Journal of Information Systems Engineering & Management, 1:4 (2016), 46
ISSN: 2468-4376

http://www.lectitojournals.com

Applying Absolute Residuals as Evaluation Criterion for Estimating the Development Time of Software Projects by Means of a Neuro-Fuzzy Approach

Noel García-Díaz[1,2]*, Alberto Verduzo-Ramirez [1,2], Juan Garcia-Virgen [1,2], Lilia Muñoz[3]

[1]Instituto Tecnológico de Colima, MÉXICO
[2]Universidad de Colima, MÉXICO
[3]Universidad Tecnológica de Panamá, REPUBLICA DE PANAMÁ
***Corresponding Author**: ngarcia@itcolima.edu.mx

Citation: García-Díaz, N., Verduzo-Ramirez, A., Garcia-Virgen, J. and Muñoz, L. (2016) Applying Absolute Residuals as Evaluation Criterion for Estimating the Development Time of Software Projects by Means of a Neuro-Fuzzy Approach, *Journal of Information Systems Engineering & Management, 1*:4 (2016), 46.

doi: http://dx.doi.org/10.20897/lectito.201646

Received: July 14, 2016; **Accepted**: September 24, 2016; **Published**: November 4, 2016

ABSTRACT

In the software development field, software practitioners expend between 30% and 40% more effort than is predicted. Accordingly, researchers have proposed new models for estimating the development effort such that the estimations of these models are close to actual ones. In this study, an application based on a new neuro-fuzzy system (NFS) is analyzed. The NFS accuracy was compared to that of a statistical multiple linear regression (MLR) model. The criterion for evaluating the accuracy of estimation models has mainly been the Magnitude of Relative Error (MRE), however, it was recently found that MRE is asymmetric, and the use of Absolute Residuals (AR) has been proposed, therefore, in this study, the accuracy results of the NFS and MLR were based on AR. After a statistical paired t-test was performed, results showed that accuracy of the New-NFS is statistically better than that of the MLR at the 99% confidence level. It can be concluded that a new-NFS could be used for predicting the effort of software development projects when they have been individually developed on a disciplined process.

Keywords: Artificial neural networks, fuzzy logic, neuro-fuzzy system, Software effort estimation, absolute residuals

INTRODUCTION

A high percentage of machine learning models have been proposed based on an accuracy asymmetric criterion Magnitude of Relative Error (MRE) (Wen et al., 2012), however, it was recently found that MRE is asymmetric and that the use of the Absolute Residual (AR) should be used instead because of AR is unbiased and it is does no lead to asymmetry (Shepperd & MacDonell, 2012). The AR criterion has been already used in recent publications for estimating the effort (López-Martín, 2015) and schedule (duration, time) (Ferreira-Santiago et al., 2015) of software projects.

Software Engineering (SE) is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software and it provides the fundamentals, principles and skills needed to develop and maintain high quality software products (Abran et al., 2004). Some of the areas of SE are: Requirement, design, construction, testing, and management. Software Engineering Management includes planning and measurement of SE, which involves to the Software Development Effort Estimation (SDEE).

The Chaos report conducted by the Standish Group (The Standish group, 2012), which is the report on the failure of projects in the field of information technologies, measures the success of projects only if completed in time, within budget, and if they met the requirements. Several research works in software development effort estimation have cited the Chaos report (De Araújo et al., 2012; Bonneti et al., 2012; De Araújo et al., 2012b; LagerstrÖm et al., 2012; Moløkken-Østvold & Jørgensen, 2003). This report found that more than half of software projects worldwide (around 61%) conducted between 2004 and 2012 were delivered with delay, were over budget and many were not even finished; just 39 percent were classified as successful. The main cause of these problems is a failure of the SDEE (De Araújo et al., 2009, 2011). Estimation of software development effort is the basis for project bidding, budgeting and planning. The consequences of poor budgets and plans can be dramatic: if they are too pessimistic, business opportunities can be lost, while over optimism may be followed by significant losses.

The SDEE activity could start using a personal level approach, starting with the development of small-size projects. The disciplined software development at the personal level based on small-scale projects, represented by the personal software process (PSP), have offered benefit for thousands of developers in academic or industrial training courses (Rombach et al., 2008).

Two of the three most important causes of Information Technology projects failure have been related to a poor resource estimation (González-Carrasco et al., 2012). In average, software developers expend from 30% to 40% more effort than is estimated (Jørgensen & Shepperd, 2007). Because that no single technique to estimate software development effort is best for all situations, it is important to propose new models to compare their results and then generate more realistic estimates (Boehm & Abts, 2000).

The objective of this paper is to present a new Neuro-Fuzzy System (NFS) for achieving higher accuracy for estimating the development time of software projects using the AR and its mean (MAR).

