A DECISION SUPPORT SYSTEM FOR ESTIMATING COST OF SOFTWARE PROJECTS USING A HYBRID OF MULTI-LAYER ARTIFICIAL NEURAL NETWORK AND DECISION TREE

Javad Pashaei Barbin[1] and Hassan Rashidi[2]

[1]Department of Computer Engineering, Naghadeh Branch, Islamic Azad University, Naghadeh, Iran
[2]Department of Mathematics and Computer Science, Allameh Tabataba'i University, Tehran, Iran

ABSTRACT

One of the major challenges for software, nowadays, is software cost estimation. It refers to estimating the cost of all activities including software development, design, supervision, maintenance and so on. Accurate cost-estimation of software projects optimizes the internal and external processes, staff works, efforts and the overheads to be coordinated with one another. In the management software projects, estimation must be taken into account so that reduces costs, timing and possible risks to avoid project failure. In this paper, a decision- support system using a combination of multi-layer artificial neural network and decision tree is proposed to estimate the cost of software projects. In the model included into the proposed system, normalizing factors, which is vital in evaluating efforts and costs estimation, is carried out using C4.5 decision tree. Moreover, testing and training factors are done by multi-layer artificial neural network and the most optimal values are allocated to them. The experimental results and evaluations on Dataset NASA60 show that the proposed system has less amount of the total average relative error compared with COCOMO model.

KEYWORDS

Estimation the Cost of SoftwareProjects, Multi-layerArtificial Neural Network, Decision Tree, COCOMO.

1. INTRODUCTION

Making sure of meeting the demands of users in software projects, reconciling the production steps of software projects with the designs and plans already made from an engineering point of view, doing a project in specific circumstances and elimination of barriers in the administration of projects are of great importance [1]. The duties of the administrator of the software projects becomes evident in the necessity of supervising the developing projects with regard to the fact that the process of completion and development of projects is not tangible and assessable from a physical point of view and renewing technique operations to remove the defections of the system in the end require high quantitative and qualitative cost. In attention to the administrating points in software projects will not only result in the supposed advantages but also in the failure of the project. Meanwhile, detecting software risks and identifying the relationships among them play crucial roles in administration decision-makings.

The project manager is supposed to monitor the fulfillment of the duties of development team based on the administrating designs of the project; it is assumed that needs have been properly identified. The project manager is not responsible for improper identification of needs; however, while facing every kind of problem, in addition to informing the development team of the issue, he tries to offer a solution for it. The management of software projects while analyzing the estimated costs of previous projects can ensure better accuracy of estimation for the new projects [2].If estimation is accurate enough in the initial stages of project; it will result in success in human resources and cost. With knowledge of the cost of previous projects, management team can control the development of software projects using efficient methods.

Estimating the cost of software projects is identifying a set of methods, techniques, analytical and system-design tools in a framework of modeling design based on a conceptual paradigm for organizing the development process of software projects adopted in an effective way. In the software production and development model, accurate analysis of cost estimation and that of each stage based on the chosen methodology is vital in administrating morphology project. Production and development processes in each software project vary according to the methodology and process model chosen. In order to analyze, design and implement the presumed system, low-cost methods should be used. Estimation is a systematic way for allocating the activities, and responsibilities in a software development team. Its purpose is to produce high-quality software which meets the demands of ultimate users with a program and anticipatable budget. As software projects grew and software engineering developed, organized methods were devised for developing software, each of which is applicable based on the type of a project and its limitations. Producing great projects without any software-engineering methods is impossible. In order to produce complex software systems we need such methods and techniques, which if we want to proceed according to the normal process of usual software, we will not succeed. Therefore, we should have accurate estimation in great and complex projects.

Various algorithmic models such as COCOMO I [3], COCOMO II [4], and FP [5] are used to estimate the cost and effort software projects.SCE aims to increase the likelihood of success of projects by identifying and assessing systemic effort and cost. Accurate estimation at initial stages is of vital importance for project managers and software development companies in order to make the projects successful and the costs more reasonable. Algorithmic models such as COCOMO depend on a number of factors, which are used for calculating impractical features. These models try to formulize the relationship between features of efforts and the size of the projects. These models operate based on such criteria as number of lines of code or the degree of factors effective for estimation. The magnitude of projects is measured by these units; afterwards, the amount of effort required and costs is calculated.

