A Novel Hybrid Algorithm for Software Cost Estimation Based on Cuckoo Optimization and K-Nearest Neighbors Algorithms

Elnaz Eskandarian Miandoab
Department of Computer Engineering, Urmia Branch, Islamic Azad University, Urmia, Iran
Elnaz_es_m@yahoo.com

Farhad Soleimanian Gharehchopogh
Department of Computer Engineering, Urmia Branch, Islamic Azad University, Urmia, Iran
bonab.farhad@gmail.com

***Abstract*—The inherent uncertainty to factors such as technology and creativity in evolving software development is a major challenge for the management of software projects. To address these challenges the project manager, in addition to examining the project progress, may cope with problems such as increased operating costs, lack of resources, and lack of implementation of key activities to better plan the project. Software Cost Estimation (SCE) models do not fully cover new approaches. And this lack of coverage is causing problems in the consumer and producer ends. In order to avoid these problems, many methods have already been proposed. Model-based methods are the most familiar solving technique. But it should be noted that model-based methods use a single formula and constant values, and these methods are not responsive to the increasing developments in the field of software engineering. Accordingly, researchers have tried to solve the problem of SCE using machine learning algorithms, data mining algorithms, and artificial neural networks. In this paper, a hybrid algorithm that combines COA-Cuckoo optimization and K-Nearest Neighbors (KNN) algorithms is used. The so-called composition algorithm runs on six different data sets and is evaluated based on eight evaluation criteria. The results show an improved accuracy of estimated cost.**

Keywords- Software Cost Estimation; COCOMO model; COA-Cuckoo optimization algorithm; KNN algorithm

I. INTRODUCTION

Nowadays, software manufacturer organizations are expected to generate computer tools to manipulate and manage usually large amounts of data. The main problem usually observed is the mismatch between the project cost estimation and the Sactual cost. In order to reduce this difference, software developers need to collect certain initial information. These so-called information can be classified as follows [1]:

- Information relating to the capabilities and major and minor objectives of the software.
- Information relating to the organization that will operate the software.
- Information relating to the organization that will produce the software.

Prevalent use of software packages ready in certain parts of software projects reduces the cost of software development. At the same time, new software development processes are presented and manufacturers are trying to adapt to these processes. Some of these processes include risk-based and dynamic software processes, new programming languages, software applications for software development as well as commercial software, which improve the quality of software products, reduces production costs, reduces the risk and the produced software life cycle [2]. However, current models of estimating software development cost do not fully cover new approaches. And this lack of coverage causes problems in the consumer and producer ends.

II. PREVIOUS WORKS

With the introduction of machine learning algorithms in the field of software engineering, many researchers have used such algorithms for SCE. Some of these models can be briefly described as follows: In [3] fuzzy logic was implemented for SCE. They introduced cost estimation projects as one of the most important challenges and activities in software development. The researchers used 14 projects included in KEMERER projects collections. According to their results, the mean absolute error percentage of relative error and productivity rate improved compared to algorithmic methods. One of the software process criteria having a direct impact on estimated cost is the number of application lines and application thousand-line. In [4], the composition of ant colony optimization algorithm and chaos optimization algorithm was used for SCE. The researchers used Lorenz mapping to generate random data for the chaos optimization algorithm and for training they used the ant colony algorithm. This composition algorithm was evaluated on NASA 63 data set, and the results showed that the composition of ant colony optimization algorithm and chaos optimization algorithm has a better performance compared to the COCOMO model and also has a relative lower error than the COCOMO model [4].

The composition of multilayer perceptron networks and COCOMO II is used in [5]. To evaluate the proposed method, 63 COCOMO II software projects have been used. The main objective of this composition is better training of the multilayer

perceptron. In the composition model, effort factors and scale factors are taught by the intermediate layer and evaluated by the COCOMO II model. The Sigmoid Function was used as the activation function for the intermediate layer. Back-propagation is used for teaching. The data training and testing phases are 80% and 20%, respectively. The experiment results show that the standard error of relative error in the composition model is less than the COCOMO II model.