The data set obtained from (Lopez-Martin et al., 2005) with forty-one modules developed in ten projects were used for training and testing the models. The accuracy of the new NFS was compared to that of a Multiple Linear Regression (MLR) model.

The rest of this study is structured as follows: Section 2 presents the related work. Section 3 defines SDEE and the AR is described. In Section 4 is described the software estimation technique where SDEE has been addressed. A brief description to MLR, Fuzzy Logic (FL), Neural Networks (NN) and NFS is presented in their respective sections. Section 5 is dedicated to the description of the data set from which the models are generated; then is carried out the generating of the MLR model and the New-NFS, whereas in Section 6 the results are presented and compared. Finally, conclusions and future work are mentioned in Section 7.

RELATED WORK

A systematic review of 157 studies published between the years 2002 and 2012 involved the application of NFS (Kar et al., 2014). This review classifies the NFS applications into ten different categories, and in none of them was found any study regarding SDEE, therefore, in this study a NFS is proposed as a new model to compare its results to a MLR.

Additional estimation techniques have been proposed into the area of SDEE to improve estimation accuracy (Wen et al., 2012; Jørgensen & Shepperd, 2007).

A systematic review of 84 studies analyzed machine learning models specifically applied to SDEE (Wen et al., 2012). This review involved empirical studies published in the last two decades (1991–2010). After analyzing these studies, by Wen et al., found eight types of machine learning techniques applied to SDEE; however, it was not found any paper using data from small projects and basing its conclusions on AR and having a NFS using a Grid partitioning method to obtain the parameters for input and output of the membership functions (MF).

A second systematic review completed in 2004 identified 304 SDEE studies in 76 journals (Jørgensen & Shepperd, 2007). It classified the studies according to their research topic, estimation approach, research approach, study context and data set. In this review, it was not including any neuro-fuzzy model used for SDEE.

A NFS to estimate software projects development time was built by (Marza et al., 2008). The forty-one modules developed from ten software projects were used as data set. The proposed approach was compared with FL and NN model and results showed that the value of MMRE applying NFS was substantially lower than MMRE applying FL and NN.

In (Garcia-Diaz et al., 2015) the accuracy of time estimation for a NFS was statistically better than the accuracy obtained from a previous NFS and statistical regression when the forty-one modules developed from ten programs were used as data set. Results showed that the value of MMRE (Mean of Magnitude of Relative Error) applying a New-NFS was substantially lower than MMRE applying a previous NFS and statistical regression.

In (Lopez-Martin et al., 2005) was proposed a FL model for SDEE whose results are compared with those of a multiple regression. Results showed that the value of MMRE applying FL was slightly higher than MMRE applying multiple regression.

SOFTWARE ENGINEERING MANAGEMENT

A. *Software development effort estimation (SDEE)*

The SDEE has been defined, at least since 1969 as the amount of time in human hours needed to design, code, and test a software project (Naur & Randell, 1969).

The SDEE process consists of specific activities:

1. Obtaining data from previous projects.
2. Generation of estimation models.
3. Checking and validating the models, based on accuracy.

One activity of software project planning is the estimation of the development effort, which was considered to be one of the three great challenges of computer science (Brooks, 2003) and effort estimation techniques have been proposed and researched over the last years (Wen et al., 2012; Jørgensen & Shepperd, 2007). Researchers aimed at (1) determining which technique had the greatest effort estimation accuracy, or (2) proposing new or combined techniques that could provide better estimates.

SDEE techniques can be classified into two general categories (López-Martín, 2015):

1) Expert judgment: This technique implies a lack of analytical argumentation and aims at deriving estimates based on the experience of experts on similar projects; this technique is based on a tacit (intuitive) quantification step.

2) Model-based technique: It is based on a deliberate (mechanical) quantification step, and it can be divided into the following two subcategories:

a) Models based on statistics: Its general form is a statistical regression model.

b) Models based on machine learning such as FL, NN, genetic programming, and genetic algorithm.

The present work uses estimations obtained with an algorithmic model and it attempts to represent the relationship between effort and one or more characteristics of a project, based on statistics (MLR) and a NFS, which is a hybrid model based on a computational technique. MLR has been the dominating technique for software estimation in recent years (Jørgensen & Shepperd, 2007).

B. *Evaluation criterion*

For evaluating the different software effort estimation machine learning models is used the Magnitude of Relative Error (MRE) which is the most popular accuracy metric when compared effort estimation models (Wen et al., 2012), however MRE is an accuracy criterion known to lead to asymmetry (Shepperd & MacDonell, 2012), therefore, in this research the absolute residuals is used as suggested in (Shepperd & MacDonell, 2012).

