Scientific Journal of Informatics
Vol. 10, No. 1, Feb 2023

p-ISSN 2407-7658 http://journal.unnes.ac.id/nju/index.php/sji e-ISSN 2460-0040

Software Effort Estimation Using Logarithmic Fuzzy Preference Programming and Least Squares Support Vector Machines

Adnan Purwanto[1*], Lindung Parningotan Manik[2]
[1]Computer Science Graduate Program,
University of Nusa Mandiri, Indonesia
[2]Research Center of Informatics,
National Research and Innovation Agency, Indonesia

Abstract.
Purpose: Effort Estimation is a process by which one can predict the development time and cost to develop a software process or product. Many approaches have been tried to predict this probabilistic process accurately, but no single technique has been consistently successful. There have been many studies on software effort estimation using Fuzzy or Machine Learning. For this reason, this study aims to combine Fuzzy and Machine Learning and get better results.
Methods: Various methods and combinations have been carried out in previous research, this research tries to combine Fuzzy and Machine Learning methods, namely Logarithmic Fuzzy Preference Programming (LFPP) and Least Squares Support Vector Machines Machine (LSSVM). LFPP is used to recalculate the cost driver weights and generate Effort Adjustment Point (EAP). The EAP and Lines of Code values are then entered as input for LSSVM. The output results are then measured using the Mean Magnitude of Relative Error (MMRE) and Root-Mean-Square Error (RMSE). In this study, COCOMO and NASA datasets were used.
Result: The results obtained are MMRE of 0.015019 and RMSE of 1.703092 on the COCOMO dataset, while on the NASA dataset the results of MMRE are 0.007324 and RMSE are 6.037986. Then 100% of the prediction results meet the 1% range of actual effort on the COCOMO dataset, while on the NASA dataset, the results show that 89,475 meet the 1% range of actual effort and 100% meet the 5% range of actual effort. The results of this study also show a better level of accuracy than using the COCOMO Intermediate method.
Novelty: This study uses a combination of LFPP and LSSVM, which is an improvement from previous studies that used a combination of FAHP and LSSVM. The method used is also different where LFPP produces better output than FAHP and all data in the dataset is used for training and testing, whereas in previous research it only used a small part of the data.

Keywords: LFPP, LSSVM, Software Effort Estimation, COCOMO
Received November 2022 / **Revised** December 2022 / **Accepted** December 2022

This work is licensed under a Creative Commons Attribution 4.0 International License.

INTRODUCTION

One of the challenges of software engineering development is to develop a practical and relevant development model. Another aspect is estimating how accurately a model can estimate the effort in developing software. Effort Estimation is a process by which one can predict the development time and cost to develop a software process or product. Estimating costs and times accurately is important for several reasons. Overestimation can lead to financial loss in an organization whereas underestimation can result in poor software quality which ultimately leads to software failure. Effort Estimation carried out in the early phases of the development of a project can help project managers [1]. But very little information is available in the early stages. There are many algorithmic and non-algorithmic existing methods for effort estimation.

Software Effort Estimation is a critical component that predicts the effort to complete a major development or maintenance task based on historical data. Accurate estimates are very important for companies and customers because they can help company personnel to classify, prioritize, and determine resources to commit to projects [2]. Since its inception, problems and issues in Software Effort Estimation have been addressed by researchers and practitioners alike. Researchers have proposed many estimation methods

*Corresponding author.
Email addresses: 14002474@nusamandiri.ac.id (Purwanto), lind004@lipi.go.id (Manik)
DOI: 10.15294/sji.v10i1.39865

since the beginning of Software Effort Estimation as a research area [3]–[5]. The application of the developed model has been found to be suitable for a specific type of development environment. The advancements in technology stacks and frequently changing user requirements have made the Software Effort Estimation process difficult. Many approaches have been tried to predict this probabilistic process accurately, but no single technique has been consistently successful. In fact, some researchers have tried to use a combination approach rather than a single approach. The main reason for inaccurate estimates is that data sets from past projects are usually sparse, incomplete, inconsistent and poorly documented. Another reason for this is that the SEE process depends on many visible and invisible factors.

Despite extensive research on it, society has been unable to develop and accept a single model that can be applied in diverse environments and which can deal with many environmental factors. Recently, the multi-criteria decision-making method (MCDM) has emerged as a qualified approach to dealing with multifactor decision-making. Also, incorporating a Machine Learning (ML) approach into an effort estimation model always improves performance.

In previous studies, MCDM and ML have been combined, namely Fuzzy Analytic Hierarchy Process (FAHP) with Least Squares Support Vector Machine (LSSVM) [6] with more accurate results than using only LSSVM. In this study, the results from FAHP were used as input for the LSSVM model. The fuzzy logic approach can improve the accuracy of solving cases of increasingly complex real-world problems. Combining the concepts of fuzzy logic and AHP (called Fuzzy AHP) can solve problems that are unclear [7]. Fuzzy AHP (FAHP) is still a widely used method in various fields such as student project evaluation [8], employee evaluation [8], [9], and others [7], [10], [11].

The technique for deriving crisp weights from pairwise comparison matrices of fuzzy numbers can be done using Extent Analysis (EA) [12]–[14] or Fuzzy Preference Programming (FPP) [15]–[18] methods. The drawbacks of most FAHP applications using the EA method are that they are often considered invalid and the weights obtained by this method do not represent the relative importance of the decision criteria or alternatives [7], [19]. This is related to the determination of zero weight on the decision criteria or alternatives in the priority vector calculation process, so that there are weights that are not considered in the decision analysis. While the shortcomings of the FAHP method using the FPP method are in the degree of membership which can be negative, so it is considered unreasonable [20]. Another finding shows that in some cases, the FPP method yields several optimal solutions when there are strong inconsistencies among fuzzy judgments. Another weakness shows that the priority vectors derived by using the upper or lower triangular elements of the fuzzy comparison matrix are not the same.

