A Review on Soft Computing-based Software Effort Estimation Models

Puja D Saraf
Assistant Professor
Department of Computer Engineering
R. C. Patel Institute of Technology, Shirpur

ABSTRACT
Accurately estimating the code size, cost, effort and schedule is probably the leading vital challenge facing code developers lately. It's major implications for the management of code development as a consequences of every the overestimates and underestimates have direct impact for inflicting damage to code companies. Heap of models square measure projected over the years by varied researchers for ending effort estimations. in addition variety of the studies for early stage effort estimations promoter the importance of early estimations. New paradigms offeralternatives to estimate the code development effort, specially the machine Intelligence (CI) that exploits mechanisms of interaction between humans and processes domain information with the intention of building intelligent systems (IS). Among IS, Artificial Neural Network and logic unit of quantity the two most popular soft computing techniques for code development effort estimation. The aim of this study is to research soft computing techniques inside the there models and to bring thorough review of code and project estimation techniques existing in trade.

Keywords
Effort Estimation, Fuzzy Logic, Soft Computing, COCOMO, LOC, Putnam model

1. INTRODUCTION
Software Estimation is responsive to the man blems y prothe software industry has experienced in creating significant cost and time estimates. Software estimation is base on measuring of software attributes which are typically related to the product, the process and the resources of software development [1]. This kind of measuring can be used as parameters in project management models [2] which provide assessments to software project managers in managing software projects to avoid problems such as cost overrun and behind the schedule. One of the most widely researched areas of software measurement is software effort estimation.

Software effort estimation models divided into two main categories: algorithmic and non-algorithmic.

The most popular algorithmic estimation models include Boehm's COCOMO [3], Putnam's SLIM [4]. These models require as inputs, accurate estimate of certain attributes such as line of code (LOC), complexity and so on which are difficult to obtain during the early stage of a software development project. Software development effort estimation is a vital aspect that deals with planning, prediction of amount of time and cost that will be incurred in developing of software project [5].

Despite considerable research and practical experience it is still a formidable challenge to understand and predict what happens in a large software projects. In 1995, Standish Group surveyed over 8,000 software projects for the purpose of budget analysis. It was found that 90% of these projects exceeded its initially computed budget. Moreover, 50% of the completed projects lack the original requirements [6]. From these statistics, it can be seen how prevalent the estimation problem is. Improving the accuracy of the cost estimation models leads to effective control of time and budget during software development. In order to make accurate estimates and avoid large errors, several cost estimation models have been proposed. Among those techniques, COCOMO is the most commonly used because of its simplicity for estimating the effort in person month for a project at different stages [7].

2. SOFTWARE EFFORT ESTIMATION MODELS
Software effort estimation models helps in estimating the quantity of effort that has to be place in to develop the package. However, the method estimation is unsure in nature because it for the most part depends upon some attributes that square measure quite unclear throughout the first stages of development; however it has to be allotted as large investments square measure concerned in building the package [8]. package effort estimation models divided into 2 main categories: recursive models and non-algorithmic models. recursive models square measure supported the applied mathematics analysis of historical information (past projects), e.g. package Life Cycle Management (SLIM) and COCOMO and Albrecht's perform purpose. These models rely on correct estimate of size of package in terms of line of code (LOC), variety of user screen, interfaces, complexity, etc., at a time once uncertainty is generally gift within the package [8]. The limitations of algorithmic models have led to the exploration of non algorithmic models which are based upon soft computing techniques. Non-algorithmic techniques are based on new approaches such as, Parkinson, Expert Judgment, Price-to-Win and machine learning approaches. The soft computing techniques include methodologies like artificial neural networks, fuzzy logic and evolutionary computations. Due to their inherent nature these techniques are used to handle imprecision and uncertainty [9].

2.1. Algorithmic Methods
2.1.1. Expert Judgment Method
Expert judgment techniques involve consulting with software package value estimation knowledgeable or a bunch of the consultants to use their expertise Associate in nursing understanding of the projected project to attain an estimate of its value. Typically speaking, a bunch agreement technique, metropolis technique, is that the best thanks to be used. The strengths and weaknesses are complementary to the strengths and weaknesses of recursive technique. to supply a sufficiently broad communication information measure for the consultants to exchange the degree of data necessary to calibrate their estimates with those of the opposite consultants, a band metropolis technique is introduced over commonplace metropolis technique [11]. The estimating steps victimization this method: organizers gift every

knowledgeable with a specification Associate in Nursing an estimation type Coordinator calls a bunch meeting during which the consultants discuss estimation problems with the organizer and every different. Consultants fill out types Associate in nursing ominously organizer prepares and distributes a outline of the estimation on an iteration form. Organizer calls a bunch meeting, specially specializing in having the consultants discuss points wherever their estimates varied wide. Consultants fill out forms, once more anonymously, and steps four and six are iterated for as several rounds as acceptable. The band metropolis Technique has after been utilized in variety of studies and price estimation activities [12].