The AR is defined as follows:

$$AR_i = \left| ActualEffort_i - EstimatedEffort_i \right| \qquad (1)$$

The AR value is calculated for each observation i whose effort is estimated. The aggregation of ARs over multiple observations (N) can be achieved through its mean (MAR) as follows:

$$\boldsymbol{MAR = (\frac{1}{N}) \sum_{1}^{N} AR_i}$$

The accuracy of an estimation technique is inversely proportional to the MAR (Shepperd & MacDonell, 2012). That is, a lower MAR indicates a more accurate estimate.

SOFTWARE ESTIMATION TECHNIQUE

A. *Multiple linear regression (MLR)*

Accuracy of statistical regressions has frequently been used to be compared to other software estimation models (Wen et al., 2012; Jørgensen & Shepperd, 2007). The comparison against a statistical regression model is suggested because it should be built as the default model construction method (Kitchenham et al., 2007).

B. *Fuzzy logic (FL)*

FL was introduced by Zadeh in 1965. FL is the definition given to a mathematical system developed to model the brain of human curious way of processing and selecting words (Zadeh, 1965). The main motivation behind the creation of FL was the existence of imprecision in the measurement process.

FL represents models or knowledge using IF–THEN rules in the form of “if X then Y”.

A fuzzy model is a modelling construct featuring two main properties (Zhiwei-Xu & Khoshgoftaar, 2004): (1) It operates at a level of linguistic terms (fuzzy sets that are sets whose elements have degrees of membership), and (2) it represents and processes uncertainty.

FL offers a particularly convenient way to generate a keen mapping between input and output spaces thanks to the natural expression of fuzzy rules (Zadeh, 2002).

In SDEE, two considerations justify the decision of implementing a fuzzy model: first, it is impossible to develop a precise mathematical model of the domain (Lewis, 2001); second, metrics only produce estimations of the real complexity.

There are two types of fuzzy inference system (FIS), these are: Mamdani (Mamdani, 1976) and Takagi-Sugeno (Takagi & Sugeno, 1983). The FIS that was used in this study is the proposed by Takagi-Sugeno (Takagi & Sugeno, 1983) once we did not find any study that used for the SDEE of small projects. The Mamdani (Mamdani, 1976) FIS expects the output MF to be fuzzy sets, whereas the Takagi-Sugeno-type system can be used to model any inference system in which the output is either linear or constant. In this research was used the constant output.

The rules in functional Takagi-Sugeno fuzzy systems have the form:

$\mathrm{R_i}$: IF $\mathrm{x_1}$is $\mathrm{A_1^j}$ and…and $\mathrm{x_n}$is $\mathrm{A_n^j}$THEN y is $\mathrm{f_j}(x)$

Where $\mathrm{f_j}(x)$ is a crisp function of the input variables, rather than a fuzzy proposition (Takagi & Sugeno, 1985). For a particular application, the effectiveness of the fuzzy system in most cases depends on the order of the function.

C. Neural networks (NN)

A NN, is a massively parallel, distributed system composed of simple processing units or artificial neurons that are interconnected to mimic a biological NN (Haykin, 1999).

Before a NN can be used, it has to undergo some training, which involves iteratively finding the appropriate weight values so that the network outputs the desired value for a given a set of input values. A number of training algorithms have been developed over the years, with Backpropagation being the most widely known (Haykin, 1999). After a NN has been trained, it is convenient to validate its performance using ideally a dataset different from the one used to train it.

In this research was used a combination of training algorithms between Backpropagation and least mean squares.

NN are used in SDEE due to its ability to learn from previous data. In addition, it has the ability to generalize from the training data set thus enabling it to produce acceptable result for previously unseen data (Su et al., 2007).

NN can model complex non-linear relationships and approximate any measurable function such that it is very useful in problems where there is a complex relationship between inputs and outputs (Aggarwal et al., 2005; Huang et al., 2007).

D. Hybrid systems

One of the most important capabilities of FL is to model the qualitative aspects of human by using the simple rules. The NN have some advantages such as its capability of learning and high computational power. As a result, it is possible to combine the advantages of NN and FL to make a better tool.

NFS is a fuzzy system augmented by NN to enhance some characteristics like flexibility and adaptability (Nauck, 1994; Nauck et al., 1997; Saliu, 2003).

The NN research started in the 1940s, and the FL research in the 1960s, but the neuro-fuzzy research area is relatively new (Jantzen, 1998).

The neuro-fuzzy hybrid systems may be divided into two major groups (Mitra & Hayashi, 2000): FNN and NFS. FNN is a NN equipped with the capability of handling fuzzy information. NFS is a fuzzy system combined with NN in order to enhance certain desirable characteristics (Nauck, 1994; Nauck et al., 1997; Saliu, 2003). This research is based on the second approach.

A NFS can be viewed as a special three layer NN (Nauck et al., 1997). The first layer represents input variables; the hidden layer represents fuzzy rules and the third layer represents output variables.

