Analytic Study of Fuzzy-based model for Software Cost Estimation

John Chibuike Nwaiwu
Computer Science Department,
Federal University of Technology Akure, Nigeria
+2348030845700
johnnwaiwu@futa.edu.ng [1]

Samuel Adebayo Oluwadare
Computer Science Department,
Federal University of Technology Akure, Nigeria
+2348034034202
saoluwadare@futa.edu.ng

ABSTRACT
The need for successful software projects has been a major area of discourse amongst researchers and software developers in academia and software industry respectively. Failure of software projects has been tied to flawed estimation at the early stages of software development life cycle. Recently, soft computing techniques such as Fuzzy logic models has been seen as an alternative to handle uncertainties and vagueness of input parameters to the early software estimation models. In order to analyze the various conditions which affect estimation accuracy of fuzzy-based models, a sample of 93 COCOMO NASA projects was used to develop two groups of fuzzy models. One was the controlled group while the other was the experimental group varying in conditions of model structure, linguistic variables, parameters of input and output variables. A comparative analysis of the Mean Magnitude of Relative Error (MMRE) and Prediction accuracy Pred(*l*) evaluation criteria for the models was made and findings recorded. Results from the experiments show that the performance of a fuzzy-based software cost estimation model utilizing Takagi-Sugeno inference, Gaussian/Sigmoid membership function with more number of input variables and linguistics variables is more efficient.

CCS Concepts
• **Software and its engineering →Software system structures →Software system models →Model-driven software engineering** • **Computing methodology→Artificial Intelligence Knowledge representation and reasoning→Vagueness and fuzzy logic**

Keywords
Fuzzy model, Software cost estimation (SCE), membership function (MF), Fuzzy Inference System, performance.

1. INTRODUCTION
Software engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software products [1]. A good software product passes through a process known as Software Development Life Cycle (SDLC) which ensures that a structured approach is followed from conception through development to maintenance and evaluation. This framework enables the planning of resources prior to development, helps understand the entire process and assists management to monitor and track the progress of software development. A key phase in the SDLC has suffered neglect from software practitioners which has always resulted in occurrence of software crisis. This phase is the feasibility study phase which is responsible for anticipation of future scenarios of software development.

In a recent update of Standish Group study (2012) conducted for ComputerWorld, Standish examined 3,555 IT projects between 2003 and 2012 that had labour costs of at least $10 million and found that only 6.4% of them were successful [36]. Notably questions have been raised in literature as to 'why most software projects fail' [2, 3, 4, 5, 6]. Flawed and inaccurate estimation of needed resources during the early stages of software development has been identified as one of the common factors why software projects fail [3, 5].

Estimation is a process which uses prediction systems and intuition for resource and cost planning. It is controlled by cost realism, which does not always insist on exactness but lays equal emphasis on logic as much on the mathematical form of the prediction system [7].

1.1 Software Cost Estimation (SCE)
Software Cost Estimation (SCE) has been identified as one of the most crucial activities in managing software projects required during the early stages of software development life cycle [8, 9, 10, 11]. It is the process of estimating the cost, work-effort, schedule and time required to develop and control a software project. Software cost estimation aids in facilitating contract negotiations, generating project proposal. However, the process of estimation is uncertain in nature as it largely depends upon some attributes that are quite unclear during the early stages of development [12].

The entire process of software cost estimation is generally classified into two broad categories [13]:

a. Expert Judgement: The technique aims at deriving estimates based on the experience of experts on similar projects. This technique is intuitive.
b. Model-based technique: This technique is further subdivided into
    i. Models based on statistic: These models are based on linear regression. They are also known as algorithmic models. Early linear regression estimation models include Halstead, Bailey-Basili, Doty, Walston Felix and Constructive Cost model (COCOMO).
    ii. Models based on computational intelligence: These models solve nonlinear, time varying, correlated discontinuous complex probabilistic real-world problems. They provide feasible way to obtain either optimal

CoRI'16, Sept 7–9, 2016, Ibadan, Nigeria.

or suboptimal solutions [14]. Recent estimation models based on computational intelligence include fuzzy logic (FL), artificial neural network (ANN), particle swam optimization (PSO), genetic algorithm (GA) and genetic programming models.

1.2 Software measurement

Measures provide a quantitative indication of amount, dimension, capacity or size of a given attribute of a product. Metrics are a quantitative measure of the degree to which a system, component or process possesses a given attribute of a product. Measurement is achieved as the result of the collection of a data point or more data points. Software metrics can be defined as the continuous application of measurement based techniques to software development process and its products to supply meaningful and timely management information, together with those techniques to improve that process and its products [1]. However, metrics ensure that we achieve a final product of high quality and productivity. Metrics are categorized into the following:

a. Product metrics: They describe characteristics of the product such as size, complexity, design features, performance, efficiency, reliability and portability.
b. Process metrics: They describe the effectiveness and quality of the processes that produce the product. These include effort required in the process, time to produce the product, effectiveness of defect removal during development and maturity of process.
c. Project metrics: Project metrics describe the characteristics of the project and its execution. They include; number of software developers, staffing pattern over life-cycle of the software, cost and schedule.

1.3 Performance Evaluation

Performance evaluation of estimates had always been a critical stage in software development. Thus, proper and accurate the evaluation criteria of software projects become, the better the software cost estimation models. The mostly used software cost estimation (SCE) criteria are listed in [15] but for the purpose of this study we will be making use of the first three because of their continued relevance in most literature.

a. Magnitude of Relative Error (MRE): An MRE criterion is value error estimated for each of the projects compared to the actual.
b. Mean Magnitude of Relative Error (MMRE): MMRE is used as the criteria error of the mean value of the project.
c. Percentage Relative Error Deviation (PRED(*l*)): PRED(*l*) criterion is used in order to estimate the accuracy of the models.

