure 1. Radial Basis Function Neural Network

Proposed Model

The most important challenge we face in development of the large and complex software projects development is value accurate SEE. First the software projects were small and the costs of producing them included a small percent of the total costs and the error of the effort estimation did not affect the software execution considerably. But by the increase of the number, size and importance of the software projects and the costs of the software development, the software production is the most expensive element in software engineering and the increase of the costs has led the software teams to be defeated in production of the software projects.

The point to be considered in SEE is the method of selecting the suitable model among the estimation models in which the most accurate effort estimation takes place for the development of the software projects.

Also, one of the important goals of studying SEE is studying like costs and utilization of the software are very important. The goal of estimation is to provide the utilization and the control factors of the project and contributes the project manager to define the problem making fields. In the proposed model it is tried to use the RBF and evaluate using NASA project database and find the more accurate value.

In this section we conduct experimental studies and show related results for the experiment. In this experiment, we compare linear regression model and Radial Basis Functional Neural Network.

A. Database

Here we have used COCOMO NASA93 dataset, containing 93 projects. These NASA projects are collected from different NASA centers. These projects were developed during 1980's and 1990's. This database contains size in term of source lines of code (SLOC); the database contains KSLOC value, which means thousand SLOC. It also contains effort in person months and 15 other effort multipliers as described in COCOMO II.

B. Estimation Models

1) Radial Basis Function Neural Network (RBFNN): A radial basis function neural network has been implemented with two input neurons, one hidden layer and eight clusters: KLOC and time to estimate project effort (see figure 2). The net uses a competitive rule with full conscience with one the hidden layer and one output layer with the Tanh function, all the learning process has been performed with the momentum algorithm. Unsupervised learning

stage is based on 100 maximum epochs and the supervised learning control uses as maximum epoch 51938, threshold 0.00001.

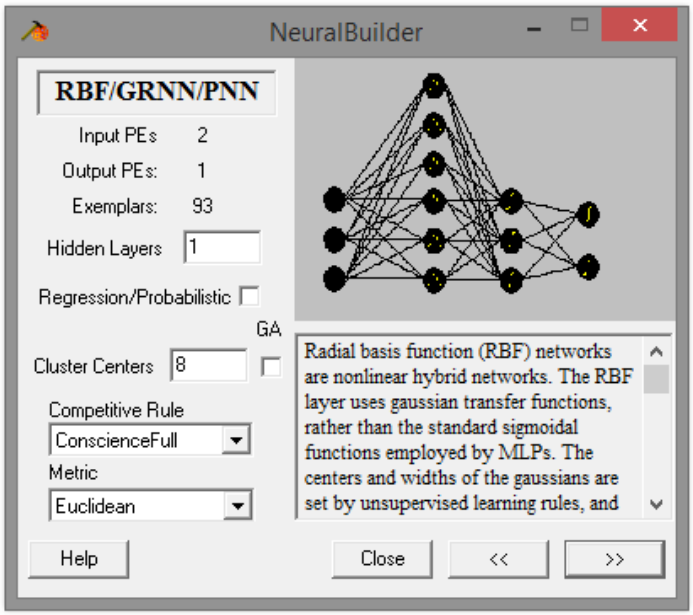

Figure 2. Structure of Radial Basis Function Neural Network Used

2) Regression Analysis Model: This is a traditional prediction method. Here we have used logarithmic function for effort estimation. Linear regression is not suitable in this prediction as effort is not linearly dependent on size (LOC) of the software. So nature log is more appropriate function.

We have performed an initial study using 93 patterns, in training set. Problem under study is prediction of software effort.

Results

The main results of the models studied are presented, first RBFNN and then Nonlinear Regression.