The first integrated hybrid NFS is ANFIS; it has lowest Root Mean Square Error (RMSE) among other NFS like the ARX model. Therefore, ANFIS was used here for implement Takagi-Sugeno NFS. By MATLAB, the ANFIS structure with (a) type: Sugeno FIS, (b) and method: prod, (c) or method: probor, (d) implication Method: min, (e) aggregation Method: max and (f) defuzzfication: Wtaver (weighted average) was implemented and its architecture is shown in Fig. 1 (Abraham, 2005).

The functioning of each layer is as follows (Abraham, 2005):

Layer-1 (input layer): No computation is done in this layer. Each node in this layer, which corresponds to one input variable, only transmits input values to the next layer directly. The link weight in layer 1 is unity.

Layer-2 (fuzzification layer): Each node in this layer corresponds to one linguistic label to one of the input variables in layer 1. In other words, the output link represents the membership value, which specifies the degree to which an input value belongs to a fuzzy set, is calculated in layer 2. A clustering algorithm will decide the initial number and type of MF to be allocated to each of the input variable. The final shapes of the MFs will be fine-tuned during network learning.

Layer-3 (rule antecedent layer): A node in this layer represents the antecedent part of a rule. Usually a T-norm operator is used in this node. The output of a layer 3 node represents the firing strength of the corresponding fuzzy rule.

Layer-4 (rule strength normalization): Every node in this layer calculates the ratio of the firing strength of the i-th rule to the sum of all rules firing strength.

$$\overline{w}_i = \frac{w_i}{w_1 + w_2},\ i=1, 2\ldots \qquad (3)$$

Layer-5 (rule consequent layer): Every node i in this layer is with a node function.

$$\overline{w}_i f_i = \overline{w_i}\,(p_i x_1 + q_i x_2 + r_i) \qquad (4)$$

Where $\overline{w}_i$ is the output of layer 4, and $\{p_i, q_i, r_i\}$ is the parameters set. A well-established way is to determine the consequent parameters using the least means squares algorithm.

Layer-6 (rule inference layer): The single node in this layer computes the overall output as the summation of all incoming signals.

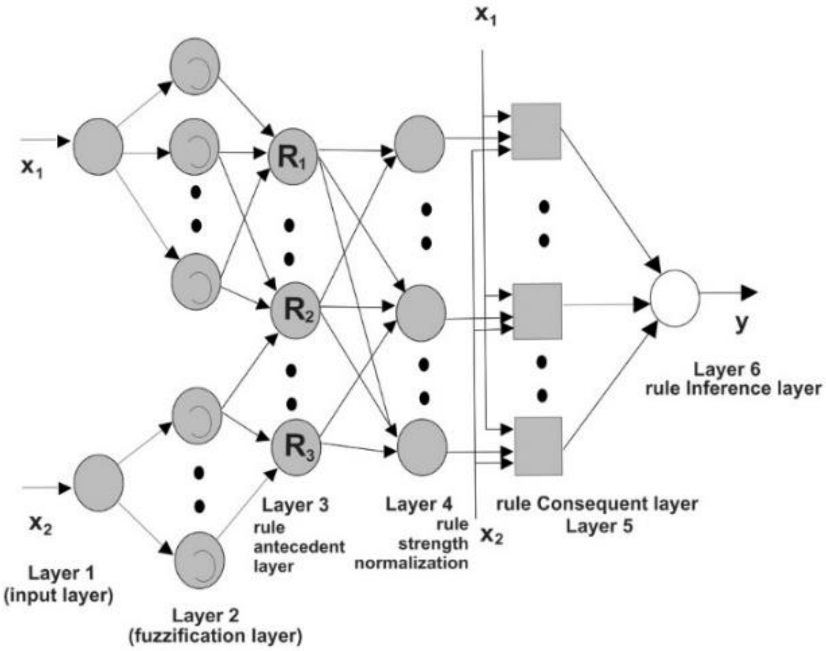


Figure 1. Neuro-fuzzy system.

GENERATION OF MODELS

A. Data sample description

The comparative study carried out here was based on the empirical study done by (Lopez-Martin et al., 2005; Marza et al., 2008; Garcia-Diaz et al., 2015). The development time of forty-one modules and for each module, coupling (Dhama), complexity (McCabe), and lines of code metrics were registered, all programs were written in Pascal, hence, module categories belong to procedures or functions.

The development time of each of the forty-one modules were registered including five phases: requirements understanding, algorithm design, coding, compiling and testing (Lopez-Martin et al., 2005).

The statistics and a brief description related to each module are described by (Lopez-Martin et al., 2005).

B. Multiple linear regression

Using the data described in (Lopez-Martin et al., 2005; Marza et al., 2008; Garcia-Diaz et al., 2015) the MLR equation considering the New & Changed (N&C) lines of code was obtained:

Time = 17.31 + (2.06*MC) – (32.94*DC) - (0.05*LOC) (5)

The equation 5 describes the relationship between the dependent variable (Time) and the independent variables (McCabe complexity, Dhama coupling as well as lines of code).

