Asian Journal of Information Technology 15 (17): 3306-3310, 2016
ISSN: 1682-3915
© Medwell Journals, 2016

Learning Conventional Fuzzy Technique for Evaluating Software Attempts

[1]B. Arthi and [2]A. Grace Selvarani
[1]Department of Information Technology, Easwari Engineering College, Chennai, India
[2]Department of Computer Science and Engineering (PG),
Sri Ramakrishna Engineering College, Coimbatore, India

Abstract: The software cost evaluation with rich consistency, efficiency and improvement attempts is a major difficult task and this driven the software field to push and investigate research in evaluating software attempts for developing complex techniques. The evaluation based on assumptions is a technique to measure the software attempts. Moreover, the technique is employed for evaluating the software by assumptions and it cannot hold the statistical information in accurate and clear manner. A fresh technique is been proposed for evaluating the software attempts for artifacts characterized by the statistical or analytical information based on explanation by assumptions and fuzzy based methods. The established datasets inspected with fuzzy based datasets produces exact results to the datasets studied with traditional techniques.

Key words: Software cost evaluation, performance, statistical information, fuzzy technique and artifacts, established

INTRODUCTION

Due to the tremendous growth in software environment it is quite difficult to evaluate its development every now and then. The use of neural networks by the present development is perceived with a doubt by many of the scholars focusing on cost assessment (Kazemifard *et al*., 2011). But, it is to be noted that the benefits of neural networks in cost assessment are briefed for overcoming diverse problems even though there are some notable demerits available which retards its usage for general usage for assessing the costs. Different models hold notable merits like ability to attain knowledge and admirable understanding by not comprising the quality of Constructive Cost Model (COCOMO) (Du *et al*., 2015).

It is simple to perform the evaluation based on similarity as compared to the algorithm based models because the similarity based models are flexible enough to suit for local data which is difficult to adapt for algorithm based models (Attarzadeh and Ow, 2010). The technique can be employed for estimating the quality and extent of information based on the input datasets. The similarity based evaluation likely alleviates the results of conventional datasets because the evaluation based on similarity does not depend on single model for adjusting itself to handle all the projects.

It is quite difficult to evaluate the initial assessment because the prevailing information about the project data is not sufficient (Jodpimai *et al*., 2010; Du *et al*., 2015). The planned technique clearly estimates the efforts of a software using similarity based technique with traditional fuzzy based methods.

Literature review: Poonam Kaushal described that the software effort evaluation and risk estimation are the two major categories for a better software project. The quantitative techniques for planning a software and cost assessment are designed for aiding the project managers for approximating the plan and cost for a software project and to perform compassion analysis using different metrics. These models are used for balancing the manager's decision and perception for performing decisions. Huge variations from the evaluation can become a reason for fractional or complete breakdown of a software project. The risk evaluation is performed by the project manager's for identifying the reasons generating risks for making correct decisions. The author presents a methodological technique for examining risks along with software effort evaluation.

Finnie and Wittig (1996) describes that the software growth entails numerous interconnected features that distress the growth and efficiency. Most of the associations are not clearly understood due to which precise evaluation and progress time become quite difficult. Many of the evaluation models employed in the planned techniques are based on regression methods and the work analyzes the likely hood of artificial intelligence techniques called the artificial neural network and case based explanation for progressing the evaluation models. The artificial neural networks offer precise approximation during difficult associations between the variables and unclear inputs during high noise levels. The case based explanation resolves the problems by modifying the solutions observed the later problems which resemble similar to the prevailing problems. The performance of the both the techniques are analyzed for evaluating the progress in software and particularly the case based explanation allows progress evaluation using the similar dataset.

Jorgensen and Shepperd (2007) studied the software effort evaluation and observed that many projects face attempt or plan elaboration. These elaborations are quite lower than the expansion accounted by few professional companies. The evaluation method generally uses a special evaluation technique and the reason for the regular usage of special evaluation is due to the lack of facts that prescribed evaluation models guides to attain more precise results. There are extensive studies and examinations performed for attempting plan expansions.