The FPP method was then improved with the Logarithmic Fuzzy Preference Programming (LFPP) and Consistent Fuzzy Preference Relation (CFPR) methods. The CFPR method answers the shortcomings of the FPP method by simplifying the expert judgment process when comparing the level of importance between criteria, a number of n-1, where n is the number of criteria [21]. CFPR guarantees that the resulting pairwise comparison matrix is consistent, because each element is calculated using the appropriate proposition. Although it is considered more effective at the stage of determining the importance between criteria in building a pairwise comparison matrix, the CFPR method only pays attention to the mean value of each element to produce a crisp paired comparison matrix. Therefore, so that the resulting matrix remains a fuzzy pairwise comparison matrix, each element retains the upper, middle and lower values [22]. The upper, middle and lower values in each matrix element were used to calculate the consistency index using the LFPP method.

The LFPP method is a refinement of the FPP method which in some cases results in a negative final value, and makes the expected solution less valid [20]. The LFPP method uses the logarithmic function of natural numbers which can produce a single solution in determining the value of the weight of importance [23]. The LFPP method produces a value of λ* which represents the degree of satisfaction and a natural indicator of the inconsistency of expert judgment, so that it can be considered as a consistency index.

SVM is a discriminatory classifier formally defined by a separator hyperplane. In other words, given the labeled training data (supervised learning), the algorithm generates an optimal hyperplane that categorizes the new examples. The hyperplane provides the greatest minimum distance to the training instance. Therefore, the optimal dividing hyperplane maximizes the margins of the training data. SVM is able to handle and map nonlinear classifications using kernel tricks. The input data is mapped to an n-dimensional feature space using functions, and then, a linear model is applied in that space [24].

LSSVM is the least squares version of SVM, a set of related supervised learning methods that analyze data and recognize patterns, and which are used for classification and regression analysis. Another synonym of the SVM standard is the LSSVM algorithm. It adopts equality constraints and linear Karush–Kuhn–Tucker system, which has more powerful computational capabilities in solving nonlinear and small sample problems using linear programming methods [25]. However, the modeling accuracy of a single LSSVM is not only affected by the input data source but also by its kernel functions and regularization parameters. LSSVM is a class of kernel-based learning methods. LSSVM has been used effectively for nonlinear estimation and classification problems. Standard SVM and LSSVM performed consistently in hyperparameter-tuned combinations [26]. But, the advantages of using LSSVM are 1) less computational burden for constrained optimization programming and 2) better for higher dimensional data. In this research, LSSVM has been used to predict software data effort, while learning from past available data

COCOMO (Constructive Cost Model) is a software effort calculation model using Lines of Code (LOC) based regression [27]. COCOMO was later developed into COCOMO Intermediate by taking into account the cost drivers. There are 15 cost drivers which are grouped into 4 attributes. Each cost driver has 4-6 ratings, namely Very Low, Low, Nominal, High, Very High and Extra High with different values. Then the value is multiplied and produces the Effort Adjustment Factor (EAF) value [28].

This study develops a calculation model on COCOMO using the LFPP method to calculate the value of EAF and Machine Learning LSSVM. EAF calculations with the value of each cost driver often still do not produce consistent results. Therefore, there needs to be improvements in calculating the Effort Adjustment Factor using the LFPP method and developing a software effort estimation model using LSSVM, resulting in better accuracy in software effort estimation calculations.

METHODS

The research methodology that will be used in this research is LFPP which is used to weight the criteria on 15 cost drivers which are grouped into 4 attributes. Then the value of the 63 projects data is converted into an ordinal scale (1-6) and then multiplied by the LFPP results. All the multiplication results are added together to produce the EAF value. The resulting EAF value along with LOC is then modeled using LSSVM. Then do the calculation of the performance of the Effort Estimation Software which can produce two parameters, namely Mean Magnitude of Relative Error (MMRE) and Root-Mean-Square Error (RMSE).

The research steps can be described as follows:

1. LFPP Method
    a. Determination of Software Effort Estimation Criteria

       Determination of usability criteria is obtained based on a systematic map that explains the number of criteria n and criteria (C1, C2, …, Cn) used in the usability evaluation model. The preliminary study was carried out by describing a systematic map using the Systematic Mapping Study (SMS) method. SMS aims to build a classification scheme to indicate the frequency of publications, determine coverage in a particular field, and to combine the results in answering more specific research questions, organizing research types and results by grouping areas. SMS is a method that was initially used in drug grouping but recently it has been widely applied in the software engineering field [29].

       Figure 1 shows the steps of the SMS method which consists of 5 steps, namely defining research questions, searching for relevant papers, filtering papers, filtering papers based on abstracts, and mapping data extraction. Each process has a result, and each data extraction result is used to create a mapping [30].

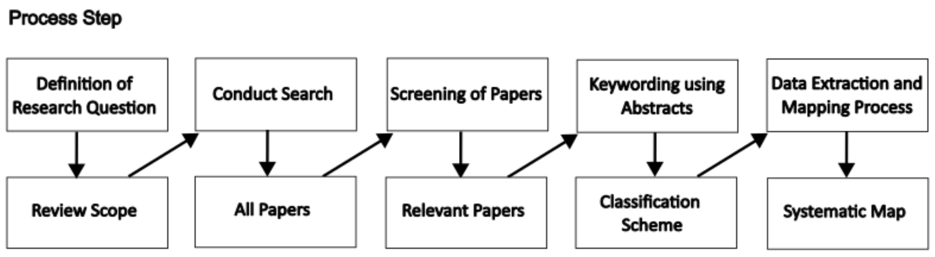

Figure 1. The systematic mapping process

The systematic map determines the criteria used to calculate software effort. These criteria are arranged in such a way as to form a hierarchical structure of the model, with the highest hierarchy being the effort adjustment factor.

