Optimized Satin Bowerbird for Software Project Effort Estimation

Resmi V[1], Anitha K L[2], Narasimha Murthy G K[3]

[1,2]Assistant Professor, Department of Computer Science, Mar Ivanios College, Trivandrum, INDIA

[3]Assistant Professor, Acharya Institute of Graduate Studies, Bangalore

[1,]***vreshmi15@gmail.com, [2]anithakl07@gmail.com, [3]Narasimha2629@acharya.ac.in***

AbstractGeneration of successful software project is the fundamental problem for many organizations. Therefore, the success of software project effort estimation is considered to be one of the foremost concerns of software engineering. Due to complicated nature of software projects, effort estimation has become a demanding issue that has to be addressed at the initial project stage. Here, Ensemble Classifier with Optimized Satin Bowerbird (ECOPB) for software project effort estimation is proposed. The ECOPB uses coherent neural network based clusters which are based on the deterministic model by updating multiple centroids. Subsequently, an ensemble of Coherent Bayes with Selective Gradient Boosting Classifier is trained on the clustered result sets to build the classifier model based on projects' attributes. Finally, Optimized Satin Bowerbird Effort Estimation is employed to optimize the classifiers. The proposed ECOPB is implemented using Cocomo81,CocomoNasa93, Cocomonasa60,desharnais,Albrechtfpa, ChinaFPA and cocomo_sdr datasets extracted from PROMISE software engineering repository. The obtained results are compared to state-of-the-art software project effort estimation approaches.

KeywordsSoftware project effort estimation; software engineering; Coherent Neural Networks, Ensemble Classifier; Satin Bowerbird optimization.

1. Introduction

Software effort estimation is one of the most important activities in software project management. In case of software development, the effort estimation is a procedure to findthe effort required to build or tomaintain software from incomplete, noisy and uncertain inputs. Effort estimates can be used as an input to formulate project plan, iteration plan, investment analyses, budgets, pricing processes and so on. Various models are being designed for effort estimation for software projects.

Software cost estimation is a challenging one wherein it must have knowledge about a number of key attributes which concern outcome of any software projects.One of the difficult tasks is the need of the required information where in it is often impossible to

get in desirable quantities. Due to this cause, the Software effort estimation has become a tough process in the IT industries. There has been a huge research led in software project effort estimation for the past two decades by machine learning techniques. But, the accuracy of the software project effort estimation was not enough and even the total time required to predict software project effort was more. To address this problem, an ensemble classifier technique with Satin Bowerbird Optimization algorithm is used with an objective of optimizing the rubrics to extract the best effort estimation with the support of the software projects.

Effort estimation of the software projects,an Ensemble Classifier with Optimized Satin Bowerbird (ECOPB) for software project effort estimation is proposed and the contribution of the work is mentioned as follows,

- Coherent neural clustering approach is designed which is based on deterministic model aimed at clustering the software projects in ECOPB. Through the aid of measuring the distance between feature vectors, coherent clusters are formed through updating the centroid values. By this the accuracy of the clustering process is improved.
- By the use of similarity function, an ensemble of the coherent Bayes and the selective gradient boosting classifier is developed for obtaining selective projects in the software project effort estimation. The classification between quantitative and qualitative attributes is completed with minimum losses by using the selective gradient classifier.
- In order to produce the optimized results while classifying software projects, Optimized Satin Bowerbird Effort Estimation algorithm is designed for each selective project. In case of Satin Bowerbird Effort Estimation algorithm, the probabilities and fitness are calculated for identifying a similar group of projects to estimate the software project effort with greater accuracy.

The foremost advantage of the proposedECOPB framework includes increase in the clustering accuracy, the prediction accuracy and minimizing the error. The rest of the paper is organized as follows: Section 2 provides a survey of software project effort estimation methods suggested by different researchers in varied fields. Section 3 discusses the proposed algorithm and software project effort estimation framework. Section 4 provides the experimental setup and discussions and conclusions of this proposed work discussed in Section 5.

2. Related Work

PrzemyslawPospieszny et al. [1] designed an ensemble of three machine learning algorithms, Support Vector Machines, Neural Networks and Generalized Linear Models for estimating duration and effort with improved accuracy. For estimating effort and finding duration of the project, ensemble of three different machine learning algorithms is applied. With this, the resultant predictions were combined to minimize bias and variance error. With the application of machine learning techniques, limitations found in conventional algorithms were minimized resulting in the improvement of project's success rate.

SoufianeEzghari and AzeddineZahi designed a Consistent Fuzzy Analogy-based Software Effort Estimation model (C-FASEE)[2] using fuzzy logic and reasoning theory with the objective of addressing imprecision and uncertainty through reasoning. Besides, the C-FASEE used possibility distribution to measure the rate of uncertainty that in turn helped the software managers for assessing risk.