The advantages of this technique are: The consultants will think about variations between past project expertise and needs of the projected project. The consultants will think about project impacts caused by new technologies, architectures, applications and languages concerned within the future project and might conjointly think about exceptional personnel characteristics and interactions, etc.

The disadvantages include: This technique can not be quantified. It’s exhausting to document the factors employed by the consultants or consultants cluster. Knowledgeable could also be some biased, optimistic, and pessimistic, even if they need been bated by the cluster agreement. The knowledgeable judgment technique perpetually compliments the opposite value estimating strategies like recursive technique [11].

2.1.2. COCOMO Models

One very widely used algorithmic software cost model is the Constructive Cost Model (COCOMO). The basic COCOMO model [4] has a very simple form MAN MONTHS = K1* (Thousands of Delivered Source Instructions) K2 Where K1 and K2 are two parameters dependent on the application and development environment. Estimates from the basic COCOMO model can be made more accurate by taking into account other factors concerning the required characteristics of the software to be developed, the qualification and experience of the development team, and the software development environment. Many of these factors affect the person months required by an order of magnitude or more. COCOMO assumes that the system and software requirements have already been defined, and that these requirements are stable. This is often not the case. COCOMO model is a regression model. It is based on the analysis of 63 selected projects. The primary input is KDSI. The problems are: In early phase of system life cycle, the size is estimated with great uncertainty value. So, the accurate cost estimate cannot be arrived at. The cost estimation equation is derived from the analysis of 63 selected projects. It usually has some problems outside of its particular environment. For this reason, the recalibration is necessary [13].

2.1.3. Putnam model

Another popular software cost model is the Putnam model. The form of this model is:

Technical constant C= size * B1/3 * T 4/3 Total Person Months B=1/T 4*(size/C) 3

T= Required Development Time in years Size is estimated in LOC Where: C is a parameter dependent on the development environment and it is determined on the basis of historical data of the past projects. Rating: C=2,000 (poor), C=8000 (good) C=12,000 (excellent) The Putnam model is very sensitive to the development time: decreasing the development time can greatly Increase the person months needed for development [13].

2.2. Non-Algorithmic Methods

2.2.1. Neural Networks

Neural networks square measure nets of process parts that square measure able to learn the mapping existent between input and output information. The vegetative cell computes a weighted add of its inputs ANd generates an output if the add exceeds an exact threshold. This output then becomes AN simulative positive) or restrictive (negative) input to alternative neurons within the network. the method continues till one or additional outputs square measure generated [14]. It reports the employment of neural networks for predicting package irresponsibleness, together with experiments with each feed forward and Jordan networks with a cascade correlation learning algorithmic rule The Neural Network is initialized with random weights and step by step learns the relationships underlying a coaching information set by adjusting its weights once conferred to those information. The network generates effort by propagating the initial inputs through ulterior layers of process parts to the ultimate output layer. every vegetative cell within the network computes a non linear operate of its inputs and passes the result hymenopter price on its output [15]. The favored activation operate is Sigmoid operate given among the many obtainable coaching algorithms the error back propagation is that the most utilized by package metrics researchers [16]. One in all the strategies is that the use of ripple Neural Network (WNN) to forecast the package development effort. The effectiveness of the WNN variants is compared with other techniques such as multiple linear regressions in terms of the error measure which is mean magnitude relative error (MMRE) obtained on Canadian financial (CF) dataset and IBM data processing services (IBMDPS) dataset [15]. Based on the experiments conducted, it is observed that the WNN outperformed all the other techniques. Another method is proposed to use radial basis neural network for effort estimation. A case study based on the COCOMO81 database compares the proposed neural network model with the Intermediate COCOMO. The results are analyzed using different criterions and it is observed that the Radial Basis Neural network provided better results

2.2.2. Genetic Programming

Genetic programming is one amongst the organic process ways for effort estimation. Organic process computation techniques area unit characterized by the very fact that the answer is achieved by suggests that of a cycle of generations of candidate solutions that area unit cropped by the factors 'survival of the fittest’ [18]. Once GA is employed for the resolution of world issues, a population comprised of a random set of people is generated. The population is evaluated throughout the evolution method. for every individual a rating is given, reflective the degree of adaptation of the individual to the atmosphere. A share of the foremost custom-made people is unbroken, whereas that the others area unit discarded. The people unbroken within the choice method will suffer modifications in their basic characteristics through a mechanism of replica [17].