MRE is evaluated as follows (for the i-th observation)

$$MRE_i = \frac{|\text{Actual Effort}_i - \text{Estimated Effort}_i|}{\text{Actual Effort}_i} \quad (1)$$

MMRE is evaluated as follows (for n observations)

$$MMRE = \frac{1}{n} \sum_{i=1}^{n} MRE_i \quad (2)$$

Pred(*l*) is calculated as follows for n observations

$$Pred(l) = \frac{1}{n} * \sum_{i=1}^{n} \begin{cases} 1, & if\ MRE \leq l \\ 0, & otherwise \end{cases} \quad (3)$$

where n is the total number of projects and *l* is the number of projects with MRE less than or equal to *l*. However, a model estimate with low MMRE has a better performance index than one with high MMRE. A model with high Pred(*l*) has a better performance than one with a low Pred(*l*). A model with high VAF is better than one with a low VAF. Alternatively, a model showing low VARE has more accuracy than one with high VARE. Likewise a model having low MARE is preferable to one with high MARE.

1.4 Related works

Various fuzzy-based models for software cost estimation have been proposed by researchers in [9, 10, 11, 12, 13, 15, 16, 17, 18, 19, 20, 21]. The issue of the compatibility of COCOMO with the fuzzy logic showed that the accuracy of estimation is very sensitive to the changes in inputs. Results showed that the 'fuzzy' intermediate COCOMO 81 model tolerates imprecision in its inputs (cost drivers) and consequently generates more gradual outputs (cost) [37]. In the paper [38], it was reported that fuzzy logic based prediction systems could produce further better estimates provided that various parameters and factors pertaining to fuzzy logic are carefully set. The paper demonstrated that the prediction accuracy of a fuzzy logic based effort prediction system is highly dependent on the system architecture, the corresponding parameters, and the training algorithms. A soft computing approach (fuzzy) for software cost estimation was presented in [39]. The paper described an enhanced soft computing model for the estimation of software cost and time estimation. Evaluation of the model was based on MMRE and PRED(25) criteria and validated on NASA 93 projects dataset. The experimental results show that the proposed software effort estimation model shows better estimation accuracy than the COCOMO model. Also, an output with more terms or fuzzy sets provided a better performance due to the high granularity demanded from the results**.** Work by Jorgensend et al in [22] did a systematic review of software development cost studies. Their work reviewed 304 software cost estimation papers and classified the papers according to estimation approach, research approach, research topic, data set and study context. Performance evaluation of software effort estimation [23] was done using Fuzzy analogy based on complexity. Comparative analysis of fuzzy models was done based on membership functions in [24] and fuzzy inference engine type in [25] respectively. This paper seeks to extend the work in [8] by analysing fuzzy-based software cost estimation models in areas of inference structure, membership function, input variables, linguistic variables and defuzzification techniques in order to evaluate more factors that affect estimation accuracy. In order to analyze the various conditions which affect estimation accuracy of fuzzy-based models, a sample of 93 COCOMO NASA projects is used to develop two groups of fuzzy models. One is the controlled group while the other is the experimental group varying in conditions of model structure, linguistic variables, parameters of input and output variables. A comparative analysis of the Mean Magnitude of Relative Error (MMRE) and Prediction accuracy Pred(*l*) evaluation criteria for the models is carried out and findings recorded.

2. SOFTWARE COST ESTIMATION TECHNIQUES

2.1 Traditional estimation models

Early algorithmic models which include COCOMO [26], Bailey-Basili [27] Doty, Halstead, Walston [28] models respectively, has

been developed for the estimation of cost and effort of software projects. Boehm [26] explored software engineering from an economic point of view, thus coming up with a Constructive Cost model. Software Life-Cycle Management (SLIM) developed by Putnam [29] accepts number of lines of code as a major input. Albrecht's Function Points [30] measures the amount of functionality in a system as described by a specification.

2.1.1 COCOMO

The COCOMO I [31] model is a regression-based software cost estimation model, developed by Boehm and the most cited of all traditional cost estimation models [27, 28, 29, 30]. COCOMO II was developed to improve on the limitation of COCOMO I. The model equation for estimating effort in the COCOMO II model is shown in Equation (4) below. The model includes several software attributes such as: 17 Effort Multipliers (EMs), 5 Scale Factors (SFs), Software Size (SZ), and Effort estimation that are used in the Post Architecture Model.

$$E = a * (SZ)^{b} + 0.01 * \sum_{j=1}^{5} SF_j * \prod_{i=1}^{17} EM_i \quad (4)$$

where a = 2.94 and b = 0.9. Other algorithmic models include:
Halstead Model Equation

$$Effort = 5.2 * (KLOC)^{1.5} \quad (5)$$

Bailey-Basili Model Equation

$$Effort = 5.5 + 0.73 * (KLOC)^{1.16} \quad (6)$$

Doty Model Equation

$$Effort = 5.288 * (KLOC)^{1.04} \quad (7)$$

Algorithmic models posses the following shortcomings;

i. They make assumptions about the form of the prediction function.
ii. They need to be adjusted or calibrated to local circumstances.
iii. They are practically model-intractable.
iv. They can't handle the vagueness and uncertainty present in the input parameters to the model.
v. The output from this set of models is almost crisp values which often lead to overconfidence in accuracy and precision of estimate.
vi. Software cost estimation is a complex non-linear stochastic problem. These models can't model non-linear and complex problems.

2.2 Fuzzy Logic

Fuzzy logic is an extension of multi-valued logic that models effectively how the human brain reasons. A systematic use of fuzzy logic in software cost estimation is a necessity when the available information is imprecise, incomplete or not totally reliable [32].
A fuzzy model is a mapping between input and output spaces described using conditional propositions and inference engine. Fuzzy systems have two distinguishing features.

a. They implore non-linear mapping between input and output vectors and can be accurately described using mathematical formulae.
b. They are also knowledge-based driven expressed in linguistic terms. These are known as intuitive systems. However, humans can make inferences in an imprecise environment and that is where fuzzy logic comes handy [40].