Estimating the effort of software project is considered to be crucial in the early stages of the software development life cycle. Many researchers have designed for software project effort predictions. Ali Bou Nassif et. [3] presents a comparative study of neural network models for software development effort estimation. Eugenio Capra et. [4] developed an empirical analysis on maintenance effort. The construction cost is considered to be one of the vital elements in construction project.Rshma Chawla et. [5] designed a review of different techniques designed for software project effort evaluation..

Mandeep Kaur [7] introduced a nature inspired optimization techniques to carry out software effort estimation process. Qiuying Li, and Hoang Pham [8] designed a software reliability model using Non Homogeneous Poisson Process to improve fitting and predictive performance. José Javier Dolado et. [9] used Minimum Interval of Equivalence for the search of the best model to estimate software development effort. Radek Silhavyet et. [10] designed an Algorithmic Optimization Method based on Use Case points and Multiple Least Square Regression to minimize the relative error score for software engineering projects.

For efficient software project effort management, two important criteria have to be selected in an optimal manner. They are dynamic staffing and rescheduling. Sumeet Kaur Sehra, et al. [11] introduced Soft Computing Techniques for reliable estimate of software development effort. R. Silhavy and Z. Prokopova [12] proposed an effort estimation method to find the level of effort for software development projects.

A. Corazza et. [13] presented an effort estimation method by applying meta-heuristic Tabu Search. With this objective,Filomena Ferrucci et. [14] developed parallel Genetic Algorithms using MapReduce to speed up the operation and therefore minimizing the execution time for challenging software engineering problem.

Software project effort estimation is considered to be the most critical activity for planning and monitoring software project development to produce the product on time with minimum budget and loss. Federica Sarro et., [15] employed a bi-objective effort estimation algorithm to improve the accuracy of effort estimation. H. Velarde et. [22] applied multiple classifier system and lines of code for development of software effort estimation that in turn provided more easily interpretable knowledge that assisted software engineer to improve skills related to decision making on project planning.

As several factors involved in software project effort estimation, optimization is considered to be one of the important factors during software engineering. Prasad Reddy.P.V.G.D and CH.V.M.K.Hari [17] designed a Particle Swarm Optimization to give good software effort estimation results. Kavita Choudhary [18] developed a genetic algorithm approach to identify the effort needed for the development of software project.

Thanh Tung Khuat and My Hanh Le [19]

designed a Directed Artificial Bee Colony Algorithm to optimize parameters needed for software effort prediction models. With this model both prediction quality and accuracy was said to be improved significantly. NazeehGhatasheh et al. [6] employed Firefly Algorithm in order to provide solution for software effort estimation model.

Martin J. Shepperd and Stephen G. MacDonell [20] developed a novel framework for evaluating competing prediction systems. William B. Langdon et al. [21] presents an improved measure of effectiveness predictors based on comparing them with random guessing. P. A. Whigham et al., [16] developed a relevant baseline model termed as Automatically Transformed Linear Model (ATLM) for evaluating against Software effort estimation methods. ATLM was simple and provided better performance while considering different project types. In addition, ATLM employed with mixed numeric and categorical data did not require parameter alteration.

F. Sarro, A and Petrozziello [23] proposed a new method based on Linear Programming to estimate the efficacy of Linear Programming for Effort Estimation (LP4EE) and ATLM. Web effort estimation models were planned to give efficient effort estimates while using cross-company datasets in [24]. Nearest Neighbour filtering technique was applied to increase the prediction accuracy of cross-company models when estimating single-company projects.

In[25], a standard analogy-based estimation method was designed for test project by means of collecting evidence from the effort values of same projects in some training set. Eight different single company datasets in the Tukutuku database were taken in [26] to empirically evaluate the accuracy of estimates obtained using cross company and within-company models.A bi-objective effortestimation algorithm was designed in [27] to integrate the confidence interval testing for providing significantly improved performance. Search-Based Software Project Management overview was presented in [28] to handle the software optimization difficulties in software project management.

In our work, the framework provides better performance in terms of the software project clustering accuracy, prediction accuracy and mean magnitude of the relative error.

3. Methodology

TheECOPB framework is designed to identify similar groups of software development projects which share similar characteristics based on the deterministic model. Deterministic model is a model which precisely determines the output value (https://en.wikipedia.org/wiki/Deterministic_system). The datasets used in this research are based on CocomoNasa93, Cocomonasa60,desharnais,ALBRECHTFPA, ChinaFPAand cocomo_sdrwhich are extracted from the PROMISE software engineering repository. The clusters obtained through the Coherent neural clustering approach are further investigated and validated through the use of Ensemble classifiers and optimization technique. Figure 1 given below shows the Ensemble Classifier with Optimized Satin Bowerbird (ECOPB) framework for software project effort estimation.