b. Comparison of Interests Between Criteria
The fuzzy pairwise comparison matrix was arranged based on the level of importance according to the research participants. Each usability criterion is prioritized according to its weight. The participants, in this case the usability experts, were asked to answer two pairs of questionnaires with wise consideration. The questionnaire is designed in such a way, in which criteria A is compared with criteria B and so on. Experts are free to decide and then mark which criteria they think are more important than others. This study uses triangular FAHP with the conversion of crisp numbers to fuzzy numbers [31]. Assessment of the elements of the problem from each level being researched priority, the assessment is stated on a numerical scale (scale 1 to 9). The results of the questionnaire were then formed into a fuzzy pairwise comparison matrix to calculate the weight of each criterion. Comparison of a criterion against the criterion itself, will produce a value of 1 (equal importance). So that the main diagonal in the matrix will contain the value 1. (Criterion C1 against criteria C1, criteria C2 against criteria C2 and so on). Next, the matrix element is filled with the comparison value of the left column with other columns.

c. Consistency Index Calculation
Research by Wang and Chin [7] states that expert judgment is considered to be inconsistently strong if the optimal value λ^* =0, or the value $\delta^* = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(\delta_{ij}^{*2} + \eta_{ij}^{*2}) = 0$ This means, the greater δ^*, the stronger the inconsistency between fuzzy judgments. Thus, the value of λ^* and δ^* can be treated as a measure of inconsistency in the fuzzy pairwise comparison matrix. The LFPP method produces values varying from -1 to 1. The results show that, even if we add the natural logarithm to the LFPP equation, there is no guarantee that the optimal value is always positive, depending on the P value used. The larger the value of P, the smaller the value of δ^*. This can be seen when the values of P = 10^2 and 10^3 there are still < 0 in both methods

d. Calculation of Criteria Weight
Calculation of weights using the equation [32]

$$w_i^* = \frac{exp(x_i^*)}{\sum_{j=1}^{n} exp(x_j^*)}, i = 1,2,\dots,n, \quad (1)$$

where exp() is the exponential function of for i=1,2, …, n and n is the number of criteria.

2. LSSVM Method

a. Changing the value of the cost driver
63 data on the COCOMO dataset and 93 data on the NASA dataset were converted using an ordinal scale from 1 to 6.

b. Calculation of Effort Adjustment Factor
Effort Adjustment Factor (EAF) is calculated by adding up all multiplication of weights with values on the ordinal scale of each cost driver.

c. Software Effort model development
Using LSSVM with inputs EAF and Line of Code, Effort is calculated. The resulting effort will then be calculated for its performance. Typical kernel functions include linear kernel function, polynomial kernel function, radial basis kernel function, sigmoid kernel

function, and multiple kernel function [25]. While in this study using three kernels, namely polynomial, radial basis and linear.

3. Performance Testing

The performance of various different approaches has been evaluated using different performance measures, in this study mean magnitude of relative error (MMRE) [2] and root-mean-square error (RMSE) [33], [34] as described in the equation below are used. Here, the actual effort is taken from the dataset and predicted effort is the effort calculated using the proposed technique. These measures have been widely used by the research community.

$$MMRE = \frac{1}{N}\sum_{i=1}^{N} \frac{Actual\ effort - Prediction\ Effort}{Actual\ Effort} \tag{2}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(Actual\ Effort - Prediction\ Effort)^2} \tag{3}$$

RESULT AND DISCUSSION

1. LFPP Method

   a. Determination of Software Effort Estimation Criteria

   At the initial stage, the hierarchical structure of the model is built to calculate the Effort Adjustment Factor (EAF) value. The results of the hierarchical structure of the model can be seen in Figure 2.

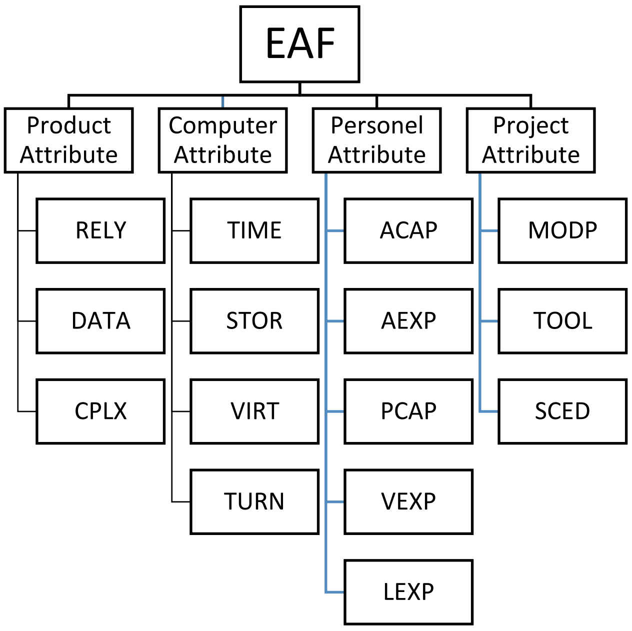

Figure 2. Hirarchy structure of EAF

   b. Comparison of Interests Between Criteria

   A fuzzy pairwise comparison matrix was formed based on the results of the FLPP questionnaire which was filled out by 3 experts (see Appendix …). The experts used are practitioners and academics who are involved in the software effort field as researchers and practitioners.

   This study sets a number of l=3, where l is the number of experts, then the pairwise comparison matrix $P_{n\times n}$ consisting of elements p_{ij} for i,j=1,2,…,n where n=4. The next step is to combine each expert judgment (p_{ij}) that corresponds to each pairwise comparison matrix into a single assessment. In this study, the methodology found by Aczbl and Saaty [35] was used. Based on the research of Aczbl and Saaty[35] it is proven that, if given a number of $l \geq 2$ experts with individual ratings of p_{ij}^{n} for n=1, 2, …, l, then the synthesis function $f\left(p_{ij}^{1}, \dots, p_{ij}^{l}\right)$, according to the aggregate values into a single assessment, using the geometric mean.