We've organized the overall structure of this paper as follows: in the Section 2, we will review the literature related to cost estimation of software projects. We will discuss proposed model in the Section 3. In Section 4, we will explain the results and evaluation of the proposed model. Finally in Section 5, we will come to the conclusions and future work.

2. REVIEW OF LITERATURE

Using artificial intelligence techniques, many researches have been carried out in recent years on cost estimation of software projects. But, we cannot claim with absolute certainty that artificial intelligence methods can accurately estimate the cost 100%.However, based on the investigations

made, we daresay that artificial intelligence methods are more accurate and effective in comparison with Algorithmic models.

Using a combination of Ant Colony Optimization (ACO) and Genetic Algorithm (GA) Algorithms, a new model is proposed for SCE [6]. Evaluation was conducted on NASA60 dataset. ACO algorithm is used for training data and GA algorithm is used to test data. MMRE standard is the fitness function of the proposed model. Test results have shown that as the number of generations increases the degree of MMRE decreases in the proposed model. The results of 10 of software projects on NASA60 dataset showed that the MRE of the proposed model is less than COCOMO models. PSO algorithm is proposed for estimating the software effort [7]. Evaluation was conducted on the dataset KEMERER. In order to test and train PSO algorithm is used COCOMO model. Experimental results show that the proposed model is more accurate in comparison with COCOMO model. The degree of MMRE is 56.57 in the proposed model and 245.39 in the COCOMO model respectively.

PSO-FCM and PSO-LA hybrid models have been proposed for the SCE [8]. Evaluation was conducted on NASA60 dataset. InPSO-FCM hybrid model, the minimum of distance between the clusters, the total of inter-cluster distance and the number of clusters have been used as fitness parameters and for improving PSO algorithm. Use the FCM (Fuzzy C-Means) makes the particles gather in the best cluster and fitness function to have a great deal of local optimum points. LA (Learning Automata) is used to adjust the behavior of the particles so that the efficiency of PSO algorithm will be improved. In PSO-LA hybrid model, all particles simultaneously search for a place in the search space. In PSO-LA LA hybrid model, LA model strategy allows the particles to achieve a local optimum given the award criterion for PSO algorithm. Their results show that PSO-FCM hybrid model has less degree of MRE error rate PSO-LA hybrid model. The degree of MMRE error in PSO-FCM model is 25.36, 24.56, 24.22 and 23.86 and 26.32 in PSO-LA model. The degree of PRED accuracy (25) in the COCOMO model is 40 and 61.6, 58.3, 65 and 68.3 in PSO-FCM. Also in the PSO-LA, it is equal to 63.3.

A case comparison has been presented for SCE using data-mining techniques [9]. Evaluation has been carried on ISBSG, COCOMO81, Desharnais and Maxwell datasets. The results show that the degree of MdMRE error in Regression Trees (CART), M5, MLP, Radial Basis Function Networks (RBFN) models on ISBSG dataset is respectively, 61.6, 115, 55.7, and 70.4. And on COCOMO81 dataset, it is respectively 89.2, 105, 107, and 76.4. And on Desharnais dataset, it is respectively, 100, 36.1, 32.4, and 33.4. And on Maxwell dataset, it is respectively 100, 65.5, 57.1, and 48.4. The degree of PRED accuracy (25) on ISBSG dataset, it is respectively 16.3, 13.9, 26.1, and 22.7. And on COCOMO81 dataset, it is respectively 9.52, 15.9, 12.7, and 11.1. And on Desharnais dataset, it is respectively 2.47, 35.8, 38.3, and 34.6. And on Maxwell dataset, it is respectively 14.5, 19.4, 14.5, and 30.6. The assessments show that M5 and CART models perform better than other models.

