Asian Journal of Information Technology 15 (7): 1207-1212, 2016
ISSN: 1682-3915

A Comparative Analysis of Algorithmic and Soft Computing Techniques in Estimating Software Effort

[1]N. Shivakumar, [2]N. Balaji and [1]K. Ansnthskumsr
[1]Departement of Computer Science and Engineering, Thiagarajar College of Engineering,
[1, 2]Departement of Information Technology, K.L.N College of Engineering, Madurai, India

Abstract: Effort estimation of project development was a very challenging problem. It is simple and valid in using algorithmic approaches in effort estimation but they are not so reliable in many cases. Based on information collected from history of projects, the process are continuously improved in the project management to eliminate difficulties in estimating effort of software. However, several methods are used to estimate the software development effort accurately. These non-algorithmic models ware build up on limited number of resources and their performance have not been well investigated. This study compares algorithmic models like Doty, Bailey, Halstead, COCOMO Iand COCOMO II with non-algorithmic approaches like particleswarm optimization k-means clustering algorithms, triangular fuzzy approach and adaptive neuro fuzzy. The results suggests that the non-algorithmic approaches works well with more accurate and reliable.

Key words: Effort estimation, algorithmic models, COCOMO, neuro fuzzy, PSOK-means clustering, triangular fuzzy

INTRODUCTION

Software effort estimation is the important activities in software project management. By increasing the total number of software projects and need of user society to get high quality software, some models based on the regression techniques and linear equations were presented in 1965 as the software effort techniques (Boehm, 1981). Afterward in 1973, the IBM researchers introduced the first automated tool, Interactive productivity and quality. Analogy Based Estimation (ABE) was explained as a comparative method in 1997. This Analogy method calculates the software project metrics by comparing the target project features with previously finished projects. Simplicity and capability of analogy estimation technique in prediction increase its usage so that Analogy based estimation was comparable with most mathematic models.

Poor project planning at the earlier stages of software development drags the companies to a collective loss and this it leads to poor software economy. COCOMO and COCOMO II are the most popular non-proprietary effort estimation techniques in literature as well as in industry (Albrecht and Gaffney, 1983). Even though, it's easy to implement it with few exceptions, these modelsare supposed to be applicable in estimating the effort, they are not always reliable. The empirical studies in estimation suggests that the COCOMO or other algorithmic models, are not a better estimation criteria. Since, all these algorithmic models are based on research and previous data history and use inputs namely source lines of code (kloc/sloc), number of functions to perform and other cost drivers.

In recent years, researchers found that former estimation techniques cannot response to dynamic behaviour of software projects. Particularly mathematical equation is unable to get accurate estimation for today's software development project. Soft computing Techniques have been used to get the software development effort because those methods can performaccurately in unstable and variable environments (Matson *et al*., 1994). Fuzzy concepts, Artificial Neural Network (ANN) and clustering algorithms are being referred as the important idea behind researches in term of software effort estimation because it can interpret the complicated relations among software project characteristics. Artificial Neural networks can be useful to interpret the relation between software project characteristics and final estimated effort. Different types of artificial neural network withvarious structures have been explained to estimate the effort value of software project development.

Literature review: Boehm (1981) explains the usage of algorithmic methods to compute software effort estimation. The study on 169 projects dataset compared

Corresponding Author: N. Shivakumar, Departement of Computer Science and Engineering, Thiagarajar College of Engineering, Madurai, India

the results of several models (the Wolverton, Doty, PRICE 5 and SLIM models) and computed the software cost estimation. This proved to be one of the pivotal research work to be done in the field of software project management.

Iman (Matson *et al*., 1994) compares the software effort estimation computed by the conventional methods like function points, regression models and COCOMO with the model designed using fuzzy logic. The parameter Mean Magnitude of Relative Error (MMRE) is used to compute the accuracy of the considered methods.

Saini and Kaur (2014) proposes a combination of PSO and K-means clustering to achieve the better clustering results (Han and Kamber, 2001). The main reason behind the method is to find the similarity in the data items and grouping them based on similarity. The result of PSO is given as an input for k-means to obtain better result.

Shiyna and Chopra (2013) compares the frameworks designed by using fuzzy logic and Neural Networks based on the accuracy of effort estimation. COCOMO NASA dataset had been used as the input for both the frameworks. These frameworks are validated using the parameters Mean Magnitude of Relative Error (MMRE) and Prediction accuracy (Pred) (Khan *et al*., 2010). The results show that fuzzy logic based framework works better when compared to the neural network framework.

