Adoption of Machine Learning Techniques in Software Effort Estimation: An Overview

IOP Conf. Series: Materials Science and Engineering **551** (2019) 012074 doi:10.1088/1757-899X/551/1/012074

Adoption of Machine Learning Techniques in Software Effort Estimation: An Overview

Siti Hajar Arbain[1], Nor Azizah Ali[1] and Noorfa Haszlinna Mustaffa[1]
[1]School of Computing, Faculty of Engineering, Universiti Teknologi Malaysia (UTM), 81310 Johor Bahru, Johor, Malaysia.

Email: shajar75@live.utm.my, nzah@utm.my

Abstract. Nowadays the significant trend of the effort estimation is in demand. It needs more data to be collected and the stakeholders require an effective and efficient software for processing, which makes the hardware and software cost development becomes steeply increasing. This scenario is true especially in the area of large industry, as the size of a software project is becoming more complex and bigger, the complexity of estimation is continuously increased. Effort estimation is part of the software engineering economic study on how to manage limited resources in a way a project could meet its target goal in a specified schedule, budget and scope. It is necessary to develop or adopt a useful software development process in executing a software development project by acting as a key constraint to the project. The accuracy of estimation is the main critical evaluation for every study. Recently, the machine learning techniques are becoming widely used in many effort estimation problems but there are limitations in some of the models and the variation research is still not enough. This paper presents an overview of the effort estimation using machine learning techniques and will be useful for researchers to provide future direction in the field of machine learning adoption in software effort estimation.

1. Introduction

Nowadays the software development system is becoming complicated. The usage of software arises in most companies. Depending on the organisation's size and accompanied tasks, each activity under a software project development should be updated regularly. They must deal in giving high-quality software with a low-cost budget. Therefore, more intelligent approaches are needed to solve the challenging problems in this domain. A software development project is one of the processes in the planning of software project management. It needs to be monitored by the manager to ensure a high-quality software can be produced at a low cost within a specified time and budget [6]. Software effort estimation (SEE) assumes an essential part of the improvement of software development. Lately, the product has turned into the costliest part of the software development efforts. The critical aspect of cost in software advancement is the human-effort, and most cost estimation techniques concentrate on this aspect and give estimates in regard to the individual [3][9]. During the early phase of software development project, there is a lot of work to be done. Software effort estimation is one of the steps in software development project which targets on the production of quality software, which can be delivered on time and within budget, and satisfying its requirements. It is also known as a feature of software engineering monetary on how to oversee restricted assets to meet the objectives and goals of the schedule, budget, and scope.

The growing concern by most developers is the complexity of estimation made in an early phase of the development process. Sometimes, the development of software product resulted differently due to uncertainties. It increases with the size of software project estimation mistakes that could cost a lot in terms of resource allocated to the project. Because software development problems have many

dimensions, we need to investigate the use of several techniques to optimise these challenging issues, not only focusing on the software effort engineering approach, but also to include the incorporation of other methods that can contribute to the enhancement in effort estimation accuracy. The following sections present the overview of the techniques implemented in the development of the software effort estimation model. The methodology used for literature review is presented in Section 2. The background and the related theory of software effort estimation techniques are presented in Section 3, followed by the summary review techniques analysis and conclusions in Section 4.

2. Methodology

This paper presents the incorporation of machine learning and soft computing techniques with software effort estimation developed up to early 2007 publications. The publication search procedures are illustrated in Figure 1.

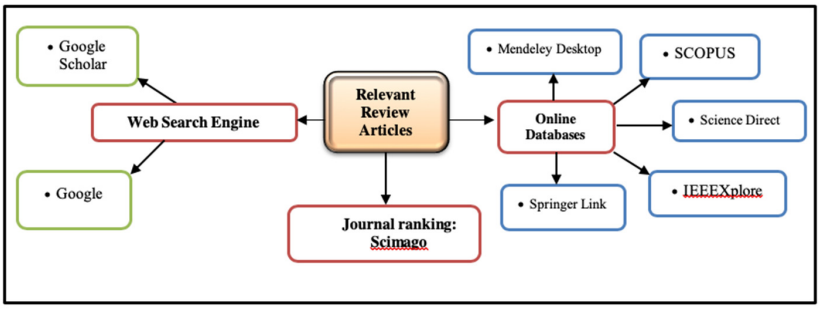

Figure 1: Publication Search

The searches include any keywords of research articles included in title, abstract and keywords. An example of the search queries used is *("software effort estimation") AND ("soft computing" OR "COCOMO II").*