2.2.1 Fuzzy sets and linguistic variables

The theory of fuzzy sets introduces a paradigm which extends the concept of the crisp set, allowing objects to partially belong to a set [33]. A fuzzy set is given below

$$S = \{(x, \mu_S(x)) \; x \in U, \mu_S(x): U \rightarrow [0, 1] \quad (8)$$

The space U represents the universe of discourse and the function $\mu_S(x)$ is called the membership function.
A linguistic variable can be defined as a variable that takes its value as terms defined by words in natural language. These terms are described by fuzzy sets defined in the universe of discourse in which the variable is defined. The number of linguistic variable was among the conditions that were varied during the experiments while others remain constant to analyze the effect on performance of estimate.

2.2.2 Fuzzy inference system (FIS)

Fuzzy Inference System (FIS) is based on the generalization of the classical inference rules to fuzzy logic. There are two basic FIS [33] namely;

a. Mamdani fuzzy inference system
It is formed by the following four components: A fuzzy rule base formed by a set of logical implications having the form described in Rule 1.

IF X_1 is $A_{j,1}$ and ... and X_m is $A_{j,m}$ THEN Y is B_j(Rule 1)

where the propositions X_1 is $A_{j,1}$,..., X_m is $A_{j,m}$; Y is B_j are all defined in the fuzzy sets $A_{j,1}$,..., $A_{j,m}$, B_j A fuzzy inference engine is defined using the sup-star operation. A fuzzifier operator and defuzzifier operator completes the system.

b. Takagi-Sugeno fuzzy inference system
It is formed by the following components: A fuzzy rule base formed by a set of logical implications having the following form:

IF f(X_1 is $A_{j,1}$,...,X_m is $A_{j,m}$) then y = gj(x^*_1,...,x^*_n)
(Rule 2)

where x^*_1,...,x^*_n are crisp values on the input universes, X_1 is $A_{j,1}$,...,X_m is $A_{j,m}$ are fuzzy propositions defined on the same input universes by the fuzzy sets $A_{j,1}$,...,$A_{j,m}$, f(.) is a logical function, y is a crisp value on output universe and g(.) is a crisp function that implies the value of y when the inputs x^*_1,...,x^*_n satisfy the premise. An algorithm of reasoning which is a modified version of Mamdani FIS completes the system.

In summary, the Tagaki-Sugeno FIS method divides the input spaces in fuzzy subspaces and next builds a relation between input variables into each subspace. In the experiments, two groups of fuzzy models were used; one generated using the Mamdani and the other using the Takagi-Sugeno FIS type.

2.2.3 Fuzzy membership functions

The only condition a membership function must really satisfy is that it must vary between 0 and 1. The function can be an arbitrary curve whose shape can be defined as a function that suits from the point of view of speed, efficiency and ease of use. The membership function maps an element, say X, to a membership value between 0 and 1. The eleven membership functions in fuzzy logic are derived from several basic functions namely;

a. piece-wise linear functions
b. Gaussian distribution function

c. sigmoid curve
d. quadratic and cubic polynomial curves

These membership functions (MFs) include: trapezoid (trapmf), triangular (trimf), Gaussian (gaussmf), generalized bell (gbellmf), combination of Gaussian (gauss2mf), sigmoid (sigmf), pi-shaped (pimf), product of sigmoid (psigmf), difference between two sigmoid (dsigmf), s-shape (smf and z-shape (zmf). In this study, we carried out experiments using five membership functions (namely triangular, trapezoid, Gaussian, generalized bell and product of sigmoid) while keeping all other conditions constant.

2.2.4 Defuzzification

It is the mapping from a fuzzy set (aggregate output fuzzy set), say S, defined on the universe of discourse, say U, to a crisp value x*ЄU. The defuzzifier's main function is to ascertain an object x* that best represents the fuzzy set S. The aggregate of a fuzzy set includes a range of output values which must be defuzzified in order to achieve a single output value from the set. Five defuzzification techniques available include: smallest of maximum, largest of maximum, middle of maximum (the average of the maximum value of the output set), centroid and bisector.

3. DESIGN OF EXPERIMENTS

3.1 Generation of fuzzy model

The dataset used in this work was sourced from [34] which consist of 93 NASA projects. This dataset consists of one kilo lines of code attribute, fourteen COCOMO I effort drivers and one actual effort (in man-months) attribute. In the experiments, the schedule constraint effort driver was excluded from the fuzzy models because it showed a U-shaped correlation to actual effort. In order to generate the fuzzy model from the data available, the following steps were taken [35].

a. Select a Fuzzy Inference System (FIS)
b. Define the input and output variables' mode.
c. Define the linguistic variables and values.
d. Set the type of the membership functions for input variables and output variable.
e. The data is now translated into a set of IF-THEN rules written in the fuzzy rule editor.
f. A certain model structure is created and parameters of input and output variables can be tuned to get the desired output.

Table 1 below shows the input variables to the fuzzy model used in this study.

Table 1: List of Effort drivers and description

S/N	Effort drivers	Description
1	RELY	Required Software Reliability
2	DATA	Database size
3	CPLX	Process Complexity
4	TIME	Time Constraint for CPU
5	STOR	Main memory constraint
6	VIRT	Machine volatility
7	TURN	Turnaround time
8	ACAP	Analyst Capability
9	AEXP	Application experience
10	PCAP	Programmers capability
11	VEXP	Virtual machine experience
12	LEXP	Language experience
13	MODP	Modern programming practice
14	TOOL	Use of software tools

The various experimental groups of models in this study are described below.