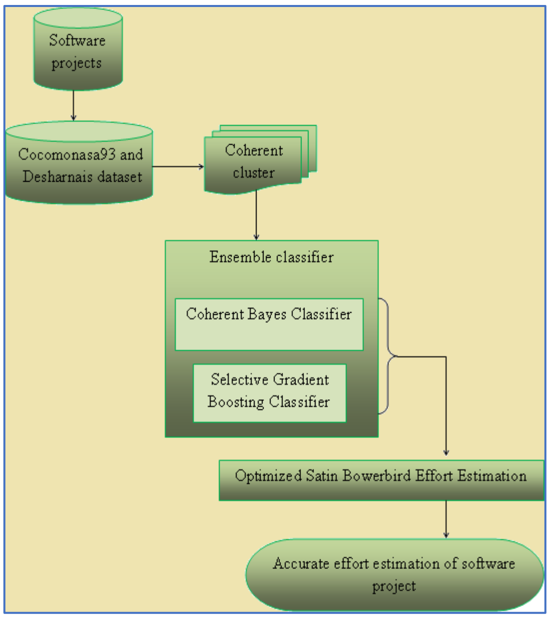
Software projects
Cocomonasa93 and Desharnais dataset
Coherent cluster
Ensemble classifier
Coherent Bayes Classifier
Selective Gradient Boosting Classifier
Optimized Satin Bowerbird Effort Estimation
Accurate effort estimation of software project

Figure 1: ECOPB framework

As illustrated in figure 1, the ECOPB framework consists of three parts. Initially, ECOPB framework clusters the software projects for effort estimation. Then with the clustered output, an ensemble of Coherent Bayes classifier with the Selective Gradient Boosting classifier is applied to perform selective classification. With the resultant selective classifiers, finally accurate prediction of software projects effort is made through the Satin Bowerbird Optimization algorithm. The detailed description is provided below.

3.5 Coherent Neural Clustering(CNC) approach

CNC algorithm is an alternative to K-means clustering which is used for cluster analysis and for obtaining coherent (i.e. rational) clusters. Figure 2 shows the schematic diagram of Neural Gas Coherent Clustering.

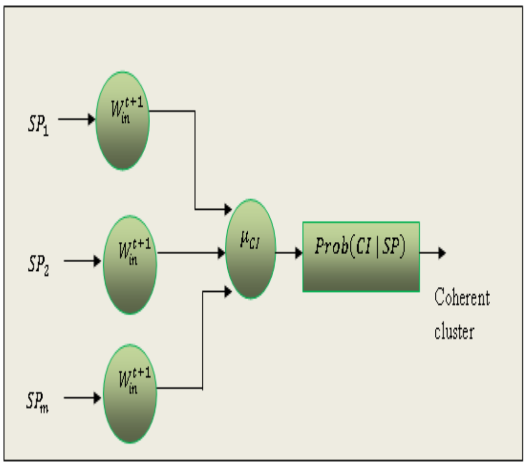

Figure 2 Schematic diagram of Neural Gas Coherent Clustering

As illustrated in figure 2, let us consider a probability distribution '$P(SP)$' that comprises software projects '$SP = \{SP_1, SP_2, \ldots, SP_m\}$' of data vectors '$SP$' with '$m$' corresponding to the number of software projects. Further at each time interval 't', a data vector 'SP' randomly selected from '$P(SP)$' is presented. Then, the distance 'Dis' from the feature vector '(p_i, p_j)' to the given data vector '(q_i, q_j)' with $i = 1, 2, \ldots, mandj = 1, 2, .., n$ is mathematically formulated as given below.

$$Dis = \sqrt{(p_i - q_i)^2 + (p_j - q_j)^2} \quad (1)$$

Let 'in_0' represent the index of the closest feature vector and 'in_1' represent the index of the second closest feature vector. Then each feature vector 'fv' is adjusted based on the following mathematical formulation.

$$W_{in}^{t+1} = W_{in}^{t} + \alpha . e^{-\lambda} . (SP - W_{in}^{t})$$

(2)

From the above equation (2), 'α' corresponds to the adaptation size, with 'λ' representing the neighbourhood range and 'W_{in}^{t+1}' representing the weight of the index of closest feature vector at time '$t+1$' and 'W_{in}^{t}' representing the weight of the index of closest feature vector at time 't'. With the objective of obtaining a coherent cluster, the adaptive size is initially set high to equalize the probabilities of software project being allocated to different clusters. It is then moderately minimized over several iterations to avoid a software project to go to its closest centroids with a high probability. To formulate this, let 'SP' represent software projects, 'CI' the cluster index and 'μ_{CI}' represent the centroid of cluster 'CI' with respect to 'N' clusters, then the coherent cluster is said to be formed by following the two steps given below.

$$\mu_{CI} = \frac{1}{N} Prob(CI \mid SP) SP \quad (3)$$

$$Prob(CI \mid SP) = e^{-r(SP,CI)} \quad (4)$$

From the above equation (3) and (4), centroid of cluster is calculated to group the software project by using assignment probability. Here '$r(SP,CI)$' represents the rank function that takes '$l-1$' value when the cluster centroid is closer to the software projects 'SP'. Then clusters are ranked based on the assignment probability '$Prob(CI \mid SP)$' where minimum distance between cluster index and 'SP' is grouped. Similarly, all the software projects are grouped and multiple centroids get updated to form coherent clusters.