   The results of the comparation matrix can be seen in tables 1 to 5.

Table 1. Comparation matrix for EAF

	C1			C2			C3			C4		
C1	1.00	1.00	1.00	1.26	2.29	3.30	0.22	0.28	0.38	6.32	7.32	8.32
C2	0.30	0.44	0.79	1.00	1.00	1.00	0.16	0.19	0.23	4.58	5.59	6.60
C3	2.62	3.63	4.64	4.31	5.31	6.32	1.00	1.00	1.00	6.65	7.65	8.65
C4	0.12	0.14	0.16	0.15	0.18	0.22	0.12	0.13	0.15	1.00	1.00	1.00

Table 2. Comparation matrix for product attribute

	C11			C12			C13		
C11	1.00	1.00	1.00	1.59	2.62	3.63	0.23	0.30	0.44
C12	0.28	0.38	0.63	1.00	1.00	1.00	0.15	0.18	0.22
C13	2.29	3.30	4.31	4.64	5.65	6.65	1.00	1.00	1.00

Table 3. Comparation matrix for computer attribute

	C21			C22			C23			C24		
C21	1.00	1.00	1.00	0.26	0.35	0.55	1.26	2.29	3.30	2.88	3.91	4.93
C22	1.82	2.88	3.91	1.00	1.00	1.00	2.88	3.91	4.93	6.65	7.65	8.65
C23	0.30	0.44	0.79	0.20	0.26	0.35	1.00	1.26	1.44	2.29	3.30	4.31
C24	0.20	0.26	0.35	0.12	0.13	0.15	0.23	0.30	0.44	1.00	1.00	1.00

Table 4. Comparation matrix for personel attribute

	C31			C32			C33			C34			C35		
C31	1	1.00	1.00	3.30	4.31	5.31	4.31	5.31	6.32	5.52	6.54	7.56	3.91	4.93	5.94
C32	0.19	0.23	0.30	1.00	1.00	1.00	2.29	3.30	4.31	3.30	4.31	5.31	2.29	3.30	4.31
C33	0.16	0.19	0.23	0.23	0.30	0.44	1.00	1.00	1.00	2.88	3.91	4.93	1.26	2.29	3.30
C34	0.13	0.15	0.18	0.19	0.23	0.30	0.20	0.26	0.35	1.00	1.00	1.00	0.44	0.55	0.79
C35	0.17	0.20	0.26	0.23	0.30	0.44	0.30	0.44	1.00	1.26	1.82	2.29	1.00	1.00	1.00

Table 5. Comparation matrix for project attribute

	C41			C42			C43		
C41	1.00	1.00	1.00	0.22	0.28	0.38	0.14	0.16	0.19
C42	2.62	3.63	4.64	1.00	1.00	1.00	0.22	0.28	0.38
C43	5.19	6.21	7.23	2.62	3.63	4.64	1.00	1.00	1.00

c. Consistency Index Calculation

The calculation of consistency in the matrix (4.2) in the LFPP model uses equation (4.2) as follows. Minimize $J = (1-\lambda)^2 + P\sum_{i=1}^{13}\sum_{j=i+1}^{14}\left(\delta_{ij}^2 + \eta_{ij}^2\right)$ with limitations

$$\begin{cases} x_1 - x_2 - \lambda\, ln(2.28/1.25) + \delta_{12} \geq ln(1.25), \\ -x_1 + x_2 - \lambda\, ln(3/2) + \eta_{12} \geq -ln(3), \\ x_1 - x_3 - \lambda\, ln(0{,}33/0{,}25) + \delta_{13} \geq ln(0{,}33), \\ -x_1 + x_3 - \lambda\, ln(0{,}5/0{,}33) + \eta_{13} \geq -ln(0{,}5), \\ \vdots \\ \quad x_3 - x_4 - \lambda\, ln(8/7) + \delta_{1314} \geq ln(7), \\ \quad -x_3 + x_4 - \lambda\, ln(9/8) + \eta_{1314} \geq -ln(9), \\ \quad \lambda, x_1, x_2, \ldots, x_4 \geq 0, \\ \quad \delta_{12}, \eta_{12}, \delta_{13}, \eta_{13}, \ldots, \delta_{34}, \eta_{34} \geq 0. \end{cases} \tag{4}$$

If P=10^{-1} is determined, then the value of λ^*= 0.307 (a value above 0, is considered consistent) for the paired matrix for EAF. The value of x is explained as follows, x_1= 1.141; x_2 =0.749; x_3 =2.001; x_4 =-0.549.
While the results of the paired matrix as a whole can be seen in table 6

d. Calculation of Criteria Weight
The weights are calculated using equation (1), so as follows
$w_1^* = exp(1{,}14)/((exp(1{,}14) + exp(0{,}75) + exp(2{,}00) + exp(-0{,}55)) = 0{,}24,$
$w_2^* = exp(0{,}75)/((exp(1{,}14) + exp(0{,}75) + exp(2{,}00) + exp(-0{,}55)) = 0{,}16,$
$w_3^* = exp(2{,}00)/((exp(1{,}14) + exp(0{,}75) + exp(2{,}00) + exp(-0{,}55)) = 0{,}56$
$w_4^* = exp(-0{,}55)/((exp(1{,}14) + exp(0{,}75) + exp(2{,}00) + exp(-0{,}55)) = 0{,}04.$
The results of w_1^* to w_4^*above are for the paired matrix EAF, then the weights for the sub-criteria are shown in the table 7.