Thus, the literature survey proves that the evolution of various techniques in software effort estimation has enabled the researchers to choose between the algorithmic and soft computing models to compute the software effort based on the dataset nature and the resources available.

MATERIALS AND METHODS

Effort estimation techniques: The effort estimation techniques used in software development are broadly classified as algorithmic and non-algorithmic methods. These algorithmic models are statistical models and use formulas for calculating effort. Some important algorithmic models are experimented in this study which uses the size in KLOC value from the data sets.

Algorithmic models

Halstead model: This model estimates the programming effort by using a formula which uses only kilo lines of code and constants:

$$\text{Effort} = 0.7 \times (\text{KLOC})^2 \tag{1}$$

Bailey-Basili model: This model developed by Bailey and Basili (1981), uses the kilo lines of source code and constants:

$$\text{Effort} = 5.5 \times (\text{KLOC})^{1.16} \tag{2}$$

Doty model: This model developed by Doty, uses the kilo lines of source code and constants:

$$\text{Effort} = 5.288 \times (\text{KLOC})^{1.047} \tag{3}$$

COCOMO: It was the first model suggested by Barry Boehm. This model has been widely accepted in practice. In the COCOMO model, the code-size S is given in Kilo Lines of Code (KLOC) and Effort is in person-month. The basic COCOMO model is simple and easy to implement. It does not contain any cost factors and only a rough estimate.

$$\text{Effort} = a \times (\text{KLOC})^b \tag{4}$$

Where a, b are complexity factors. This formula uses three sets of a andb depending on thecomplexity of the project.

COCOMO II: It is just a formula in three variants, post architecture model, application composition model and early design model (Bailey and Basili, 1981). This COCOMO II model is derives from the COCOMO and is defined as:

$$\text{Effort} = 2.9 \times (\text{KLOC})^{1.10} \tag{5}$$

Non algorithmic approaches

Fuzzylogic techniques

Triangular fuzzy techniques: Fuzzy logics are widely used on the observations and the fuzzy numbers are derived and from this derived fuzzy numbers the defuzzification is applied to get the rate factor which is the effort in person hours (Lopez-Martin, 2011). Triangular fuzzy function are used to represent the data terms in fuzzy complexity matrixes. Every fuzzy set is represented by membership function, which corresponds to a point in the fuzzy set in the interval (Bailey and Basili, 1981), called degree or grade of membership.

There triangular, trapezoidal and parabolic membership function. A Triangular Fuzzy Number (TFN) is represented in form of a triplet (α, m, β) where α and β are right and left boundary valuesand m is the modal value. It is defined as TFN (α, m, β), $\alpha \le m$, $\beta \ge m$ and $\beta \ge m$. The membership function ($\mu(x)$):

$$\mu(x) = \begin{cases} 0, & x \le \\ x - \alpha / m - \alpha, & \alpha \le x \le m \\ \beta - x / \beta - m, & m \le x \\ 0, & x \ge \beta \end{cases} \tag{6}$$

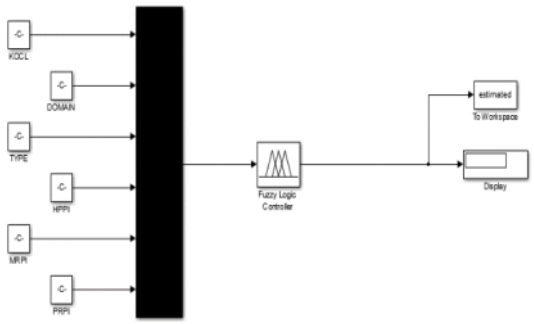

Fig. 1: Fuzzy rules generated

Defuzzification transforms from fuzzyvalue to crisp value. The fuzzy results obtained has to be converted to the two crisp quantities for further processing. This is done by the defuzzification process. The defuzzification process creates asingle-valued quantity called as a set, or in other words converting it to the form which fuzzy quantity is available. This process can also called as rounding off method. It reduces the group of values in membership function to a single sealer quantity. Fuzzy logic, fuzzy sets and corresponding membership degrees are obtained. It applies the rules that transform a number of variables into a fuzzy resultwhich is described in form of memberships in fuzzy sets. The interference method uses a base rules to correlate the inputs to a fuzzy output which can be a crisp value or left as a fuzzy value (Reddy and Raju, 2009; Attarzadeh and Ow, 2010).