3.6 The Ensemble of Coherent Bayes with Selective Gradient Boosting Classifier

Although many classical classification mechanisms have been used to improve software project effort estimation accuracy, the experiments are underway to maximize the flexibility of the model. As a matter of fact, the current classical classification mechanisms [2] are incompetent to reinforce a wide range of software projects and the operation is limited to a specific type of software projects. Also, every classification algorithm has its own disadvantages. Therefore to strengthen the weak classifier, ensemble classification technique is applied in this work and thereby improving the software project effort estimation accuracy.

The problems related to the classical classification methods support the idea of the ensemble classification of software projects. The ensemble classification is conducted on the basis of the undisclosed project attributes. The undisclosed project attributes refer to those attributes that are very predominant in the software project effort estimation process and which cannot be detected in the early stage as they possess a sizeable result on the estimation effort.

For example, two software projects may be positioned in the similar or same cluster due to similar file size and inputs. However, they may differ in the project attributes, namely, type of organization or the method of development and so on. Subsequently, the significance of attributes is not taken into consideration during the classification process. To address this issue, an Ensemble of Coherent Bayes with Selective Gradient Boosting Classifier is used in this work.

Let us consider the Coherent clusters '$CC = CC_1, CC_2, \dots, CC_k$' as input represented by a vector denoting certain key attributes (i.e., type of organization, a method of development) are examined to identify the corresponding clusters. To do this, initially instance probabilities are measured using Bayes Optimal and is mathematically formulated as given below.

$$Prob(Cl_m \mid CC_1, CC_2, \dots, CC_k), foreachofmpossibleclasses \quad (5)$$

From above equation (5), '$CC_1, CC_2, \dots, CC_k$', represent the coherent cluster with 'm', possible classes 'Cl'. In order to obtain a conditional probability to estimate the software project effort for a new project, two clusters including the type of organization and the method of development are selected from the available clusters. The other clusters are discarded and therefore the conditional probability is mathematically formulated as given below.

$$Prob(Cl_m \mid CC) = \frac{Prob(Cl_m) Prob(CC \mid Cl_m)}{Prob(CC)} \quad (6)$$

$$CP = Prob(Cl_m | CC_1, CC_2, \dots, CC_k) \quad (7)$$

In the above equation (6), '$Prob(Cl_m)$' corresponds to the prior class labels with likelihood function represented as '$Prob(CC \mid Cl_m)$' with an evidence of '$Prob(CC)$' respectively. However, an undisclosed project attribute obtained from equation (7) includes both qualitative and quantitative data. The quantitative data (i.e. attributes) cannot be used for classification.

Besides the qualitative data includes design architecture of the software project, programming or designing language used for the concerned software and standards applied should not be taken into consideration because they cannot be identified in the early stages of a software project. Hence, improper selection of qualitative and quantitative attributes causes significant loss. However, the project attributes that are fundamental to software project estimation have to be determined. Hence, Gradient Boosting Classifier (GBC) is applied that involves a supervised learning model for making an efficient prediction with a minimum loss (i.e. Mean Squared Error) for the generated conditional probability values. Hence, the loss or MSE is mathematically designed as given below.

$$LOSS = MSE = \left(CP_i - CP_i^p\right)^2 \quad (8)$$

In the above equation (8), 'CP_i' represents the 'ith' target value, with 'CP_i^p' representing the 'ith' prediction. The objective of the proposed work lies in minimizing the loss function (MSE). By using GBC and modifying the predictions according to the learning rate, the resultant values are identified with minimum MSE. This is achieved by modifying the above equation (8) as given below.

$$CP_i = CP_i + LR * \beta * \sum \frac{\left(CP_i - CP_i^p\right)^2}{\beta * CP_i^p} \quad (9)$$

$$CP_i^p = CP_i^p - LR * 2 * \sum\left(CP_i - CP_i^p\right) \quad (10)$$

From the above equation (10), 'LR' represents the learning rate and '$\sum\left(CP_i - CP_i^p\right)$' representing the sum of residuals. Therefore, the predictions are modified in such a way that the sum of the residuals is close to '0' or minimum and hence the predicted values are closer to the actual values.

Optimized Satin Bowerbird Effort Estimation

Finally, the optimization of software project effort estimation is done with selective classifiers as input by applying the Satin Bowerbird Optimization algorithm. This algorithm is inherited from the concept of bowerbird's mating process. Here male birds attract the females by constructing specialized stick structures called bowers. These bowers are decorated by male bird by flowers, feathers, berries, etc. Female bird visits many bowers to choose her partner for mating.

