International Journal of Advances in Intelligent Informatics
Vol. 4, No. 3, November 2018, pp. 251-260
ISSN 2442-6571

Analogy-based model for software project effort estimation

Ardiansyah [1,*], Murein Miksa Mardhia [2], Sri Handayaningsih [3]

Software Engineering and Data Knowledge Research Group, Department of Informatics, Universitas Ahmad Dahlan, Yogyakarta, Indonesia
[1] ardiansyah@tif.uad.ac.id; [2] murein.miksa@tif.uad.ac.id; [3] sriningsih@tif.uad.ac.id
* corresponding author

ARTICLE INFO

Article history
Received August 25, 2018
Revised October 21, 2018
Accepted November 10, 2018
Available online November 11, 2018

Keywords
Analogy estimation
Effort estimation
Software project
Similarity distance
Machine learning

ABSTRACT

Accurate effort estimation of software development plays an important role to predict how much effort should be prepared during the works of a software project so that it can be completed on time and budget. Some sectors, e.g. banking sectors, were renowned fields of software projects, not only due to its huge size of project, but also extremely expensive and takes a long time to completion. Project estimation is essential for software development project able to run on time and budget with maximum quality. This study aims to investigate the accuracy of software project effort estimation with the Analogy method using three parameters: Euclidean, Manhattan and Minkowski distance. Analogy based estimation consists several stage included similarity measure, analogy adaptation, estimation calculation and model evaluation. The results showed that the best combination of Analogy methods was using Manhattan distance with an accuracy of 50% MMRE, 28% MdMRE and Pred(25) 48%. Thus, we can concluded that this model can be used to predict accurately.

1. Introduction

In term of software development, the first step is likely started from estimating the project effort. A project manager must be careful in considering the three main factors of a project: system functionality, duration and project costs [1]. Project estimation is essential for software development project able to run on time and budget with maximum quality. In 2015, Standish Group released a survey result that 52.7% of software projects were always late from initial estimation and its costs also exceeded the budget.

Software effort estimation has been researched and developed both in algorithmic and machine learning method since the 1960s. Estimation based on expert judgement is one of the earliest and most widely used methods. Expert judgement is a process of estimating the software that results from an assessment process conducted by experts who are experienced in software projects. One of the well-known estimation techniques is Planning Poker which is often used in Agile software development methodologies [2]. There is also a Function Point, an estimation method proposed by [3] using the function points as a unit of size of the software to be developed. COCOMO or Constructive Cost Model is one of the most popular algorithmic method [4]. COCOMO I classified three project classes of Organic, Semidetached and Embedded [5]. The Use Case Point proposed by [6] estimates the effort of the software with several effort drivers including UCW, UUCW, ECF, TCF. The UCP itself is derived from the Function Points method using 20 or 28 productivity factors. Moreover, there is a regression analysis introduced by [7] and [8] which analyzes the relationship between two or more independent and dependent variables. Bayesian Belief Network (BNN) is a method with a causal-relationship approach described as directed acyclic graph. Nodes symbol represent discrete variables or random continuum, and edges represent a probabilistic dependence between the connected variables [9]-[10]. In addition, other approach have been widely used such as Artificial Neural Networks [11]-[12].

To summarize, there are two types of approaches for estimating software development effort: algorithmic and machine learning approaches. The machine learning method used to estimate software effort include KNN, Support Vector Machine (SVM) [13], Decision Tree [14], Analogy, Deep Learning [15], Ensemble [16] and Neural Network (NN) [17]. One important issue of software project estimation is related to estimation accuracy. Accurate estimation is necessary for the project can be completed on time and on budget. Unfortunately no estimating approach has proved consistently accurate. This is a major challenging problem should be solved in effort estimation field.

Among those various methods of estimating software effort, Analogy is the most commonly used method. Analogy compares the effort driver of a new software project to the previous project data to find the most similar project. This is possible because Analogy is able to learn from previous experiences autonomously. Analogy evaluation results have shown the highest accuracy compared to the other machine learning and non-machine learning methods where the average mean magnitude of relative error (MMRE) is 49.8%, the median magnitude of relative error (MdMRE) of 29.37% and Pred(25) of 51.23% [18].

Data sets play a vital role in the implementation of Analogy for software project estimation. There are currently a number of data sets of software projects publicly available including COCOMO81 [19], Miyazaki, Albrecht, China, ISBSG [20], Desnarhais, NASA, Maxwell [21], Kemerer [22], Finnish, Cosmic [23], Kitchenham, UCP, Telecom, Atkinson and Tukutuku. This paper aims to investigate the accuracy of the Analogy method on data sets of software projects. This research has one major contribution namely the implementation of Analogy framework to gain the consistent accuracy result in software project effort prediction.