Radial Basis Function Neural Network has approximated in a good manner tested examples, getting a small mean squared error, see Figures 3 and 4 below:

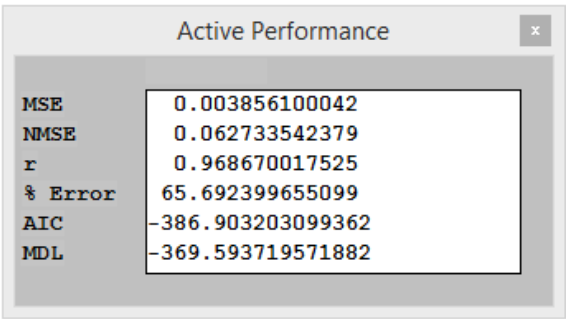

Figure 3. Table of mistake obtained

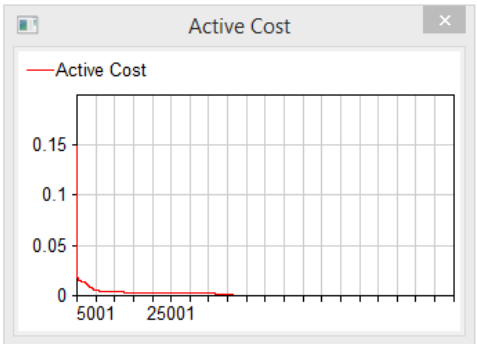

Figure 4. Graph of mistake

After training the network, a study of sensitivity was made. This study shows the influence that each input variable has on the output of the network, see Figure 5.

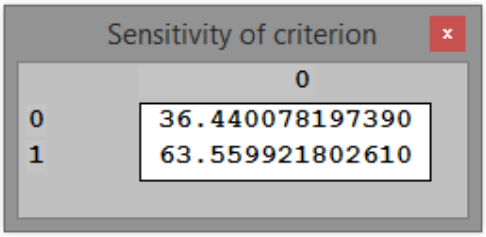

Figure 5. Sensitivity analysis

Now, it will be showed the results of regression model, see Tables 3 and 4. Effort is the dependent variable and Kloc and Time are independent variables. A nonlinear model is used due to the influence of COCOMO exponential models. Function to be estimated is: $a*Kloc^b*Time^c$. Marquardt defined estimation method used [Marquardt, 1983]. Estimation stopped due to convergence of residual sum of squares.

Table 3. Estimation Results

			Asymptotic	*95.0%*
		Asymptotic	*Confidence*	*Interval*
Parameter	*Estimate*	*Standard Error*	*Lower*	*Upper*
a	0.982764	0.731052	-0.469601	2.43513
b	-0.473374	0.164031	-0.799251	-0.147498
c	2.63606	0.386384	1.86844	3.40368

Table 4. Analysis of Variance

Source	*Sum of Squares*	*Df*	*Mean Square*
Model	1.02321E8	3	3.41071E7
Residual	5.26492E7	90	584991.
Total	1.5497E8	93	
Total (Corr.)	1.18711E8	92	
R-Squared = 55.6491 percent			

The output shows the results of fitting a nonlinear regression model to describe the relationship between Effort and 2 independent variables. The equation of the fitted model is $Effort = 0.982764*Kloc^{(-0.473374)}* Time^{2.63606}$.

In performing the fit, the estimation process terminated successfully. The estimated coefficients appeared to converge to the current estimates.

The R-Squared statistic indicates that the model as fitted explains 55.6491% of the variability in Effort. The adjusted R-Squared statistic, which is more suitable for comparing models with different numbers of independent variables, is 54.6636%. The standard error of the estimate shows the standard deviation of the residuals to be 764.847. This value can be used to construct prediction limits for new observations by selecting the Forecasts

option from the text menu. The mean absolute error (MAE) of 397.278 is the average value of the residuals. The Durbin-Watson (DW) statistic tests the residuals to determine if there is any significant correlation based on the order in which they occur in your data file.

Table 5 shows estimated correlations between the coefficients in the fitted model. These correlations can be used to detect the presence of serious multicollinearity, i.e., correlation amongst the predictor variables. In this case, there are 3 correlations with absolute values greater than 0.5.

Table 5. Asymptotic correlation matrix for coefficient estimates

	a	b	c
a	1.0000	0.7110	-0.9021
b	0.7110	1.0000	-0.9430
c	-0.9021	-0.9430	1.0000

Using response surface methodology, we explore the relationship between the variables (see Figure 6).