An acceptable value for the coefficient of determination is $r^2 \geq 0.5$ (Humphrey, 1995). In this case, the r^2 of equation 5 was 0.7223. The ANOVA for this equation had a statistically significant relationship between the variables at the 95% confidence level.

C. Neuro-fuzzy system (NFS)

In according to (Lopez-Martin et al., 2008) a Gaussian MF have two parameters, one of them (k) determines the curve shape and the other one (m) corresponds to curve central position. Their scalar parameters (k, m) are defined as follows:

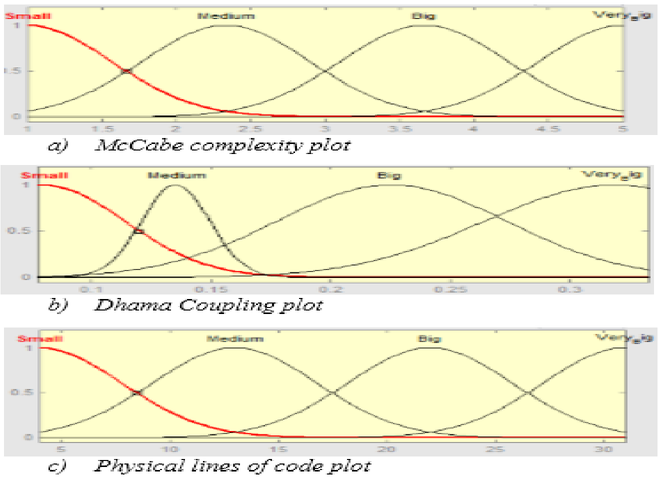
$$\mathrm{MF}(x) = e^{-k(x-m)^2} \qquad (6)$$

Table 1 shows the final parameters of the MF of input variable. The NFS parameters are obtained by the Grid Partitioning method. Grid partition divides the data space into rectangular subspaces using axis-paralleled partition based on predefined number of MF and their types in each dimension (Wei et al., 2007). According to (Wei et al., 2007) grid partition is only suitable for cases with small number of input variables (e.g. less than 6).

In Fig. 2a to 2c are shown the three input variables with its respective small, medium, big and very big MFs and its parameters with each one of the MFs from table 1.

Table 1. Parameters of mf of input variables

Input Variable	Range	MF	Parameters	
McCabe	1-5	Small	0.5662	1
		Medium	0.5658	2.33
		Big	0.5665	3.66
		Very_Big	0.5663	5
DC	0.077-0.333	Small	0.0344	0.0794
		Medium	0.0144	0.135
		Big	0.0483	0.2255
		Very_Big	0.0547	0.3191
LOC	4-31	Small	3.822	4
		Medium	3.822	13
		Big	3.821	22
		Very_Big	3.822	31



a) McCabe complexity plot

b) Dhama Coupling plot

c) Physical lines of code plot

Figure 2. Input variables of Takagi-Sugeno New-NFS.

RESULTS

Mean Magnitude of Relative Error (MMRE) assesses the validation results of estimation accuracy of the 41 projects in a previous work (Garcia-Diaz et al., 2015) whose results are shown in Table 2.

Table 2. MRE of Each Technique (**MLR**: Multiple Linear Regression, **NFS**: Neuro-Fuzzy System, **MMRE**: Mean of Magnitude of Error Relative).

	MLR	**New-NFS**
MMRE	0.1005	0.0163

Table 3. MAR of each module (**MLR**: Multiple Linear Regression, **NFS**: Neuro-Fuzzy System, **MAR**: Mean of Absolute Residuals).

Module	**Actual Time**	**AR - MLR**	**AR -New-NFS**
1	13	2.06	0.0003
2	13	2.36	0.0000
3	9	0.80	0.0001
4	15	3.20	0.0001
5	15	1.63	0.0000
6	15	1.87	0.4999
7	16	0.87	0.5001
8	16	0.62	0.0001
9	16	0.42	0.0073
10	18	2.47	0.0129
11	15	0.43	1.0068
12	15	0.43	1.0068
13	18	2.57	1.9932
14	13	1.35	0.9952
15	14	0.35	0.0048
16	15	0.65	1.0048
17	13	1.10	0.0009
18	12	0.70	0.0003
19	12	0.70	0.0003
20	22	2.09	0.0079
21	19	0.17	0.0058
22	18	0.63	0.5227
23	19	0.37	0.4773
24	21	2.86	0.1219
25	20	1.91	0.3850
26	21	2.91	0.6150
27	19	0.96	0.5191
28	20	1.96	0.4809
29	15	2.35	0.0506
30	13	4.30	0.0898
31	19	2.99	0.0018
32	13	1.81	0.0006
33	12	2.66	0.0010
34	12	2.41	0.0003
35	21	1.22	0.0115
36	21	0.48	0.0001
37	19	2.10	0.0117
38	18	0.23	0.0000
39	24	2.19	0.4999
40	25	3.19	0.5001
41	18	1.94	0.0001
		MLR	**New-NFS**
MAR		1.6173	**0.2765**

Mean ARs assesses the validation results of estimation accuracy of the 41 projects in this study, whose results are shown in the Table 3.

