Investigating the Use of an Adaptive Neuro-Fuzzy Inference System in Software Development Effort Estimation

Zainab Rustum Mohsin^{1,*}

¹Department of Computer Science, College of Pure Science, University of Thi-Qar, Thi-Qar, Iraq.

*Corresponding Author: Zainab Rustum Mohsin

DOI: <https://doi.org/10.52866/ijcsm.2021.02.02.003>

Received March 2021; Accepted Jun 2021; Available online July 2021

ABSTRACT: Modeling software development effort estimation models has been a hot research topic over the last three decades. Numerous models were proposed in these decades to predict the effort. The key challenges for future software development is providing accurate software estimation. Failure to acknowledge the accuracy of effort estimation can cause inaccurate estimation, customer disappointment, and poor software development or project failure. This research presents a novel computational technique, named adaptive neuro-fuzzy inference system (ANFIS), for the modeling of software effort estimation. It was developed utilizing the Constructive Cost Model (COCOMO) dataset. The data were randomly divided into two sets: 83% for training and 17% for testing. The mean magnitude relative-error (MMRE) and the coefficient of correlation (R) were used as assessment indices. Results showed that the accuracy of the proposed model is quite satisfactory in comparison with actual values. Moreover, a comparison study was conducted with another approach. The results showed that ANFIS produced better results in comparison with Function Point Analysis, Software Lifecycle Management, and COCOMO methods. ANFIS was found to be a potential predictive model for software development effort estimation.

Keywords: Soft Computing; ANFIS; Fuzzy logic; Software effort estimation; COCOMO dataset.

1. INTRODUCTION

In software project management, the estimation of software development efforts is an important topic. This approach predicts development time and cost needed for software development. Precisely estimating cost and time is an essential step in project management for two reasons. Overestimating can result in financial loss for a company, and underestimation may result in a software with poor quality program, can lead to the failure of the project. Moreover, estimation at the initial stages of project development may benefit project managers. However, little information is available at the initial stage.

Many methods for effort estimation methods have been proposed, and they are divided into two main groups: algorithmic and non-algorithmic methods. Algorithmic methods include the Software Lifecycle Management (SLIM) model [1], Constructive Cost Model (COCOMO) model [2], and Function Points Analysis [3], which are the most widely used methods. The non-algorithmic methods include estimation methods, such as fuzzy logic (FL) [4], artificial neural networks (ANNs) [5], and regression trees [6].

In fact, many variables need to be considered during the software development process, and thus the software development process is complex. The complexity of the product developed, platforms available, human behavior, and environment [7] are some of these variables. Given that the accuracy of effort estimation is greatly influenced by modeling accuracy, discovering effective and accurate software estimation models has become one of the most difficult goals for the software engineering community.

An alternative technique to modeling has recently appeared under the rubric of “soft computing” and its main components “neural network” and “fuzzy logic.” These are observational models that have been developed on the basis of

available data. Soft computing includes some effective methods utilizing the human tolerance for uncertainty, imprecision, approximation, and partial truth in decision making for solving problems under uncertainty or complex problems where available information is imprecise [8]. Using these approaches would save a significant amount of time and cost and improve accuracy.

Neural networks (NNs) and FL techniques are model-free numerical estimators. They share the common capability for improving the predictive ability of a system that works in uncertain, imprecise, and noisy environments. NNs and fuzzy systems are complementary approaches. The capability of these approaches can be exploited by merging into an integrated system known as the adaptive neuro-fuzzy inference system (ANFIS). The integrated model combines the learning ability of ANNs with the power of linguistic representation offered by FS. This approach provided an impetus for the current research.

This study investigated the capability of ANFIS, which is one of the most common soft computing approaches for software effort estimation. The COCOMO database [2] is used as the dataset for training and testing the proposed model. To the best of our knowledge, no study has explored the application of the ANFIS approach to estimate the software development effort on the basis of COCOMO datasets for training and testing. This situation provided an impetus for the present study. The performance of the proposed ANFIS model is assessed and analyzed compared to the traditional methods. With this investigation we have attempted not only to develop a better effort estimation tool, but also to integrate soft computing techniques to software engineering.

The structure of the paper is as follows: Section 2 presents some background on the various prediction methods that were used as the basis for the research conducted. Section 3 briefly explains the ANFIS method and the architecture of the proposed model used in the present study. Section 4 describes the datasets and performance measures that were used to assess the effectiveness of the proposed model. Section 5 presents the results obtained by the ANFIS model and compares the results. Finally, Section 6 presents the conclusions and future works.