To perform this optimization, the population is initially initialized with a set of selective classifiers. Each classifier is defined as an 'n' dimensional vector of parameters that has to be optimized. Then, the probability includes the optimized parameters. This probability is mathematically formulated as given below.

$$Prob = \frac{Fit_i}{\sum_{n=1}^{CP_i^p(x)} Fit_n} \tag{11}$$

In the above equation (11), 'Fit_i' represents the fitness of the 'ith' solution and '$CP_i^p(x)$' is the number of selective classifiers used as input. In this equation, the value of 'Fit_i' is mathematically formulated as given below.

$$Fit_i = \begin{cases} \frac{1}{1+MSE}, & MSE \geq 0 \\ 1 + MSE, & MSE < 0 \end{cases} \tag{12}$$

From the above equation (11) and (12), final fitness is measured, if the value of the error is greater than zero and the value of error less than zero respectively. During the each iteration, new changes occurring at any selective classifiers are calculated by mathematically forming the given equation as below.

$$x_{ij}^{new} = x_{ij}^{old} + \left[\frac{x_{kj} + x_{best,j}}{2}\right] - x_j \tag{13}$$

From the above equation (13), the updated optimal values for software project effort to be estimated, 'x_{ij}^{new}' are obtained for the 'x_i' solution vector with 'x_{ij}' representing the members of the selective classifier vector. Besides, 'x_j' is determined as the target solution vector among all the solutions in the current iteration. Also, it specifies that the solutions possessing largest probability have more chance to be selected with 'x_j' denoting the best software project attribute whose fitness is highest in the current iteration. In this manner, accurate prediction of software projects effort is said to be performed with higher accuracy.

4. Experimental Evaluation

In order to evaluate the effectiveness of proposed Neural Coherent Clustered Ensemble Classifier with Optimized Satin Bowerbird (ECOPB) for software project effort estimation, proposed method is applied on Cocomo81,CocomoNasa93, Cocomonasa60,desharnais,Albrechtfpa, ChinaFPAand cocomo_sdrdatasets. These datasets are mined from the PROMISE software engineering repository. The two datasets are partitioned into two subsets: 70% of the historical projects as training set and 30% of the projects as test set. The training set is employed to get the model without the participation of the test set, which is employed to assess the accuracy of the estimation model. The ECOPB carried out experiments with a 10-fold cross-validation approach in order to train models with the training data and to make sure that our findings will generalize adequately with the independent test set and to solve over fitting problem.

5. Results

Theresulting analysis of proposed ECOPB for software project effort estimation is compared against two existing methods. They areEnsemble of three machine learning algorithms such as Support Vector Machines, Neural Networks, and Generalized Linear Models Ensemble (SVM, NN & GLM) [1],Consistent Fuzzy Analogy-based Software Estimation (C-FASEE) [2] and Linear Programming as a Baseline for Software Effort Estimation(LP4EE)[23].The detailed result analysis with the respective graphs and tables are given below.

5.5 *Scenario 1: Impact of clustering accuracy*

. Clustering accuracy for software project effort estimation is defined as the ratio of percentage of accurate coherent clusters generated. It is mathematically formulated as given below.

$$CA = \frac{CC_a}{SP} * 100 \tag{14}$$

From the above equation (14), clustering accuracy ‘CA’ is measured with respect to accurate coherent clusters generated ‘CC_a’ to the number of software projects ‘SP’ provided as input.

Table 1: Tabulation for clustering accuracy

SI.No	Datasets	ECOPB (%)	Ensemble of SVM, NN & GLM (%)	C-FASEE (%)	LP4EE (%)
1	Cocomo81	72.6	73.13	71.25	68.20
2	Cocomonasa60	73.78	70.83	72.00	69.50
3	CocomoNasa93	72.5	72	72.10	74.56
4	Cocomo_sdr	69.2	65.04	65.54	64.32
5	Desharnais	79.3	76.22	75.32	76.21
6	ALBRECHT FPA	81.22	80.12	74.23	78.33
7	Kemerer FPA	50.31	45.66	71.55	67.87
8	miyazaki1 cobol	98.1	95.21	90.12	90
9	MAXWELL	79	76.97	73.85	75.22
10	China FPA	98.99	96.64	95.03	92.43

Table 1 shows the performance analysis of clustering accuracy of different methods. The proposed ECOPB framework improves the clustering accuracy more than the existing Ensemble of SVM, NN & GLM [1], C-FASEE [2] and LP4EE [23]. The graphical representation of the proposed and existing methods is illustrated in figure 3.