A comparison is usually recommended by supported the accepted Desharnais knowledge set of eighty one computer code comes derived from a Canadian computer code house. It shows that Genetic Programming a suggestion some important enhancements quality and has the potential to be a legitimate extra tool for computer code effort estimation. A genetic formula needs genetic illustration of the answer domain and also the fitness performs for that. However the genetic formula works will be clearly understood by the pseudo code as follows:

Step 1 Initialize: To initialize the program me give initial random values to genes in population

Step 2 Evaluation: Evaluate This Gene Population. Each gene is tested in the present population and its fitness is calculated as a solution to the problem. If any gene has solved the problem, or it provides a good enough fit, depending on the application and its requirements, then terminate the programmer, go to SOLVED

Step 3 Next population: Generation of new genes by crossover from pairs of the highest fitness (scoring) last population genes. Randomly mutate or modify the values of a small fraction of a small number of these new genes

Step 4 Go to Evaluation.

Step 5 Solved: Finished

2.2.3. Fuzzy Logic

Fuzzy logic could be a valuable tool, which may be wont to solve extremely complicated issues wherever a Mathematical model is simply too troublesome or not possible to form. It’s additionally wont to cut back the complexness of existing solutions likewise as increase the accessibility of management theory [21]. The event of software package has forever been characterized by parameters that possess sure level of Fuzziness study showed that mathematical logic model includes a place in software package effort estimation [16]. the applying of mathematical logic is in a position to beat a number of the issues that area unit inherent in existing effort estimation techniques. Mathematical logic isn't solely helpful for effort prediction, however that it's essential so as to boost the standard of current estimating models. Mathematical logic permits linguistic illustration of the input and output of a model to tolerate inexactness. it's significantly appropriate for effort estimation as several software package attributes area unit measured on Nominal or ordinal scale kind that could be a specific case of linguistic values. a technique is projected as a Fuzzy Neural Network (FNN) approach for embedding artificial neural network into fuzzy abstract thought processes so as to derive the software package effort estimates [23]. Artificial neural network is employed to see the numerous fuzzy rules in fuzzy abstract thought processes. The results showed that applying FNN for software package effort estimates resulted in slightly smaller mean magnitude of relative error (MMRE) and chance of a project having a relative error of but or adequate zero.25 (Pred (0.25)) as compared with the results obtained by simply exploitation Artificial neural network and therefore the original model. Another proposal is that the use of set choice rule supported mathematical logic for analogy software package effort estimation models. Validation exploitation 2 established datasets shows that exploitation fuzzy options set choice rule in analogy software package effort estimation contribute to vital results [11]. Empirical study is finished not solely on the ten comes of NASA however additionally compared their results to the present models. Comparative study shows higher results therefore methodology projected is general enough to be applied to different models supported perform purpose ways and to different areas of quantitative software package Engineering. Fuzzy logic is a logic that is represented by fuzzy expressions which satisfies the following: Truth values, 0 and 1, and variables xi (Î[0,1], i = 1, 2, ..., n) are fuzzy expression

If f is a fuzzy expression, ~f (not f) is also a fuzzy expression If f and g are fuzzy expressions, f Ù g and f Ú g are also fuzzy expressions As in fuzzy expression, a fuzzy proposition can have its

Truth value in the interval [0, 1]

f: [0, 1] →[0,1]

3. CONCLUSIONS

This paper has totally different models for estimation however there’s no estimation technique which might present the most effective estimates all varied things and every technique will be appropriate. The square compute several code value estimation strategies accessible as well as algorithmic strategies, estimating by correspondence, professional judgment technique, high down technique, and bottom up technique. No technique is essentially higher or worse than the opposite, in fact their strengths and weaknesses are measure usually complimentary to every different. In Associate in nursing absolute sense, none of the models perform well at estimating code development effort, particularly long the MMRE dimension. However in a very relative sense ANN approach is competitive with ancient models. Once more as a comparative analysis, genetic programming will be wont to match advanced functions and might be simply taken. Genetic Programming will realize a lot of advanced function between KLOC and energy. Particle crowd improvement alone provides virtually same results as basic models. Project data and the traditional algorithmic model into one general framework that can have a wide range of applicability in software cost estimation, software quality estimation and risk analysis.