2. RELATED WORK

One of the most important responsibilities for everyone involved in software project management is estimating software effort. In the software industry, there is a lot of competition today to produce software with high quality at a specified cost and time. Therefore, in the key stages of software projects it is very important to estimate software development effort accurately. Regardless the breadth of research conducted in the last three decades, effective resource estimation in the software industry is remaining a major challenge. Time to time, researchers have discussed different approaches for the same. Rijwani and Jain [5] developed a model based on multi layered feed forward ANN approach for estimating software effort. They used the COCOMO dataset to train and test the proposed model. Kumar et al. [9] utilized wavelet neural network approach to predict the software development effort. Kamal et al. [10] developed a FL model to predict software effort. De Barcelos Tronto et al. [11] used two techniques for software effort estimation. They used ANN and multiple regression methods and finally concluded that the ANN model outperformance the regression model. Wittig and Finnie [12] proposed a model to estimate the software development effort using the ANN method. Karunanithi et al. [13] investigated the ability of ANN method in the estimation of software reliability growth. Reddy and Raju [14] developed FL model to estimate software effort. The presented model is based on COCOMO dataset. Karimi and Gandomani [15] presented a predictive model utilizing a hybrid of ANFIS and differential evolution algorithm (ANFIS-DE) to determine the software development effort estimation. Nassif et al. [16] implemented ANN to build a prediction model for software project cost estimation. They developed four different NN models, such as multi-layer perceptron, general regression NN, radial basis function NNs, and cascade correlation NNs. Nanda et al. [17] modelled the software effort estimation based on hybrid PSO- ANFIS method. The NASA datasets have been used to develop the model. The cost driver parameters comprise of 17 features which were optimized utilizing PSO algorithm in order to improve prediction accuracy. Moosavi et al. [18] proposed a model to achieve more accurate software development effort estimations using a combination of ANFIS and satin bower bird optimization algorithm (SBO) methods. The hybrid ANFIS-SBO model is an optimized neuro-fuzzy-based estimation model which is capable of producing accurate estimations in software project.

Finally, in the last years, soft computing techniques particularly ANFIS have been the prominent approaches for developing predictive models. Therefore, newer soft computing-based techniques can always be used to develop more accurate predictive precision models. In this vision, this paper presents a predictive model based on ANFIS method for software effort estimation.

3. ANFIS BASED METHODOLOGY

The ANFIS was developed to serve as a basis for constructing a fuzzy inference system (FIS), and its structure was obtained by incorporating fuzzy inference system into the ANN framework [19]. This model utilizes the capability of

ANN in training by some samples to produce the unknown parameters of a fuzzy model. The membership functions of the ANFIS model depend on the fuzzy rules and membership functions of the data [19]. Assuming that a first-order Sugeno FIS have two inputs (X_1 & X_2) and one output (y) will lead to two IF-THEN fuzzy rules [20]:

Rule 1: IF X_1 is A_1 and X_2 is B_1 , THEN $y_1 = m_1x_1 + n_1x_2 + q_1$

Rule 2: IF X_1 is A_2 and X_2 is B_2 , THEN $y_2 = m_2x_1 + n_2x_2 + q_2$

where A_1, A_2 and B_1, B_2 are membership functions for inputs X_1 and X_2 , respectively; m_1, n_1, q_1, m_2, n_2 , and q_2 are the design parameters for the output function.

The structure of a typical ANFIS is presented in Figure 1. The characteristics of each layer in the architecture are described as follows:

Layer 1: Input nodes. The main function of this layer is the fuzzification of the received inputs using membership functions. The output of the i th node is defined by

$$O_i^1 = \mu_{A_i}(x) \quad \text{for } i = 1, 2 \quad (1)$$

where x = input to the i th node, A_i = linguistic label associated with the node, and μ_{A_i} = membership function. Parameters in this layer are generally defined as premise parameters.

FIGURE 1. Architecture of an ANFIS approach [19]

Layer 2: Rule nodes. Each node in this layer is a fixed node and denoted as M (Figure 2). In each node, the incoming signals are multiplied, and the output O_i^2 , which represents the firing strength of a rule, is calculated as follows:

$$O_i^2 = w_i = \mu_{A_i}(x)\mu_{B_i}(x) \quad \text{for } i = 1, 2 \quad (2)$$

Layer 3: Average nodes. Each i th node in this layer is a fixed node (symbol N in Figure 2). The i th node in this layer calculates the normalized firing strength as follows:

$$O_i^3 = \bar{w}_i = \frac{w_i}{w_1 + w_2} \quad \text{for } i = 1, 2 \quad (3)$$

Layer 4: Consequent nodes. All the nodes in this layer are adaptive node with a node function

$$O_i^4 = \bar{w}_i y_i = \bar{w}_i (m_i x_i + n_i x_2 + q_i) \quad \text{for } i = 1, 2 \quad (4)$$

where $\bar{w}_i$ is the output of layer 3, and m_i, n_i, q_i is the consequent parameter set.