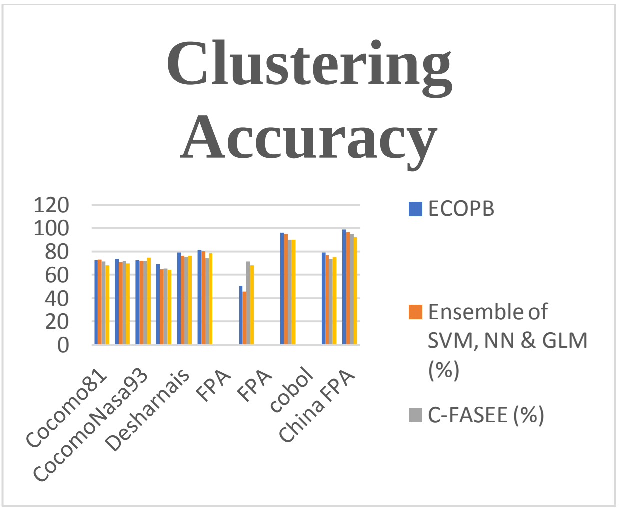

Figure 3 Performance measure of clustering accuracy

Figure 3 given above illustrates the performance measure of clustering accuracy. In the above figure, the values obtained from the proposed ECOPB framework is compared with the existing Ensemble of SVM, NN & GLM [1], C-FASEE [2] and LP4EE & ATLM [23] using Cocomonasa93 and Desharnais dataset. When applied with both the datasets, the clustering accuracy obtained using the ECOPB framework is found to be better than [1], [2] and [23].

5.6 *Scenario 2: Impact of Mean Magnitude of Relative Error*

The second experiment conducted to analyze the software project effort estimation is the Mean Magnitude of Relative Error (MMRE) or MRE. The Mean Magnitude of Relative Error (MMRE) is defined to calculate the difference between the actual and the predicted software project effort. A formula for calculating MMRE is given below.

$$MMRE = \frac{1}{n}\sum_{i=1}^{n}\frac{Actualeffort_i - Predictedeffort_i}{Actualeffort_i} \qquad (15)$$

From the equation (15), the Mean Magnitude Relative Error '$MMRE$' is evaluated based on the actual effort '$Actualeffort_i$' and predicted effort '$Predictedeffort_i$'.

Table 2: Tabulation for Mean Magnitude of Relative Error

SI. No	Datasets	ECOPB	Ensemble of SVM, NN & GLM	C-FASEE	LP4EE
1	Cocomo81	0.20	0.23	0.22	0.25
2	Cocomonasa60	0.15	0.20	0.21	0.26
3	CocomoNasa93	0.18	0.24	0.25	0.28
4	Cocomo_sdr	0.34	0.62	0.43	0.79
5	Desharnais	0.17	0.245	0.26	0.26
6	ALBRECHT FPA	0.29	0.35	0.48	0.55
7	Kemerer FPA	0.25	0.28	0.27	0.29
8	miyazaki1 cobol	0.26	0.25	0.25	0.26
9	MAXWELL	0.19	0.23	0.25	0.24
10	China FPA	0.15	0.22	0.23	0.25

Table 2 describes the performance analysis of mean magnitude of relative error which is done by comparing proposed ECOPB framework and existing Ensemble of SVM, NN & GLM [1], C-FASEE [2] and LP4EE & ATLM [23]. From the above table, it is clearly seen that the value of mean magnitude of relative error is reduced in ECOPB when compared to the state-of-the-art methods.

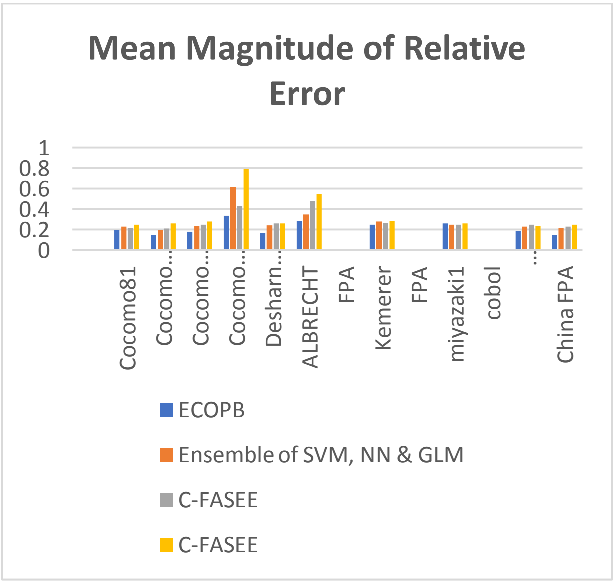

Figure 4 Performance measure of MMRE

Figure 4 shows the performance measure of MMRE using ECOPB framework and the existing Ensemble of SVM, NN & GLM, C-FASEE, LP4EE & ATLM respectively.

5.7 *Scenario 3: Impact of prediction*

The third experiment conducted to analyze the software project effort estimation is the prediction accuracy. The prediction accuracy is defined as the percentage ratio of the number of correct prediction (SP_{cp}) of software projects effort to the total number of software projects considered for experimentation. Here, SP_{cp} is the number of projects whose MMRE [20] is 0.25 or less. Then, the formula for prediction accuracy is given below.

$$Pred = \left[\frac{SP_{cp}}{SP} * 100\right] \; subject\ to\ MMRE \leq 0.25 \quad (16)$$

