Analyzing Correlation of the relationship between Technical Complexity Factors and Environmental Complexity Factors for Software Development Effort Estimation

Ho Le Thi Kim Nhung[1], Vo Van Hai[1] and Huynh Thai Hoc[1]

[1] Tomas Bata University in Zlin, Faculty of Applied Informatics, Nad Stranemi 4511, Zlin 76001, Czech Republic
{lho, vo_van, huynh_thai}@utb.cz

Abstract. In this paper, a new method called Correlation-based Feature Selection in Correction Factors is proposed. The method is based on the feature selection method used in software development effort estimation to reduce redundant correction factors. In this paper, the impact of correlation-based feature selection on the method's estimation accuracy is investigated. Multiple linear regression was used as the basic technique for the correction factors preprocessed by the feature selection method. The results were evaluated using six unbiased accuracy measures through the 5-fold cross-validation over the historical dataset. The proposed method leads to a significant improvement in estimation accuracy by simplifying the evaluation of correction factor values in the use case points method, thus increasing the usefulness of the proposed method in practice.

Keywords: Software Development Effort Estimation, Use Case Points, Correlation-based Feature Selection, Multiple Linear Regression.

1 Introduction

Software Development Effort Estimation (further only SDEE) is a critical factor in the early stages of the software development life cycle. SDEE can help project managers make early software development decisions, i.e., to prepare the necessary project plan and budget within the expected completion time [1], [2]. The application of appropriate SDEE methods determines the success or failure of a software project. In this context, various approaches have been proposed and evaluated to address this problem. SDEE approaches can be categorized as expert judgment, parametric techniques, and machine learning techniques [3].

The Use Case Points (further only UCP) method can be used as a functional size metric in the early stages of the software lifecycle [4]. The UCP method is based on a Use Case Model (further only UCM) structured scenario and actor analysis. The method estimates effort based on software size and fixed productivity factors (20 person-hours). However, the original UCP method was analyzed for its low precision [5]-[8]. Combining ML techniques to produce SDEE models based on the original UCP formula could be a method to improve accuracy. Some estimation approaches [9]-[18] have also explored variant models, especially the use of statistical regression or ML

models to obtain better effort estimation based on historical data. The benefits of using UCP-based effort estimation methods are numerous. These methods have made many advances in reducing the influence of human error during UCM analysis and simplifying the original principles of UCP. We recently conducted a literature review on UCP-based SDEE methods in the context of system development [19]. This review highlighted that there are some challenges in estimating the components of UCP, especially the correction factors - which are thirteen technical complexity factors (TCF) and eight environmental complexity factors (ECF). According to Luis et al. [20], a small variation in the weighting value of the correction factors can dramatically affect the software size. Nassif et al. [21] also pointed out the need to refine correction factors that are directly related to estimates computed by the UCP method.

In the UCP method, not all correction factors are crucial for finding the hidden knowledge among the important data. In many cases, some of the factors under consideration are irrelevant and unnecessary. By selecting appropriate subsets of correction factors, we can reduce the complexity of the UCP method and UCP-based SDEE methods. The motivation for us to evaluate whether or not the influence of the feature selection method on correction factors improves the accuracy of the UCP method. To address these issues, this paper proposes the Correlation-based Feature Selection in Correction Factors (named CFSiCF) method for SDEE. We evaluate the construction of correction factors based on the machine learning technique preprocessed with a feature selection method. The method used is the Correlation-based Feature Selection (further only CFS) [22], [23]. The Multiple Linear Regression (further only MLR) technique is chosen as the base method for the selected correction factors to minimize human error's influence during the analysis of these factors. The research questions are answered as follows:

- RQ1: What is the correlation and benefit of the number of technical complexity and environmental complexity factors for size estimation?
- RQ2: Is the proposed method more estimation accurate than the UCP method and other tested methods?

The main contributions of this study are:

- Evaluating the construction of the best correction factors based on MLR models preprocessed with the CFS algorithm.
- The results achieved by the proposed method are compared with three tested estimation methods to verify their accuracy. The methods are run on the dataset of projects from three data donators. The project by each data donator differs in size (measured in UCP). Unbiased evaluation measures (8-13) were used to evaluate the accuracy of the methods.

The rest of the article is organized as follows: Section 2 introduces the background of the methods used. Section 3 describes the research methodology. Section 4 demonstrates the experimental evaluation. Finally, Section 5 concludes the paper and suggests future work.