Upon completion of searching the publications, manual search was carried out to identify the redundant results. The redundancies in similar article publications were eliminated. Besides, we used the inclusion and exclusion characteristics to access the potential direction of the study as follows:

i. **Inclusion**: Publications with a clear focus on software effort estimation and machine learning.
ii. **Exclusion**: Publications of unrelated area of studies, or not peer-reviewed, for example, lecture notes and tutorials.

3. Background and Related Works

Software Effort Estimation (SEE) emphasises on how to estimate the effort, time and cost with the project activities plan. Generally, in SEE, two types of traditional approaches being implemented to calculate the effort estimation which are the non-algorithmic method, focusing on human expertise and the algorithmic approach which consists of mathematical modelling [1] [12].

The algorithmic process is produced by utilising some numerical demonstrations to carry out the software estimation. These scientific, mathematical models are based on the historical data and use the inputs such as size, number of attributes functions and other cost drivers. The Constructive Cost Model (COCOMO) model, Function Point Analysis (FPA) model, and Putnam Model are some of the models that use the algorithmic approach. A non-algorithmic method is used by an expert for software project estimation. In its process, the estimators must have the knowledge on an earlier completed project that is similar to the current project [4][5]. These estimation techniques are used to characterise the spending plan and the items are balanced with the goal in the spending figure. Table 1 shows the advantages and disadvantages of both the algorithmic and the non-algorithmic techniques in software effort estimation.

Table 1: Advantages and disadvantages of software effort estimation

Methods	Advantages	Disadvantages
Non-Algorithmic / Heuristic	Basic info parameter.	No formal/substantial motivation to judge that the estimation is adequate.
	Expert's understanding and learning give a high esteem.	Conflicting. For a situation, the blunder edge could be huge. Then again, it could likewise create a superior estimation than utilising the algorithmic technique.
	For the most part, estimation is made of the group accord.	
	Useful without measured, experimental information.	Estimate is just as great master's supposition.
	Can figure in contrasts between past venture encounters and necessities of the proposed undertaking.	Hard to archive the components utilised by the masters.
	Can calculate in effects brought on by new innovations, applications and dialects.	
Algorithmic	Objectively aligned to experience.	Confused information parameter.
	Reliable, since the conditions are consistent.	
	Utilising standard unit of estimation.	Requirements of having broad information as to each info parameter.
	Could be utilised as a part of each venture as long as the info parameter could be satisfied.	Unable to manage remarkable conditions.
	Generate repeatable estimations.	
	Easy to adjust information.	Some experience and components cannot be measured.

Studies show that both expert and formal methods have drawbacks and still, the best method is yet to be discovered. The efficiency of the processes depends on a specific context data, the techniques used and the problems domain to be solved. Software analysts and experts are giving numerous effort estimation procedures in quite a few years. The early software estimation models depend on relapse examination or numerical determinations. The present models depend on simulation, neural network, Genetic Algorithm (GA), fuzzy logic and so forth [2]. There are many implementation and improvement of effort estimation model using soft computing in order to overcome some limitations in the estimation accuracy which is applicable for every standard software development. Soft computing procedure is investigated to defeat the vulnerability and imprecision in estimation. In classical formulation, machine learning is one of the techniques under soft computing, included evolutionary computation and fuzzy logic. In previous research, Particle Swarm Optimization (PSO) method is used to tune function parameters *f(a, b)* consists in COCOMO II and take advantage of Fuzzy Logic to build a set of linear models over the domain of possible software Lines of Codes (LOC). The PSO algorithm starts by generating particle locations randomly throughout the designed space of the parameters, and the performance of the developed model was evaluated using NASA software projects dataset [10][13]. Moreover, some researchers suggest using the regression model to minimise the sum of relative errors and impose the non-negative coefficients, which is a favourable technique for calibrating the COCOMO model parameters [9].