Layer 5: Output nodes. This layer consists of single fixed node labeled Σ , which computes the overall output of the ANFIS model as the summation of all incoming signals, that is,

$$O_i^5 = \Sigma_i \bar{w}_i y_i = \frac{\Sigma_i \bar{w}_i y_i}{\Sigma_i \bar{w}_i} \quad \text{for } i = 1, 2 \quad (5)$$

More details about the ANFIS algorithm can be found in [19].

4. DATA DESCRIPTION AND PERFORMANCE MEASURES

4.1 DATA DESCRIPTION

In this study, effort was calculated with the COCOMO 81 database [2] and ANN and ANFIS approaches. COCOMO81 was selected because it is a public domain database that has already been used for various methods. Moreover, this database was utilized in describing and testing one of the most significant approaches in the area, that is, the approaches provided in Ref. [21]. These data consisted 63 projects and 17 cost drivers that were used in training and testing the proposed models. Table 1 lists the cost driver variables considered in this study. Size, denoted as ADJKDSI, was the only numerical independent variable in the analysis. Better fitting and prediction results were obtained by transforming this variable into its corresponding logarithmic value. The output of the ANFIS model was the software effort, which was measured in man-months. The data were randomly divided into two sets: 52 (83%) were used for the training process, and the remaining 11(17%) were for testing the proposed models.

Table 1. COCOMO cost drivers

Variable	Full name
ACAP	Analyst capability
TOOL	Use of software tools
AEXP	Applications experience
RELY	Reliability
MODP	Use programming modern practices
DATA	Database size
VEXP	Experience with virtual machine
PCAP	Programmers capability
CPLX	Process complexity
RVOL	Requirements volatility
LEXP	Programming language experience
TIME	Restriction of time
TURN	Computer turnaround
SCED	Schedule constraint
VIRT	Virtual machine volatility
STOR	Main storage constraint
ADJKDSI	Software size

4.2 PERFORMANCE MEASURES

The following performance indicators have been used in evaluating and comparing the predictive accuracy of the ANFIS model.

a. The mean magnitude of relative error (MMRE): MMRE is the mean measurement of the absolute values of the relative errors over N projects [22].

$$MMRE = \frac{1}{N} \sum_{i=1}^N \left| \frac{\text{Actual Effort} - \text{Predicted Effort}}{\text{Actual Effort}} \right| \quad (6)$$

b. The correlation coefficient (R): R is used in determining how strong a relationship is between actual values and predicted values.

$$R = 1 - \sqrt{\frac{\sum_{i=1}^N (\text{Actual Effort} - \text{Predicted Effort})^2}{\sum_{i=1}^N (\text{Actual Effort})^2}} \quad (7)$$

The R values are between -1 and $+1$, and the best models provided R values that are as close to $+1$.

5. RESULTS AND DISCUSSION

Modeling using ANFIS approach comprises three phases: (a) data preprocessing, (b) model construction (structure), (c) training, and (d) ANFIS model testing. MATLAB software was utilized for the implementation of the proposed model.

A total of 17 input parameters and effort as output variable were used in building the predictive model. Table 1 presents the input variables.

5.1 GENERATION OF THE ANFIS MODEL

ANFIS models can be composed of various structures that govern output and estimation performance. The subtractive clustering method introduced by Chiu [23] has been adopted to generate the FIS structure with the MATLAB function (*genfis2*). The *genfis2* generates a FIS utilizing subtractive clustering method and needed separate pairs of an input–output dataset as input argument. When only one output is available, *genfis2* can be used to create an initial FIS for ANFIS training.

The most common FIS are Mamdani and Sugeno [24, 25]. The Sugeno FIS was used in this study. The training and testing datasets provided good predictions in terms of MSE and the type and the number of membership functions.