4. REFERENCES

[1] N. E. Fenton, S. L. Pfleeger, Software Metrics, A PWS Publishing Company, Thomso Publishing, Boston, 1997.

[2] A. R. Gray, S. G. MacDonell, “Applications of Fuzzy Logic to Software Metric Models for Development Effort Estimation”. Fuzzy Information Processing Society 1997 NAFIPS’ 97, Annual Meeting of the North American, 21 – 24, September 1997, pp. 394 – 399, 1997.

[3] B. W. Boehm, Software Engineering Economics, Englewood Cliffs, NJ,Prentice Hall, 1981.

[4] L. H. Putnam, “A general empirical solution to the macrosoftware sizing and estimating problem”. IEEE Transactions on Software Engineering, SE-4(4) pp 345-361, 1987.

[5] S. G. MacDonell and A. R. Gray, “A comparison of techniques for software development effort prediction,” International Conference on Neural Information Processing and Intelligent Control Systems, New Zealand, pp. 869-872, 1997.

[6] A. C. Hodgkinson and P. W. Garratt, “A neurofuzzy cost estimator,”Proceedings of Third International Conference on Software Engineering and Applications, pp. 401-406, 1999.

[7] A. Idri, T. M. Khoshgoftaar, A. Abran. “Can neural networks be easily interpreted in software cost estimation”, IEEE Trans. Software Engineering,Vol. 2, pp. 1162 – 1167,2002

[8] Z. Fei, X. Liu f-COCOMO: fuzzy constructive cost model in software engineering. In: IEEE international conference on fuzzy systems, pp 331–337, 2001.

[9] M. W. Nisar, J. W. Yong and M. Elahi, "Software development effort estimation using fuzzy logic – A survey," IEEE International Conference on Fuzzy Systems and Knowledge Discovery, vol. 1, pp. 421-427, 2008.

[10] P. Musilek, W. Pedrycz, G. Succi, and M. Reformat," Software cost estimation with fuzzy models," Applied Computing Review, vol. 2, pp. 24-29, 2000.

[11] Finnie, G. R., G.E. Wittig and J-M. Desharnais, "A Comparison of Software Effort Estimation Techniques Using Function Points with Neural Networks, Case-Based Reasoning and Regression Models", Journal of Systems and Software, Vol. 39, pp. 281-289, 1997.

[12] K.Ramesh and Karunanidhi, " Literature Survey On Algorithmic And Non- Algorithmic Models For Software Development Effort Estimation", International Journal Of Engineering And Computer Science ISSN:2319-7242 Volume 2 Issue Page No. 623-632, 3 March 2013.

[13] C.L. Martin, J.L. Pasquier, M.C. Yanez, and T.A. Gutierrez, "Software Development Effort Estimation Using Fuzzy Logic: A Case Study", IEEE Proceedings of the Sixth Mexican International Conference on Computer Science (ENC'05), pp. 113-120, 2005.

[14] X. Huang, D. Ho, L. Capretz and J. Ren "Novel Neuro-Fuzzy Models for Software Cost Estimation", Proc.of the Third International Conference on Quality Software, IEEE Computer Society Press, Dallas, TX, USA, 2003.

[15] N. Karunanitthi, D. Whitley, and Y. K. Malaiya, (1992), "Using Neural Networks in Reliability Prediction", IEEE Software, Vol. 9, no.4, pp. 53-59.

[16] Finnie, G. R., G.E. Wittig and J-M. Desharnais, (1997), "A Comparison of Software Effort Estimation Techniques Using Function Points with Neural Networks, Case- Based Reasoning and Regression Models", Journal of Systems and Software, Vol. 39, pp. 281-289.

[17] A.P. Engelbrecht, (2006), Fundamentals of Computational Swarm Intelligence, JohnWiley & Sons, New Jersy.

[18] Urkola Leire , Dolado J. Javier , Fernandez Luis and Otero M. Carmen , (2002), "Software Effort Estimation: the Elusive Goal in Project.

[19] "Fuzzy systems and neural networks" in software engineering project management, Journal of Applied Intelligence, no. 4, pp. 31-52.