Use Case Points based software effort prediction using regression analysis

By Ardiansyah

Use Case Points based software effort prediction using regression analysis

Ardiansyah
Department of Electrical Engineering and Information Technology
Universitas Gadjah Mada
Yogyakarta, Indonesia
ardiansyah2018@mail.ugm.ac.id

Ridi Ferdiana
Department of Electrical Engineering and Information Technology
Universitas Gadjah Mada
Yogyakarta, Indonesia
ridi@ugm.ac.id

Adhistya Erna Permanasari
Department of Electrical Engineering and Information Technology
Universitas Gadjah Mada
Yogyakarta, Indonesia
adhistya@ugm.ac.id

Department of Informatics, Faculty of Industrial Technology, Universitas Ahmad Dahlan Yogyakarta, Indonesia
ardiansyah@tif.uad.ac.id

***Abstract*— Software development effort prediction was an important stages in project planning. Poor prediction would lead to project failure, losing tenders and reduced profits. Several studies have improved Use Case Points as the effort prediction model using regression analysis. However, evaluation performance on the prediction models were biased and produce an asymmetric error distribution. Moreover, the dataset used were primarily from industrial, and less from universities. This study aims to investigate the performance of the regression model in terms of software development effort prediction based on Use Case Points using standardized accuracy (SA) and effect size (Δ) as the evaluation measurement. From the experiment results, regression model yielded 92% - 0.64, 96% - 1.86, and 69% - 0,53 in term of SA and (Δ) over dataset DS1, DS3, and DS4, respectively. Experiment results shows that regression model yielded the best accuracy compared with the Karner model over three dataset. In the future, our results maybe used in development of effort prediction framework for calculating software project costs.**

Keywords—effort prediction, regression, standardized accuracy, effect size, use case points

I. Introduction

Software industry was sixty-four years old after the founding of Computer Usage Company (CUC) in 1955 [1]. The industry has continued to grow fantastically and predicted by Gartner's business consulting in 2022 to reach a capitalization of $310.2 billion [2]. These conditions indicated that the software industry was a very important sector globally. However, time and cost overruns were serious problems for software organizations [3], [4]. According to a report released by [5], software project with a budget of more than $15 million, 66% over budget, and 33% over time. Consequently, the software was released longer and the costs needed are greater. Moreover, if the project was carried out over time and budget the existence of an organization can be threatened [5]. Therefore, the software project will succeed if the functions created are aligned with user requirements, met the set of quality standards, and completed on time and budget.

Software development effort predictions were a solution for solving these problems. Effort prediction was an important stages in software project planning. Project planning was consists of allocating costs, duration, and team [6]. Poor estimates would lead to project failure, losing tenders and reduced profits [7]. Conversely, if the prediction was accurate, the project will be completed on time, lower cost, and reduce the potential loss of profits. In other words, accurate predictions have the potential for higher project success.

Use Case Points (UCP) was one of software effort prediction methods for an object-oriented development paradigm. UCP was constructed based on functional requirements modeled through a use case diagram. Reference [8] first proposed the method and gained wide attention from the industry and academia until now. UCP calculated effort based on software size and fixed productivity factors (20 person-hours). However, original UCP has widely criticized because of having low accuracy performance [9]. Moreover, this method impractical because of ignoring several software project factors such as complexity, type, domain, and environmental.

Several studies have improved UCP using regression analysis. Reference [10], [11] proposed a non-linear regression and used a dataset from industry. Reference [12] proposed a multiple linear regression (MLR) analysis using industrial datasets. Reference [13] used MLR, support vector regression, and regression trees without productivity factors as a predictor using industrial dataset. Simple linear regression and MLR were employed by [14] to evaluate the proposed prediction model. Reference [15] investigated the significance of UCP variables using MLR using industrial dataset. Moreover, [16] applied least square regression to predict a value of correction parameters, and finally [9] investigates the significance of using subset selection methods for the prediction accuracy of MLR.