Table 3: Tabulation for Prediction

SI.No	Datasets	ECOPB (%)	Ensemble of SVM, NN & GLM (%)	C-FASEE (%)	LP4EE (%)
1	Cocomo81	85.65	81.31	80.45	80.22
2	Cocomonasa60	84.11	79.21	78.34	78.65
3	CocomoNasa93	83.31	78.88	75.98	70.62
4	Cocomo_sdr	82.61	76.32	77.21	77.22
5	Desharnais	81.51	75.72	73.83	72.34
6	ALBRECHT FPA	75.3	68.76	69.32	67.43
7	Kemerer FPA	72.43	70.23	68.93	67.88
8	miyazaki1cobol	81.45	78.64	79.54	75.73
9	MAXWELL	83.23	80.84	76.48	77.92
10	China FPA	89.21	87.32	88.65	87.93

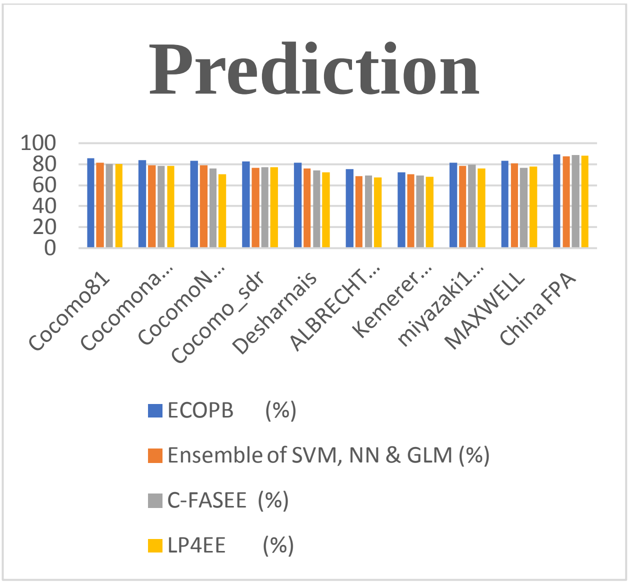

Figure 5 Performance measure of prediction

Table 3 and Figure 5 show the performance measure of prediction using ECOPB framework and the existing Ensemble of SVM, NN & GLM, C-FASEE, LP4EE respectively. It is clearly seen that the prediction of ECOPB is better when compared to the state-of-the-art methods.

6. Conclusion

Accurate estimation of software project effort with minimum time and effort for both the user and software organizations is one of the challenges in IT industry. In order to realize the above-said objective, it is paramount to evaluate and estimate the project's success level of the software organizations. The success of the software project depends upon various factors which also includes effective software project effort management. This work, Neural Coherent Clustered Ensemble Classifier with Optimized Satin Bowerbird (ECOPB) for software project effort estimation is designed with the objective of optimizing the selected classified software projects. Here, Neural Gas Coherent Clustering algorithm is applied for clustering them in a coherent manner using an index of the closest feature vector as the decisive parameter. In order to analyze the choice of the best classifier, an ensemble of Coherent Bayes with Selective Gradient Boosting Classifier is applied. Finally, the Satin Bowerbird Optimization algorithm is applied to the best classifier for optimizing the resultant output. Several experiments were conducted to measure the efficacy of ECOPB framework. An ensemble of two classifiers with coherent cluster ensures accurate prediction of software project effort. This framework, in addition, speeds up the process of software project effort estimations and henceforth improves the quality with minimum loss function.

7. References

[1] PrzemyslawPospieszny , Beata Czarnacka-Chrobot , Andrzej Kobylinski, "An effective approach for software project effort and duration estimation with machine learning algorithms", The Journal of Systems & Software, Elsevier, Nov 2017 (Ensemble of SVM, NM & GLM) Metrics – Mean Magnitude Relation Error (MMRE), Percentage Relative Error Deviation (PRED), Accuracy

[2] SoufianeEzghari, AzeddineZahi, "Uncertainty management in Software effort estimation using a consistent fuzzy analogy-based method", Applied Soft Computing, Elsevier, Mar 2018 (Consistent Fuzzy Analogy-based Software Estimation (C-FASEE)) Metrics – Accuracy, Mean Absolute Residual (MAR), Magnitude Relative Error (MRE)

[3] Ali Bou Nassif1, Mohammad Azzeh, Luiz Fernando Capretz, Danny Ho4, "Neural network models for software development effort estimation: a comparative study", Neural Computing and Applications, Springer, Nov 2015

[4] Eugenio Capra, Chiara Francalanci, and Francesco Merlo, "The Economics of Community Open Source Software Projects: An Empirical Analysis of Maintenance Effort", Hindawi Publishing CorporationAdvances in Software EngineeringVolume 2010

[5] Rshma Chawla, Deepak Ahlawat, Mukesh Kumar, "Software Development Effort Estimation Techniques: A Review", International Journal of Electronics Communication and Computer Engineering, Volume 5, Issue 5, Pages 2278–4209, 2015

[6] NazeehGhatasheh, Hossam Faris, Ibrahim Aljarah, Rizik M. H, Al-Sayyed, "Optimizing Software Effort Estimation Models Using Firefly Algorithm", Journal of Software Engineering and Applications, Volume 8, Issue 3, Pages 133-142, March 2015