2. Method

2.1. Analogy-based Estimation

The essence of Analogy-based method is to compare the projects that will be estimated with all the software project's historical data. Project data can come from primary data that is internal to the company or widely available public data. Comparison is done to find out which projects are the most similar that will be estimated. Similar projects will be selected to be adapted so that the estimated effort of the new project can be identified. Fig. 1 shows the Analogy-based estimation model framework.

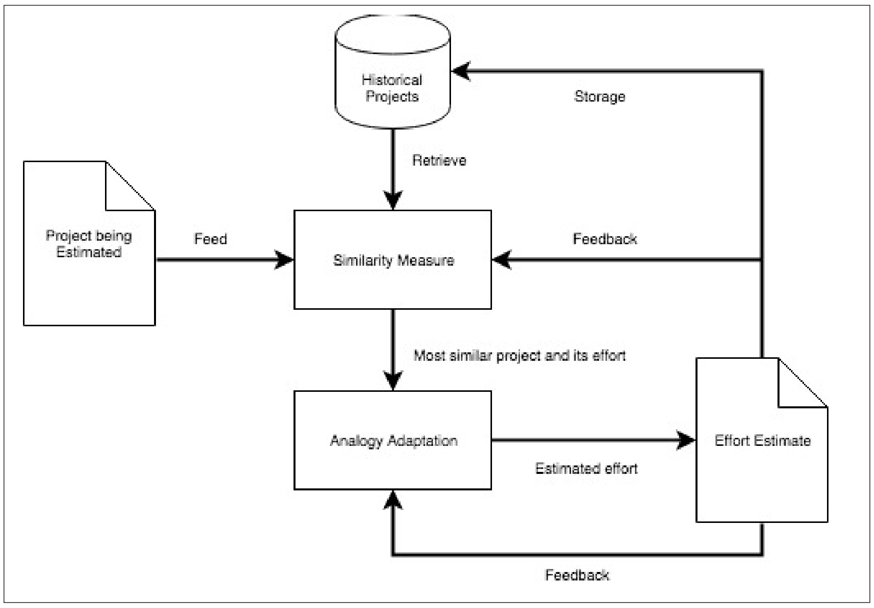

Fig. 1. Analogy-based estimation framework

2.2.1. Similarity Measure

Similarity measures calculates the similarity between projects based on how close the distance between projects according to the type of each attribute. The measurement techniques used in this experiment are Euclidean, Manhattan and Minkowski which are proven to produce good results according to similarity measurements [24]-[25].

Euclidean distance is measuring the distance D between two software projects notated in equation (1) and (2). Where p is a new project that will be estimated and p' is the older project that has been completed. The f_i and f'_i show *i-th* attribute/feature value of a project, w_i = {0, 1} is the weight of the *i-th* attribute.

$$D(p,p') = \sqrt{\sum_{i=1}^{n} w_i\, Dis(f_i, f'_i)} \tag{1}$$

$$Dis(f_i, f'_i) = \begin{cases} (f_i - f'_i)^2, & \text{if } f_i \text{ and } f'_i \text{ are numeric or ordinal} \\ 1, & \text{if } f_i \text{ and } f'_i \text{ are nominal and } f_i = f'_i \\ 0, & \text{if } f_i \text{ and } f'_i \text{ are nominal and } f_i \neq f'_i \end{cases} \tag{2}$$

Manhattan distance calculates the absolute distance on each pair of attributes without rooting as denoted by equation (3) and (4).

$$D(p,p') = \sum_{i=1}^{n} w_i Dis(f_i, f'_i) \tag{3}$$

$$Dis(f_i, f'_i) = \begin{cases} |f_i - f'_i|, & \text{if } f_i \text{ and } f'_i \text{ are numeric or ordinal} \\ 1, & \text{if } f_i \text{ and } f'_i \text{ are nominal and } f_i = f'_i \\ 0, & \text{if } f_i \text{ and } f'_i \text{ are nominal and } f_i \neq f'_i \end{cases} \tag{4}$$

Minkowski distance is a generalization of Euclidean and Manhattan distance that calculates the rank of each attribute pair as denoted by equation (5) and (6).

$$D(p,p') = \sqrt[h]{\sum_{i=1}^{n} w_i\, Dis(f_i, f'_i)} \tag{5}$$

$$Dis(f_i, f'_i) = \begin{cases} |f_i - f'_i|^h, & \text{if } f_i \text{ and } f'_i \text{ are numeric or ordinal} \\ 1, & \text{if } f_i \text{ and } f'_i \text{ are nominal and } f_i = f'_i \\ 0, & \text{if } f_i \text{ and } f'_i \text{ are nominal and } f_i \neq f'_i \end{cases} \tag{6}$$

2.2.2. Number of Selected Analogy

The selected analogy is determined by how many most similar projects used as analogues to model software project effort estimation. There are two types of analogy selection: fixed and dynamics analogy selection. Some studies that adopt fixed analogy selection suggest using one closest analogy (K = 1), two closest analogy K = {1, 2}, three closest analogy K = {1, 2, 3}, four closest analogy K = {1, 2, 3, 4} or five closest analogy K = {1, 2, 3, 4, 5}. In this study, the fixed analogy selection category was chosen by applying all of the combinations, K = 1, K = {1, 2}, K = {1, 2, 3}, K = {1, 2, 3, 4} and K = {1, 2, 3, 4, 5}.