There are two important characteristics found in these studies. First, most of the dataset used is typically from industrial organizations. Whereas, beside industrial there was another source that is universities. Second, most of the methods were evaluated using the mean magnitude of relative error (MMRE). Unfortunately, this popular prediction accuracy statistic is a biased estimator of the central tendency of the residuals of a prediction system because it is an asymmetric measure [17]. Another studies

None of above studies emphasized the use of datasets from universities. Likewise, very few studies that use accuracy statistic technique other than MMRE. This study aims to investigate the performance of the regression model in terms of software development effort prediction based on Use Case Points using evaluation framework proposed by [17]. The model ignored productivity factor as one of the independent variables and used software size only. Moreover, most of the dataset came from universities, while the rest came from industry.

II. Theoretical Background

Regression analysis is a statistical method for analyzing the relationship between dependent variables with one or more independent variables [18]. In terms of software effort predictions, the effort is a dependent variable, while software size is an independent variable. UCP is the unit of software size. Equation (1) shows formal notation for the regression model.

$$Y = a + b * X \tag{1}$$

where Y is equal to effort as a dependent variable, and X is equal to size as an independent variable. b was regression coefficient, while a was intercept or constant parameter. When (1) is associated with Use Case Points, the equation becomes (2).

$$effort = a + b * size \tag{2}$$

The regression model requires data normally distributed. If the data is not normal, then the data must be normalized first. Logarithmic is one of the normalization techniques for regression analysis. Equation (3) shows the equation for the logarithmic technique.

$$Y = a + b * ln(X) \tag{3}$$

Thus, we can translate (3) into UCP based equation as stated in (4).

$$ln(effort) = a + b * ln(size) \tag{4}$$

Karner proposed UCP in 1993 which consisted of six steps. First, determine Unadjusted Actor Weighting (UAW). Second, determine Unadjusted Use Case Weighting (UUCW). Third, multiply UAW and UUCW to get Unadjusted Use Case Points (UUCP). Fourth, Determine Technical complexity factors (TCF) and environmental complexity factors (ECF). Fifth, multiply UUCP, TCF, and ECF to get use case point (UCP) size. Finally, use fixed productivity factors (20 person-hours) and multiply it to UCP size to get effort estimation in person-hours. Fig. 1 shows the original Karner use case points effort estimation framework.

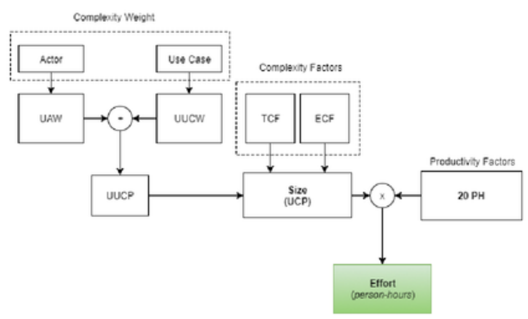

Fig. 1. Use Case Points framework

III. Methodology and Experimental Setup

The experimental procedure was carried out in four stages, which are data collection, normality test, model validation, and evaluation.

A. Dataset

This study collected four datasets whose project employed Use Case Points as the effort estimation model. The first dataset (DS1) consists of ten software projects in the field of business such as MLM, sales systems, education and training management, IDs System electronic vehicles, labor and workforce systems, online ticketing, building rental systems, mall search engine systems, cookies, food trading system, and data dictionary bank [19]. The second dataset (DS2) consists of fourteen projects developed by software companies, universities, independently developed from scratch, enhanced features and developed by students for outside campus organizations [14]. The third dataset (DS3) consists of eight website development projects [20]. The last dataset (DS4) consists of seven software development projects for educational purpose [21].

DS1, DS3 and DS4 were collected using interviews, questionnaires and document review. Interviews were used to obtain the number of team members and the duration of the project. Questionnaires were used to obtain factors that influence the project. There were two factors identified, technical and environmental factors. Document review was used to obtain a list of use cases and the number of actors on each project.

B. Normality test

Due to the data was less than fifty, then the Shapiro-Wilk test has been chosen. If the significance value was greater than 0.05, then the normality requirement has been fulfilled. Whereas, if the significance value less than 0.05, then the normality requirement has not been fulfilled, so that the data have to transform until normally distributed. If the data were still not normally distributed, then regression analysis can not be used.

C. Validation

Leave one-out cross validation (LOOCV) technique was employed to validate the models. LOOCV takes each project as a test set, while the rest was used as a training set. Each test data entered the prediction model to obtained predicted effort. Each time the model successfully predicted the effort, accuracy would be able to calculate. The difference between using LOOCV compared with other n-fold cross-validation techniques is that LOOCV uses deterministic procedures that can be easily applied in other studies with various datasets. LOOCV produces lower estimation bias and higher variance values [22], especially for relatively small dataset [23]. Moreover, LOOCV ensures that any prediction model constructed from the same set of training data. To verify the performance of the regression model, it compared with Karner model [8].