3. PROPOSED MODEL

Using the test, repetition method and a comparison of the results and the results obtained, the number of layers used in this network indicates that three-layer network has better accuracy. In this paper, in order to enhance the performance of decision-helping system, we have considered hidden layers of neural network to be 15, which equals the parameters of software projects; then, we calculated the performance of network for the existent topology with different number of

neurons in the hidden layer. Finally, we chose the structure with higher-performance topology. Another parameter used by the user to get utmost performance is transfer functions in order to apply a threshold for each neural network layer. In this paper, tensing transfer function has been used for the neurons of hidden and output layers to improve the efficiency of the neural network [10]. Artificial Neural Networks, which are regarded to be one of the important methods for data-mining, is more efficient in estimating software cost in comparison with other methods. Neural networks are able to be taught without combining extracted knowledge with the rules, process them and extract their core knowledge. Hence, the use of neural networks is more efficient and useful than conventional methods for cost estimation. They also provide better and powerful tools to help managers to analyze, model and find relationships between resources, of doing which COCOMO and statistical methods are not capable.

There are several algorithms in neural network learning field; post-transmission algorithm is one of the most popular ones, which has various functions. Neural network utilizes the concept of learning to solve problems; gradient descending methods are widely used to train networks. By gradient descending method we mean the altercations of weight according to network error derivatives in a way that network error is minimized. This algorithm has two phases, the first of which is called feedback. In this phase, through neurons data are applied first from hidden layers into outer layer. The processed data is conveyed from previous layers into next hidden and outer layers by application on transfer functions. The second phase of post-transmission algorithm is the reduction of errors. In the first phase, once the application is due and calculation over, the actual output is obtained. Then, we will find the difference between actual and target output values of test. This value is called error. In the second phase, the error is run backward so that the communications which are the same as weights can be updated. In this paper, considering the top speed of post-transmission algorithm gradient combination is measured, and the supporting decision system is designed by this algorithm. Some of the advantages of this algorithm compared with other algorithms of search and minimization are its high accuracy, reduction of repetitions in learning process, hence reduction of learning time. This algorithm is fully automatic and independent of the parameters definition user. It prevents any waste of time in computational and learning processes. From an optimal point of view, learning in artificial neural network is equivalent to minimizing an error function, which is a multivariate function and is dependent on the weights of network.

Some of the most important decision tree algorithms can be C4.5 algorithm [11]. The algorithm uses an entropy-based criterion. It also uses pruning technique for removing extra branches. The process of training and testing is usually continued till all the groups are purified; namely, all the samples are placed in a category when the tree stops growing. The algorithm chooses a quality for each given item with regard to irregularities of each of them. After choosing the best quality, the qualities lacking value are allocated with values on the part of those that provide the data and the algorithm is continued. The choice of which quality should be in the stem depends on the GAIN of every feature. For calculating GAIN formula (1) is used:

$$Gain(S, A) = Entropy(s) - \sum_{v \in values(A)} \frac{|S_v|}{|S|} Entropy(S_v) \quad (1)$$

Where Values (A) is a collection of all the features of A and $S_V$is subset of S for which A has V. Entropy defines the purity (disorder or lack of purity) of a set of examples. If the S set consists of

a set of positive and negative examples of the target concept of entropy, *S* is defined to the category of Boolean as formula (2):

$$Entropy\ (S) = -P_{\oplus} \log_2 P_{\oplus} - P_{\Theta} \log_2 P_{\Theta} \quad (2)$$

In formula (2), $P_{\oplus}$ isthe proportion of positive examples to all of the examples and $P_{\ominus}$ is the proportion of negative examples to all of the examples. log0 = 0 is assumed to be zero. C4.5 algorithm is able to use disintegrated, noisy and valueless features; also it is able to sue features with high vales because of the use of Gain factor. Even if there is no error in the training data, pruning is done which makes the tree more perfect and less dependent on the training set. The deduction of tree begins with the node of the tree, which offers all the given data and classifies the data into smaller collections retrospectively. This is done by testing the degree of each of the groups. Sub-trees indicate the classifications of original data sets that complete the tests specified ratios measurement. This process usually continues till purification of sets is due; it means that all samples are put in one category and at this very moment the tree stops growing.