$$D(y)=\begin{cases}\mu(x)\times w1 & 0<c(x)<1\\ \mu(x)\times w1+(1-\mu(x))\times w2 & 1<c(x)\le 2\\ \mu(x)\times w2+(1-\mu(x))\times w1 & 2<c(x)\le 3.5\\ \mu(x)\times w3+(1-\mu(x))\times w3 & 3.5<c(x)\le 5\\ \mu(x)\times w4+(1-\mu(x))\times w2 & 5<c(x)\le 6.5\\ \mu(x)\times w5+(1-\mu(x))\times w4 & 6.5<c(x)\le 8\end{cases} \quad (7)$$

Adaptive neuro fuzzy interface system: ANFIS is a hybrid supervised technique which uses both Artificial Neural network and fuzzy inference system (Mewada *et al*., 2013). An ANFIS has the similar layers as in artificial neural network. Every layer in ANFIS is a fuzzy inference system which consists of fuzzification rule generation, training, execution and defuzzification. This adaptive neuro fuzzy interface system is implemented using the Matlab tool. The input arguments for ANFIS is a training data represented in form of a matrix with N inputs columns (Du *et al*., 2015; Dhingra and Mann, 2013). The first N columns contains the input parameters and the final column contains the actual effort (Fig. 1).

PSO and K-means clustering hybrid method

Particle Swarm Optimization (PSO): It was proposed by Kennedy and Eberhart (1995) and Pena *et al*. (1999). It is simple, fast and easy to understand. It is a population-based method similar to genetic algorithm. It represents the state of the algorithm by a population, which gets modified iteratively (Kennedy and Eberhart, 1995; Dong and Qi, 2009). This will continue until a termination criterion is satisfied (Komarasamy and Wahi, 2011). In PSO, the potential solution called as particle. This particle searches the whole space by referring to the p best, the best previous position and gbest, the swarm best position. The fitness function is arrived using the following Eq. 8:

$$f_n=\sum_{j=1}^{k}\sum_{i=1}^{n}\left\|x_i^j-c_j\right\|^2 \quad (8)$$

Initially, position and velocity of the particles are expressed as follows:

$$X_i=LB+r\ and(UB-LB) \quad (9)$$

$$V_i=\frac{LB+r\ and(UB-LB)}{\Delta t} \quad (10)$$

The velocity and positions values are updated during each iteration. Velocity update Eq. 11 is given below:

$$V_{i+1}=w\,V_i+c_1\,r\ and\frac{pbest_i-X_i}{\Delta t}+c_2\ r\ and\frac{pbest_i-X_i}{\Delta t} \quad (11)$$

Where:

X_i	=	Current position
V_{i+1}	=	The velocity of next iteration
V_i	=	Current velocity
Δt	=	The time interval
Rand	=	A uniformly distributed random variable and the values between 0 and 1
$pbest_i$	=	The location of the particle which exhibits the best fitness
$gbest_i$	=	The location of the particle that experiences a global best fitness value
c_1 and c_2	=	Two positive constants for acceleration responsible for degree of information
w	=	Inertia weight and will linearly decrease during the iterations

Updating position is the last step in each iteration, it is updated using velocity vector. Position update equation is given below:

$$X_{i+1} = X_i + V_{i+1}\Delta t \quad (12)$$

Where

X_{i+1} = The next position
X_i = The current position
V_{i+1} = The next velocity
Δt = The time interval

Update the best fitness values at each generation, based on Eq. 13:

$$P_i(t+1) = \begin{cases} P_i(t) f(X_i(t+1)) \le f(X_i(t)) \\ X_i(t+1) f(X_i(t+1)) > f(X_i(t)) \end{cases} \quad (13)$$

Where:

f = Denotes the fitness function eqn (1)
$P_i(t)$ = Stands for best fitness value and the coordination where the values are computed
$X_i(t)$ = The current position
t = Denotes the generation step

Update the velocity, position and fitness computations are repeated until the termination criteria is met.

K-means clustering: This is a simple unsupervised learning algorithms that solve the well-known clustering problem is K-means clustering. Through a certain number of clusters, it is simple and easy to classify a given data set. Initially, select the k centroids then calculate the distance between each cluster centre and each object. Repeat the steps until no more changes are done. The objective function of k-means clustering is given as:

$$f_n = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^j - c_j \right\|^2 \quad (14)$$

Where

n = The number of data points
K = The number of clusters
$\|x_i^j - c_j\|$ = The distance between each cluster centre and each object

The steps of k-means clustering is shown below:

- Place K points into the space which denotes initial group centroids
- Each object is assigned to the group which has the closest centroid
- Then recalculate the positions of the K centroids when all the objects are assigned
- Repeat steps 2 and 3 until the centroids no longer move

Experimental setup

Data set: We have considered 91 instances of historical project data from the investigated projects, case studies and experiments. These data sets consists of 7 variables size in KLOC, domain, type of work, Human Perception and Performance Index (HPPI), Machine Requirement and Performance Index (MRPI), Process Requirement and Performance Index (PRPI) and Effort. Adaptive Neuro-Fuzzy model (Fig. 1) for software development effort estimation is perfect in the learning and good interpretability. Artificial neural networks are made up of processing units in a parallel manner called as neurons these neurons are inter linked by connections. The input for this model is six grouped attributes. Each attribute represents one factor which leads to the development effort. Variables in the data sets is described there.