A statistical test for comparing the two sets of MARs by model should be selected taking into account the assumptions of dependence and normality of data (Ross, 2004).

In this work, each of the two sets of ARs by model, $ARMLR_i$ and $ARNew\text{-}NFS_i$, is obtained from the corresponding project *i*; therefore $ARMLR_i,\ldots, ARMLR_n$, and $ARNew\text{-}NFS_i,\ldots, ARNew\text{-}NFS_n$, are dependent (also termed related or paired) samples. The above means that a procedure to test the differences between the two sets $ARMLR_i$ and $ARNew\text{-}NFS_i$ should be selected for determining whether the mean of the set of 41 differences is equal or not to zero with a difference statistically significant. Figure 3 shows a normal probability plot for the set of 41 differences for ARs.

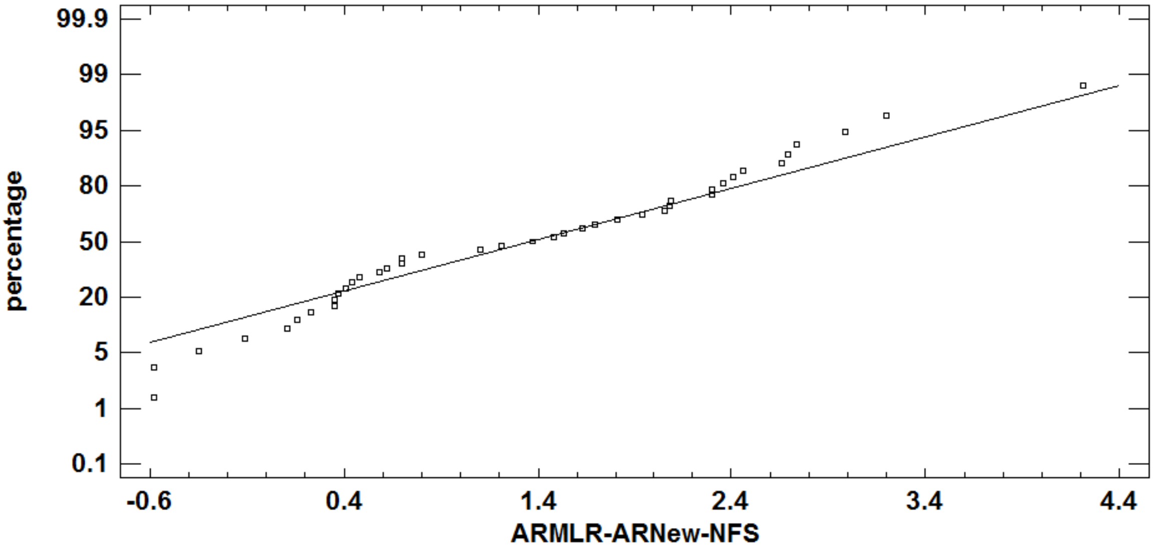


Figure 3. Normal probability plot.

The normality statistical tests of Chi-Squared, Shapiro-Wilk, Skewness, and Kurtosis were applied to the data from Figure 3, had *p*-values equal to 0.567, 0.290, 0.580 and 0.444, respectively. Since the smallest amongst the test is greater than 0.01, we cannot reject the idea that the data presented in Figure 3 comes from a normal distribution, with 99% of confidence.

Because the two sets of ARs by model are dependent and the set of their differences is normally distributed, the suitable statistical test for comparing their estimation accuracy is the paired *t*-test (Ross, 2004), which tests the null hypothesis that the mean $ARMLR_i$ – $ARNew\text{-}NFS_i$ equals 0.0, versus the alternative hypothesis that the mean $ARMLR_i$ – $ARNew\text{-}NFS_i$ is not equal to 0.0. After applying the paired *t*-test, the resulting *p*-value was equal to 3.26E-9, meaning that since the *p*-value for this test is less than 0.05, we can reject the null hypothesis at the 99.0% confidence level.

CONCLUSIONS AND FUTURE RESEARCH

This paper presented a research aimed at comparing the New-NFS and the MLR model to estimate software projects development time using as accuracy criteria ARs and a paired t-test.

The paper proposes a new approach NFS with four MFs for estimating of software projects development time. The major difference between our work and previous works is that New- NFS used ARs and a paired t-test as evaluation criteria.

The new-NFS with four MF presented the best (lowest) MMRE value in a previous work, as well as the lowest (best) MAR value in present study showing better results than the MLR model for both research works.