2.2.3. Analogy Adaptation

After determining the number of selected analogy, the next step is predicting the effort of the new project by calculating certain statistical techniques based on the selected project. There are four analogy adaptations applied here: closest analogy (CA), mean of closest analogies, median of closest analogy, and inverse rank weighted mean (IRWA) of closest analogy.

Closest analogy means choosing one (K = 1) from the closest project. Mean of closest analogies is adaptation analogy obtained by calculating the average effort driver from as many as K > 1 selected analogy. The median of closest analogy is an adaptation analogy obtained by calculating the median effort driver from as many as K > 2 selected analogies. Inverse rank weighted mean is an adaptation analogy that gives the highest weight in the selected analogy most similar to other analogy. For example, if four closest analogy are selected, the first closest analogy (CA) is given a weight of four, the second closest analogy (SC) is given a weight of three, the third closest analogy (TC) is given weight two and the fourth closest analogy (LA) are given a weight of one [26]. The calculation of inverse rank weighted mean is formulated as in (7).

$$Inverse\ Rank\ Weighted\ Mean = \frac{4CA+3SC+2TC+LA}{10} \quad (7)$$

2.2.4. Adaptation Rules

Adaptation rules are the last step taken to calculate the amount of effort estimated on a new project according to the most similar selected project. The calculation is done by dividing the old project effort with the size of old project then multiply with the size of new project. Equation (8) denoted the formulations of these adaptation rules.

$$Effort_{new\ project} = \frac{Effort_{old\ project}}{Size_{old\ project}} Size_{new\ project} \quad (8)$$

2.2.5. Model Evaluation

Three evaluation criteria used in this study are Mean Magnitude of Relative Error (MMRE), Median Magnitude of Relative Error (MdMRE) and Pred(25). These three criteria are most widely used to measure the accuracy of software project estimation models resulting from the Magnitude of Relative Error (MRE) measurement. MMRE is generated by calculating the average MRE of each project in the data set. MMRE is one of evaluation technique that is used to assess the efficiency of the effort to be estimated.

MRE is a statistical technique used to measure the accuracy of project estimates obtained from dividing the absolute value from e_i subtracted by $\hat{e}_i$ with e_i, as denoted in equation (9). In equation (9), *e* shows the actual effort of the old project and $\hat{e}$ is the estimated effort of the new project obtained using (8).

$$MRE = \frac{|e_i - \hat{e}_i|}{e_i} \quad (9)$$

MMRE as denoted by equation (10) is one of the accuracy measurements for software project estimation models that calculate the average of MRE. The accuracy of the estimation model is categorized as good if the MMRE is less than equal to 0.25.

$$MMRE = \frac{1}{n}\sum_{i=1}^{i=n} \frac{|e_i - \hat{e}_i|}{e_i} \quad (10)$$

MdMRE as denoted by equation (11) is an accuracy measurement of a software project estimation model that calculates the median of MRE. The accuracy of the estimation model is categorized as good if MdMRE is less than equal to 0.25.

$$MdMRE = median(MRE) \quad (11)$$

Pred(25) is an aggregate of the percentage of MRE which is less than equal to 0.25, as denoted by equation (12). The accuracy of the estimation model is categorized as good if Pred(25) is more than equal to 0.75.

$$Pred(25) = \frac{1}{n} \text{x} \sum_{i=1}^{n} (MRE \leq 0.25) \quad (12)$$

2.2. Data Set Description

The experiment uses Maxwell's data set consists of 62 banking software project data in Finland from 1985 to 1993 and has often been used in research related to software project estimation [27]–[29]. There are 16 attributes owned by Maxwell's data set [30]. In order to develop the estimation model, three attributes were chosen which had a major influence on the project, namely Duration, Size and Effort. Duration is a numeric type attribute that shows the duration of the project from the specification stage until it is sent to the client and measured in months. Size is a numeric type attribute that shows the size of a software project that is calculated by the unit function point (FP). Effort is a numeric type attribute that shows how long a software developer works on a project starting from the specification stage to being sent to the client and measured in hours.