Input: an attribute-valued dataset D

1. Tree={}
2. **If** D is "pure" OR other stopping criteria met **then**
3. Terminate
4. **End if**
5. **For all** attribute aϵ D **do**
6. Compute information-theoretic criteria if we split on a
7. **End for**
8. a_{best} = Best attribute according to above computed criteria
9. Tree = Create a decision node that tests a_{best} in the root
10. D_s = Induced sub-datasets from D based on a_{best}
11. **For all** D_s**do**
12. $Tree_c = C4.5(D_v)$
13. Attach $Tree_c$ to the corresponding branch of tree
14. **End for**
15. Return Tree

Figure 1.Shows the pseudo code of decision tree

Figure (2) shows the model of decision supporting system for estimating the cost of software projects using a combination of multi-layered artificial neural network and C4.5 decision tree. In the proposed model,we have 15 effort estimation factors for network; testing and training data are done by C4.5 decision tree. C4.5 decision tree chooses the most optimal ones from among the factors and test the projects based on those values. This cycle goes on until MMRE function is more than COCOMO. We have chosen 80 percent of data for network training and 20% for data testing.

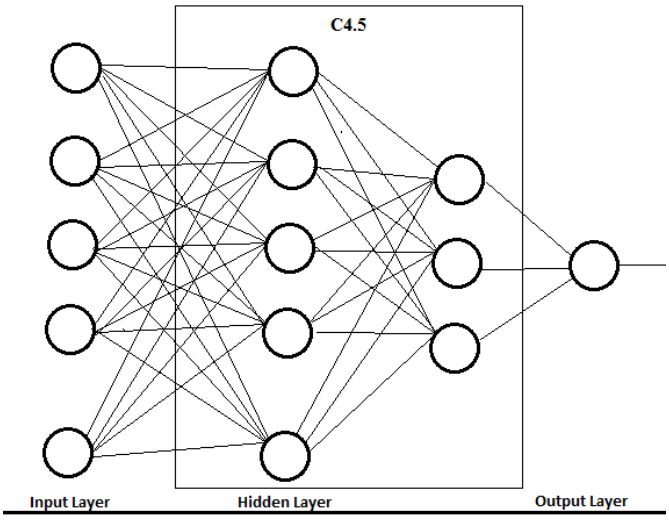

Figure 2. The proposed model

In the proposed model, MMRE is considered as error function for comparing with COCOMO model. The purpose of error function is to minimize the degree of error in the proposed model in comparison with COCOMO model; the proposed model is repeated until the degree of MMRE decreases. Fitness function is defined for the proposed model as equation (4) [12].

$$MRE_i = \frac{|a(i) - e(i)|}{a(i)} \times 100 \tag{3}$$

$$MMRE = \frac{1}{n}\sum_{i=1}^{n} MRE_i, i = 1,2,...,n \tag{4}$$

Using equation (4),we can obtain total error by the proposed and COCOMO models. PRED is also an important criterion in the accuracy of estimation. The most common methods for prediction accuracy are PRED and MMRE. PRED (x) is defined according to equation (5) [12].

$$PRED(x) = \frac{1}{n} \times \sum_{i=1}^{n} \begin{cases} 1, & if MRE \le x \\ 0, & otherwise \end{cases} \tag{5}$$

PRED (x) criterion, which is defined on the basis of MRE, is the most widely used in the accuracy of estimations and is a good measure of how well the models operate.

4. EVALUATION AND RESULTS

In this section, we have investigated the proposed model on NASA60 dataset. The proposed model is also compared with COCOMO model. Simulation of models has been conducted in VC#.NET 2013 programming. The results of the implementation of the proposed model while training and testing the data are shown in Table (1). The test results on the data sets of NASA60 [13] dataset show that the proposed model has less MRE error than COCOMO model.