D. Evaluation

Prediction models must be evaluated using reliable accuracy measurement techniques. The measurement results must be unbiased and do not produce an asymmetric error distribution [22]. The measurement was based on Absolute Error (AE). From AE we can measure another metric Mean Absolute Error (MAE), Mean Balanced Relative Error (MBRE), Mean Inverted Balanced Relative Error (MIBRE), respectively. The minimum score shows the best prediction model performance.

Besides the four measure, we also used the evaluation framework proposed by [17]. The framework consists of two metrics, standardized accuracy (SA) and effect size (Δ). SA was used to evaluate whether the prediction model produces meaningful predictions or not, and the value must be better than the prediction model derived from random guessing. Similarly, the effect size is used to ensure the results does not produce by chance. The larger metric score indicates good model performance.

Finally, the significance test was carried out using t-Test and Mann-Whitney. Both tests employ AE values from each prediction model based on normality check. If AE is normally distributed, then use the t-test. Otherwise, use Mann-Whitney instead.

TABLE I. SIX ACCURACY MEASUREMENTS FORMULA

Accuracy Metrics	Formula
Absolute Error (AE)	$AE_i = \lvert y_i - \hat{y}_i \rvert$
Mean Absolute Error (MAE)	$\frac{1}{n}\sum_{i=1}^{n} \lvert y_i - \hat{y}_i \rvert$
Mean Balanced Relative Error (MBRE)	$MBRE = \frac{1}{n}\sum_{i=1}^{n} \frac{AE_i}{\min(y_i, \hat{y}_i)}$
Mean Inverted Balanced Relative Error (MIBRE)	$MIBRE = \frac{1}{n}\sum_{i=1}^{n} \frac{AE_i}{\max(y_i, \hat{y}_i)}$
Standardized Accuracy (SA)	$SA = 1 - \frac{MAE_{P_i}}{\overline{MAE}_{P_0}}$
Effect Size (Δ)	$\Delta = \frac{MAE_{P_i} - \overline{MAE}_{P_0}}{S_{P_0}}$

Where y_i and $\hat{y}_i$ is actual effort and predicted effort from a single case of project. S_{P_0} is the standard deviation from the random guessing prediction model. $\overline{MAE}_{P_0}$ is the mean value of a large number runs of random guessing. This defined as, predict a $\hat{y}$ for the target case t by randomly sampling (with equal probability) over all the remaining n - 1 cases and take $\hat{y}_t = y_r$ where r is drawn randomly from $1..n \wedge r \neq t$.

IV. Results and Discussions

This section describes empirical results from model validation and comparison with another model. We validated the two models to obtain the value of SA and Δ. The objective of validation was to ensure that the proposed model produces meaningful prediction and better than random guessing. Another objective was to ensure that all results have been produced by predictions, not by chance. Table II shows the results of SA and ES over the four datasets using random guessing as to the baseline model. The model with greater SA indicates a more reliable model and produce a more meaningful prediction. Furthermore, the model with greater effect size indicates that prediction results unlikely produced by chance. From Table II, we can observe that SA for regression model outperformed the random guessing over DS1, DS3, and DS4. Whereas, Karner model superior over DS2 only. The quality of data in the original datasets (DS1, DS3, and DS4) contributed significantly to the obtained performance. DS1, DS3, and DS4 has the same characteristics, data collection technique, and also used regression to analyzing the productivity factor. Moreover, the datasets were collected by the students who conducted research under the same supervisor. Therefore, it was not surprising when the regression model showed better performance. This result contrast with analysis by [22] who stated that datasets collected by a practitioner in an organization give better result than collected by students in universities. Consequently, we can conclude that it does not matter collected by whom, as long as the datasets were collected using a proper manner.