2 Background

2.1 Use Case Points

The UCP method was introduced by Karner [4] to estimate the size of object-oriented software projects. The UCP method is calculated by computing four basic size metrics - Unadjusted Actor Weight (UAW), Unadjusted Use Cases Weight (UUCW), Technical Complexity Factors (TCF), and Environmental Complexity Factors (ECF), as shown in Eq. (1).

$$UCP = (UAW + UUCW) \times TCF \times ECF \tag{1}$$

The UAW is calculated by taking the weighted sum of the number of actors in each type, as shown in Eq. (2). The actors are classified based on their complexity (see Table 1).

$$UAW = \sum_{i=1}^{3} a_i \times w_i \tag{2}$$

where, a_i is the number of actors in the i^{th} actor type and w_i is the associate complexity weight for each type.

The UUCW is calculated by taking the weighted sum of number of use cases as shown in Eq. (3). The use cases are classified based on the number of transactions in the use case (see Table 2).

$$UUCW = \sum_{j=1}^{3} uc_j \times w_j \tag{3}$$

where, uc_j is the number of use case in the j^{th} use case type and w_j is the associate complexity weight for each type.

The TCF is calculated from thirteen factors representing the complexity of software projects, as shown in Eq. (4). Table 3 presents the technical factors as defined in the UCP.

$$TCF = 0.6 + (0.01 \times \sum_{i=1}^{13} t_i \times fw_i) \tag{4}$$

where, t_i is the value of complexity factor i, and fw_i is the weight of factor i.

The ECF is calculated from a set of eight factors that describe the non-functional requirements, as shown in Eq. (5). The environmental factors are listed in Table 4.

$$ECF = 1.4 - (0.03 \times \sum_{i=1}^{8} e_i \times ew_i) \tag{5}$$

where, e_i is the value of complexity factor i, and ew_i is the weight of factor i.

Table 1. Actor Classification and Their Weights.

Actor Classification	Complexity Weight
Simple	1
Average	2
Complex	3

Table 2. Use Case Classification and Their Weights**.**

Use Case Classification	Number of Transactions	Complexity Weight
Simple	(0 - 4)	5
Average	<4 -7>	10
Complex	(7 - ∞)	15

Table 3. Technical Complexity Factors

Factor ID	Description	Weight
T1	Distributed System	2
T2	Response Adjectives	2
T3	End-User Efficiency	1
T4	Complex Processing	1
T5	Reusable Code	1
T6	Easy to Install	0.5
T7	Ease of Use	0.5
T8	Portable	2
T9	Easy to Change	1
T10	Concurrent	1
T11	Security Feature	1
T12	Access for Third Parties	1
T13	Special Training Required	1

Table 4. Environmental Complexity Factors

Factor ID	Description	Weight
E1	Familiar with RUP	1.5
E2	Application Experience	0.5
E3	Object-oriented Experience	1
E4	Lead Analyst Capability	0.5
E5	Motivation	1
E6	Stable Requirements	2
E7	Part-time Workers	-1
E8	Difficult Programming Language	2

2.2 **Correlation-based Feature Selection**

Performing feature selection is considered a step of data preprocessing to determine the best subset of features to improve estimation accuracy [24], [25]. Feature selection techniques can be classified: Filter, Wrapper, and Hybrid algorithms. Filter methods select the most relevant features based on the properties of a dataset. In contrast, wrapper methods select the best feature subset based on assessing the effects of various

subsets of features on the performance of an estimation system. Embedded or hybrid methods determine the best feature subset by performing the selection step and model building concurrently or combining filtering and wrapper techniques.

In this study, we use the CFS algorithm, which uses correlation to evaluate a feature subset derived from the Pearson correlation coefficient. The CFS evaluates different subsets of features according to the heuristic evaluation function and selects the best one.

$$Heuristic_{fs} = \frac{nr_{fcc}}{\sqrt{n+n(n-1)r_{ffi}}} \quad (6)$$

where, $Heuristic_{fs}$ is the heuristic value of a feature subset fs containing n features, r_{fcc} is the average feature-class correlation, and r_{ffi} is the average feature-feature intercorrelation [26]. To apply $Heuristic_{fs}$, a correlation matrix and heuristic search are computed to find a best subset of features. In this research, the CFS algorithm is used with Greedy stepwise forward search by the space of attribute subsets.