A new methodology for SCE based on particle swarm optimization algorithm and fuzzy logic was proposed in [6]. Estimation of the software cost depends on the estimation of the size of the project and its parameters. For measurements uncertainty, fuzzy logic is applied and particle swarm optimization is used for parameters. In the proposed method, the triangular membership function is used. Evaluation is implemented on NASA63 dataset. The results show that the proposed method has lower error compared to other models. A composition of firefly and genetic algorithms was been proposed in [7]. Evaluation is implemented on NASA93 dataset. In the proposed method, the genetic algorithm, via elitism operation, tries to obtain the best answer for effort factors and evaluate the fitness function, and provide the lowest error solution as the final answer. The results show that the average value of the relative error in COCOMO model is 58.80, and in genetic and firefly algorithms is 38.31 and 30.34, respectively and in the composition model is 22.53. Note that (25) PRED on genetic and firefly algorithm models is 77.41 and 80.64 respectively and in the composition model is 88.17. Comparisons show that the composition model compared with COCOMO model increased the efficiency of the estimated precision about 2.88%.

III. BASIC CONCEPTS

A. SCE and COCOMO II

Among the algorithmic models presented for SCE, COCOMO II is the most popular. This model provides a basis method to predict the number of people needed per month for software development in the industry. This model can also estimate the development time, the amount of effort required in each software development phase and the required cost [8]. The formulas used are listed in Table I.

TABLE I. COCOMO MODEL FORMULAS

Formula	Project Type
$PM=2.4\cdot(size)^{1.05}\cdot\prod_{i=1}^{15}EM_i$	Organic
$PM=3\cdot(size)^{1.12}\cdot\prod_{i=1}^{15}EM_i$	Semi Organic
$PM=3.6\cdot(size)^{1.2}\cdot\prod_{i=1}^{15}EM_i$	Embedded

The "size" parameter is the project's size in thousands of lines of KSLOC code, and the "EM" parameter is the effort coefficient

B. KNN Algorithm

The KNN algorithm was described for the first time in [9]. The selection of K in this method is very crucial. If the value of K is selected too small, the algorithm becomes sensitive to noise. If the value of K is selected too large, the records of other classes may also be included among the nearest neighbors. When K is selected as a large number, it leads to a classification error [10]. The algorithm searches the test space for k samples close to the unknown sample. This closeness is defined by the Euclidean distance. Then, the label having the majority among k neighbors is given to the unknown sample.

C. COA-Cuckoo optimization algorithm

COA-Cuckoo optimization meta-heuristic algorithm was first developed in 2009 [11]. In the first step of the Cuckoo algorithm, the first primary settlement sites are produced. In an N-dimensional optimization problem, settlement areas will be an N*1 array, showing the current position of cuckoo living areas. The appropriateness of the current living location is obtained by assessing the utility function at the location. Each cuckoo has a specified range for the number of laying eggs. After cuckoo chicks' growth, theses chicks migrate to other areas for laying eggs and the assumed migration location is also calculated [12].

D. Evaluation Criteria in SCE

Finally, any research work to determine the accuracy of the work done should be evaluated by a series of criteria. A number of the criteria used to assess SCE techniques is shown in Table II [12, 13].