Table 1: Comparison MRE of models

No.	KSLOC	Actual Effort	MRE COCOMO	MRE Proposed
1	2.2	8.4	24.15	21.36
2	3.5	10.8	3.95	2.65
3	5.5	18	7.36	6.98
4	6	24	58.88	35.26
5	9.7	25.2	20.05	15.14
6	7.7	31.2	23.91	21.65
7	11.3	36	30.83	15.98
8	8.2	36	29.55	32.54
9	6.5	42	28.22	11.24
10	8	42	22.22	9.87
11	20	48	27.21	19.65
12	10	48	41.66	32.85
13	15	48	46.19	14.98
14	10.4	50	34.90	25.64
15	13	60	9.36	1.23
16	14	60	25.88	16.98
17	19.7	60	6.10	7.85
18	32.5	60	93.91	52.65
19	31.5	60	3.81	6.54
20	12.8	62	27.96	20.98
21	15.4	70	22.51	13.24
22	20	72	60.76	45.65
23	7.5	72	41.75	32.10
24	16.3	82	29.79	21.98
25	15	90	39.54	16.84
26	11.4	98.8	42.04	32.65
27	21	107	36.75	12.98
28	16	114	34.48	25.74
29	25.9	117.6	27.85	20.35
30	24.6	117.6	31.65	15.32
31	29.5	120	18.94	16.54
32	19.3	155	35.78	13.62
33	32.6	170	29.88	14.98
34	35.5	192	32.10	12.65
35	38	210	28.46	32.62
36	48.5	239	24.31	16.57
37	47.5	252	37.81	12.95
38	70	278	21.28	16.52
39	66.6	300	23.76	17.54
40	66.6	352.8	35.17	20.38
41	50	370	36.90	16.54
42	79	400	45.74	32.51
43	90	450	38.29	19.84
44	78	571.4	24.50	20.36
45	100	215	120.66	95.84
46	150	324	49.50	30.26
47	100	360	44.97	29.54
48	100	360	15.85	12.32
49	190	420	1.89	6.95

50	115.8	480	11.37	8.97
51	101	750	19.87	20.65
52	161.1	815	4.76	3.02
53	284.7	973	38.36	21.54
54	227	1181	3.93	6.54
55	177.9	1228	3.64	1.98
56	282.1	1368	17.21	11.49
57	219	2120	29.00	8.74
58	423	2300	25.78	59.30
59	302	2400	0.46	1.06
60	370	3240	25.21	8.41

Figure (3) shows the comparative charts of MRE of the proposed and COCOMO model. As you consider, the degree of MRE in the proposed model is less in the majority of the projects; this is indicative of the efficiency of the proposed model and the accurate assessment of effort factors in software projects. The degree of MMRE in the COCOMO model is 29.64 and 19.98 in the proposed model. The results show that the proposed model has lower amount of MMRE in comparison to COCOMO model, and has reduced 1.45 times of that.

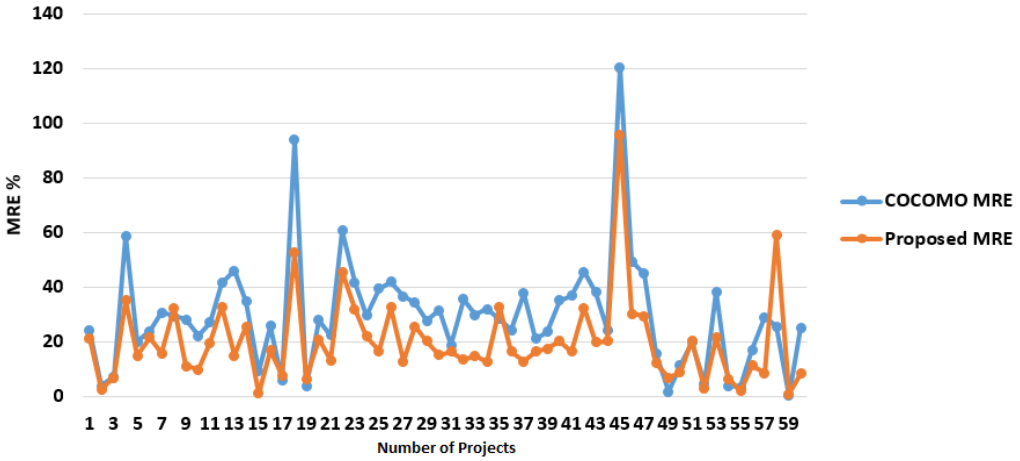

Figure 3. Comparison of the values of MRE

The results of evaluations have shown that enforcement of and running the proposed model in the face of the uncertainty factor values has better cost performance. Because the exact amount of cost factors affect the total amount of effort and cost estimates. The results of the implementation of the proposed model on NASA60 dataset showed that the proposed model is of a high caliber of estimates in comparison with COCOMO model. Figure (3) indicates the MRE diagram of the proposed model compared with COCOMO model on the basis of the number of iterations. As observed, MRE value reduces as the number of repetitions increases; this is the result of training and testing most of the data.