Five from the total number of 62 project data have been removed from the data set since those are considered as outliers due to the very large values. Those are projects with ID numbers 62, 38, 26, 21 and 18. This is done in accordance with the recommendations from [31] which state that outlier data need to be eliminated. Reference [24] had also once discarding data because it indicates an outlier with very little value. Thus, of the 62 data now, 57 data sets are left to be used for experiments. Descriptive statistics for effort driver size, duration attributes and the amount of effort of software development on the Effort attribute are shown in Table 1. The average project size is 478 function points, with a work duration of 5.6 months with an effort of 5910.2 hours. The smallest project size is 48 FP and the largest is 1849 FP. The fastest project duration is one month and a maximum of nine months with a standard deviation of 2.2 months. The least deployed efficiency is 583 hours and the largest is 25919 hours with a standard deviation of 4968.8 hours.

Table 1. Descriptive Statistic Maxwell Data Set (*N* = 57)

Attribute	Minimum	Maximum	Mean	Median	Standard Deviation
Size	48	1849	478.0	366	397.9
Duration	1	9	5.6	6	2.2
Effort	583	25910	5910.2	4557	4968.8

3. Results and Discussion

Data sets are randomly divided into training data and testing data, with a percentage of 87% and 13% respectively. This division differs from what was done by [30] which divided 50 training data from projects prior to year 1992 and 12 testing data from the project between year 1992 and 1993. Fig. 2 shows the framework for cross-validation process.

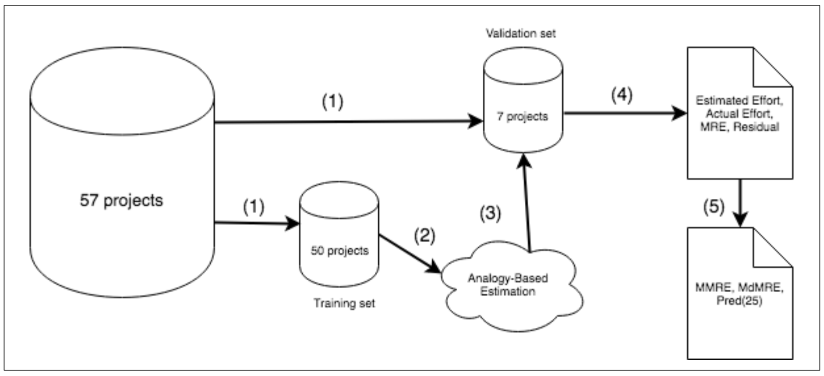

Fig. 2. Framework for cross-validation process

The results of the evaluation process use three-fold cross validation techniques to form a composition as shown in Table 2.

Table 2. Three-fold Cross Validation Technique

Set	Training Data	Testing Data
1	1,2,3,4,5,6,7,10,11,12,13,14,15,16,22,23,24,25,27,28,29,31,32,33,34,35,36,37, 50,51,52,53,54,55,56, 8,17,19,30,39,48,49	41,42,43,44,45,46,47
2	1,2,3,4,5,6,7,10,11,12,13,14,15,16,22,23,24,25,27,28,29,31,32,33,34,35,36,37, 8,17,19,30,39,48,49, 41,42,43,44,45,46,47	50,51,52,53,54,55,56
3	1,2,3,4,5,6,7,10,11,12,13,14,15,16,22,23,24,25,27,28,29,31,32,33,34,35,36,37, 41,42,43,44,45,46,47,50,51,52,53,54,55,56	8,17,19,30,39,48,49

3.1. MMRE Results

The results of MMRE accuracy obtained by Manhattan distance had the lowest MMRE of 0.39 with *K* = 2 using the mean of closest analogies and *K* = 3 using IRWM, Euclidean distance with the lowest MMRE of 0.44 with *K* = 2 and *K* = 3 using IRWM, Minkowski distance with MMRE was 0.42 with *K* = 3 using IRWM. So that the MMRE with Manhattan distance has the best MMRE value compared to Euclidean and Minkowski distance. Table 3 shown the MMRE results from Manhattan, Euclidean and Minkowski distance.

Table 3. Mean Magnitude of Relative Error Results

Model	Manhattan	Euclidean	Minkowski
CA-K1	0.54	0.54	0.54
Mean-K2	**0.39**	0.51	0.50
IRWM-K2	0.46	**0.44**	0.44
Mean-K3	0.40	0.48	0.44
IRWM-K3	**0.39**	**0.44**	**0.42**
Median-K3	0.46	0.46	0.56
Mean-K4	0.43	0.50	0.48
IRWM-K4	0.40	0.46	0.45
Median-K4	0.50	0.56	0.52
Mean-K5	0.47	0.51	0.50
IRWM-K5	0.43	0.48	0.47
Median-K5	0.50	0.56	0.54

3.2. Pred(25) Results

The results of Pred(25) obtained by Manhattan distance had the highest value of 0.48 with *K* = 2 using the mean of closest analogies and *K* = 4 using the median of closest analogies. The highest Pred(25) value using Euclidean distance of 0.43 with *K* = 2 using mean of closest analogies, *K* = 3 uses IRWM and the median of closest analogies, while Minkowski with highest Pred(25) is 0.48 with *K* = 2 using mean of closest analogies as shown in Table 4.