3.2 Experimental group with varying fuzzy inference system

The controlled conditions for experiment 1 include: input and output variable, linguistic variable and value, membership function and defuzzification technique respectively. This experimental group utilized triangular membership, five linguistic variables (very low, low, nominal, high, very high), centroid defuzzification technique and fourteen effort drivers including Kilo line of code size (KLOC). The varied condition is in inference type as shown below.

a. Model with Mamdani inference system (Experiment model 1a)
b. Model with Tagaki-Sugeno inference system (Experiment model 1b)

3.3 Experimental group with varying defuzzification technique

The controlled conditions for experiment 2 include number of input variables (fourteen effort drivers including Kilo line of code size (KLOC)), membership function (triangular), inference type (Mamdani inference) and five linguistic variables (very low, low, nominal, high, very high) respectively. The varied condition is in the defuzzification type as shown below.

a. Model with centroid defuzzification technique (Experiment model 2a)
b. Model with bisector defuzzification technique (Experiment model 2b)
c. Model with Middle of Maximum (MOM) defuzzification technique (Experiment model 2c)
d. Model with Largest of Maximum (LOM) defuzzification technique (Experiment model 2d)
e. Model with Smallest of Maximum (SOM) defuzzification technique (Experiment model 2e)

3.4 Experimental group with varying membership functions

The defuzzification technique implored here is the middle of maximum (MOM). Membership function is the varied condition in this setup as described below.

a. Model with triangular membership function (Experiment model 3a)
b. Model with trapezoid membership function (Experiment model 3b)
c. Model with Gaussian membership function (Experiment model 3c)
d. Model with generalized bell membership function (Experiment model 3d)
e. Model with sigmoid membership function (Experiment model 3e)

3.5 Experimental group with varying linguistic variables

In this setup, the controlled conditions for experiment 4 are product of sigmoid membership function, number of input variables (fourteen effort drivers including Kilo line of code size (KLOC)), inference type (Mamdani inference) and defuzzification technique (middle of maximum) respectively. The varied condition is linguistic variable. For experimental model 4a, five linguistic variables (very low, low, nominal, high, very high) are

used while three linguistic variables (low, nominal, high) are used in experimental model 4b respectively.

a. Model with increased linguistic variables (Experiment model 4a)
b. Model with reduced linguistic variables (Experiment model 4b)

3.6 Experimental group with varying number of input variables

All other conditions remained the same as in experiment 4a above except for the number of input variables to the model. For experiment 5b, 'main memory constraint' and 'time constraint for CPU' input variables were merged together prior to fuzzification because both had the same correlation coefficient respectively with effort. The same also was done for 'analyst capability' and 'programmer capability'. The correlation coefficient between effort and these named drivers was approximately 0.35. Also 'required software reliability' and 'process complexity' were combined together because both had approximately same correlation (0.2) to development effort (output). Thus, the number of input variables in experiment model 5b reduced to twelve.

a. Model with increased number of input variables (Experiment model 5a)
b. Model with reduced number input variables (Experiment model 5b)

3.7 Evaluation criteria

For this study, we made use of three evaluation criteria due to their widespread relevance in most related literature. They include

a) Magnitude of Relative Error (MRE)
b) Mean Magnitude of Relative Error (MMRE)
c) Prediction accuracy criteria (PRED)

There expressions are explained in equations 1 to 3.

4. EXPERIMENTAL RESULTS AND ANALYSIS

Below shows the results from the validation of the experimental models using one third of the project data.

Table 2: Estimated efforts (in man-months) of Experiment 1 models

Proj. No.	Size in KLOC	Actual Effort	Model 1a	Model 1b
8	66.6	352.8	192	0.5
28	48.5	239	192	250
38	90	444	198	250
10	20	72	192	250
40	16	114	192	0.5
50	78	571.4	473	417
60	350	720	802	750
70	151	432	4130	465
90	233	8211	5920	5250
14	100	215	203	250
54	219	2120	2580	2500
84	24	430	470	500
16	100	360	203	250
46	423	2400	2590	2500
56	227	1181	1580	1500
66	150	882	817	750
76	162	756	4130	750
17	150	324	229	250
87	70	1645.9	1580	1500
23	29.5	120	192	250
43	282.1	1368	1590	1500
53	101	750	802	750
63	90	162	198	250
25	38	210	192	250
65	137	636	808	750
85	165	4178.2	5140	4280
32	35.5	192	192	250
59	980	4560	4080	4000
79	60	409	470	500
81	32	1350	1580	1500
68	240	192	198	250

The MMRE of experimental model 1a and 1b were 0.4958 and 0.082 while the prediction accuracy, Pred (30) were 67.74% and 70.96% respectively.

Table 3: Estimated efforts (in man-months) of Experiment 2 models

Prj No	Actual Effort	Model 2a	Model 2b	Model 2c	Model 2d	Model 2e
8	352.8	192	165	124	248	0
28	239	192	165	124	248	0
38	444	198	165	124	248	0
10	72	192	165	124	248	0
40	114	192	165	124	248	0
50	571.4	473	413	124	248	0
60	720	802	825	784	825	743
70	432	4130	4130	4130	4130	4130
90	8211	5920	6190	7080	8250	3960
14	215	203	165	124	248	0
54	2120	2580	2560	2520	2560	2480
84	430	470	413	124	248	0
16	360	203	165	124	248	0
46	2400	2590	2560	2520	2810	2230
56	1181	1580	1570	1490	1490	1490
66	882	817	825	825	1070	578
76	756	4130	4130	4130	4130	4130
17	324	229	248	206	413	0
87	1645.9	1580	1570	1490	1490	1490
23	120	192	165	124	248	0
43	1368	1590	1570	1570	1820	1320
53	750	802	825	784	825	743
63	162	198	165	124	248	0
25	210	192	165	124	248	0
65	636	808	825	784	908	660
85	4178.2	5140	5280	5830	8250	3300
32	192	192	165	124	248	0
59	4560	4080	4040	4040	4040	4040
79	409	470	413	124	248	0
81	1350	1580	1570	1490	1490	1490
68	192	198	165	124	248	0

The MMRE and Pred(30) of experimental models 2a, 2b, 2c, 2d, 2e respectively were 0.4958, 0.4233, 0.2623, 0.6377, 0.1215 and 67.74%, 67.74%, 48.38%, 54.83%, 38.70% respectively.