5. CONCLUSION

Some of the most challenging items projects in the development of software projects, and defining projects have always been accurate estimations of projects and their due completion. As complexity of projects increases, the cost of software development projects continues to grow.

Therefore, accurate estimation of costs in the initial stages of development seems to be very important. One of the most important decisions on software projects is a detailed estimate of the cost and reduction of risks. Proper evaluation of the project cost estimates traditionally with the help of experts has such limitations as high costs and time-consuming process of validation. In most cases, estimating the costsoftware depends on the number of line codes of software projects. The most important algorithmic model for estimating the cost of software projects is the COCOMO model. Today, with the development of artificial intelligence models, hybrid models have become widely spreading; in this article, based on a combination of artificial neural networks and decision tree model we have offered a supporting system to estimate the cost and effort of software projects. C4.5 decision tree is used to optimize multi-layer artificial neural network. It follows that this could help managers of projects in deciding the financial allocation for various projects. The results showed that the proposed model has less degree of MMRE error compared with COCOMO.

REFERENCES

1. P.C. Pendharkar, "Probabilistic Estimation of Software Size and Effort", Expert Systems with Applications, Vol. 37, Issue 6, pp. 4435-4440, 2010.
2. Maleki. I, Ebrahimi. L, Jodati. S, Ramesh. I, Analysis of Software Cost Estimation Using Fuzzy Logic, International Journal in Foundations of Computer Science & Technology (IJFCST), 4(3) 27-41, 2014.
3. B.W. Boehm, "Software Engineering Economics", Prentice-Hall, Englewood Cliffs, New Jersy, 1981.
4. B.W. Boehm, "Software Cost Estimation with COCOMO II", Prentice Hall PTR, Englewood Cliffs, New Jersy, 2000.
5. A.J. Albrecht, J. Gaffney, "Software Function, Source Lines of Code, and Development Effort Prediction: a software science validation", IEEE Transactions on Software Engineering SE, 9(6), pp. 639-648, 1983.
6. I. Maleki , A. Ghaffari, M. Masdari, A New Approach for Software Cost Estimation with Hybrid Genetic Algorithm and Ant Colony Optimization, International Journal of Innovation and Applied Studies, 5(1), 72-81, 2014.
7. F.S. Gharehchopogh, I. Maleki, S.R. Khaze, A Novel Particle Swarm Optimization Approach for Software Effort Estimation, International Journal of Academic Research, Part A, 6(2), 69-76, 2014.
8. F.S. Gharehchopogh, L. Ebrahimi, I. Maleki, S.J. Gourabi, A Novel PSO based Approach with Hybrid of Fuzzy CMeans and Learning Automata in Software Cost Estimation, Indian Journal of Science and Technology, 7(6), 795-803, 2014.
9. Z.A. Khalifelu, F.S. Gharehchopogh, Comparison and Evaluation Data Mining Techniques with Algorithmic Models in Software Cost Estimation, Elsevier, Procedia-Technology Journal, ISSN: 2212-0173, Vol. 1, pp. 65-71, 2012.
10. A.B. Nassif, D. Ho, L.F. Capretz, "Towards an Early Software Estimation using log-linear Regression and a Multilayer Perceptron Model", Journal of Systems and Software, Vol. 86, Issue 1, pp.144-160, 2013.
11. Quinlan J.R. (1993). C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Mateo.
12. MacDonell. S.G, Gray. A.R, A Comparison of Modeling Techniques for Software Development Effort Prediction, in Proceedings of International Conference on Neural Information Processing and Intelligent Information Systems, pp. 869-872. 1997.
13. Menzies. T, Port. D, Chen. Zh, Hihn. J, Validation Methods for Calibrating Software Effort Models, ICSE ACM, USA, 2005.