TABLE II. EVALUATION CRITERIA

Criteria Formula	Criteria Name
$MMER=\frac{1}{N}\sum_{i=1}^{n}\frac{\lvert act_i-est_i\rvert}{est_i}\cdot 100$	Mean Magnitude Error Relative (MMER)
$MMER=\frac{1}{N}\sum_{i=1}^{n}\frac{\lvert act_i-est_i\rvert}{act_i}\cdot 100$	Mean Magnitude of Relative Error (MMRE)
$MDMRE=Median(\frac{1}{N}\sum_{i=1}^{n}\frac{\lvert act_i-est_i\rvert}{act_i}\cdot 100)$	Median Magnitude of Relative Error (MDMRE)
$PRED(n)=\frac{1}{N}\sum_{i=1}^{n}\begin{cases}1, \text{if MRE}\le m\\ 0, \text{otherwise}\end{cases}$	PRED(m)
$MSE=\frac{1}{N}\sum_{i=1}^{n}(act_i-est_i)^2$	Mean Squared Error (MSE)
$RMSE=\sqrt{\frac{1}{N}\sum_{i=1}^{n}(act_i-est_i)^2}$	Root Mean Squared Error (RMSE)
$MAE=\frac{1}{N}\sum_{i=1}^{n}\lvert act_i-est_i\rvert$	Mean of Absolute Errors (MAE)
$MAPE=\frac{1}{n}\sum_{i=1}^{n}\frac{\lvert act_i-est_i\rvert}{act_i}\cdot 100$	Mean Absolute Percentage Error (MAPE)

In these criteria the "N" variable denotes the total number of data in the data set; the "act" variable represents the real cost; the "est" variable represents the estimated cost; the "i" variable indicates any data index, which ranges from 1 to N.

IV. PROPOSED METHOD

In the first step of the algorithm, the data is normalized and vague and empty data are deleted. After this stage, a composition algorithm is implemented, and in the first stage of the implementation, the data are classified randomly into two subsets of training and test (20% and 80% respectively). After this initial classification, data in training and testing data set are classified into three different categories using KNN based on the features available in the data set, and the training data set is sent to COA-Cuckoo algorithm for training. In the first stage of training, the primary parameters of COA-Cuckoo optimization algorithm such as the number of initial cuckoo population, the number of algorithm iterations, the minimum and maximum eggs to be laid for each cuckoo, and random variables are initialized. After determining the number of cuckoos, the cuckoos are randomly initialized. Upon completion of this stage, cuckoos lay eggs in different nests. At this stage, if alien eggs are identified by an adult cuckoo, these eggs will be destroyed; otherwise, the eggs grow and then are converted to mature cuckoo chicks. After the growth of cuckoos, if the number of cuckoos is greater than the the initial population predetermined, cuckoos that live in the worst place will be lost. After that, the optimal amount of each living location is assessed and the best place is selected. Then, the adult cuckoos migrate to places close to this location. It should be noted that the condition of completion of the algorithm is the number of iterations or a minimum value that reaching this minimum value indicates obtaining the goal. In Table III and Figure 1, the proposed scheme is shown. As shown in Figure 1, after finishing each training phase, the results of this phase are stored, and then are applied as peer to peer test on test data set that is divided into three categories, and finally, the results obtained are displayed in the form of charts and text data.

TABLE III. THE OUTLINE OF THE PROPOSED APPROACH

Inputs: User selected data sets include: factors affecting the estimate and the actual cost of any software project
Outputs: Constant amounts of COMOCO model and classified data
Step 1: Read the existed data in the dataset
Step 2: Breakdown to training and testing data
Step 3: Classification of training and testing data based on KNN
Step 4: Invoking Cuckoo optimization algorithm for each class of software
Step 5: Giving the initial value to Cuckoo optimization algorithm parameters and constant parameters of COMOCO model
Step 6: Ovipositing of Cuckoos in different nests
Step 7: Killing the alien known eggs
Step 8: Checking the size of Cuckoos' population and in the case of large population, eliminating the worst places of Cuckoo residence
Step 9: Checking Cuckoo Fitness Function
Step 10: Finding the best places to grow Cuckoos (Maximum Fitness)
Step 11: Determining cuckoo population and migration to new residential areas
Step 12: Determining the laying radius for each Cuckoo
Step 13: In the case of non-completion algorithm go to the sixth stage
Step 14: Finalizing Cuckoo optimization algorithm
Step 15: Save the generated values for fixed parameters of Cocomo model (by Cuckoo optimization algorithm) and store classified samples