In order to achieve more accurate estimation, others MFs should also be used such as triangular, trapezoidal and Gaussian using the Takagi-Sugeno model with a linear output and with different numbers of MF. Also, another training algorithm as is Backpropagation could be used in order to obtain the parameters of the MF.

ACKNOWLEDGMENTS

Authors of this paper would like to thank Instituto Tecnológico de Colima, Universidad de Colima, Consejo Nacional de Ciencia y Tecnología (CONACYT), as well as Programa para el Desarrollo Profesional Docente

(PRODEP). In addition, we would like to express our gratitude to Ing. Saturnino Castro Reyes, Director of Instituto Tecnológico de Colima, for his valuable support obtained.

REFERENCES

J. Wen, S. Li, Z. Lin, Y. Hu, C. Huang, "Systematic literature review of machine learning based software development effort estimation models," Information and Software Technology, 54 (2012) 41–59.

M. Shepperd, S. MacDonell, "Evaluating prediction systems in software project estimation", Information and Software Technology, Volume 54, pp. 820–827, 2012. doi:10.1016/j.infsof.2011.12.008

C. López-Martín Predictive accuracy comparison between neural networks and statistical regression for development effort of software projects Appl. Soft Comput., 27 (2015), pp. 434–449. http://dx.doi.org/10.1016/j.asoc.2014.10.033

A. Ferreira-Santiago, López-Martín, C., & Yáñez-Márquez, C. (2015). Metaheuristic optimization of multivariate adaptive regression splines for predicting the schedule of software projects. Neural Computing and Applications, 1-12.

A. Abran, J. W. Moore; editors, P. Bourque, R. Dupuis, "Guide to the Software Engineering Body of Knowledge (SWEBOK)," IEEE Computer Society, Pp. 1–1, 2004.

Standishgroup.com, "The Standish Group ©," www.standishgroup.com, 2012.

R.A. De Araújo, S. Soares, L.I. Oliveira, "Hybrid morphological methodology for software development cost estimation," Expert Systems with Aplications, Pp. 6129-6139, 2012.

A. Bonneti, S. Bortot, M. Fedrizzi, R.A. Marques Pererira, A. Molinari, "Modelling group process and effort estimation in project management using the Choquet integral: An MCDM approach," Expert Systems with Aplications, 2012.

R.A. De Araújo, A.L.I. Oliveira, S. Soares, S. Meira, "An evolutionary morphological approach for software development cost estimation," Neural Networks, Pp. 285-291, 2012b.

R. LagerstrÖm, L. Marcks von Würtemberg, H. Holm, O. Luczak, "Identifying factors affecting software development cost and productivity," Software Quality Journal, Pp. 395-417, 2012.

K. Moløkken-Østvold, M. Jørgensen, "A review of surveys on software effort estimation," in: International Symposium on Empirical Software Engineering (ISESE 2003), IEEE Computer Society, Rome, Italy. Pp. 223–230, 2003.

R.A. De Araújo, A.L.I. de Oliveira, S. C.B. Soares, "A morphological-rank linear approach for software development cost estimation," In IEEE international conference on tools with artificial intelligence, IEEE, 2009.

R.A. De Araújo, A.L.I. de Oliveira, S.C.B. Soares, "A shift-invariant morphological system for software development cost estimation," Expert Systems with Applications, 38(4), 4162–4168, 2011.

D. Rombach, J. Münch, A. Ocampo, W.S. Humphrey, D. Burton, "Teaching disciplined software development," Journal Systems and Software, Elsevier, Pp. 747-763, 2008.

I. González-Carrasco, R. Colomo-Palacios, J.L. López-Cuadrado, F.J. García-Peñalvo, SEffEst: Effort estimation in software projects using fuzzy logic and neural networks International Journal of Computational Intelligence Systems, 5:4, (2012) 679-699.

M. Jørgensen, M. Shepperd, "A systematic review of software development cost estimation Studies," IEEE Transactions on Engineering Management 33(1), 2007.

B. Boehm, C. Abts, "Software Development Cost Estimation Approaches – A Survey," University of Southern California. Los Angeles, CA. 2000.

C. Lopez-Martin, J. Leboeuf- Pasquier, C. Yañez-Marquez, A. Gutierrez-Tornes, "Software Development Effort Estimation Using Fuzzy Logic: A Case Study," IEEE Proceedings of the Sixth Mexican International Conference on Computer Science (ENC'05), 2005, pp. 113-120.

S. Kar, S. Das, G.P. Kanti, "Applications of neuro fuzzy systems: A brief review and future outline," Applied Soft Computing 15 (2014) 243–259.

V. Marza, A. Seyyedi, L.F. Capretz, "Estimating Development Time of Software Projects Using a Neuro Fuzzy," Approach World Academy of Science, Engineering and Technology. Vol: 2 2008-10-27.