Variables:

- KLOC
- Domain of work
- Complexity
- Human Perception and Performance Index (HPPI)
- Machine Requirement and Performance Index (MRPI)
- Process Requirement and Performance Index (PRPI)
- Actual effort

Evaluation criteria: The commonly used accuracy measures Magnitude of Relative Error (MRE) for every data sets between estimated and actual effort. And finally Mean Magnitude of Relative Error (MMRE), Mean Absolute Error (MAE), Mean Balanced Relative Error (MBRE) and Mean Inverted Balanced Relative Error (MIBRE) are shown in the below Eq.

$$AE_i = |y_i - x_i| \quad (15)$$

$$MRE_i = \frac{|y_i - x_i|}{y_i} \quad (16)$$

$$MAE = \frac{\sum_i^n AE_i}{n} \quad (17)$$

$$MRE_i = \frac{|y_i - x_i|}{y_i} \quad (18)$$

$$MIBRE = \frac{1}{n} \sum_1^n \frac{AE_i}{\max(y_i, x_i)} \quad (19)$$

Table 1s: Accuracy of all models in all experiments

Models	MAE	MBRE	MIBRE
Halstead model	295	32	20
Bailey-Basili model	290	26	16
Doty model	212	17.5	13.2
COCOMO I	244	35.4	20.6
COCOMO II	282	29	21
Triangular fuzzy	254	14.3	10.2
PSO and k-means clustering hybrid method	220	12.7	9.2
ANFIS	218	12.3	8.9

Where

x_i and y_i = The actual and estimated effort of a particular data set and n is the total no of data sets

MAE = The mean absolute error of the prediction model

Experiments: We have designed and performed 4 experiments. Experiment 1 with all five algorithmic models which involves the KLOC and in some cases the complexity variable in the data sets are used. The estimates effort is calculated for all the data sets. Experiment 2 is for triangular fuzzy model. Experiment 3 is for K-means clustering and PSO hybrid model.

And experiment 4 is ANFIS model. All the experiments are investigated on the performance reliability and validity of the algorithmic and non-algorithmic models. All the 90 data sets are used in the experiments (Table 1). This ensures that all the experiments are demonstrated on the same data sets

RESULTS AND DISCUSSION

This study reports the obtained results from all the four conducted experiments. The first experiment is conducted to compare all five algorithmic models which uses the KLOC attribute in the data sets and other three experiments are for the non-algorithmic models.

The performance of the models improved in experiment 3 and 4 with lower MAE, MBRE and MIBRE. We can see better performance in experiment 4 with the lowest value. A notable accuracy is achieved in Anfis model and followed by PSO and K-means clustering (Fig. 2 and 3). This study presents a comparative analysis between algorithmic and non-algorithmic models in the software effort estimation techniques.

These experiment shows the importance of the three variables Human Perception and Performance Index (HPPI), Machine Requirement and Performance Index (MRPI) and Process Requirement and Performance Index (PRPI). When the size factor alone is used, it will definitely leads to uncertainty in estimation.