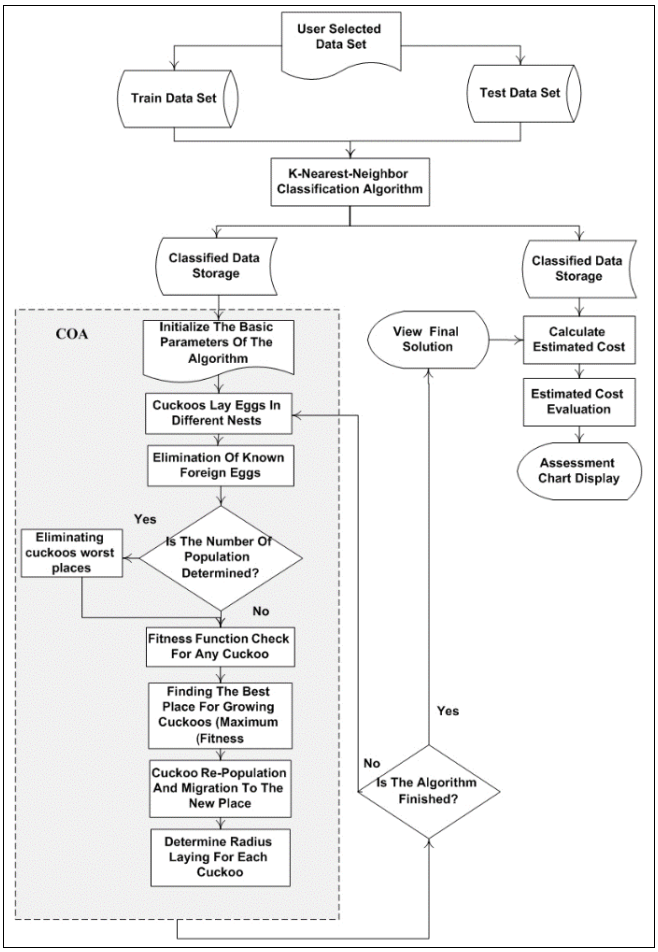

Fig. 1. The way of performance of the proposed approach

V. DISCUSSING AND EVALUATING THE RESULTS

The NASA60, NASA63, NASA93, MAXWELL, KEMERER, and MIYAZAKI data sets are used. The results are displayed in Tables IV to IX in terms of error rate. For the NASA60 dataset (Table IV) the proposed method has acted in all criteria better than comparative approaches. For the NASA63 dataset (Table V) the proposed method performs equally to KNN for the PRED(N) criterion, and better than the COCOMO model and KNN for the MDMRE criterion and worse than the COA-Cuckoo optimization algorithm. For the NASA93 dataset (Table VI) the proposed method performs equally to KNN for the PRED(N) evaluation criterion, and better than all comparative algorithms for rest of the criteria. For the MAXWELL dataset (Table VII) the proposed method performs better for all criteria, except for RMSE, MSE, MAE. For these criteria, it has performed better than COA-Cuckoo and worse than COCOMO model and KNN. For the KEMERER dataset (Table VIII) the proposed method has performed better in all criteria, and only in PRED(N) criterion has performed equally to COA-Cuckoo. For the MIYAZAKI dataset (Table IX) the proposed method has performed better in all criteria and only in PRED(N) criterion it has performed equally to the KNN.