2.3 Multiple Linear Regression

MLR is one of the types of effort estimation methods used to determine how the dependent variable is related to a change in the independent variables and which independent variable is relevant to the dependent variable [27]. The regression models are built based on historical data. The regression models are then evaluated and compared with alternative models [9]-[18]. The MLR model is shown in Eq. (7):

$$Y = \alpha_0 + \alpha_1 X_1 + \alpha_2 X_2 + \cdots + \alpha_n X_n + \varepsilon \quad (7)$$

where Y is the dependent variable is related to the independent variables $X_1, \ldots, X_n$; α_0 is the intercept parameter; $\alpha_1, \ldots, \alpha_n$ are the regression coefficients; and ε_i are the error residuals.

3 Research Methodology

3.1 Data Description

The methods are evaluated on a dataset of 70 observations (each observation represents a description of a realized software system) from three data donators [27]. All data donators work in different government, health, and business sectors. The projects were installed in Java and C# programming languages. For comparison and estimation accuracy of the methods, a standard of 20 man-hours per UCP [4] is used, without considering the productivity factor (PF - man-hours per 1 UCP). The statistical properties of the dataset are described in Table 5. Fig. 1 shows the boxplot of Real_P20 in the dataset. Real_P20 is the real effort in man-hours divided by productivity.

Table 5. Statistical characteristics of the dataset

	Median man-hours	Median Real_P20	Standard deviation	Minimum Real_P20	Maximum Real_P20	n

Dataset	6406	320.3	33.21	288.75	398.5	70

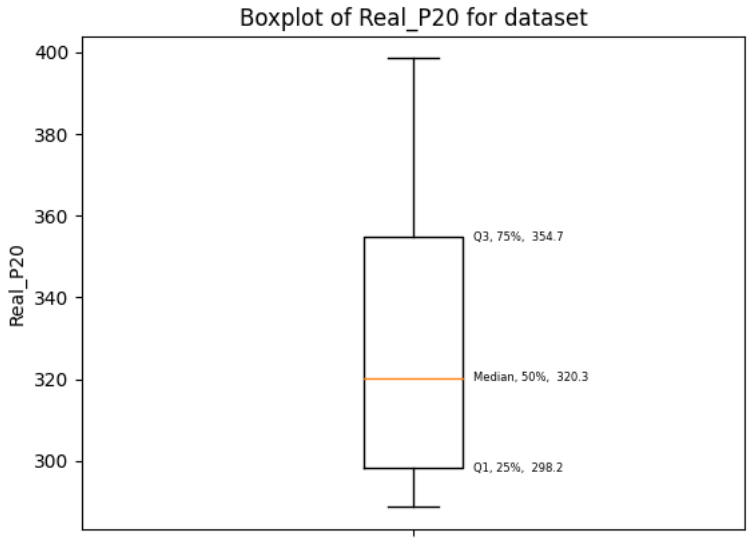

Fig. 1. Boxplot of Real_P20 in the dataset

3.2 **Evaluation Criteria**

To evaluate the accuracy of the methods, we used evaluation criteria commonly used in the field of the software system size estimation [28]. The accuracy measure that used in this paper are Mean Absolute Residual (MAR), Mean of Magnitude of Relative Error (MMRE), Percentage of prediction within x% (PRED(x)), Mean Absolute Percentage Error (MAPE), Sum of Squared Errors (SEE), and the Mean Squared Error (MSE). These measures were chosen because they behave very differently. The best method is the one where MAR, MMRE, MAPE, SSE and MSE are minimized and PRED (x) is maximized.

Mean absolute residual (MAR)

$$MAR = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{8}$$

Mean magnitude of relative error (MMRE)

$$MMRE = \frac{1}{n}\sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{y_i} \tag{9}$$

Percentage of prediction within x% (PRED(x))

$$PRED(x) = \frac{1}{n}\sum_{i=1}^{n} \{1 \; if \; \frac{|y_i - \hat{y}_i|}{y_i} \leq x; \; 0 \; otherwise \tag{10}$$

Mean absolute percentage error (MAPE)

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{y_i} \times 100 \quad (11)$$

Sum of square error (SSE)

$$SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)_i^2 \quad (12)$$

Mean squared error (MSE)

$$MSE = \frac{1}{n}\sum_{i}^{n}(y_i - \hat{y}_i)_i^2 \quad (13)$$

where, n is the number of observations, y_i is the known real value, $\hat{y}_i$ is the estimated value.

3.3 **Methodology used**

This section describes the procedure for investigating the accuracy of the proposed method. As described in Section 2, the CFS algorithm selects the most appropriate factors for the correction factors to be used as input to the MLR models. The methodology for the historical dataset is as follows:

Step 1: Using the CFS algorithm to determine the correlation and benefit of the number of technical complexity and environmental complexity factors for size estimation.