N. Garcia-Diaz, J. Garcia-Virgen, N. Farias-Mendoza, A. Verduzco-Ramirez, R. Martinez-Bonilla, E. Chavez-Valdez, & Soriano-Equigua, L. "Software development time estimation based on a new Neuro-fuzzy approach". In Information Systems and Technologies (CISTI), 2015 10th Iberian Conference on (pp. 1-7). IEEE.

P. Naur, B. Randell, "Software Engineering: Report on a conference sponsored by the NATO Science Committee," Garmisch, Germany, 1969.

F.P. Brooks, "Three Great Challenges for Half-Century-Old Computer Science," Journal of the ACM, Vol. 50, No. 1, Pp. 25-26, 2003.

B.A. Kitchenham, E. Mendes, G.H. Travassos, "Cross versus within-company cost estimation studies: a systematic review," IEEE Trans SoftwEng 33(5):316–329. 2007.

L.A. Zadeh, "Fuzzy Sets, Information and Control," Pp. 338–353, 1965.

Z. Zhiwei Xu, T.M. Khoshgoftaar, "Identification of fuzzy models of software cost estimation," Elsevier Fuzzy Sets and Systems, Pp.141-163, 2004.

L.A. Zadeh, "From computing with numbers to computing with words–from manipulation of measurements to manipulation of perceptions," Journal AMCS, Pp. 307-324, 2002.

J.P. Lewis, "Large Limits to Software Estimation," ACM Software Engineering Notes, Pp. 54–59, 2001.

E.H. Mamdani, “Advances in the linguistic synthesis of fuzzy controllers,” International Journal of Man-Machine Studies, Pp. 669–678, 1976.

T. Takagi, M. Sugeno, “Derivation of fuzzy control rules from human operator’s control actions,” In: Proc. the IFAC Symp. On Fuzzy Information, Knowledge Representation and Decision Analysis, Pp. 55–60. 1983.

T. Takagi, M. Sugeno, “Fuzzy Identification of Systems and its Application to Modelling and Control,” IEEE Transactions on Systems, Man and Cybernetics, Pp.116-132, 1985.

S. Haykin, “Neural networks, a comprehensive foundation,” Second Edition, Pearson. 1999.

M.T. Su, T.C. Ling, K.K. Phang, C.S. Liew, P.Y. Man, “Enhanced Software Development Effort and Cost Estimation Using Fuzzy Logic Model,” Malaysian Journal of Computer Science, Vol. 20, No. 2, Pp. 199-207, 2007.

K.K. Aggarwal, Y. Singh, P. Chandra, M. Puri, “Sensitivity Analysis of Fuzzy and Neural Network Models,” ACM SIGSOFT Software Engineering Notes, Vol. 30, Issue 4, Pp. 1-4, 2005.

X. Huang, D. Ho, J. Ren, L.F. Capretz, “Improving the COCOMO model using a neuro-fuzzy approach,” Applied Soft Computing, Vol.7, Issue 1, Pp. 29-40, 2007.

D. Nauck, “A Fuzzy Perceptron as a Generic Model for Neuro-Fuzzy Approaches,” In Proceedings of Fuzzy-Systeme’94, 2nd GI-Workshop, Munich, 1994.

D. Nauck, F. Klawonn, R. Krus, “Foundations of Neuro-Fuzzy Systems,” Wiley, Chichester, 1997.

M.O. Saliu, “Adaptive Fuzzy Logic Based Framework for Software Development Effort Prediction,” A Thesis Presented to the DEANSHIP OF GRADUATE STUDIES, King Fahd University of Petroleum & Minerals Dhahran. 2003.

J. Jantzen, “Neuro-fuzzy modelling,” Report no 98-H-874. 1998.

S. Mitra, Y. Hayashi, “Neuro-Fuzzy Rule Generation: Survey in Soft Computing Framework,” IEEE Transactions on Neural Networks, Vol.11, No.3, Pp.748-768, 2000.

A. Abraham, “Adaptation of Fuzzy Inference System Using Neural Learning,” Springer-Verlag Berlin Heidelberg, Pp. 59-83, 2005.

W. Humphrey, “A Discipline for Software Engineering,” Addison Wesley, 1995.

C. Lopez-Martin, C. Yañez-Marquez, A. Gutierrez-Tornes, “Predictive accuracy comparison of fuzzy models for software development effort of small programs,” The journal of systems and software, Pp. 949-960, 2008.

M. Wei, B. Bai, A.H. Sung, Q. Liu, J. Wang, M.E. Cather, “Predicting injection profiles using ANFIS,” Information Sciences 177 (2007) 4445–4461.

S.M. Ross, 2004. “Introduction to Probability and Statistics for Engineers and Scientists”, Third Edition, Elsevier Press.