Table 4. Pred(25) Results

Model	Manhattan	Euclidean	Minkowski
CA-K1	0.24	0.24	0.24
Mean-K2	**0.48**	**0.43**	**0.48**
IRWM-K2	0.43	0.38	0.43
Mean-K3	0.38	0.33	0.33
IRWM-K3	0.43	**0.43**	0.43
Median-K3	0.43	**0.43**	0.43
Mean-K4	0.29	0.19	0.19
IRWM-K4	0.43	0.33	0.38
Median-K4	**0.48**	0.29	0.38
Mean-K5	0.29	0.24	0.24
IRWM-K5	0.29	0.24	0.24
Median-K5	0.33	0.29	0.29

Thus can be seen that Manhattan has consistent accuracy because there are two models that have the highest value when using the mean and the median of closest analogies compared to Euclidean and Minkowski distance. Though Minkowski has the same value as Manhattan, it only happens on the mean of closest analogies model.

3.3. MdMRE Results

The evaluation results of MdMRE accuracy obtained by Manhattan distance had the lowest MdMRE of 0.26 with *K* = 3 using the mean of closest analogies, Euclidean distance with the lowest MdMRE of 0.31 with *K* = 3 using the median of closest analogies, Minkowski with the lowest MdMRE of 0.30 with *K* = 2 using IRWM. These scores show Manhattan has the best MdMRE accuracy compared to Euclidean and Minkowski as shown by Table 5.

Table 5. MdMRE Results

Model	Manhattan	Euclidean	Minkowski
CA-K1	0.40	0.40	0.40
Mean-K2	0.34	0.34	0.34
IRWM-K2	0.30	0.37	**0.30**
Mean-K3	**0.26**	0.35	0.34
IRWM-K3	0.36	0.36	0.36
Median-K3	0.31	**0.31**	0.31
Mean-K4	0.41	0.47	0.45
IRWM-K4	0.31	0.41	0.41
Median-K4	0.28	0.47	0.33
Mean-K5	0.44	0.45	0.45
IRWM-K5	0.37	0.40	0.39
Median-K5	0.35	0.46	0.44

3.4. Absolute Residual Results

Good estimation accuracy is directly related to how well the absolute residual (AR) value is. Absolute residual is the absolute difference between actual and estimated effort. The smaller the absolute residual value shows the actual estimated value is the same, which means a good sign. Table 6 shows the absolute residual score from Euclidean, Manhattan and Minkowski distance using three-fold cross validation.

Table 6. Absolute Residual results

	Euclidean Distance			Manhattan Distance			Minkowski Distance		
	MRE	*Absolute Residual*	*Actual Effort*	*MRE*	*Absolute Residual*	*Actual Effort*	*MRE*	*Absolute Residual*	*Actual Effort*
Set 1	1.05	1155.9	1100	0.80	880.4	1100	0.80	835.4	1100
	0.31	1744.6	5578	0.20	1217.4	5578	0.22	1700.6	5578
	0.50	533.9	1060	0.60	598.2	1060	0.56	230.8	1060
	0.04	229.7	5279	0.30	1499.7	5279	0.28	1927.4	5279
	0.34	2724.9	8117	0.20	1444.3	8117	0.18	2977.5	8117
	1.90	**16589.5**	**8710**	**1.10**	**9961.1**	**8710**	**1.14**	**15883.2**	**8710**
	0.36	287.4	796	0.40	319.4	796	0.40	186.8	796
Set 2	**1.81**	**10752.0**	**5931**	**2.70**	**16228.5**	**5931**	**2.74**	**4671.2**	**5931**
	0.42	1885.5	4456	0.90	3893.7	4456	0.87	791.4	4456
	0.01	24.0	3600	0.00	90.0	3600	0.02	144.2	3600
	0.41	1872.7	4557	0.90	3908.9	4557	0.86	2411.4	4557
	0.10	898.5	8752	0.20	1622.5	8752	0.19	4067.4	8752
	0.72	2491.0	3440	0.70	2267.3	3440	0.66	2428.5	3440
	0.18	356.0	1981	0.00	35.8	1981	0.02	97.2	1981
Set 3	0.11	1044.0	9125	0.00	453.2	9125	0.05	2293.5	9125
	0.07	1922.0	25910	0.20	4750.2	25910	**0.18**	**5866.6**	**25910**
	0.19	2842.5	15052	**0.30**	**4995.9**	**15052**	0.33	5819.4	15052
	0.48	871.2	1798	0.60	1034.4	1798	0.58	1763.6	1798
	0.08	455.0	5787	0.00	172.0	5787	0.03	1309.7	5787
	0.31	**3430.0**	**11023**	0.10	1402.4	11023	0.13	2654.1	11023
	0.24	423.8	1755	0.20	342.4	1755	0.20	21.2	1755
MMRE	0.46			0.50			0.44		
MdMRE	0.31			0.28			0.30		
Pred(25)	0.42			0.48			0.43		