The Artificial Neural Network (ANN), Support Vector Machine (SVM), Linear Regression (LR), K-Nearest Neighbour (KNN) are some of the data mining techniques combined with the algorithmic COCOMO II model in the quest for the precise estimation software development effort and time estimation [5]. Simulated annealing (SA) had been used to overcome the limitation of GA, which produced premature convergence population and was proposed to advance the COCOMO II PA model coefficients for achieving the accurate software effort estimation and to diminish the dubiousness of COCOMO II post architecture model. The study, however, suggested combining different techniques for calculating the best estimate for software engineering field. Recently, COCOMO II model is widely studied by most software researchers and practitioners in the enhancement of its ability to estimate software cost project precisely. COCOMO II is one of the numerous techniques in software effort estimation. It is widely accepted as an industry standard because of its applicability at diverging the stage of software engineering studies. The accuracy of COCOMO II is highly affected by of its input

parameters which are size of projects, coefficients and attributes cost drivers. The small changes in these parameters bring huge differences in their effort estimation [11].

Incorporating COCOMO II with novel soft computing and machine learning model becomes popular in research nowadays because it offers an added advantage with the ability to improve performance in tuning the parameters and to learn from experience data, which mainly focus on predictive accuracy [7]. Most of the research use many models with the combination with COCOMO II, to improve the signs of the particular model. However, the accuracy of the models is still questionable until now. The COCOMO model, either COCOMO I or COCOMO II is one of the established software effort estimation related to software engines, but it is not utilised in the practice even though it is widely known to the software industry. Table 2 shows the summary of the machine learning techniques that have been applied for ten years from 2007 until early 2018.

Table 2: Some of ML techniques for SEE for 10 years

YEAR	TECHNIQUES	DATASET	EVALUATION METHOD
2007, 2015	NeuroFuzzy	NASA &COCOMO	MMRE
2008	PSO	NASA	MMRE
2008	Regression Techniques	COCOMO	MMRE and PRED
2010, 2013,2015	Genetic Algorithm	-Turkish & Industry data set -COCOMO & NASA	Not specified
2011, 2014	ANN	COCOMO & NASA, PROMISE repository	Not specified
2011, 2018	Fuzzy Logic	NASA	MMRE, PRED
2013	PSO-ANN-COCOMO II	COCOMO I & NASA93	MRE and PRED (25%)
2014	Radial Basis function NN	COCOMO & NASA	MSE
2014	Bayesian with PSO	NASA 93	MMRE
2014	PCA with ANN	COCOMO 81	
2014	Simulated Annealing	NASA	
2014	Hybrid COCOMO and FP	NASA	MRE
2015	Bee Colony Optimization	Interactive Voice Response Software project dataset	MMRE
2016, 2017, 2018	Cuckoo, Hybrid Cuckoo Optimization	NASA60, NASA63, NASA93, MAXWELL, KEMERER, and MIYAZAKI	MRE, MMRE
2016	COCOMO TLBO	NASA	MMRE, PRED (30)
2016	Differential Evolution Method	NASA	MMRE, MRE
2016	Hybrid PSO-ANFIS	NASA	MSE, MMRE, PRED
2016	Bat Algorithm	NASA 93	MMRE
2016	COCOMO-GA	NASA	MMRE
2017	ANN-Dragonfly Algorithm	NASA	MSE, MMRE, MAE

The summary shows that the trends of estimation software effort are more on the exploration of enhancement techniques rather than proposing a new ML technique. The latest publication on 2017 enhances the previous ANN algorithm techniques using dragonfly to provide the optimal training of neural network, which offered more enhanced and accurate software effort estimation. In software effort estimation studies, there are several questions regularly asked; which data should be used? Which datasets are more reliable to be used for studies? How to choose a specific project to estimate a future project? These questions are important for the researchers in making their decision before analysing the historical data. Most researchers usually did not show the explicit ways on how to make a choice from the specific studies. The software process improvement is facing the difficulties with such effort estimation because of lack of data for analysis and lack of information regarding the data types.