Table 6. Result of consistency index calculation

No	Comparation Matrix	Results
1	EAF	λ = 0.307345388704000, δ_{12} = 0.0160653500691174, δ_{13}= -7.90200195643976*10^{-9}, δ_{14} = 0.196634206091134, δ_{23}= 2.26007054164823*10^{-9}, δ_{24}= 0.158681802774347, δ_{34}= 6.34032454916013*10^{-8}, η_{12} = 3.34834934306670*10^{-8}, η_{13} = 0.212699513598144, η_{14} = 1.79213928929813*10^{-8}, η_{23} = 0.142616549074326, η_{24} = -4.43300239670117*10^{-8}, η_{34} = 0.355316063821643, x_1 = 1.14130209637158, x_2 = 0.749499514762662, x_3 = 2.00122167849278, x_4 = -0.549412216180030
2	Product Attribute	λ = 0.811253956233567, δ_{12} = 0.000247186142258240, δ_{13} = -1.51320741369652*10^{-14},δ_{23} = 0.000247186141459889, η_{12}= -1.03861341148107*10^{-13},η_{13} = 0.000247186142119083, η_{23} = -6.50147612331208*10^{-13}, x_1 = 1.02845674260699, x_2 = 0.160987822780843, x_3 = 2.15989408861230
3	Computer Attribute	λ = 0.905639669376814, δ_{12} = 3.27454277762186*10^{-11}, δ_{13} = 0.000111456506979897, δ_{14} = -2.84688891585358*10^{-12}, δ_{23} = 3.00660346821763*10^{-12}, δ_{24} = 2.20057478436442*10^{-11}, δ_{34} = 6.24858668437129*10^{-12}, η_{12}= 0.000111456513235545, η_{13} = 1.30408570521382*10^{-11}, η_{14} = 7.55581320087213*10^{-12}, η_{23} = 0.000111456507595695, η_{24} = 1.12754646342450*10^{-12}, η_{34} = 1.68477602235859*10^{-11}, x_1 = 1.44162997314509, x_2 = 2.47494343884749, x_3 = 0.669564257366483, x_4 = 0.168390392341068
4	Personel Attribute	λ = 1.31736448245890, δ_{12} = 0.118971522449942, δ_{13} = 0.0526578480040824, δ_{14} = -1.08783693446225*10^{-8}, δ_{15} = -2.15554509206035*10^{-9}, δ_{23} = 0.0869824797266322, δ_{24} = -3.53825256836739*10^{-9}, δ_{25} = 6.09934883813902*10^{-9}, δ_{34} = 0.0680990736803731, δ_{35} = 1.58497004108922*10^{-9}, δ_{45} = -7.13799733378960*10^{-9}, η_{12} = 0.0319890397660950, η_{13}= 0.0715412374113577, η_{14} = 0.0680990783793618, η_{15} = -3.33598092460354*10^{-9}, η_{23} = 3.52391948301901*10^{-9}, η_{24} = 2.41257303717067*10^{-9}, η_{25} = -1.42899388646602*10^{-9}, η_{34} = 2.64483589359638*10^{-9}, η_{35}= 1.75343463968512*10^{-10}, η_{45}= -2.59185019484599*10^{-9}, x_1 = 2.59825201114059, x_2 = 1.17460449591612, x_3 = 0.914361476603041, x_4 = 0.319418132124238, x_5 = 0.662955719815310
5	Project Attribute	λ = .342385311410742, δ_{12} = 1.71526533657727*10^{-10}, δ_{13} = 0.00492479475340633, δ_{23} = -1.64086235600608*10^{-10}, η_{12}= 0.00492479485914775, η_{13} = 1.08696945793720*10^{-10}, η_{23} = 0.00492479446014590, x_1 = -0.336473117955605, x_2 = 0.743196051730407, x_3= 1.58884534532510

2. LSSVM Method
The modified dataset is then entered into a Python-based Machine Learning program. The dataset is divided into 2, namely training data and testing data with a composition of 80:20. Then the training data is used to train the model. The metrics used are Root Mean Squarred Error (RMSE) and Mean Magnitude of Relative Error (MMRE) with hyperparameters tested for γ, σ, c are [1.0e-

6, 1.0e-5, 1.0e-4, 1.0e- 3, 1.0e-2, 0.1, 1.0, 10.0, 100.0, 1.0e3] while the parameters for d are 1 to 24. The tested kernels include Radial Basis Function, Polynomial and Linear. The test results can be seen in the table 8.

Table 7. Result of weight calculation

No	**Name of Sub Category**	**Weight**
1	Rely	0.0523851944777905
2	Data	0.0220024780014056
3	Cplx	0.162399824859281
4	Time	0.0351532963013055
5	Stor	0.0987935044039464
6	Virt	0.0162428477952745
7	Turn	0.00984022730783198
8	ACAP	0.334388440267549
9	PCAP	0.0805320946933312
10	AEXP	0.0620793110165558
11	VEXP	0.0342425640469561
12	LEXP	0.0482794990725185
13	MODP	0.00404227067551933
14	TOOL	0.0118992545127162
15	SCED	0.0277191925680181

Table 8. Result of LSSVM

Dataset	**Kernel**	**Parameter**	**RMSE**	**MMRE**
COCOMO	RBF	$\gamma = 1000, \sigma = 0.1$	1.703092	0.015019
	Linear	$\gamma = 1000$	845.807708	4.164992
	Linear	$\gamma = 0.000001$	592.256069	8.421516
	Poly	$\gamma = 0.000001, c= 0.000001, d = 24$	413.362013	0.923077
NASA	RBF	$\gamma = 1000, \sigma = 0.000001$	6.039682	0.007324
	RBF	$\gamma = 1000, \sigma = 0.1$	6.037986	0.007378
	Linear	$\gamma = 0.1$	375.478774	0.743051
	Linear	$\gamma = 10$	360.064139	1.204707
	Poly	$\gamma = 0.000001, c = 1000, d = 24$	615.852924	1

3. Performance Testing
From table 8, it can be concluded that the best value in the COCOMO and NASA datasets for RMSE and MMRE is the RBF kernel with parameters $\gamma = 1000$ and $\sigma = 0.1$. This can be seen from the lowest MMRE and RMSE values obtained. Then the value is entered into the LSSVM model to get the value of the test results. The results of testing the test data on the COCOMO dataset can be seen in table 9, while the NASA dataset can be seen in table 10. Then the results of this test are also compared with the results of the Intermediate COCOMO test.