The better performance showed by the Karner model than regression over DS2 was also not surprising. DS2 was constructed by using the Karner model and used fixed productivity factor (20 person-hours). Irrespectively, the SA score (>0%) showed that both models (regression and Karner) were able to produce better meaningful predictions than random guessing. Nevertheless, rely on SA alone cannot give us the full picture of the accuracy superiority. Therefore, we needed to use the effect size to build the final decision. Effect size is a metric which able to show the meaningfulness of both prediction models. Table II shows that effect size is considerably reasonable over all datasets. This result indicates a good improvement against random guessing.

TABLE II. SA AND Δ ANALYSIS OF REGRESSION AND KARNER MODEL, CONSIDERING RANDOM GUESSING AS THE BASELINE

Dataset		Karner	Regression
DS1	SA	26%	88%
	Δ	0.26	1.06
DS2	SA	2.13%	1.20%
	Δ	20.9	0.48
DS3	SA	20%	65%
	Δ	0.45	0.58
DS4	SA	23%	51%
	Δ	0.23	0.62

Table III shows the number of accuracy improvements in term of SA and effect size on the regression model, considering the Karner model as the baseline. Interestingly, regression model generates better improvement over DS1, DS3, and DS4 with good SA. Regression model generates fair improvement over DS2 and DS3 with good effect size. Surprisingly, the SA results in DS2 were bad, perhaps due to the effect of outliers in size variable.

TABLE III. SA AND Δ ANALYSIS OF REGRESSION MODEL, CONSIDERING KARNER MODEL AS THE BASELINE

Dataset	Karner as the baseline	
DS1	SA	92%
	Δ	0.64
DS2	SA	-65%
	Δ	10.39
DS3	SA	96%
	Δ	1.86
DS4	SA	69%
	Δ	0.53

Table IV shows accuracy results with respect to MAE, MBRE, and MIBRE. These measures have used because they behave very differently from each other, and they can effectively evaluate how well a model performed. The result showed that regression performed better than the Karner model over DS1, DS3, and DS4. Whereas, the Karner model performed better than regression over DS2, which suggested that regression outperformed the Karner model.

TABLE IV. MAE, MBRE, AND MIBRE RESULTS

Dataset	Karner			Regression		
	MAE	MBRE	MIBRE	MAE	MBRE	MIBRE
DS1	1268.2	0.430	0.281	103.533	0.046	0.043
DS2	2.4363	0.547	0.349	4.02134	1.406	0.582
DS3	3138.5	1.273	0.552	123.498	0.055	0.051
DS4	1262.1	0.417	0.259	388.884	0.135	0.112

To justify whether the results obtained were significant or not, we used t-test and Mann-Whitney based on absolute residual (AE) at a significant level of 0.05. The statistical test results are shown in Table V. DS1, DS2, and DS3 showed significant results. Whereas, DS4 is not significant. We can generally notice that the regression model generates statistically different and better prediction than the Karner model over the three datasets. Meanwhile, the Karner model generates statistically different and better prediction than the regression model over one dataset.

TABLE V. STATISTICAL SIGNIFICANCE TEST RESULTS

Dataset	Sig. value	Technique	Result
DS1	0.00 < 0.05	Mann-Whitney	Significant
DS2	0.00 < 0.05	Independent sampel t-test	Significant
DS3	0.01 < 0.05	Independent sampel t-test	Significant
DS4	0.08 > 0.05	Independent sampel t-test	Not Significant

V. Conclusion

In this article we presented the performance of regression model in term of software project effort prediction. In the first phase we collected three datasets from universities and one dataset from industry. The dataset then was tested for normality checked. For evaluation purposes, the regression model is compared with the Karner model using leave one out cross validation (LOOCV) technique. MAE, MBRE, MIBRE, SA, and effect size were used to measure model performance. The results obtained were promising and show better improvements over random guessing.
There was a reason that contributed to the regression model to produced good performance results. This result was because the dataset used has the same characteristics, data collection technique, and also employed regression to analyzed productivity factor.
In term of SA and effect size, the results gained by the regression model are encouraging and show significant improvement over the Karner model (SA=96%, Δ=1.86). We can conclude that when estimation is conducted in enough time, carefully, and use proper techniques the performances tends to be better. For future works, we will collect more datasets from universities. We want to investigate the prediction models performance using a large dataset.

Acknowledgment

Ardiansyah thanks to the Universitas Ahmad Dahlan for their valuable sponsorship that help us to publish this work.

References

[1] J. W. Sheldon, "Recollections of the First Software Company," *IEEE Ann. Hist. Comput.*, vol. 16, no. 2, pp. 65–71, 1994.