[7] Mandeep Kaur, "Estimation of effort using nature inspired optimization techniques", International Journal of Academic Research and Development, Volume 3, Issue 1, Page No. 197-199, January 2018,

[8] Qiuying Li, Hoang Pham, "A testing-coverage software reliability model considering fault removal efficiency and efficiency and error generation", PLOS ONE, https://doi.org/10.1371/journal.pone.0181524 July 27, 2017

[9] José Javier Dolado, Daniel Rodriguez, Mark Harman, William B. Langdon, Federica Sarro, "Evaluation of estimation models using the Minimum Interval of Equivalence", Applied Soft Computing, Elsevier, May 2016

[10] Radek Silhavy, Petr Silhavy, ZdenkaProkopova, "Algorithmic Optimisation Method for Improving Use Case Points Estimation", PLOS ONE | DOI:10.1371/journal.pone.0141887 November 9, 2015

[11] Sumeet Kaur Sehra, Yadwinder Singh Brar, and Navdeep Kaur, "Soft Computing Techniques for Software Project Effort Estimation", International Journal of Advanced Computer and Mathematical Sciences, Volume 2, Issue 3, Pages 160-167, 2011

[12] R. Silhavy, Z. Prokopova, P. Silhavy, "Algorithmic optimization method for effort estimation", Programming and Computer Software, Springer, Volume 42, Issue 3, Pages 161–166, April 2016

[13] A. Corazza, S. Di Martino, F. Ferrucci, C. Gravino, F. Sarro, E. Mendes, "Using tabu search to configure support vector regression for effort estimation", Empirical Software Engineering, Springer, Dec 2011

[14] Filomena Ferrucci, Pasquale Salza, Federica Sarro, "Using Hadoop MapReduce for Parallel Genetic Algorithms: A Comparison of the Global, Grid and Island Models", Evolutionary Computation, May 2017

[15] Federica Sarro, Alessio Petrozzielloy and Mark Harman, "Multi-objective Software Effort Estimation", ACM, June 2016

[16] PETER A. WHIGHAM, CAITLIN A. OWEN, and STEPHEN G. MACDONELL, "A Baseline

Model for Software Effort Estimation", ACM Transactions on Software Engineering and Methodology, Vol. 24, No. 3, Article 20, May 2015.

[17] Prasad Reddy.P.V.G.D, CH.V.M.K.Hari, "Software Effort Estimation Using Particle Swarm Optimization with Inertia Weight", International Journal of Software Engineering (IJSE), Volume 2, Issue 4, Pages 87-96, 2011

[18]Kavita Choudhary, "GA Based Optimization of Software Development Effort Estimation", International Journal of Computer Science and Technology, Volume 1, Issue 1, Pages 38-40, September 2010

[19] Thanh Tung Khuat, My Hanh Le, "Optimizing Parameters of Software Effort Estimation Models using Directed Artificial Bee Colony Algorithm", Informatica, Volume 40, Pages 427–436, 2016

[20] Martin J. Shepperd, Stephen G. MacDonell, " Evaluating prediction systems in software project estimation" Information & Software Technology 54(8): 820-827 (2012)

[21] William B. Langdon, José Javier Dolado, Federica Sarro, Mark Harman, "Exact Mean Absolute Error of Baseline Predictor, MARP0" Information & Software Technology 73: 16-18(2016)

[22] H. Velarde, C. Santiesteban, A. García and J. Casillas," Software Development Effort Estimation based on Multiple Classifier system and Lines of Code", IEEE Latin America Transactions, Vol. 14, No. 8, Aug. 2016

[23] F. Sarro, A. Petrozziello, "Linear Programming as a Baseline for Software Effort Estimation", ACM Transactions on Software Engineering and Methodology (TOSEM), 2018, http://www0.cs.ucl.ac.uk/staff/F.Sarro/projects/LP4EE/.

[24] F. Ferrucci, E. Mendes, F. Sarro, "Web Effort Estimation: the Value of Cross-company Data Set Compared to Single-company Data Set". In Proceedings of the International Conference PROMISE 2012, pp.29-38.

[25] Kocaguneli, E., Gay, G., Menzies, T., Yang, Y., Keung, J. W, "When to use data from other projects for effort estimation" In proceedings of the International COnference ASE 2010, pp. 321-324.

[26] L. Minku, F. Sarro, E. Mendes, F. Ferrucci, "How to Make Best Use of Cross-Company Data for Web Effort Estimation?", in Proceedings of the 9th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM 2015).

[27]ZahraShahpar, Vahid Khatibi, Asma Tanavar and Rahil Sarikhani, "Improvement of effort estimation accuracy in software projects using a feature selection approach", Journal of Advances in Computer Engineering and Technology, Volume 2, Issue No.4, 2016, Pages 31-38.

[28] M.Harman, F. Ferrucci, F. Sarro, "Search-Based Software Project Management" in Software Project Management in a Changing World, Günther Ruhe and ClaesWholin (Editors), Springer, 2014, pp.373-399.