From table 9 below, it can be concluded that the difference between predicted effort and actual effort data is very good, it is proven that the prediction level that meets the range of 1% actual effort is 13 of 13 data (100%). Compared with Intermediate COCOMO which produces a prediction rate of 1 of 13 data in the range of 5%, the LFPP method with LSSVM is better.

Table 9. Result of performance test for COCOMO dataset

No	Actual	Result Test of LFPP with LSSVM				Result Test of Intermediate COCOMO			
		Predicted	Pred			Predicted	Pred		
			10	5	1		10	5	1
1	1075	1074.61777979	1	1	1	339.5465	0	0	0
2	423	423.26912843	1	1	1	171.8695	0	0	0
3	201	201.49090666	1	1	1	99.59108	0	0	0
4	40	40.3763407	1	1	1	38.0595	1	1	0
5	6600	6594.09829927	1	1	1	12370.85	0	0	0
6	539	539.15324432	1	1	1	343.8561	0	0	0
7	702	701.99040716	1	1	1	2634.893	0	0	0
8	82	82.60971467	1	1	1	73.79905	0	0	0
9	6	6.68571186	1	1	1	3.518121	0	0	0
10	87	87.60479277	1	1	1	99.65634	0	0	0
11	126	126.56583174	1	1	1	270.8005	0	0	0
12	176	176.51588169	1	1	1	86.45069	0	0	0
13	15	15.67672085	1	1	1	10.39778	0	0	0

Table 10. Result of performance test for NASA dataset

No	Actual	Result Test of LFPP with LSSVM				Result Test of Intermediate COCOMO			
		Predicted	Pred			Predicted	Pred		
			10	5	1		10	5	1
1	25.2	25.81078534	1	1	0	22.91249	1	0	0
2	352.8	326.55828025	1	1	0	290.5114	0	0	0
3	360.0	360.1462867	1	1	1	418.2046	0	0	0
4	60.0	60.58253625	1	1	1	48.2116	0	0	0
5	120.0	120.52259619	1	1	1	73.66737	0	0	0
6	192.0	192.45066811	1	1	1	98.73482	0	0	0
7	60.0	60.58253625	1	1	1	33.68189	0	0	0
8	444.0	444.1346985	1	1	1	360.4753	0	0	0
9	114.0	114.528588	1	1	1	56.30644	0	0	0
10	1248.0	1247.39572306	1	1	1	1202.737	1	1	0
11	973.0	972.67044833	1	1	1	1353.757	0	0	0
12	252.0	252.39072805	1	1	1	118.5991	0	0	0
13	571.0	571.47164714	1	1	1	548.6515	1	1	0
14	150.0	150.49262616	1	1	1	61.45643	0	0	0
15	192.0	192.45066811	1	1	1	62.04336	0	0	0
16	300.0	300.34277601	1	1	1	395.6275	0	0	0
17	756.0	755.88723155	1	1	1	1086.801	0	0	0
18	599.0	599.04407471	1	1	1	136.752	0	0	0
19	1645.9	1644.8980046	1	1	1	843.2518	0	0	0

From table 10 above, it can be concluded that the difference between predicted effort and actual effort data is well proven that the prediction level that meets the range of 1% actual effort is 17 of 19 data (89.47%) while in the range of 5% of actual effort 19 out of 19 data (100%). While Intermediate COCOMO which produces a prediction rate of 2 of 19 data in the 5% range and 3 of 19 data in the 10% range of Actual Effort S, the LFPP method with LSSVM can predict better.
In a previous study conducted by Sehra [6], a comparison table of MMRE and RMSE results for both the COCOMO and NASA datasets using the BCO, RBF-LSSVM, FAHP-RBF-LSSVM

methods has been shown. Tables 11 and 12 display these comparisons plus the results of the current study.

Table 11. MMRE and RMSE comparation of BCO, RBF-LSSVM, FAHP- RBF-LSSVM and LFPP- RBF-LSSVM using COCOMO dataset

S.no.	Method	Result
MMRE		
1	BCO	0.63
2	RBF-LSSVM	0.82
3	FAHP-RBF-LSSVM	0.57
4	LFPP-RBF-LSSVM	0.015
RMSE		
1	BCO	763.07
2	RBF-LSSVM	630.78
3	FAHP-RBF-LSSVM	569.43
4	LFPP-RBF-LSSVM	1.703

Table 12. MMRE and RMSE comparation of BCO, RBF-LSSVM, FAHP- RBF-LSSVM and LFPP- RBF-LSSVM using NASA dataset

S.no.	Method	Result
MMRE		
1	BCO	0.21
2	RBF-LSSVM	0.5
3	FAHP-RBF-LSSVM	0.19
4	LFPP-RBF-LSSVM	0.0073
RMSE		
1	BCO	9.9
2	RBF-LSSVM	19.74
3	FAHP-RBF-LSSVM	5.99
4	LFPP-RBF-LSSVM	6.039

From table 11 above, the results of this study show that the performance of MMRE and RMSE is much better than the three methods above. Whereas in table 12, the results of this study show better performance on MMRE but on RMSE the results of the FAHP-RBF-LSSVM method show slightly better performance.