Borade and Khalkar (2013) described that the primitive task of software project outlay and attempt evaluation is employed for precise calculation for the necessary workload and its associated outlays in software lifecycle. The software cost evaluation is a difficult task which requires understanding about the key elements affecting the results for software outcomes in terms of performance and isolation. The major problem addressed is that it requires huge number of information which is quite tough to obtain with required extents. These features makes software cost assessment a major problem among industries. The authors studied diverse conventional techniques for software project attempt and cost evaluation along with their related metrics.

The project crash is a primitive problem faced presently among software project managers due to vague evaluations because the size and need of software are subjected for change and it should be precisely estimated for analyzing the cost for progression in software. This is a major drawback in software industry and it is quite difficult to design suitable metric models for attaining good precision for improving software in all aspects. The implementation of useful models is required to precisely estimate the cost of software based products (Keung, 2008). The researchers examined diverse conventional models for software cost evaluation and techniques that suits best for all the situation and produces reasonable outcomes.

MATERIALS AND METHODS

Necessitate for software estimation: Small level projects can be easily evaluated and accurate precision is not required but as the size of project increases accuracy is required. It is noted that with increase in project size the accurate precision analysis are difficult (Huang *et al*., 2007). The better evaluation should have rough estimates for better decision making because the attempts performed for a project plays a major role and it is quite important. This examination begins with the progress of software projects and it serves as a schedule for upcoming activities. The evaluation with consistency remains a problem to be solved (Jorgensen and Shepperd, 2007).

RESULTS AND DISCUSSION

Evaluating software attempts: The fuzzy logic is based on the performance and interpretation and it is applied in cases where the decision making process is difficult. The technique is viewed as the expansion of conventional hypothesis by allocating values for an entity between two diverse limits characterized by an association function:

$$X = \int_y \mu_X(y)/y \tag{1}$$

Where:

y	=	A factor in Y
U_x (y)	=	The association function which differentiates the fuzzy set between the time [0, 1]

The fuzzy based system is categorized into three types as. The fuzzy technique alters the input into an association value. The purpose of assumption engine is to expand the density matrix for generating a fresh association value for resolving the efficiency rules based on fuzzy. The defuzzy brings the process for merging the outcomes as per the required statistical value.

Fuzzy correlation: The process is a conventional correlation process containing the following procedures.

Case classification: The primary objective is to classify all the software projects based on the element sets. The choice of elements portrays the software projects and this technique is quite difficult in the classification based process. The goal is to evaluate the attempts for a software project where the elements must be appropriate for evaluating the attempts and the technique is purely dependent on the classified information. The classification process for each software project is portrayed by an element set calculated by the statistical values.

For the statistical value y_0, the fuzzy rule is executed by the association function with values 1 for y and 0 otherwise. For classification purpose C elements are measured and for each element C_n the measure is described as X^n_m. The value X^n_m is characterized by based on the fuzzy set with a association function.

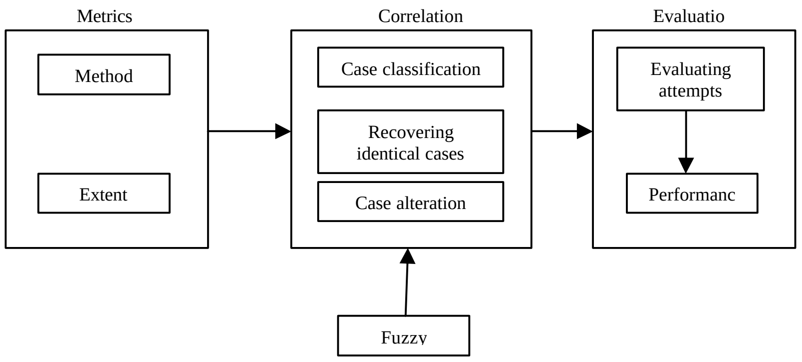

Fig. 1: Evaluating software attempts

It is observed that the fuzzy sets assure the standard conditions and it characterize the statistical data either as minimum or very minimum. It agrees to deal with ambiguity and vagueness for the classification process (Fig. 1).