TABLE IV. EVALUATION OF THE PROPOSED METHOD ON THE NASA60 DATASET

MSE	PRED(N)	MAE	MAPE	RMSE	MDMRE	MMRE	MMER	
8521.4	0.83	50.34	16.09	92.31	16.27	16.09	19.43	**COCOMO II**
6995.79	0.83	43.3	15.52	83.64	14.68	15.52	16.67	**KNN**
6924.33	0.75	47.14	18.6	83.21	15.7	18.6	14.77	**Cuckoo**
4592.82	0.92	37.47	14.86	67.77	8	14.86	13.85	**Proposed Method**

TABLE V. EVALUATION OF THE PROPOSED METHOD ON THE NASA63 DATASET

MSE	PRED(N)	MAE	MAPE	RMSE	MDMRE	MMRE	MMER	
8521.4	0.83	50.34	16.09	92.31	16.27	16.09	19.43	**COCOMO II**
6633.15	0.92	43.64	15.16	81.44	14.68	14.52	16.67	**KNN**
9925.84	0.85	54.4	16.76	99.63	10.56	16.76	15.21	**Cuckoo**
4483.69	0.92	36.49	12.94	66.96	13.75	12.94	13.25	**Proposed Method**

TABLE VI. EVALUATION OF THE PROPOSED METHOD ON THE NASA93 DATASET

MSE	PRED(N)	MAE	MAPE	RMSE	MDMRE	MMRE	MMER	
5933.62	0.84	44.49	18.23	77.03	20.98	18.23	23.01	**COCOMO II**
4970.08	0.89	40.04	17.24	70.5	18.18	17.24	21.27	**KNN**
2671.27	0.84	31.28	14.87	51.68	11.59	14.87	15.84	**Cuckoo**
1686.23	0.89	23.77	11.55	41.06	8.69	11.55	11.03	**Proposed Method**

TABLE VII. EVALUATION OF THE PROPOSED METHOD ON THE MAXWELL DATASET

MSE	PRED(N)	MAE	MAPE	RMSE	MDMRE	MMRE	MMER	
145135	0.38	2603	43.76	3809.67	36.58	43.76	118.44	**COCOMO II**
145135	0.38	2603	43.76	3809.67	36.58	43.76	118.44	**KNN**
242060	0.38	3111.74	39.99	4919.96	34.77	39.99	103.81	**Cuckoo**
206051	0.54	2851.59	35.82	4539.28	27.67	35.82	91.91	**Proposed Method**

TABLE VIII. EVALUATION OF THE PROPOSED METHOD ON THE KEMERER DATASET

MSE	PRED(N)	MAE	MAPE	RMSE	MDMRE	MMRE	MMER	
145135	0.38	2603	43.76	3809.67	36.58	43.76	118.44	**COCOMO II**
145135	0.38	2603	43.76	3809.67	36.58	43.76	118.44	**KNN**
242060	0.38	3111.74	39.99	4919.96	34.77	39.99	103.81	**Cuckoo**
206051	0.54	2851.59	35.82	4539.28	27.67	35.82	91.91	**Proposed Method**

TABLE IX. EVALUATION OF THE PROPOSED METHOD ON THE KEMERER DATASET

MSE	PRED(N)	MAE	MAPE	RMSE	MDMRE	MMRE	MMER	
35460	0	1375.95	687.64	1883.09	626.6	687.64	79.23	**COCOMO II**
35460	0	1375.95	687.64	1883.09	626.6	687.64	79.23	**KNN**
694	0.44	121.56	50.67	260.16	39.34	50.98	57.31	**Cuckoo**
676	0.44	39.34	49.98	57.31	35.61	49.67	56.69	**Proposed Method**

TABLE X. EVALUATION OF THE PROPOSED METHOD ON THE MIYAZAKI DATASET

MSE	PRED(N)	MAE	MAPE	RMSE	MDMRE	MMRE	MMER	
18960.4	0	122.16	272.79	137.7	277.48	272.79	70.83	**COCOMO II**
18960.4	0	122.16	272.79	137.7	277.48	272.79	70.83	**KNN**
699.54	0.4	21.3	34.45	26.45	42.95	34.45	62.95	**Cuckoo**
447.04	0.4	17.33	30.2	21.14	38.64	30.2	45.54	**Proposed Method**