[2] N. Gupta *et al.*, "Forecast: Enterprise Application Software, Worldwide, 2016-2022, 1Q18 Update," *Research*, 2018. [Online]. Available: https://www.gartner.com/doc/3869907/forecast-enterprise-application-software-worldwide. [Accessed: 23-Apr-2019].

[3] S. Grimstad, M. Jørgensen, and K. Moløkken-Østvold, "Software effort estimation terminology: The tower of Babel," *Inf. Softw. Technol.*, vol. 48, no. 4, pp. 302–310, 2006.

[4] P. Pospieszny, B. Czarnacka-Chrobot, and A. Kobylinski, "An effective approach for software project effort and duration estimation with machine learning algorithms," *J. Syst. Softw.*, vol. 137, pp. 184–196, 2018.

[5] M. Bloch, S. Blumberg, and J. Laartz, "Delivering large-scale IT projects on time, on budget, and on value," *McKinsey Digital*, 2012. [Online]. Available: https://mck.co/2yXnefx. [Accessed: 23-May-2019].

[6] M. Azzeh, A. B. Nassif, and S. Banitaan, "Comparative analysis of soft computing techniques for predicting software effort based use case points," *IET Softw.*, vol. 12, no. 1, pp. 19–29, 2017.

[7] A. W. M. M. Parvez, "Efficiency factor and risk factor based use case point test effort estimation model compatible with agile software development," in *2013 International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2013, pp. 113–118.

[8] G. Karner, "Resource Estimation for Objectory Projects," University of Linköping, 1993.

[9] R. Silhavy, P. Silhavy, and Z. Prokopova, "Evaluating subset selection methods for use case points estimation," *Inf. Softw. Technol.*, vol. 97, no. June 2017, pp. 1–9, 2018.

[10] A. B. Nassif, D. Ho, and L. F. Capretz, "Regression Model for Software Effort Estimation Based on the Use Case Point Method," *Int. Conf. Comput. Softw. Model.*, vol. 14, no. January, pp. 117–121, 2011.

[11] A. B. Nassif, D. Ho, and L. F. Capretz, "Towards an early software estimation using log-linear regression and a multilayer perceptron model," *J. Syst. Softw.*, vol. 86, no. 1, pp. 144–160, Jan. 2013.

[12] 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.

[13] M. Azzeh and A. B. Nassif, "Project productivity evaluation in early software effort estimation," *J. Softw. Evol. Process*, vol. 30, no. 12, pp. 1–12, 2018.

[14] M. Ochodek, J. Nawrocki, and K. Kwarciak, "Simplifying effort estimation based on Use Case Points," *Inf. Softw. Technol.*, vol. 53, no. 3, pp. 200–213, Mar. 2011.

[15] R. Silhavy, P. Silhavy, and Z. Prokopova, "Analysis and selection of a regression model for the Use Case Points method using a stepwise approach," *J. Syst. Softw.*, vol. 125, pp. 1–14, 2017.

[16] R. Silhavy, P. Silhavy, and Z. Prokopova, "Applied Least Square Regression in Use Case Estimation Precision Tuning," no. April, pp. 10–17, 2015.

[17] M. Shepperd and S. Macdonell, "Evaluating Prediction Systems in Software Project Estimation," *Inf. Softw. Technol.*, vol. 54, no. 8, 2012.

[18] A. Trendowicz and R. Jeffery, *Software Project Effort Estimation: Foundation and Best Practice Guidelines for Success*. Springer, 2014.

[19] Subriadi, A. Pribadi, and P. A. Ningrum, "Critical Review of the Effort Rate Value in Use Case Point Method for Estimating Software Development Effort," vol. 59, no. 3, pp. 735–744, 2014.

[20] Sholiq, T. Sutanto, A. P. Widodo, and W. Kurniawan, "Effort Rate on Use Case Point Method for Effort Estimation of Website Development," *J. Theor. Appl. Inf. Technol.*, vol. 63, no. 1, pp. 209–218, 2014.

[21] I. D. Kenestie and Sholiq, "Determining Effort Rate (ER) Value for Use Case Points based Educational Software Development Effort Estimation," *J. Tek. POMITS*, pp. 1–11, 2011.