Recovery of identical cases: The technique is based on the selection of software projects resemblance determination. A set of nominee calculation for software project comparison is planned for software project comparison. The calculation evaluates the entire comparison for two projects S_1 and S_2, e (S_1, S_2) by linking all the entity comparisons of S_1 and S_2 linked with several variables r_n entailing the projects S_1 and S_2, e (S_1, S_2). Furthermore, the obvious rationales for entity calculation for the entity distance:

$$e_{r_n}(S_1, S_2) = \left(\mu_{X_m^n} S_1\right)\left(\mu_{X_m^n} S_2\right) \quad (2)$$

Where:
$X^n{}_m$ = Represents the fuzzy rules linked with r_n
$X^n{}_m$ = The association function

The level based features are employed for understanding the goal of the software artifact in terms of elasticity and unity. The increase in attempts is the software dependability, difficulty and reusable techniques. The vagueness in outlays extensively concerns the precision in evaluating attempts that are resultant from designing the attempt evaluation. The vagueness in software attempt evaluation cannot be ignored because designing the fuzzy system attains merits in validating the outlays by modifying the fuzzy based sets:

$$\text{Attempts} = X * (\text{Range})^{b+0.01*\sum_{i=1}^{n} LF_i} * \prod_{i=1}^{n} AM_i \quad (3)$$

Where:
X = The constant
LF = The level based factor

Equation 3 is used for evaluating the attempts for a software projects. The technique is designed with cost evaluation factor in the ancestor part and the related attempt estimator resultant part. The value of defuzzy for every attempt evaluation is attained from the entity fuzzy assumption system after performing similarity, combining assumptions and related defuzzy techniques. The overall attempt is attained based on the product and the maximum values for the cost evaluation is higher than the preliminary evaluation. The minimum value is used for minimizing the evaluation about one half to the initial value.

Case alteration: The goal is to obtain an approximation for a fresh artifact based on the attained attempt values of the identical artifacts. The technique is not satisfied by fitting the similarity and so, it is planned that all the artifacts are used to obtain evaluation for new artifacts. All the conventional artifacts are used to evaluate the attempts for new artifacts in accordance to likeness.

Performance evaluation: Figure 1 depicts the artifacts gathered based on datasets with maximum attempts and contrasted with the planned technique. It is concluded that the datasets has a very minimum highest attempts as contrasted with the original highest attempts for the planned technique.

Here, the datasets are employed for calculating the standard attempts as contrasted with the original values. It is viewed that the conventional average attempts using the chosen factors are large as contrasted to the planned technique for each and every datasets. The conclusion is achieved by considering all the aspects for all the planned technique as contrasted to the conventional methods as depicted in Fig. 2 and 3.