CONCLUSION

predicting effort in software development with an MMRE score of 0.015019 and an RMSE score of 1.703092 on the COCOMO dataset, while the NASA dataset obtained an MMRE score of 0.007324 and an RMSE of 6.037986. Then 100% of the prediction results meet the 1% range of actual effort on the COCOMO dataset, while the NASA dataset results in 89.47%. However, in the range of 5% of the actual effort on the NASA dataset, the prediction results are 100%. In both datasets it can be concluded that the method with the combination of LFPP and LSSVM produces more accurate predictions than the COCOMO Intermediate method. The results of this reseach also show a better increase in performance compared to previous research that has been done using the BCO, RBF-LSSVM and FAHP-RBF-LSSVM methods.

REFERENCES

[1] A. B. Nassif, M. Azzeh, A. Idri, and A. Abran, "Software Development Effort Estimation Using Regression Fuzzy Models," *Comput. Intell. Neurosci.*, vol. 2019, pp. 1–17, Feb. 2019, doi: 10.1155/2019/8367214.

[2] H. Azath, M. Mohanapriya, and S. Rajalakshmi, "Software Effort Estimation Using Modified Fuzzy C Means Clustering and Hybrid ABC-MCS Optimization in Neural Network," *J. Intell. Syst.*, vol. 29, no. 1, pp. 251–263, Feb. 2018, doi: 10.1515/jisys-2017-0121.

[3] S. K. Sehra, Y. Singh Brar, and N. Kaur, "Applying Fuzzy-AHP for software effort estimation in data scarcity," *Int. J. Eng. Trends Technol.*, vol. 45, no. 1, pp. 4–9, Mar. 2017, doi: 10.14445/22315381/IJETT-V45P202.

[4] H. Rastogi, S. Dhankhar, and M. Kakkar, "A survey on software effort estimation techniques," in *2014 5th Int. Conf. - Conflu. Next Gener. Inf. Technol. Summit (Conflu.)*, Sep. 2014, pp. 826–830. doi: 10.1109/CONFLUENCE.2014.6949367.

[5] A. Trendowicz, J. Münch, and R. Jeffery, “State of the practice in software effort estimation: A survey and literature review,” *Lect. Notes Comput. Sci. (incl. subser. Lect. Notes Artif. Intell. Lect. Notes Bioinform.)*, vol. 4980 LNCS, pp. 232–245, 2011, doi: 10.1007/978-3-642-22386-0_18.

[6] S. K. Sehra, Y. S. Brar, N. Kaur, and S. S. Sehra, “Software effort estimation using FAHP and weighted kernel LSSVM machine,” *Soft Comput.*, vol. 23, no. 21, pp. 10881–10900, Nov. 2019, doi: 10.1007/s00500-018-3639-2.

[7] Y.-M. Wang and K.-S. Chin, “Fuzzy analytic hierarchy process: A logarithmic fuzzy preference programming methodology,” *Int. J. Approx. Reason.*, vol. 52, no. 4, pp. 541–553, Jun. 2011, doi: 10.1016/j.ijar.2010.12.004.

[8] C. Ayca and K. Hasan, “An application of fuzzy analytic hierarchy process (FAHP) for evaluating students project,” *Educ. Res. Rev.*, vol. 12, no. 3, pp. 120–132, Feb. 2017, doi: 10.5897/ERR2016.3065.

[9] R. P. Kusumawardani and M. Agintiara, “Application of Fuzzy AHP-TOPSIS Method for Decision Making in Human Resource Manager Selection Process,” *Procedia Comput. Sci.*, vol. 72, pp. 638–646, 2015, doi: 10.1016/j.procs.2015.12.173.

[10] E. Iryanti and R. Pandiya, “Evaluating the quality of e-learning using consistent fuzzy preference relations method,” in *2016 Int. Conf. Front. Inf. Technol. (FIT)*, Dec. 2016, pp. 61–66. doi: 10.1109/FIT.2016.7857539.

[11] Z. Turskis, E. K. Zavadskas, J. Antucheviciene, and N. Kosareva, “A Hybrid Model Based on Fuzzy AHP and Fuzzy WASPAS for Construction Site Selection,” *Int. J. Comput. Commun. Control*, vol. 10, no. 6, p. 113, Oct. 2015, doi: 10.15837/ijccc.2015.6.2078.

[12] Y.-M. Wang, Y. Luo, and Z. Hua, “On the extent analysis method for fuzzy AHP and its applications,” *Eur. J. Oper. Res.*, vol. 186, no. 2, pp. 735–747, Apr. 2008, doi: 10.1016/j.ejor.2007.01.050.

[13] T. Wahyuningrum, Azhari, and Suprapto, “A Comparison of Extent Analysis and Fuzzy Preference Programming for Evaluating B2C Website Usability,” in *2018 8th IEEE Int. Conf. Control Syst. Comput. Eng. (ICCSCE)*, Nov. 2018, pp. 150–155. doi: 10.1109/ICCSCE.2018.8684999.

[14] S. Dožić, T. Lutovac, and M. Kalić, “Fuzzy AHP approach to passenger aircraft type selection,” *J. Air Transp. Manag.*, vol. 68, pp. 165–175, May 2018, doi: 10.1016/j.jairtraman.2017.08.003.

[15] L. Mikhailov, “A Fuzzy Programming Method for Deriving Priorities in the Analytic Hierarchy Process,” *J. Oper. Res. Soc.*, vol. 51, no. 3, p. 341, Mar. 2000, doi: 10.2307/254092.

[16] F.-Y. Meng, Q.-X. An, C.-Q. Tan, and X.-H. Chen, “An Approach for Group Decision Making With Interval Fuzzy Preference Relations Based on Additive Consistency and Consensus Analysis,” *IEEE Trans. Syst. Man Cybern. Syst.*, vol. 47, no. 8, pp. 2069–2082, Aug. 2017, doi: 10.1109/TSMC.2016.2606647.