Set 1 for Euclidean distance shows the highest AR of 16589 man-hours with an actual effort of 8710 man-hours, indicates a wide enough difference between actual effort and estimation. MRE of this project is 1.90, means there is an error of 190% in the estimation relative to actual effort. Set 2 shows the largest AR value is 10725 man-hours with an actual effort of 5931 man-hours which indicates a wide enough difference between the actual effort and the estimate. Project's MRE is 1.81 which means there is a 181% error in the estimation relative to the actual effort. The largest AR value is 3430 man-hours with an actual effort 11023 man-hours which indicates a very slight difference between actual effort and effort estimation. MRE of this project is 0.31 which means there is an error in the estimation effort of 31% relative to the actual effort. Means that the model in set 3 is the best estimation model with Euclidean distance parameters.

Set 1 for Manhattan distance shows the highest AR of 9961.1 man-hours with an actual effort of 8710 man-hours which indicates a very large difference between actual effort and estimation. MRE project is 1.1, means there is an estimated error of 110% relative to actual effort. Set 2 shows the largest AR value is 16228.5 man-hours with an actual effort of 5931 man-hours which indicates a slight difference between actual effort and estimation. MRE is 2.7, means there is a 270% error in the estimation relative to the actual effort. Set 3 shows the largest AR value is 4995.9 man-hours with an actual effort of 15052 man-hours which indicates a very slight difference between actual effort and effort estimation. MRE of this project is 0.3, means there is an estimated error of 30% relative to the actual effort. Model in set 3 is the best estimation model in Manhattan distance parameters.

Set 1 for Minkowski distance shows the highest AR of 15883.2 man-hours with an actual effort of 8710 man-hours which indicates a wide enough difference between actual effort and effort estimation. MRE of this project is 1.14 which means there is an error of 114% in the estimation relative to the actual effort. Set 2 shows the largest AR value is 4671.2 man-hours with an actual effort of 5931 man-hours which indicates a very large difference between the actual effort and the estimation. MRE of this project is 2.74 which means there is an estimated error of 274% relative to the actual effort. Set 3 shows the largest AR value is 5866.64 man-hours with an actual effort of 25910 man-hours which indicates a very small difference between actual effort and effort estimation. MRE of this project is 0.18 which means there is an estimated error of 18% relative to the actual effort. The model in set 3 is concluded as the best estimation model on the Minkowski distance parameter.

3.5. Model Comparison

The last stage is comparing the accuracy between models using Manhattan distance parameters with the research conducted by Idri [7]. As shown in Fig. 3, the accuracy of MMRE, MdMRE and Pred (25) at Manhattan distance are 50%, 28% and 48% respectively. While Idri has an accuracy of 49.9% for MMRE, 29.37% for MdMRE and 51.23% for Pred (25). Based on these comparisons, MMRE and MdMRE and Pred (25) have a very slight difference in accuracy. On the other hand, it also can be concluded that Manhattan and Idri have almost similar results of accuracy.