The data quality concept is large. It does not only relate to consistency, but also the quality of completeness datasets used. There was a study warned about using the unbalance datasets in the research that it might cause the impact factor to be concealed. In software effort estimation, the whole data project effort is of utmost important to the developers to make reasonable plan or activities in project management. The occurrences of missing data in project data can bring significant impact on future estimation effort. There is an analysis showing that the pre-processing data analysis such as treatment of missing data can also influence the performance of the prediction model [8]. There are numbers of alternative ways of dealing with missing data. In some cases, deletion or elimination the missing variable is the default method for most procedures. However, there is only a few studies investigating the

imputation of missing data in software effort estimation area. There should be an empirical investigation of the robustness and the accuracy in handling the missing data.

4. Conclusion and Future Research

There are still many problems regarding on how to compare and evaluate estimation methods. Most researchers either do not cooperate closely with the software industry and current issues or believe that it is better to focus on a replacement rather than improvement the of current approaches employed by the industry. Therefore, it is recommended that the future research works put a focus on the software enhancement as opposed to proposing new replacement techniques. It is good to note that some articles were based on datasets that are too old to be representative for the current or future projects. Future research should focus on understanding the relationship between project characteristics (dataset quality) and estimation evaluation. Most articles evaluated the estimation by employing historical datasets, but only a few evaluated based on the completed real-life estimation situation. Therefore, more studies on the methods used in real-life situation should be conducted.

Acknowledgements
This work was financially supported by RUG-Tier 2 Grant (Vot. No:15J16) from Universiti Teknologi Malaysia.

References

[1] Borade, J. G., & Khalkar, V. R. (2013). Software Project Effort and Cost Estimation Techniques. *International Journal of Advanced Research in Computer Science and Software Engineering*, *3*(8), pg. 730–739.

[2] Choudhary, K. (2010). GA Based Optimization of Software Development Effort Estimation. *International Journal Of Computer Science And Technology,* Vol.(1), pg. 38–40.

[3] Choudhary, K. (2011). Parametric Estimation of Software Systems. *International Journal of Soft Computing and Engineering*, *1*(2), pg. 17–20.

[4] Clause, J., Li, W., & Orso, A. (2007). Dytan: a generic dynamic taint analysis framework. In *Proceedings of International Symposium on Software Testing and Analysis* (pp. 196–206). ACM.

[5] Dan, Z. (2013). Improving the accuracy in software effort estimation: Using artificial neural network model based on particle swarm optimization. In *Proceedings of 2013 IEEE International Conference on Service Operations and Logistics, and Informatics, SOLI 2013*(pp. 180–185).

[6] Garbajosa, J. (2008). The emerging ISO International Standard for Certification of Software Engineering Professionals. In *IFIP International Federation for Information Processing* (Vol. 280, pp. 173–178).

[7] Maleki, I., Ghaffari, A., & Masdari, M. (2014). A New Approach for Software Cost Estimation with Hybrid Genetic Algorithm and Ant Colony Optimization. *International Journal of Innovation and Applied Studies*, *5*(1), 72–81

[8] Missier, P., Lalk, G., Verykios, V., Grillo, F., Lorusso, T., & Angeletti, P. (2003). Improving data quality in practice: A case study in the italian public administration. *Distributed and Parallel Databases*, *13*(2), 135–160.

[9] Rajper*, S., & and Zubair A. Shaikh. (2016). Software Development Cost Estimation Approaches - A Survey. *Indian Journal of Science and Technology*, *9*((31)), pg.1–5.

[10] Sandhu, G. S. (2014). A Bayesian Network Model of the Particle Swarm Optimization for Software Effort Estimation tool. *International Journal of Computer Applications*, *96*(4), 52–58.

[11] Vu Nguyen, Bert Steece, B. B., Nguyen, V., Steece, B., Boehm, B., & Vu Nguyen, Bert Steece, B. B. (2008). A Constrained Regression Technique for COCOMO Calibration. *Acm 978-1-59593-971-5/08/10*, 1–10.

[12] Zaid, A., Selamat, M. H., Azim, A., Ghani, A., Atan, R., Koh, T. W., … Albrecht, A. (2008). Issues in Software Cost Estimation. *International Journal of Computer Science and Network Security*, *8*(11), 350–356.

[13] Zhang. "Improving the accuracy in software effort estimation: Using artificial neural network model based on particle swarm optimization." Service Operations and Logistics, and Informatics (SOLI), 2013 IEEE International Conference.