[22] M. Azzeh and A. B. Nassif, "A hybrid model for estimating software project effort from Use Case Points," *Appl. Soft Comput. J.*, vol. 49, pp. 981–989, 2016.

[23] Q. Li, Q. Wang, Y. Yang, and M. Li, "Reducing biases in individual software effort estimations," in *Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement - ESEM '08*, 2008, p. 223.

Use Case Points based software effort prediction using regression analysis

ORIGINALITY REPORT

19%

SIMILARITY INDEX

PRIMARY SOURCES

1	Mohammad Azzeh, Ali Bou Nassif. "A hybrid model for estimating software project effort from Use Case Points", Applied Soft Computing, 2016 Crossref	198 words — 7%
2	aut.researchgateway.ac.nz Internet	68 words — 2%
3	Mohammad Azzeh, Ali Bou Nassif. "Analyzing the relationship between project productivity and environment factors in the use case points method", Journal of Software: Evolution and Process, 2017 Crossref	39 words — 1%
4	Yulia Ery Kurniawati, Adhistya Erna Permanasari, Silmi Fauziati. "Adaptive Synthetic-Nominal (ADASYN-N) and Adaptive Synthetic-KNN (ADASYN-KNN) for Multiclass Imbalance Learning on Laboratory Test Data", 2018 4th International Conference on Science and Technology (ICST), 2018 Crossref	34 words — 1%
5	insightsociety.org Internet	18 words — 1%
6	ir.lib.uwo.ca Internet	16 words — 1%
7	dspace.vsb.cz Internet	16 words — 1%

8 Pichai Jodpimai, Peraphon Sophatsathit, Chidchanok Lursinsap. "Re-estimating software effort using prior phase efforts and data mining techniques", Innovations in Systems and Software Engineering, 2018
Crossref
16 words — 1%

9 Software Project Effort Estimation, 2014.
Crossref
14 words — < 1%

10 link.springer.com
Internet
14 words — < 1%

11 Adhi Prahara, Dewi Pramudi Ismi, Achmad Imam Kistijantoro, Masayu Leylia Khodra. "Parallelized k-means clustering by exploiting instruction level parallelism at low occupancy", 2017 2nd International conferences on Information Technology, Information Systems and Electrical Engineering (ICITISEE), 2017
Crossref
13 words — < 1%

12 www.ijetr.org
Internet
12 words — < 1%

13 Ali Idri, Ibtissam Abnane, Alain Abran. " Evaluating Pred( ) and standardized accuracy criteria in software development effort estimation ", Journal of Software: Evolution and Process, 2018
Crossref
11 words — < 1%

14 Ali Idri, Ibtissam Abnane. "Fuzzy Analogy Based Effort Estimation: An Empirical Comparative Study", 2017 IEEE International Conference on Computer and Information Technology (CIT), 2017
Crossref
11 words — < 1%

15 repository.tudelft.nl
Internet
10 words — < 1%

16 publikace.k.utb.cz
Internet
10 words — < 1%

Swarnima Singh Gautam, Vrijendra Singh. "The state-of-the-art

17	in software development effort estimation", Journal of Software: Evolution and Process, 2018 Crossref	9 words — < 1%
18	Mohamed Hosni, Ali Idri, Alain Abran. "Evaluating filter fuzzy analogy homogenous ensembles for software development effort estimation", Journal of Software: Evolution and Process, 2018 Crossref	9 words — < 1%
19	Saraswathi, S., and N. Kannan. "A Hybrid Associative Classification Model for Software Development Effort Estimation", Circuits and Systems, 2016. Crossref	9 words — < 1%
20	springerplus.springeropen.com Internet	8 words — < 1%
21	Idri, Ali, Mohamed Hosni, and Alain Abran. "Systematic literature review of ensemble effort estimation", Journal of Systems and Software, 2016. Crossref	8 words — < 1%
22	publications.lib.chalmers.se Internet	8 words — < 1%
23	www.rroij.com Internet	8 words — < 1%
24	Mohammad Azzeh, Ali Bou Nassif. "Project productivity evaluation in early software effort estimation", Journal of Software: Evolution and Process, 2018 Crossref	7 words — < 1%

EXCLUDE QUOTES	OFF	EXCLUDE MATCHES	OFF
EXCLUDE BIBLIOGRAPHY	ON		