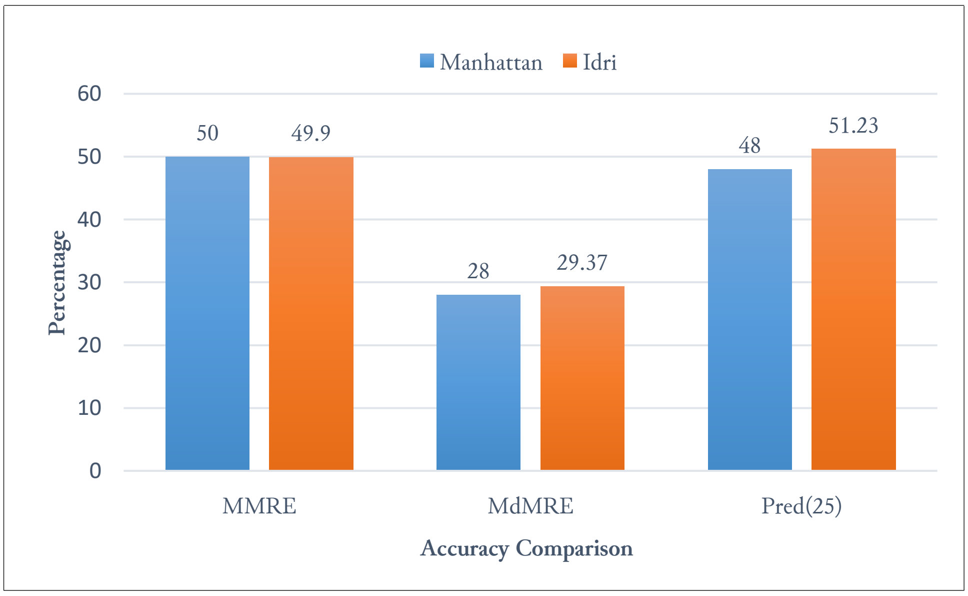

Fig. 3. Accuracy comparison between Manhattan and Idri

4. Conclusion

Analogy-based estimation requires past project history data as an analogous. Accuracy of effort estimation is very dependent on the similarity of the project history data. In addition to rigorous data, other problems that fluctuate the accuracy of the overall analogy are the number of selected analogies, distance measurements, and solution adaptation. The similarity of project to be estimated is the key to improving the accuracy of the Analogy-based estimation. This paper proposes Analogy as an estimation model of the effort of software by adjusting three distance measurements, namely Euclidean, Manhattan and Minkowski distance. The results of the evaluation of the accuracy of all three have been described in this article. The best results are obtained with Manhattan distance with a 50% MMRE, 28% MdMRE and Pred(25) at 48%. These results are not as far off as observed by [18] that the mean accuracy of the analogy method is MMRE 49.9%, MdMRE 29.37% and Pred(25) 51.23%.

Acknowledgment

This research is supported by "Penelitian Hibah Bersaing Universitas Ahmad Dahlan (PHB-UAD)" in academic year 2017/2018 with the contract number of PHB-028/SP3/LPPM-UAD/VI/2018.

References

[1] A. Dennis, B. H. Wixom, and D. Tegarden, *Systems analysis and design: An object-oriented approach with UML*, 5th ed. Wiley, 2015, available at: Google Scholar.

[2] M. Cohn, *Agile estimating and planning*. Prentice Hall, 2006, available at: Google Scholar.

[3] A. J. Albrecht and J. E. Gaffney, "Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation," *IEEE Trans. Softw. Eng.*, vol. SE-9, no. 6, pp. 639–648, Nov. 1983, doi: https://doi.org/10.1109/TSE.1983.235271.

[4] B. W. Boehm et al., *Software Cost Estimation with Cocomo II with Cdrom*, 1st ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2000, available at: https://dl.acm.org/citation.cfm?id=557000.

[5] L. Laird and M. C. Brennan, *Software Measurement and Estimation: A Practical Approach*. Wiley, 2006, doi: https://doi.org/10.1002/0471792535.

[6] C. F. Kemerer, "An empirical validation of software cost estimation models," *Commun. ACM*, vol. 30, no. 5, pp. 416-429, 1987, doi: https://doi.org/10.1145/22899.22906.

[7] Y. Miyazaki, M. Terakado, K. Ozaki, and H. Nozaki, "Robust regression for developing software estimation models," *J. Syst. Softw.*, vol. 27, no. 1, pp. 3–16, Oct. 1994, doi: https://doi.org/10.1016/0164-1212(94)90110-4.

[8] F. Yücalar, D. Kilinc, E. Borandag, and A. Ozcift, "Regression Analysis Based Software Effort Estimation Method," *Int. J. Softw. Eng. Knowl. Eng.*, vol. 26, no. 05, pp. 807–826, Jun. 2016, doi: https://doi.org/10.1142/S0218194016500261.

[9] A. Trendowicz and R. Jeffery, *Software Project Effort Estimation: Foundation and Best Practice Guidelines for Success*. Springer, 2014, doi: https://doi.org/10.1007/978-3-319-03629-8.

[10] F. Zare, H. Khademi Zare, and M. S. Fallahnezhad, "Software effort estimation based on the optimal Bayesian belief network," *Appl. Soft Comput.*, vol. 49, pp. 968–980, 2016, doi: https://doi.org/10.1016/j.asoc.2016.08.004.

[11] C. . Dawson, "A Neural Network Approach to Software Project Effort Estimation," *Int. Conf. Appl. Artif. Intell. Eng.*, vol. 16, no. 9, 1996, doi: https://doi.org/ 10.2495/AI960161.

[12] Q. M. Yousef, Y. A. Alshaer, and N. K. Alhammad, "Dragonfly Estimator: A Hybrid Software Projects' Efforts Estimation Model using Artificial Neural Network and Dragonfly Algorithm," *Int. J. Comput. Sci. Netw. Secur.*, vol. 17, no. 9, pp. 108–120, 2017, available at: http://paper.ijcsns.org/07_book/201709/20170916.pdf.