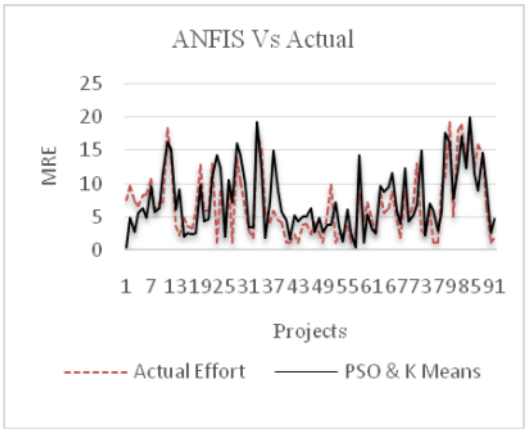

Fig. 2: PSO and K-means vs actual effort

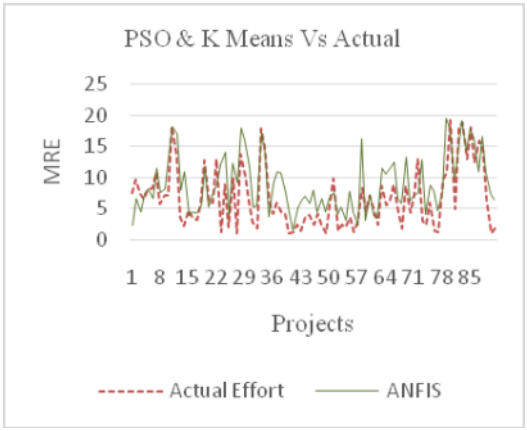

Fig. 3: ANFIS vs actual effort

CONCLUSION

Early effort estimation is a vital task in earlier stages of project development. A good number of collected data sets conforms that non-algorithmic soft computing techniques are the best in effort estimation. Success of any prediction model depends up on validity and reliability in various accuracy metrics. Specifically ANFIS model and K-Means PSO hybrid model are the best models in effort estimations. They came close to the actual effort in many cases. Future work is planned to study with other soft computing models and with large data sets.

REFERENCES

Albrecht, A.J. and J.E. Gaffney, 1983. Software function, source lines of code and development effort prediction: A software science validation. IEEE Trans. Software Eng., SE-9: 639-648.

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.

Bailey, J.W. and V.R. Basili, 1981. A meta-model for software development resource expenditures. Proceedings of the IEEE 5th International Conference on Software Engineering, March 9, 1981, IEEE, New Jersey, USA., ISBN: 0-89791-146-6, pp: 107-116.

Boehm, B.W., 1981. Software Engineering Economics. Prentice-Hall, Englewood Cliffs, New Jersey, ISBN: 9780138221225, Pages: 767.

Dhingra, S. and P.S. Mann, 2013. An adaptive neuro fuzzy approach for software development time estimation. Intl. J. Adv. Res. Comput. Sci. Software Eng., 3: 586-590.

Dong, J. and M. Qi, 2009. A new algorithm for clustering based on particle swarm optimization and k-means. Proceeding of the IEEE International Conference on Artificial Intelligence and Computational Intelligence, November 7-8, 2009, IEEE, Shanghai, China, ISBN: 978-0-7695-3816-7, pp: 264-268.

Du, W.L., D. Ho and L.F. Capretz, 2015. Improving software effort estimation using neuro-fuzzy model with SEER-SEM. Comput. Sci. Software Eng., 10: 51-63.

Kennedy, J. and R.C. Eberhart, 1995. Particle swarm optimization. Proceedings of the IEEE International Conference on Neural Networks, Volume 4, November 27-December 1, 1995, Perth, WA., USA., pp: 1942-1948.

Khan, I.R., A.H.A. Afshar and H. Anwar, 2010. Efficient software cost estimation using Neuro-fuzzy technique. Proceeding of the ISCET International Symposium on Computer Eng & Technology, March 19-20, 2010, ISCET, Punjab, India, ISBN: 978-81-910304-0-2, pp: 32-37.

Komarasamy, G. and A. Wahi, 2011. A improving the cluster performance by combining PSO and K-means algorithm. ICTACT. J. Soft Comput., 1: 206-208.

Lopez-Martin, C., 2011. A fuzzy logic model for predicting the development effort of short scale programs based upon two independent variables. Applied Soft Comput., 11: 724-732.

Matson, J.E., B.E. Barrett and J.M. Mellichamp, 1994. Software development cost estimation using function points. IEEE Trans. Software Eng., 20: 275-287.

Mewada, K.M., A. Sinhal and B. Verma, 2013. Adaptive Neuro-Fuzzy Inference System (ANFIS) based software evaluation. IJCSI. Intl. J. Comput. Sci., 10: 244-250.

Pena, J.M., J.A. Lozano and P. Larranaga, 1999. An empirical comparison of four initialization methods for the K-Means algorithm. Pattern Recognit. Lett., 20: 1027-1040.

Reddy, C.S. and K. Raju, 2009. Improving the accuracy of effort estimation through fuzzy set representation of size. J. Comput. Sci., 5: 451-455.

Saini, G. and H. Kaur, 2014. A novel approach towards k-mean clustering algorithm with PSO. Intl. J. Comput. Sci. Inf. Technol., 5: 5978-5986.

Shiyna, K. and V. Chopra, 2013. Neural network and fuzzy logic based framework for software development effort estimation. Intl. J. Adv. Res. Comput. Sci. Software Eng., 3: 1-5.