Step 2: Building the two MLR models based on the best correction factors preprocessed by the algorithm CFS. The regression model on CFS-based selected technical factors is called TCF_{CFS}. The regression model on CFS-based selected environmental factors is called ECF_{CFS}.

Step 3: The effort estimation result of the proposed CFSiCF method is calculated as the aggregate of four metrics - UAW (Eq. (2), UUCW (Eq. (3), TCF_{CFS}, and ECF_{CFS}

$$UCP_{CFSiCF} = (UAW + UUCW) \times TCF_{CFS} \times ECF_{CFS} \quad (14)$$

Step 4: Evaluate the methods using six performance measures (MAR, MMRE, PRED (0.25), MAPE, SEE, and MSE).

4 Results

This section evaluates and discusses the experimental results according to the methodology detailed in Section 3.

4.1 **RQ1**

The accuracies of experimental validation for the proposed CFSiCF method and other tested methods are listed at Fig. 2-7. As the results, we can confirm that the proposed CFSiCF method gives the best MAR, MMRE, MAPE, SSE, MSE, PRED (x) values in four selected different subsets of factors.

Specifically, the correlation and benefit of the number of technical complexity and environmental complexity factors in descending order of effectiveness (see Table 6): Case 1 - eight selected technical factors (T1, T2, T3, T4, T5, T7, T9, T10) and four selected environmental factors (ENV5, ENV6, ENV7, ENV8); Case 2 - eight selected

technical factors (T1, T2, T3, T4, T5, T7, T9, T10) and five selected environmental factors (ENV4, ENV5, ENV6, ENV7, ENV8); Case 3 - eight selected technical factors (T1, T2, T3, T4, T5, T7, T9, T10) and seven selected environmental factors (ENV1, ENV3, ENV4, ENV5, ENV6, ENV7, ENV8); and Case 4 - eight selected technical factors (T1, T2, T3, T4, T5, T7, T9, T10) and eight selected environmental factors (ENV1, ENV2, ENV3, ENV4, ENV5, ENV6, ENV7, ENV8).

Based on the above results, RQ1 can be answered using the CFS method to lead different subsets of factors that build diverse CFSiCF methods.

Table 6. Selected factors on TCF and ECF

Case	Selected technical factors	Selected environmental factors
1	T1, T2, T3, T4, T5, T7, T9, T10	ENV5, ENV6, ENV7, ENV8
2	T1, T2, T3, T4, T5, T7, T9, T10	ENV4, ENV5, ENV6, ENV7, ENV8
3	T1, T2, T3, T4, T5, T7, T9, T10	ENV1, ENV3, ENV4, ENV5, ENV6, ENV7, ENV8
4	T1, T2, T3, T4, T5, T7, T9, T10	ENV1, ENV2, ENV3, ENV4, ENV5, ENV6, ENV7, ENV8

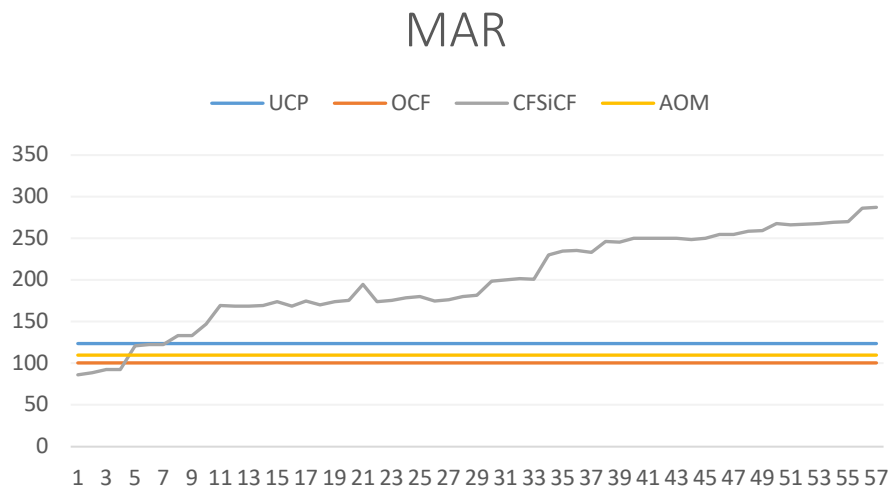

Fig. 2. The MAR results of the UCP, AOM and CFSiCF methods

4.2 **RQ2**

We measured the accuracy improvements achieved by the proposed CFSiCF method over the baseline method - UCP and other tested methods - AOM[14] and OCF[29]. As the results in Table 7, the proposed method outperforms all other methods with superior accuracy in MAR, MMRE, MAPE, SSE, MSE, PRED (x).