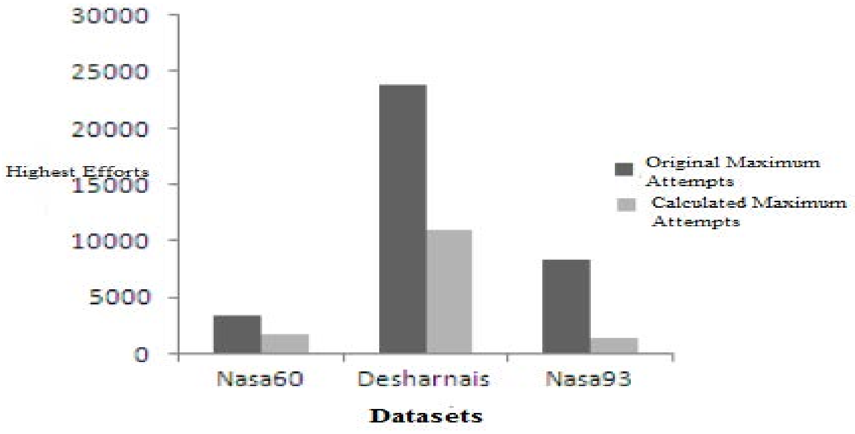

Fig. 2: Contrast between original and calculated highest attempts

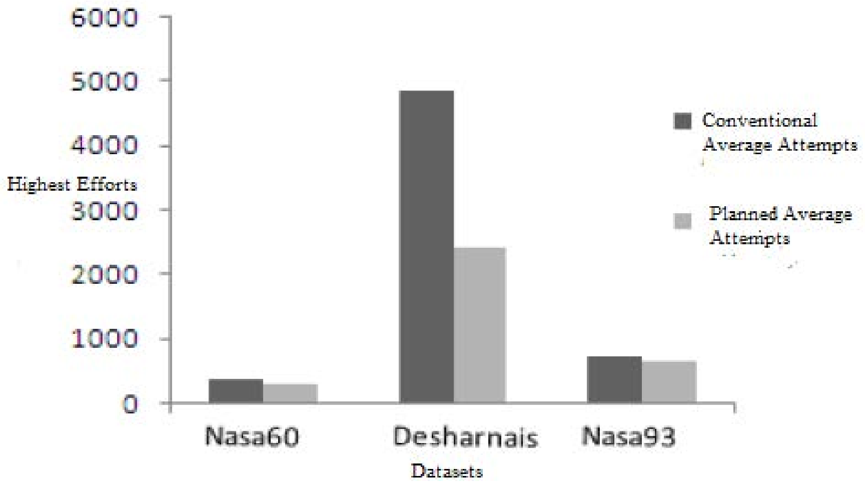

Fig. 3: Contrast between outcomes of average attempts in conventional and proposed techniques

CONCLUSION

A fresh conventional technique is planned for evaluating the average attempts for software artifacts. The technique is based on assumptions and fuzzy based technique for efficient utilization for briefing the software artifacts by either using statistical or definite information. The planned approach enhances the traditional assumption process while the employing the statistical information. For fuzzy based technique both the statistical and definite information are characterized by fuzzy based datasets. The key pro of the technique is that it can hold the performance and vagueness brilliantly while relating the software artifacts. Based on the evaluation obtained results proves that the planned technique is effective for evaluating the average attempts for the software project artifacts.

REFERENCES

Attarzadeh, I. and S.H. Ow, 2010. Improving the accuracy of software cost estimation model based on a new fuzzy logic model. World Appl. Sci. J., 8: 177-184.

Borade, J.G. and V.R. Khalkar, 2013. Software project effort and cost estimation techniques. Intl. J. Adv. Res. Comput Sci. Software Eng., 3: 730-739.

Du, W.L., D. Ho and L.F. Capretz, 2015. Improving software effort estimation using neuro-fuzzy model with SEER-SEM. J. Comput. Sci. Technol., 10: 52-64.

Finnie, G.R. and G.E. Wittig, 1996. AI tools for software development effort estimation. Proceeding of the International Conference on Software Engineering: Education and Practice January 24-27, 1996, IEEE, Dunedin, New Zealand, ISBN: 0-8186-7379-6, pp: 346-353.

Huang, X., D. Ho, J. Ren and L.F. Capretz, 2007. Improving the COCOMO model using a neuro-fuzzy approach. Appl. Soft Comput., 7: 29-40.

Jodpimai, P., P. Sophatsathit and C. Lursinsap, 2010. Estimating software effort with minimum features using neural functional approximation. Proceeding of the International IEEE. Conference on Computational Science and its Applications March 23-26, 2010, IEEE, Fukuoka, Japan, ISBN: 978-0-7695-3999-7, pp: 266-273.

Jorgensen, M. and M. Shepperd, 2007. A systematic review of software development cost estimation studies. IEEE Trans. Software Eng., 33: 33-53.

Kazemifard, M., A. Zaeri, N.G. Aghaee, M.A. Nematbakhsh and F. Mardukhi, 2011. Fuzzy emotional COCOMO II software cost estimation (FECSCE) using multi-agent systems. Appl. Soft Comput., 11: 2260-2270.

Keung, J., 2008. Empirical evaluation of analogy-x for software cost estimation. Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement October 09-10, 2008, ACM, New York, USA., ISBN: 978-1-59593-971-5, pp: 294-296.