Table 4: Estimated efforts (in man-months) of Experiment 3 models

Proj No.	Actual Effort	Model 3a	Model 3b	Model 3c	Model 3d	Model 3e
8	352.8	124	82.5	371	289	289
28	239	124	124	248	248	289
38	444	124	124	248	248	248
10	72	124	82.5	248	248	248
40	114	124	124	248	248	248
50	571.4	124	424	495	248	495
60	720	784	825	784	743	701
70	432	4130	4130	784	701	578
90	8211	7080	6030	5900	5240	5690
14	215	124	124	289	248	248
54	2120	2520	2640	2520	2520	2430
84	430	124	513	495	495	454
16	360	124	124	289	248	248
46	2400	2520	2640	2520	2520	2430
56	1181	1490	1650	1490	1490	1400
66	882	825	825	784	743	701
76	756	4130	4130	784	743	743
17	324	206	206	289	248	330
87	1646	1490	1650	1490	1490	1400
23	120	124	82.5	248	248	248
43	1368	1570	1650	1530	1490	1400
53	750	784	825	743	743	743
63	162	124	124	289	248	248
25	210	124	124	248	248	248
65	636	784	825	743	743	743
85	4179	5830	5820	5550	5510	5690
32	192	124	124	248	248	289
59	4560	4040	4170	4000	4000	3960
79	409	124	513	495	495	495
81	1350	1490	1650	1490	1490	1400
68	192	124	82.5	289	248	248

From Table 4, the MMRE values for model 3a, 3b, 3c, 3d, 3e were 0.2623, 0.3176, 0.2549, 0.1894 and 0.2 respectively. The prediction accuracy were 48.38%, 54.83%, 70.96%, 67.74% and 67.74% respectively.

Table 5: Estimated efforts (in man-months) of Experiment 4 models

Proj. No.	Size in KLOC	Actual Effort	Model 4a	Model 4b
8	66.6	352.8	289	495
28	48.5	239	289	578
38	90	444	248	495
10	20	72	248	495
40	16	114	248	578
50	78	571.4	495	578
60	350	720	701	495
70	151	432	578	578
90	233	8211	5690	4620
14	100	215	248	495
54	219	2120	2430	1530
84	24	430	454	578
16	100	360	248	495
46	423	2400	2430	2020
56	227	1181	1400	2020
66	150	882	701	578
76	162	756	743	578
17	150	324	330	536
87	70	1645.9	1400	2020
23	29.5	120	248	495
43	282.1	1368	1400	1980
53	101	750	743	578
63	90	162	248	495
25	38	210	248	495
65	137	636	743	578
85	165	4178.2	5690	4620
32	35.5	192	289	578
59	980	4560	3960	6190
79	60	409	495	578
81	32	1350	1400	2020
68	240	192	248	495

Experimental model 4a and 4b had MMRE values of 0.20 and 0.8291 respectively while their prediction accuracy, Pred (30), were 67.74% and 29.03% respectively.

Table 6: Estimated efforts (in man-months) of Experiment 5 models

Proj. No.	Size in KLOC	Actual Effort	Model 5a	Model 5b
8	66.6	352.8	289	289
28	48.5	239	289	289
38	90	444	248	248
10	20	72	248	248
40	16	114	248	248
50	78	571.4	495	495
60	350	720	701	701
70	151	432	578	578
90	233	8211	5690	5570
14	100	215	248	248
54	219	2120	2430	2430
84	24	430	454	454
16	100	360	248	248
46	423	2400	2430	2430
56	227	1181	1400	1400
66	150	882	701	701
76	162	756	743	743
17	150	324	330	330
87	70	1645.9	1400	1400
23	29.5	120	248	248
43	282.1	1368	1400	1400
53	101	750	743	743
63	90	162	248	248
25	38	210	248	248
65	137	636	743	743
85	165	4178.2	5690	4800
32	35.5	192	289	289
59	980	4560	3960	3920
79	60	409	495	743
81	32	1350	1400	1400
68	240	192	248	248

Models 5a and 5b had MMRE values of 0.2 and 0.2119 respectively. Their prediction accuracy were 67.74% and 64.51% respectively.