VI. CONCLUSION

Before presenting the new estimation methods, software projects' cost estimation process of software projects was done with some simple algorithms. But with the passage of time and presenting new methods in the field of artificial intelligence, many methods have been proposed to solve this problem but none of the proposed methods are able to resolve this issue with one hundred percent accuracy. In this paper, a hybrid of two algorithms of Cuckoo optimization and KNN was used to solve this problem. This hybrid algorithm was evaluated the 6 different data set and based on eight evaluation criteria that the data set includes: KEMERER, MAXWELL, MIYAZAKI 1, NASA 60, NASA 63, NASA93 and evaluation criteria include: MMER, MMRE, MDMRE, RMSE, MAPE, MAE, PRED (N), MSE. Based on the obtained results, it can be concluded that the proposed method in KEMERER, MIYAZAKI1, NASA60, NASA93 datasets outperformed in all criteria better than

COCOMO, KNN and Cuckoo optimization algorithm but in NASA63 dataset in MDMRE criterion and in the Maxwell dataset MDMRE in the evaluation criteria of MSE, MAE, MSE has not a good performance compared to the comparative algorithms, and its reason can be the lack of consistent data..

REFERENCES

[1] F. S. Gharehchopogh, "Neural networks application in software cost estimation: a case study", 2011 IEEE International Symposium on Innovations in Intelligent Systems and Applications, pp. 69-73, Istanbul, Turkey, June 15-18, 2011

[2] B. Boehm, B. Clark, E. Horowitz, R. Shelby, C. Westland, "An overview of the COCOMO 2.0 software cost model", Software Technology Conference, 1995

[3] K. Parkash, H. Mittal, "Software cost estimation using fuzzy logic", ACM SIGSOFT Software Engineering, Vol. 35, No. 1, pp. 1-7, 2010

[4] Z. A. Dizaji, F. S. Gharehchopogh, "A hybrid of ant colony optimization and chaos optimization algorithms approach for software cost estimation", Indian Journal of Science and Technology, Vol 8, No. 2, pp. 128–133, 2015

[5] C. S. Reddy, P. S. Rao, K. Raju, V. V. Kumari, "A new approach for estimating software effort using RBFN network", International Journal of Computer Science and Network Security, Vol. 8, No. 7, pp. 237-241, 2008

[6] A. B. Krishna, T. K. R. Krishna, "Fuzzy and swarm intelligence for software effort estimation", Advances in Information Technology and Management, Vol. 2, No. 1, pp. 246-250, 2012

[7] I. Maleki, L. Ebrahimi, F. S. Gharehchopogh, "A hybrid approach of firefly and genetic algorithms in software cost estimation", MAGNT Research Report, Vol. 2, No. 6, pp. 372-388, 2014

[8] S. Sarwar, "Proposing effort estimation of cocomo ii through perceptron learning rule", Int. J. Comput. Appl., Vol. 7, No. 1, pp. 22–32, 2013

[9] T. M. Cover, P. E. Hart, "Nearest neighbor pattern classification", IEEE Trans. Inform. Theory, Vol. IT-13, pp 21-27, 1967

[10] T. Bailey, A. K. Jain, "A note on distance weighted k-nearest neighbor rules", IEEE Trans. Systems, Man Cybernatics, Vol. 8, pp. 311-313, 1978

[11] X. S. Yang, S. Deb, "Cuckoo search via levy flights", World Congress on Nature &Biologically Inspired Computing (NaBIC2009). IEEE Publications, pp. 210–214, 2009

[12] R Rajabioun, "Cuckoo optimization algorithm", Applied Soft Computing, Vol. 11, pp. 5508–5518 , 2011

[13] L. F. Capretz, V. Marza, "Improving effort estimation by voting software estimation models", Advances in Software Engineering, Article ID 829725, pp. 1-8, 2009

[14] S. Kumari, S. Pushkar, "Performance analysis of the software cost estimation methods: a review", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 3, No. 7, pp. 229-238, 2013