[17] C. García-Diéguez, M. Herva, and E. Roca, “A decision support system based on fuzzy reasoning and AHP–FPP for the ecodesign of products: Application to footwear as case study,” *Appl. Soft Comput.*, vol. 26, pp. 224–234, Jan. 2015, doi: 10.1016/j.asoc.2014.09.043.

[18] J. Rezaei, R. Ortt, and V. Scholten, “An improved fuzzy preference programming to evaluate entrepreneurship orientation,” *Appl. Soft Comput.*, vol. 13, no. 5, pp. 2749–2758, May 2013, doi: 10.1016/j.asoc.2012.11.012.

[19] L. Sun, Z. Ma, Y. Shang, Y. Liu, H. Yuan, and G. Wu, “Research on multi-attribute decision-making in condition evaluation for power transformer using fuzzy AHP and modified weighted averaging combination,” *IET Gener. Transm. Distrib.*, vol. 10, no. 15, pp. 3855–3864, Nov. 2016, doi: 10.1049/iet-gtd.2016.0381.

[20] M. Moayeri, A. Shahvarani, M. H. Behzadi, and F. Hosseinzadeh-Lotfi, “Comparison of Fuzzy AHP and Fuzzy TOPSIS Methods for Math Teachers Selection,” *Indian J. Sci. Technol.*, vol. 8, no. 13, Jul. 2015, doi: 10.17485/ijst/2015/v8i13/54100.

[21] T. Wahyuningrum, A. Azhari, and S. -, “An Extended Consistent Fuzzy Preference Relation to Evaluating Website Usability,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 9, 2019, doi: 10.14569/IJACSA.2019.0100915.

[22] M. Salaheldin, “Solution of fuzzy analytic hierarchy process using simulation,” in *PEDAC’09 Int. Conf. Prod. Eng. Des. Control*, 2009, pp. 1–11. doi: 10.13140/RG.2.1.2892.3049.

[23] T. Wahyuningrum, A. Azhari, and S. Suprapto, “Modified LFPP to Improve the Accuracy of Matrix Pairwise Comparison Consistency Index in the Usability Evaluation,” *Int. J. Intell. Eng. Syst.*, vol. 13, no. 3, pp. 397–406, Jun. 2020, doi: 10.22266/ijies2020.0630.36.

[24] U. I. Larasati, M. A. Muslim, R. Arifudin, and A. Alamsyah, “Improve the accuracy of support vector machine using chi square statistic and term frequency inverse document frequency on movie

review sentiment analysis," *Sci. J. Informatics*, vol. 6, no. 1, pp. 138–149, 2019.

[25] D. Du, X. Jia, and C. Hao, "A New Least Squares Support Vector Machines Ensemble Model for Aero Engine Performance Parameter Chaotic Prediction," *Math. Probl. Eng.*, vol. 2016, pp. 1–8, 2016, doi: 10.1155/2016/4615903.

[26] M. Afshin, A. Sadeghian, and K. Raahemifar, "On Efficient Tuning of LS-SVM Hyper-Parameters in Short-Term Load Forecasting: A Comparative Study," in *2007 IEEE Power Eng. Soc. Gen. Meet.*, Jun. 2007, pp. 1–6. doi: 10.1109/PES.2007.385613.

[27] R. K. Yadav, "Optimized Model for Software Effort Estimation Using COCOMO-2 Metrics with Fuzzy Logic," *Int. J. Adv. Res. Comput. Sci.*, vol. 8, no. 7, pp. 121–125, Aug. 2017, doi: 10.26483/ijarcs.v8i7.4113.

[28] M. Baiquni, R. Sarno, Sarwosri, and Sholiq, "Improving the accuracy of COCOMO II using fuzzy logic and local calibration method," in *2017 3rd Int. Conf. Sci. Inf. Technol. (ICSITech)*, Oct. 2017, pp. 284–289. doi: 10.1109/ICSITech.2017.8257126.

[29] M. Oriol, J. Marco, and X. Franch, "Quality models for web services: A systematic mapping," *Inf. Softw. Technol.*, vol. 56, no. 10, pp. 1167–1182, Oct. 2014, doi: 10.1016/j.infsof.2014.03.012.

[30] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic mapping studies in software engineering," in *Proc. 12th int. conf. Eval. Assess. Softw. Eng.*, 2008, pp. 68–77. doi: 10.5555/2227115.2227123.

[31] P. D. D. Dominic and H. Jati, "A comparison of Asian airlines websites quality: using a non-parametric test," *Int. J. Bus. Innov. Res.*, vol. 5, no. 5, p. 599, 2011, doi: 10.1504/IJBIR.2011.042451.

[32] T. Wahyuningrum and R. Pandiya, "Logaritmic Fuzzy Preference Programming Approach for Evaluating University Ranking Optimization," *Sci. J. Inform.*, vol. 4, no. 1, pp. 1–7, May 2017, doi: 10.15294/sji.v4i1.6738.

[33] T. Chai and R. R. Draxler, "Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature," *Geosci. Model Dev.*, vol. 7, no. 3, pp. 1247–1250, Jun. 2014, doi: 10.5194/gmd-7-1247-2014.

[34] W. Wang and Y. Lu, "Analysis of the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE) in Assessing Rounding Model," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 324, p. 012049, Mar. 2018, doi: 10.1088/1757-899X/324/1/012049.

[35] J. Aczél and T. L. Saaty, "Procedures for synthesizing ratio judgements," *J. Math. Psychol.*, vol. 27, no. 1, pp. 93–102, Mar. 1983, doi: 10.1016/0022-2496(83)90028-7.