4.1 Performance Evaluation

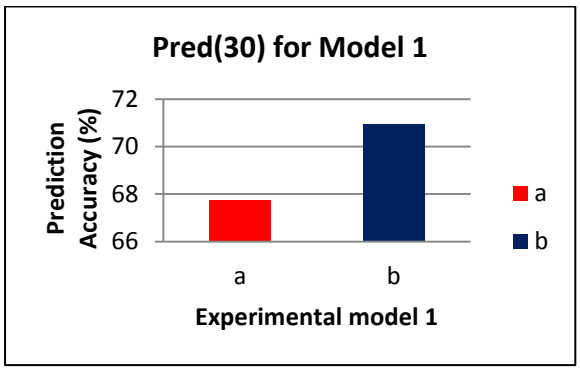

Figure 1: Prediction accuracy for model 1a and 1b

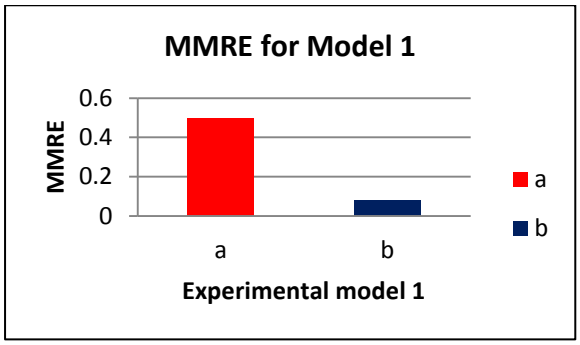

Figure 2: Mean magnitude relative error for models 1a and 1b

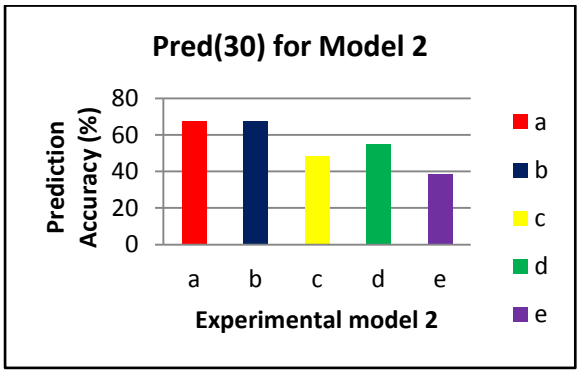

Figure 3: Prediction accuracy for models 2a, 2b, 2c, 2d and 2e

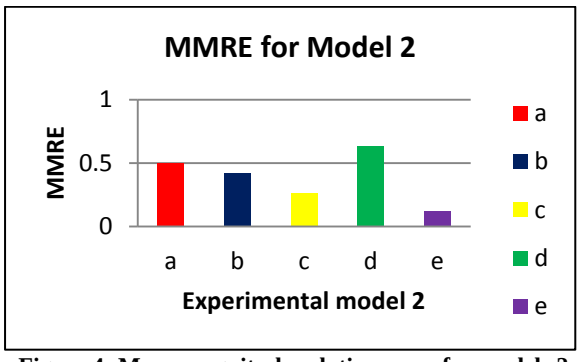

Figure 4: Mean magnitude relative error for models 2a, 2b, 2c, 2d and 2e.