After the extermination of various learning algorithms with different numbers of epochs, the optimum correlations were archived by using a hybrid learning algorithm (a combination of least squares method and back-propagation algorithm for learning the premise and consequent parameters, respectively). The membership function of type Gaussian membership functions (*gaussmf*) was used for input data, and a linear function was used for output parameters. The ANFIS model achieved final errors of 221.74 for the training stage and 126.08 for the testing stage after 300 epochs. After the training process, the proposed model was tested with a dataset that had not been used for training. The purpose was to assess the accuracy and efficiency of the trained ANFIS model. Table 2 presents the development effort estimates for each testing dataset.

Table 2. Estimated effort using the proposed technique

Project ID	Actual effort	Computed effort with ANFIS
6	43	59.9
12	201	76.44
21	2455	2449.13
24	453	171.83
30	5.9	7.14
36	55	15.7
43	83	90.26
48	1272	992.54
54	20	14.71
61	50	48.84
62	38	22.47

Figure 2 shows the R values of the ANFIS model for training and testing datasets. A good agreement was observed between the outputs of the proposed model and actual (targets) values. The R values were 1.0 for training and 0.989 for the testing phase. Figure 3 displays a comparison of the actual and predicted efforts from the ANFIS model during the testing stage. The results of the proposed ANFIS model matched well with the actual values.

FIGURE 2. Comparison between ANFIS results and actual data for (a) training datasets; (b) testing datasets

FIGURE 3. Comparative charts of actual and estimated effort values using ANFIS model for testing datasets

5.2 COMPARISON STUDY

Table 3 summarizes the MMRE and correlation R values resulting from the ANFIS model, and results obtained by Kemerer [26] with Function Points, COCOMO-Basic, and SLIM models.

Table 3. Comparison of performance measured for ANFIS model with traditional models

Model	MMRE	R
ANFIS	32.4	0.989
COCOMO Basic	610	0.70
Function Points Analysis	103	0.58
SLIM	772	0.89

According to Table 3, ANFIS produced a scientifically higher MMRE and correlation R values than other models. In terms of correlation R, the ANFIS model (0.989) outperformed COCOMO (0.70), SLIM (0.89), and Function Point Analysis models (0.58) in Kemerer's experiments. On the MMRE dimension, the ANFIS was significantly better than the other methods. The reasons for the superior positive results of ANFIS may be attributed to its structure and ability to eliminate noisy data and its application of the "IF-THEN" rules to generate an output for each rule and learn from data [27]. The results revealed that the ANFIS model is effective in software effort estimation.

6. CONCLUSION AND FUTURE WORKS

A novel method based on soft computing approaches, named ANFIS, was presented, to predict effort in software development. The Boehm's COCOMO dataset was used for training and testing the presented model. Various models can be proposed within the ANFIS framework by using various learning algorithms with various membership functions and epochs. After different ANFIS models were tested, the model with the best potential for prediction was selected. An ANFIS model with a hybrid learning algorithm, 300 epochs, and Gaussian membership functions was selected and then tested. The results of the ANFIS model were compared with those from other methods, namely, Function Points Analysis, COCOMO-Basic, and SLIM models. Two assessment criteria: MMRE and correlation coefficient R, were used for the comparison and evaluation. The values of MMRE and R for testing set were 32.4 and 0.989, respectively, which indicated that the proposed ANFIS model predicted software development effort with high accuracy. Moreover, the ANFIS model provided better results than the other models. This study has an important implication for practical applications. This can be attributed to the architecture of the ANFIS. The ANFIS captures the benefits of the learning capability of neural network and the simplifying function of fuzzy reasoning, thus having an ability to eliminate noise [28]. The ANFIS is a potentially reliable and simple tool for the software effort estimation. For future work, other soft computing methods can be used for software effort estimation.

ACKNOWLEDGMENT

The author of the article is grateful to the University of Thi-Qar.

REFERENCES

- [1] L. H. Putnam, "A general empirical solution to the macro software sizing and estimating problem," *IEEE transactions on Software Engineering*, vol. 4, no. 4, pp. 345–361, 1978.
- [2] B. Boehm, *Software engineering economics*. Englewood Cliffs, NJ: Prentice-Hall, 1981.