[13] P. L. Braga, A. L. I. Oliveira, and S. R. L. Meira, "A GA-based feature selection and parameters optimization for support vector regression applied to software effort estimation," in *Proceedings of the 2008 ACM symposium on Applied computing - SAC '08*, 2008, p. 1788, doi: https://doi.org/10.1145/1363686.1364116.

[14] S.-J. Huang, C.-Y. Lin, and N.-H. Chiu, "Fuzzy Decision Tree Approach for Embedding Risk Assessment Information into Software Cost Estimation Model," *J. Inf. Sci. Eng.*, vol. 22, pp. 297–313, 2006, available at: https://pdfs.semanticscholar.org/7a1a/34bc97e67a9253debdc7b5b5e9f3ec149b52.pdf.

[15] M. Choetkiertikul, H. K. Dam, T. Tran, T. T. M. Pham, A. Ghose, and T. Menzies, "A deep learning model for estimating story points," *IEEE Trans. Softw. Eng.*, pp. 1–1, 2018, doi: https://doi.org/10.1109/TSE.2018.2792473.

[16] E. Kocaguneli, T. Menzies, and J. W. Keung, "On the Value of Ensemble Effort Estimation," *IEEE Trans. Softw. Eng.*, vol. 38, no. 6, pp. 1403–1416, Nov. 2012, doi: https://doi.org/10.1109/TSE.2011.111.

[17] C. López-Martín and A. Abran, "Neural networks for predicting the duration of new software projects," *J. Syst. Softw.*, vol. 101, pp. 127–135, 2015, doi: https://doi.org/10.1016/j.jss.2014.12.002.

[18] A. Idri, F. A. Amazal, and A. Abran, "Analogy-based software development effort estimation: A systematic mapping and review," *Inf. Softw. Technol.*, vol. 58, pp. 206–230, 2015, doi: https://doi.org/10.1016/j.infsof.2014.07.013.

[19] D. J. Reifer, B. Boehm, and S. Chulani, "The Rosetta Stone: Making COCOMO 81 Estimates Work with COCOMO II," *Crosstalk. J. Def. Softw. Eng.*, pp. 11–15, 1999, available at: http://sunset.usc.edu/TECHRPTS/1998/usccse98-516/usccse98-516.pdf.

[20] I. Limited, "isbsg10," Nov-2012, doi: https://doi.org/10.5281/zenodo.268485.

[21] Y. Li, "Effort Estimation: Maxwell," Mar-2009, doi: https://doi.org/10.5281/zenodo.268461.

[22] J. W. Keung, "kemerer," Apr-2010, doi: https://doi.org/10.5281/zenodo.268464.

[23] I. Limited, "Effort Estimation: Cosmic," Nov-2012, doi: https://doi.org/10.5281/zenodo.268482.

[24] N.-H. Chiu and S.-J. Huang, "The adjusted analogy-based software effort estimation based on similarity distances," *J. Syst. Softw.*, vol. 80, no. 4, pp. 628–640, Apr. 2007, doi: https://doi.org/10.1016/j.jss.2006.06.006.

[25] M. Shepperd and C. Schofield, "Estimating software project effort using analogies," *IEEE Trans. Softw. Eng.*, vol. 23, no. 11, pp. 736–743, Nov. 1997, doi: https://doi.org/10.1109/32.637387.

[26] E. Mendes, *Cost Estimation Techniques for Web Projects*. IGI Global, 2008, doi: https://doi.org/10.4018/978-1-59904-135-3.

[27] M. Azzeh, A. B. Nassif, and L. L. Minku, "An empirical evaluation of ensemble adjustment methods for analogy-based effort estimation," *J. Syst. Softw.*, vol. 103, pp. 36–52, May 2015, doi: https://doi.org/10.1016/j.jss.2015.01.028.

[28] M. Hosni and A. Idri, "Software effort estimation using classical analogy ensembles based on random subspace," in *Proceedings of the Symposium on Applied Computing*, 2017, pp. 1251–1258, doi: https://doi.org/10.1145/3019612.3019784.

[29] S. Mensah, J. Keung, M. Bosu, K. E. Bennin, and P. K. Kudjo, "A Stratification and Sampling Model for Bellwether Moving Window," 2017, pp. 481–486, available at: http://ksiresearchorg.ipage.com/seke/seke17paper/seke17paper_126.pdf.

[30] Y. F. Li, M. Xie, and T. N. Goh, "A study of mutual information based feature selection for case based reasoning in software cost estimation," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 5921–5931, Apr. 2009, doi: https://doi.org/10.1016/j.eswa.2008.07.062.

[31] M. Tsunoda, T. Kakimoto, A. Monden, and K. Matsumoto, "An empirical evaluation of outlier deletion methods for analogy-based cost estimation," in *Proceedings of the 7th International Conference on Predictive Models in Software Engineering*, 2011, pp. 1–10, doi: https://doi.org/10.1145/2020390.2020407.