Specifically, the proposed CFSiCF method outperforms the UCP, OCF and AOM methods by 33.45 - 43.50%, 8.42 - 16.58% and 18.48 - 27.41% respectively for MAR,

by 33.57 - 43.85%, 8.57 - 16.92% and 16.78 - 25.77% respectively for MMRE, by 37.47 - 44.48%, 12.43 - 22.24% and 24.87 - 33.28% respectively for PRED (0.25), by 33.43 - 43.72%, 8.43 - 16.79% and 16.66 - 25.66% respectively for MAPE, by 1.69-2.24 times, 1.19-1.57 times and 1.39-1.85 times respectively for SSE and MSE.

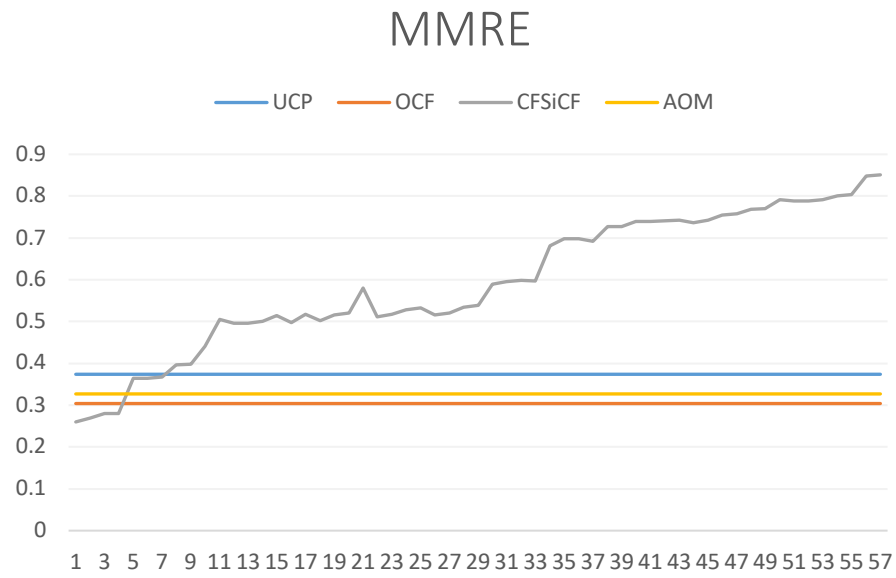

Fig. 3. The MMRE results of the UCP, AOM and CFSiCF methods

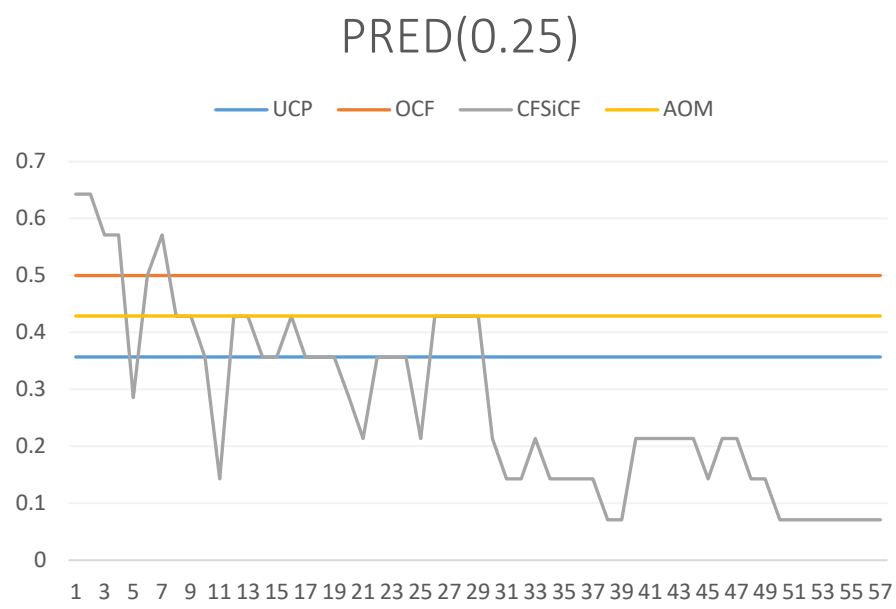

Fig. 4. The PRED(0.25) results of the UCP, AOM and CFSiCF methods

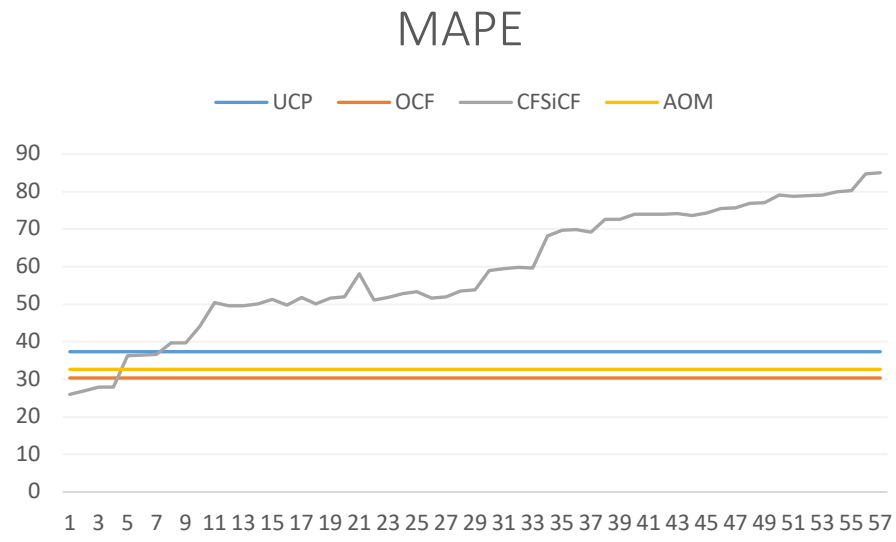

Fig. 5. The MAPE results of the UCP, AOM and CFSiCF methods

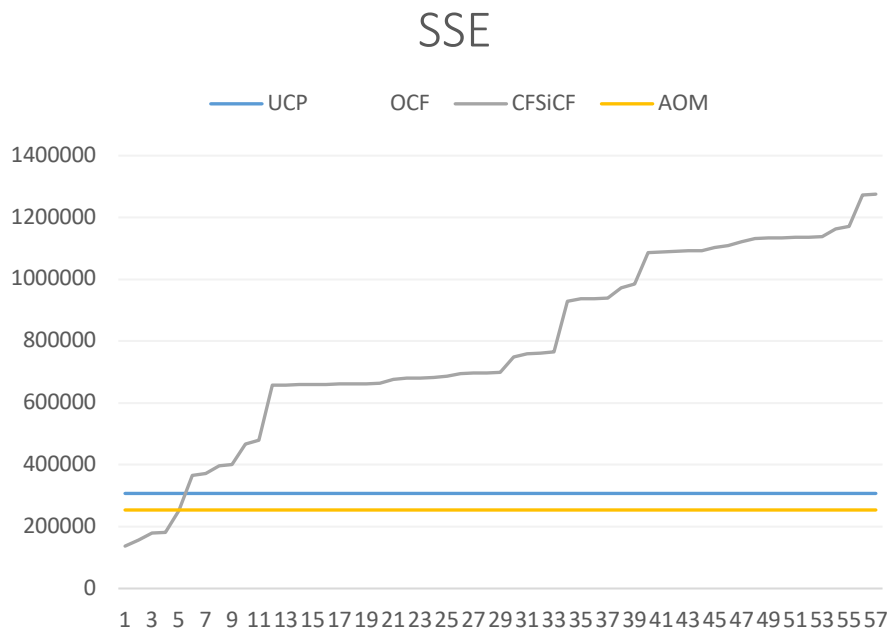

Fig. 6. The SSE results of the UCP, AOM and CFSiCF methods

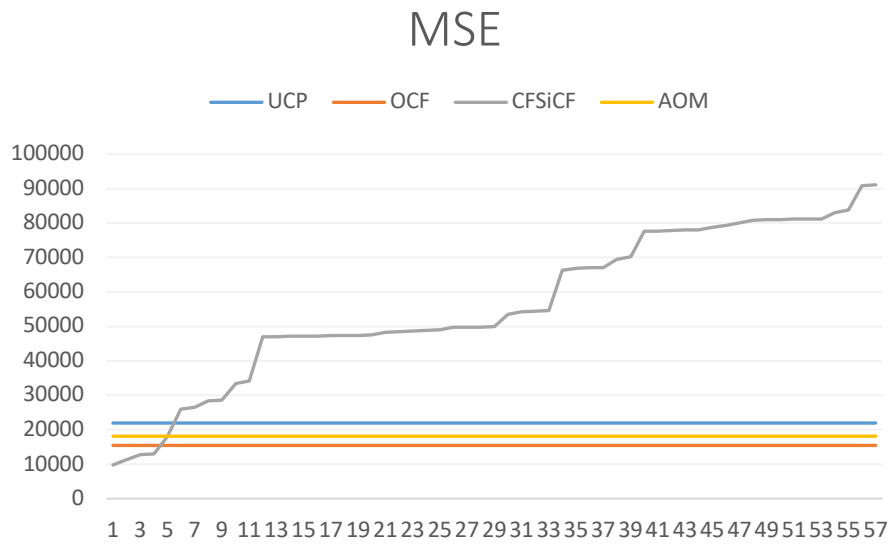

Fig. 7. The MSE results of the UCP, AOM and CFSiCF methods

Table 7. Comparison of the proposed method to other method variants

Method	MAR	MMRE	PRED(0.25)	MAPE	SSE	MSE
UCP	123.753	0.374	0.35	37.371	307472.9	21962.3