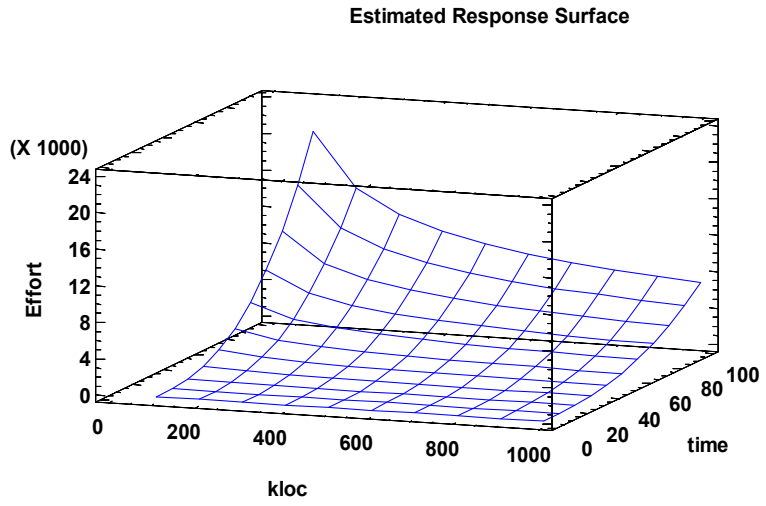

Figure 6. Estimated Response Surface

Conclusion and future work

This paper presents the results about the application of Radial Basis Function Neural Networks on Software Effort Estimation.

Radial basis function neural network learns with only a few patterns are really excellent. The dataset consists of 93 projects. We have obtained less Mean Square Error estimated using RBF than the regression method.

In the future work can further replicate this study using other software repositories in order to contrast the results.

Bibliography

[Bailey, 1981] Bailey J.W., Basili, V.R. "A meta model for software development resource expenditure," in Proceedings of the International Conference on Software Engineering, pp. 107–115, 1981

[Boehm, 1981] Boehm, B., "Software Engineering Economics. Englewood Cliffs", NJ, Prentice-Hall, 1981

[Boehm, 2000] Boehm, B. W., Madachy, R., & Steece, B. "Software Cost Estimation with Cocomo II with Cdrom", Prentice Hall PTR, 2000

[Halstead, 1977] Halstead, M. H. Elements of Software Science. New York, NJ, Elsevier, 1977

[Laird, 2006] Laird, L. M., & Brennan, M. C., Software measurement and estimation: a practical approach (Vol. 2). John Wiley & Sons, 2006

[Malhotra, 2011] Malhotra, R., & Jain, A. Software Effort Prediction using Statistical and Machine Learning Methods. International Journal of Advanced Computer Science and Applications, 2(1), 2011, pp. 1451-1521.

[Marquardt, 1983] Marquardt, D. W. An algorithm for least-squares estimation of nonlinear parameters. Journal of the Society for Industrial & Applied Mathematics, 11(2), 1963, pp. 431-441.

[Minku, 2013] Minku, L. L., & Yao, X. Ensembles and locality: Insight on improving software effort estimation. Information and Software Technology, 55(8), 2013, pp. 1512-1528.

[Mohanty, 2010] Mohanty, R., Ravi, V., & Patra, M. R. The application of intelligent and soft-computing techniques to software engineering problems: a review. International Journal of Information and Decision Sciences, 2(3), 2010, pp. 233-272.

[Moody, 1989] Moody, J. and Darken C. Fast learning in networks of locally-tuned processing units. Neural Computation, 1, 1989, pp. 281-294

[Shannon, 1949] Shannon, C.E. The Mathematical theory of communication. In: The Mathematical Theory of Communication. Ed. C.E.Shannon and W.Weaver. University of Illinois Press, Urbana, 1949.

[SWEBOK, 2004] IEEE "Guide to the Software Engineering Body Of knowledge- SWEBOK." Los Alamitos, California: IEEE Computer Society, 2004, 204 p., http://www.computer.org/portal/web/swebok (last accessed on 30/01/2013)

Authors' Information

Ana María Bautista *– E.T.S. Ingenieros Informáticos, Universidad Politécnica de Madrid, Madrid; e-mail: am.bautista@alumnos.upm.es*

Major Fields of Scientific Research: Artificial Intelligence

Tomas San Feliu *– E.T.S. Ingenieros Informáticos, Universidad Politécnica de Madrid, Madrid; e-mail: tomas.sanfeliu@upm.es*

Major Fields of Scientific Research: Software Engineering, Computer Science

Angel Castellanos *– Applied Mathematics Department. Universidad Politécnica de Madrid, Madrid; e-mail: angel.castellanos@upm.es*

Major Fields of Scientific Research: Artificial Intelligence