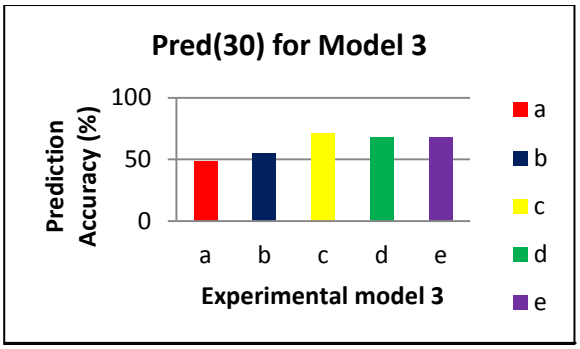

Figure 5: Prediction accuracy for models 3a, 3b, 3c, 3d and 3e

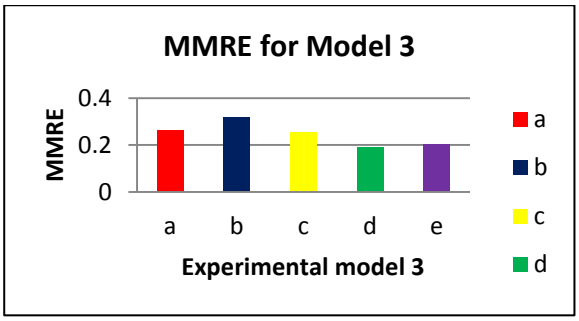

Figure 6: Mean magnitude relative error for models 3a, 3b, 3c, 3d and 3e

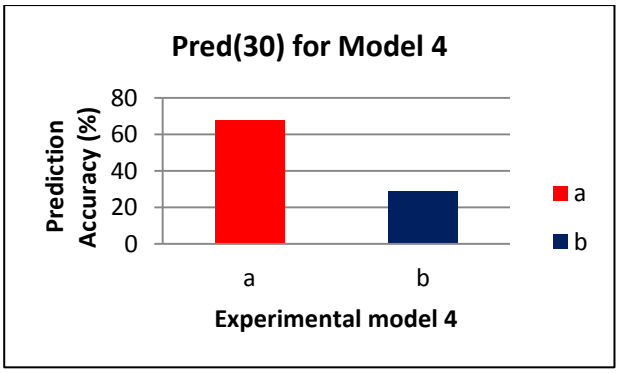

Figure 7: Prediction accuracy for models 4a and 4b

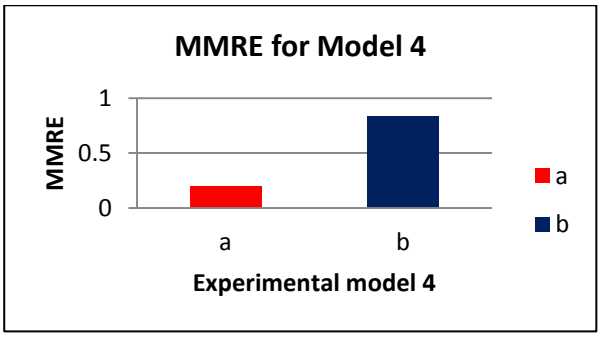

Figure 8: Mean magnitude relative error for models 4a and 4b

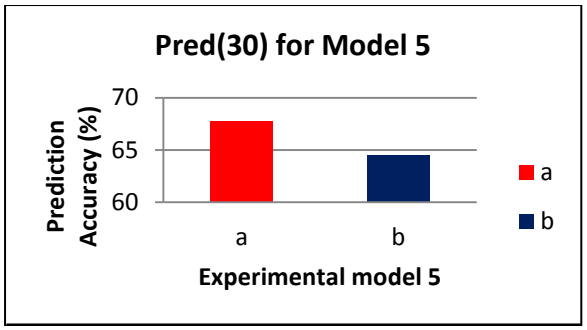

Figure 9: Prediction accuracy for models 5a and 5b

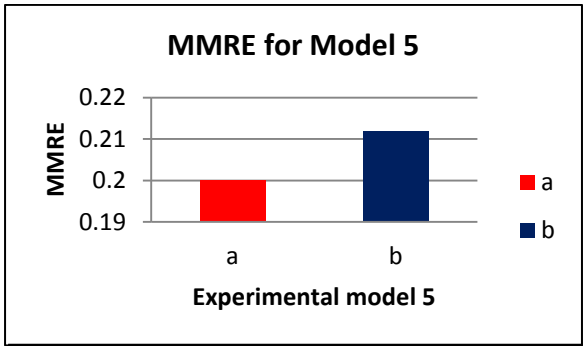

Figure 10: Mean magnitude relative error for models 5a and 5b

From figures 1 and 2, it is observed that in experiment 1 models, inference type had a great effect on performance of the fuzzy model. The model with Takagi-Sugeno inference performed better than the model with Mamdani inference in terms of mean magnitude relative error (0.0820 against 0.4958) and prediction accuracy (70.96% against 67.74%) respectively. Figures 3 and 4 shows that, no one choice of defuzzification technique enjoyed absolute preference in both evaluation criteria. While the models with centroid (67.74%) and bisector (67.74%) defuzzification technique show better predictive accuracy than the models with largest of maximum (54.83%), middle of maximum (48.38%), and smallest of maximum (38.7%) respectively. Alternatively, the models with smallest of maximum (0.1215) and middle of maximum (0.2623) defuzzification technique shows better results of mean magnitude relative error than models with bisector (0.4233), centroid (0.4958) and largest of maximum (0.6377) defuzzification technique respectively.

In terms of prediction accuracy from figure 5, the model with Gaussian membership function (70.96%) shows promising results than models with generalized bell (67.74%), product of sigmoid (67.74%), trapezoid (54.83%) and triangular (48.38%) membership functions respectively. From figure 6, it could be observed that generalized bell membership function shows better results of MMRE (0.1894) than product of sigmoid (0.2), Gaussian (0.2549), triangular (0.2623) and trapezoid (0.3176) membership functions respectively.

Another important observation from experiment 4 is that the number of linguistic variable used in a fuzzy model has effect on its performance. As seen in figures 7 and 8, the model with more linguistic variables (five) performed better in terms of MMRE (0.2) and prediction accuracy (67.74%) than the one with lesser (three) linguistic variable of MMRE of 0.8291 and prediction accuracy of 29.03%.

Also from figures 9 and 10, the model with increased number of input variables to the fuzzy model (fifteen) outperformed the one with lesser input variables (twelve) in terms of lower MMRE and better prediction accuracy respectively.

5. CONCLUSION AND FUTURE RESEARCH

This paper explored the various factors that enhance high performance of fuzzy-based models for software development cost estimation. From the comparative analysis of fuzzy models used in the experiments, it could be observed that using Tagaki-Sugeno inference type is preferred than Mamdani inference type for a fuzzy-based software cost estimation model. Also, increasing the number of linguistic variables and input variables to the fuzzy model has positive results on performance. Generalized bell, sigmoid and Gaussian membership functions performs better than triangular and trapezoid membership functions for this area of software engineering application.

The choice of a suitable defuzzification technique depends on whether smaller or larger projects are being modelled. While centroid and bisector defuzzification technique would favour modelling of medium software projects effort, smallest of maximum and largest of maximum defuzzification technique would show favourable results for small projects and large projects respectively. It is unlikely to achieve a fuzzy model which can give 100% Pred(30) but by suitably adjusting the values of the parameters in fuzzy inference system (FIS), estimated effort could be optimized.

Future research will involve a comparative study of fuzzy models using data from in-house software projects and also investigating the performance of fuzzy models with customized membership functions.

6. ACKNOWLEDGEMENTS

Special thanks to lecturers and postgraduate students of Computer Science department, Federal University of Technology Akure, Nigeria for their various inputs and suggestions to the success of this work, most especially Dr. (Mrs.) B. A. Ojokoh.

7. REFERENCES

[1] Agarwal B. B., Tayal S. P., Gupta M., 2009. Software engineering and testing. Jones and Bartlett publishers, ISBN 978-0-7637-8302-0.

[2] Dorsey P., 2000. Top 10 Reasons Why Systems Projects Fail. Dulcian Inc, www.ksg.havard.edu [Accessed November 13, 2015].

[3] Charette R. N., 2005. Why Software Fails. IEEE Spectrum, http://spectrum.ieee.org/computing/software/why-software-fails [Accessed November 13, 2015].

[4] Agbons O-L. A., 2013. Combating and preventing failed Projects in Nigeria. Project Management in Nigeria; Challenges and Prospects, 3rd International Project Management Professionals Conference, Lagos, Nigeria. Pp 6-7.

[5] Putnam D., and Putnam-Majarian C. T., 2015. The Most Common Reasons Why Software Projects Fails. InfoQ Article July 2015. http://www.infoq.com/articles/software-failure-reasons [Accessed October 30, 2015].

[6] Calleam Consulting Ltd., 2015**.** Why Do Projects Fail; A resource article. International Project Leadership Academy**,** www.calleam.com/WTPF/?page_id=1445 [Accessed November 13, 2015].

[7] Ravindranath C. P., 2003. Software Metrics: A Guide to Planning, Analysis and Application. CRC Press Inc. Boca Raton, FL, ISBN: 0849316618.

[8] Maleki I., Ebrahimi L., Jordati S., Ramesh, I., 2014. Analysis of Software Cost Estimation Using Fuzzy Logic. International Journal in Foundations of Computer Science & Technology (IJFCST), Vol. 4, No. 3.

[9] Ziauddin N., Shahid K., Shafiullah K., Jamal A. N., 2013. A Fuzzy Logic based Software Cost Estimation Model. International Journal of Software Engineering and its Applications, Vol. 7, No. 2.

[10] Ravishankar S. and Latha P., 2012. Software Cost Estimation using Fuzzy Logic. International Conference on Recent Trends in Computational Methods, Communication and Controls (ICON3C), Proceedings published in International Journal of Computer Applications (IJCA).