AOM	109.872	0.327	0.42	32.676	253954.5	18139.6
OCF	100.535	0.304	0.5	30.37	216512.7	15465.1
CFSiCF (Case 1)	86.234	0.26	0.64	26.002	130774.5	9791.04
CFSiCF (Case 2)	88.474	0.269	0.64	26.87	157356.1	11239.7
CFSiCF (Case 3)	92.454	0.28	0.57	27.954	178883.7	12777.4
CFSiCF (Case 4)	92.73	0.28	0.57	28.008	181691.4	12977.9

5 Conclusions

In this experimental study, we investigated the influence of the feature selection method on the correction factors to improve the estimation accuracy of the UCP method. For this purpose, the MLR method was chosen for correction factors preprocessed by the CFS algorithm. The proposed CFSiCF method was compared with the other tested methods on six accuracy measures using the 5-fold cross-validation method.

In the evaluation of RQ1, it can be concluded that the effect of the CFS algorithm on the correction factors was demonstrated. The most accurate estimation results were achieved with eight technical factors (T1, T2, T3, T4, T5, T7, T9, T10) and four environmental factors (ENV5, ENV6, ENV7, ENV8). It also follows from the analysis that in answer to RQ2, it can be said that the proposed CFSiCF method leads to a significant increase in estimation accuracy compared to the UCP, AOM, and OCF methods. It is also worth highlighting the simplification of the evaluation of the correction factor values, which increases the usefulness of the proposed CFSiCF method in practice.

Future research will further investigate other feature selection methods that can increase diversity and produce more estimation accuracy of the CFSiCF method. In addition, further research direction using clustering approaches will be considered a solution in our future work.

Acknowledgment. This study was supported by the Faculty of Applied Informatics, Tomas Bata University in Zlin, under Project IGA/CebiaTech/2021/001.

References

1. G. Kotonya and I. Sommerville: Requirements engineering: Processes and techniques. Wiley, 1st edition, 1998.
2. A. Trendowicz and R. Jeffery: Software project effort estimation. Foundations and Best Practice Guidelines of Success, Springer, 2014.
3. B. Boehm, C. Abts, and S. Chulani: Software development cost estimation approaches - A survey. Annals of Software Engineering, vol. 10, pp. 177-205, doi: 10.1023/A:1018991717352, 2000.