- [3] A. J. Albrecht and . E. Gaffney, "Software function, source lines of code, and development effort prediction: a software science validation," *IEEE transactions on software engineering*, vol. 9, no. 6, pp. 639–648, 1983.
- [4] S. Kumar, B. A. Krishna, and . S. Satsangi, "Fuzzy systems and neural networks in software engineering project management," *Applied Intelligence*, vol. 4, no. 1, pp. 31–52, 1994.
- [5] P. Rijwani and Jain, "Enhanced software effort estimation using multi layered feed forward artificial neural network technique," *Procedia Computer Science*, vol. 89, pp. 307–312, 2016.
- [6] R. W. Selby and . A. Porter, "Learning from examples: generation and evaluation of decision trees for software resource analysis," *IEEE Transactions on Software Engineering*, vol. 14, no. 12, pp. 1743–1757, 1988.
- [7] N. K. L. Alhammad, E. Alzaghou, and F. A. Akour, "Evolutionary neural network classifiers for software effort estimation," *International Journal of Computer Aided Engineering and Technology*, vol. 12, no. 4, pp. 495–512, 2020.
- [8] D. Nauck, F. Klawonn, and Kruse, 1997.
- [9] K. V. Kumar, V. Ravi, M. Carr, and . R. Kiran, "Software development cost estimation using wavelet neural networks," *Journal of Systems and Software*, vol. 81, no. 11, pp. 1853–1867, 2008.
- [10] S. Kamal and . A. Nasir, "A fuzzy logic based software cost estimation model," *International Journal of Software Engineering and Its Applications*, vol. 7, no. 2, pp. 7–18, 2013.
- [11] I. F. D. B. Tronto, J. D. S. D. S. Andn, Sant, and Anna, "An investigation of artificial neural networks based prediction systems in software project management," *Journal of Systems and Software*, vol. 81, no. 3, pp. 356–367, 2008.
- [12] G. E. Wittig and Finnic, "Using artificial neural networks and function points to estimate 4GL software development effort," *Australasian Journal of Information Systems*, vol. 1, no. 2, 1994.
- [13] N. Karunanithi, D. W. Andy, and . K. Malaiya, "Using neural networks in reliability prediction," *IEEE Software*, vol. 9, no. 4, pp. 53–59, 1992.
- [14] C. S. Reddy and Raju, "An improved fuzzy approach for COCOMO's effort estimation using gaussian membership function," *Journal of software*, vol. 4, no. 5, pp. 452–459, 2009.
- [15] A. Karimi and . J. Gandomani, "Software development effort estimation modeling using a combination of fuzzy-neural network and differential evolution algorithm," *International Journal of Electrical & Computer Engineering*, vol. 11, no. 1, pp. 2088–8708, 2021.
- [16] A. B. Nassif, M. Azzeh, L. F. Capretz, and Ho, "Neural network models for software development effort estimation: a comparative study," *Neural Computing and Applications*, vol. 27, no. 8, pp. 2369–2381, 2016.
- [17] S. Nanda and Soewito, "Modeling software effort estimation using hybrid PSO-ANFIS," *2016 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, pp. 219–224, 2016.
- [18] S. H. S. Moosavi and . K. Bardsiri, "Satin bowerbird optimizer: A new optimization algorithm to optimize ANFIS for software development effort estimation," *Engineering Applications of Artificial Intelligence*, vol. 60, pp. 1–15, 2017.
- [19] J. S. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE transactions on systems, man, and cybernetics*, vol. 23, pp. 665–685, 1993.
- [20] T. Takagi and Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE transactions on systems, man, and cybernetics*, vol. 15, pp. 116–132, 1985.
- [21] B. Kitchenham, "A procedure for analyzing unbalanced datasets," *IEEE transactions on Software Engineering*, vol. 24, no. 4, pp. 278–301, 1998.
- [22] D. R. Baker, "A hybrid approach to expert and model based effort estimation," 2007. Citeseer.
- [23] S. L. Chiu, "Fuzzy model identification based on cluster estimation," *Journal of Intelligent & fuzzy systems*, vol. 2, no. 3, pp. 267–278, 1994.
- [24] M. Sugeno, "An introductory survey of fuzzy control," *Information sciences*, vol. 36, no. 1-2, pp. 59–83, 1985.
- [25] E. H. Mamdani and Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *International journal of man-machine studies*, vol. 7, no. 1, pp. 1–13, 1975.
- [26] C. F. Kemerer, "An empirical validation of software cost estimation models," *Communications of the ACM*, vol. 30, no. 5, pp. 416–429, 1987.
- [27] A. A. Tofigh, M. R. Rahimpour, M. O. Shabani, and Davami, "Application of the combined neuro-computing, fuzzy logic and swarm intelligence for optimization of compocast nanocomposites," *Journal of Composite Materials*, vol. 49, no. 13, pp. 1653–1663, 2015.
- [28] T. Rajae, S. A. Mirbagheri, M. Z.-K. Andv, and Nourani, "Daily suspended sediment concentration simulation using ANN and neuro-fuzzy models," *Science of the total environment*, vol. 407, no. 17, pp. 4916–4927, 2009.