[11] Mittal A., Parkash K. and Mittal H., 2010. Software Cost Estimation Using Fuzzy Logic. ACM SIGSOFT Software Engineering, Vol. 35, No. 1, pp. 1-7.

[12] Ashita M., Varun P., Anupama K., 2013. An Analysis of Fuzzy Approaches for COCOMO II. International Journal of Intelligent Systems and Applications, Vol. 5, pp. 68-75.

[13] Lopez-Martin C., Isaza C., Chavoya A., 2012. Software development effort prediction of industrial projects applying a general regression neural network. Empir Software Eng. Pp 738-756. DOI 10.1007/s10664-011-9192-6.

[14] Tirimula R. B., Satchidananda D., Rajib M., 2012. Computational Intelligence in Software Cost Estimation: An Emerging Paradigm. ACM SIGSOFT Software Engineering Notes, Vol. 37, No. 3.

[15] Isa M., Laya E., Saman J., Iraj R., 2014. Analysis of Software Cost Estimation Using Fuzzy Logic. International Journal in Foundations of Computer Science & Technology (IJFCST), Vol. 4, No. 3.

[16] Hamdy A., 2012. Fuzzy Logic for Enhancing the Sensitivity of COCOMO Cost Model. Journal of Emerging Trends in Computing and Information Sciences, Vol. 3, No. 9, pp. 1292-1297.

[17] Balakrishna A. and Rama-Krishna T. K., 2012. Fuzzy and Swarm Intelligence for Software Effort Estimation. Advances in Information Technology and Management (AITM), Vol. 2, No. 1, pp. 246-250.

[18] Agarwa R., Alam Q. and Sarwar S., 2012. Efficient Estimation of Software System using Fuzzy Technique. International Journal of Electronics and Computer Science Engineering, Vol. 1, No. 3, pp. 1006-1012.

[19] Swarup K. N. V. R., Mandala A., Chaitanya M. V., Prasad G. V. S. N. R. V., 2011. Fuzzy logic for Software Effort Estimation Using Polynomial Regression as Firing Interval. International Journal Computer Technology Application, vol. 2, no. 6, pp. 1843-1847.

[20] Prasad R. P. V. G. D., Sudha K. R. and Rama S. P., 2011. Application of Fuzzy Logic Approach to Software Effort Estimation. International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 2, No. 5, pp. 87-92.

[21] Sharma V. and Verma H. K., 2010. Optimized Fuzzy Logic Based Framework for Effort Estimation in Software Development. IJCSI International Journal of Computer Science issues, Vol. 7, Issue 3, No. 2, pp. 30-39.

[22] Jorgensend M., and Shepperd M., 2007. A Systematic Review of Software Development Cost Estimation Studies. IEEE transactions on Software Engineering, Vol. 33, No. 1.

[23] Malathi S. and Sridhar S., 2012. Performance evaluation of Software Effort Estimation using Fuzzy Analogy based on Complexity. International Journal of Computer Applications (IJCA), Vol. 40, No. 3, pp. 32-37.

[24] Jha P. and Patnaik K. S., 2012. Comparative Analysis of COCOMO 81 using Various Fuzzy Membership Functions. International Journal of Computer Applications (IJCA), Vol. 58, No. 14, pp. 220-227.

[25] Garcia-Diaz N., Lopez-Martin C., Chavoya A., 2013. A comparative Study of two Fuzzy logic models for software development effort estimation. The 2013 Iberoamerican Conference on Electronics Engineering and Computer Science, Procedia Technology 7, pp. 305-314.

[26] Boehm B. W., 1981. Software Engineering Economics. Englewood Cliffs, NJ, Prentice-Hall.

[27] Bailey J. W., and Basili, 1981. MetaModel for Software Development Resource Expenditure. Proceedings of International Conference on Software Engineering.

[28] Walston C. E., and Felix, A. P., 1977. A Method of Programming Measurement and Estimation. IBM Systems Journal, Vol. 16, No. 1.

[29] Putnam, L. H., 1978. A General Empirical Solution to the Macro Software Sizing and Estimation Problem. IEEE Transaction on Software Engineering, Vol. 4, No. 4.

[30] Albrecht A. .,, and Gaffney J., 1983. Software Function, Source Lines of Code, and Development Effort Prediction: a Software Science Validation. IEEE Transaction on Software Engineering, Vol. 9, No. 6.

[31] Boehm, B. W., 2000. Software Cost estimation with COCOMO II, Prentice Hall.

[32] Ruan D., D'hondt P., Fantoni P. F., Cock M. D., Nachtegael M., and Kerre E. E., (eds), 2006. Applied Artificial Intelligence. Proceedings of the 7th International FLINS Conference Genova, Italy.

[33] Claudio B., Cesare F., Riccardo R., (eds) 1998. Fuzzy Logic Control: Advances in Methodology. Proceedings of the International Summer School, Ferraro, Italy.

[34] PROMISE repository. 2006. PROMISE Software Engineering Repository data set. http://promise.site.uottawa.ca/SERepository.

[35] Braz M., and Vergilio S., 2004. Using Fuzzy Theory for Effort Estimation of Object-Oriented Software. Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2004.

[36] Standish Group, (2012). CHAOS Report, West Yarmouth MA. The Standish Group International Inc.

[37] Idri, A., Abran, A., and Kjiri, L., (2000). COCOMO Cost Model Using Fuzzy Logic. 7th International Conference on Fuzzy Theory & Technology, Atlantic City, New Jersey, February 27- March 3, 2000.

[38] Zeeshan, M., and Moataz A. A., (2009). Software development effort prediction: A study on the factors impacting the accuracy of fuzzy logic systems. Information and Software Technology, Vol. 52.

[39] Attarzadeh, I., and Ow, S. H., (2010). Soft Computing Approach for Software Cost Estimation. International Journal of Software Engineering (IJSE), Vol. 3, No. 1.

[40] MathWorks, (2015). Fuzzy Logic Toolbox: User's Guide. The MathWorks Inc.