4. Karner: Resource Estimation for Objector Projects. Objective Systems SFAB, 1993.
5. M. Azzeh, A.B. Nassif, and I.B. Attili: Predicting software effort from use case points: A systematic review. Science of Computer Programming, 2021.
6. Y. Singh, P.K. Bhatia, A. Kaur, and O.P. Sangwan: A review of studies on effort estimation techniques of software development. 2nd National Conference Mathematical Techniques: Emerging Paradigms for Electronics and IT Industries, 2008.
7. Y. Mahmood, N. Kama, and A. Azmi: A systematic review of studies on use case points and experts-based estimation of software development effort. Journal of Software: Evolution and Process, 2019.
8. A.P. Subriadi, Sholiq, and P.A. Ningrum: Critical review of the effort rate value in use case point method for estimating software development effort. Journal of Theoretical and Applied Information Technology, vol. 59, 2005.
9. A.B. Nassif, L.F. Capretz, and D. Ho: Enhancing Use Case Points Estimation Method Using Soft Computing Techniques. Journal of Global Research in Computer Science, 2010.
10. A.B. Nassif, L.F. Capretz, and D. Ho: Estimating Software Effort Based on Use Case Point Model Using Sugeno Fuzzy Inference System. 23rd IEEE International Conference on Tools with Artificial Intelligence, 2011.
11. A.B. Nassif, D. Ho, and L.F. Capretz: Regression Model for Software Effort Estimation Based on the Use Case Point Method. International Conference on Computer and Software Modeling, IPCSIT, vol.14, 2011.
12. M. Jorgensen: Regression models of software development effort estimation accuracy and bias. Empirical Software Engineering, vol. 9, pp. 297-314, 2004.
13. M. Ochodek, J. Nawrocki, and K. Kwarciak: Simplifying effort estimation based on use case points. Information Software Technology, vol. 53, pp. 200-213, 2011.
14. R. Silhavy, P. Silhavy, and Z. Prokopova: Algorithmic Optimisation Method for Improving Use Case Points Estimation. PLoS ONE, 2015.
15. A.B. Nassif, and L.F. Caprets: Software effort estimation in the early stages of the software life cycle using a cascade correlation neural network model. ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 2012.
16. M.S. Iraji and H. Motameni: Object oriented software effort estimate with adaptive neuro fuzzy use case size point (ANFUSP). International Journal of Intelligent Systems and Applications, vol.4, pp. 14-24, 2012.
17. V.K. Bardsiri, D.N.A. Jawawi, S.Z.M. Hashim, and E. Khatibi: A flexible method to estimate the software development effort based on the classification of projects and localization of comparisons. Empirical Software Engineering, vol. 19, pp. 857–884, 2014.
18. N.H. Chiu and S.J. Huang: The adjusted analogy-based software effort estimation based on similarity distances. Journal of Systems and Software, vol. 80, pp. 628-640, doi: 10.1016/j.jss.2006.06.006, 2007.
19. H.L.T.K. Nhung, H.T. Hoc, and V.V. Hai: A review of Use Case-based development effort estimation methods in the system development context. In: CoMeSySo Springer Series: Advances in Intelligent Systems and Computing Springer, 2019.
20. M.H. Luis and B.O. Sussy: Factors affecting the accuracy of Use Case Points. Trends and Applications in Software engineering, Advances in Intelligent Systems and Computing 537, 2017.
21. A.B. Nassif, D. Ho, and L.F. Caprets: Towards an early software estimation using Log-linear regression and a multilayer perceptron model. Journal of systems and software, vol 86, pp. 144-160, 2013.
22. M.A. Hall: Correlation-based feature selection for machine learning. Citeseer 113, pp.1–8, 1999.

23. N. Sánchez-Maroño, A. Alonso-Betanzos, and M. Tombilla-Snaromán: Filter methods for feature selection. A comparative study. In Intelligent Data Engineering and Automated Learning (IDEAL), pp. 178–187, doi: 10.1007/978-3-540-77226-2, 2007.
24. E. Chandra Blessie and E. Karthikeyan: Sigmis: A feature selection algorithm using correlation based method. Journal of Algorithms and Computational Technology, vol. 6, pp. 385–394, 2012.
25. A. Idri and S. Cherradi: Improving effort estimation of Fuzzy Analogy using feature subset selection. IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1-8, doi: 10.1109/SSCI.2016.7849928, 2016.
26. M.A. Hall and G. Holmes: Benchmarking attribute selection techniques for discrete class data mining. IEEE Transactions on Knowledge and Data Engineering, vol. 15, pp. 1437–1447, 2003.
27. R. Silhavy, P. Silhavy, and Z. Prokopova: Analysis and selection of a regression model for the Use Case Points method using a stepwise approach. Journal of Systems and Software, vol. 125, pp. 1-14, doi. 10.1016/j.jss.2016.11.029, 2017.
28. R. Silhavy, P. Silhavy, and Z. Prokopova: Using actors and use cases for software size estimation. Electronics 2021, vol.10, doi: 10.3390/electronics10050592, 2021.
29. H.L.T.K. Nhung, H.T. Hoc, and V.V. Hai: An Evaluation of Technical and Environmental Complexity Factors for Improving Use Case Points Estimation. CoMeSySo 2020. Advances in Intelligent Systems and Computing Springer, 2020.