Engineering Applications of Artificial Intelligence 26 (2013) 2624–2640

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

LMES: A localized multi-estimator model to estimate software development effort

Vahid Khatibi Bardsiri [a,*], Dayang Norhayati Abang Jawawi [b], Amid Khatibi Bardsiri [a], Elham Khatibi [a]

[a] Department of Computer Engineering, Bardsir Branch, Islamic Azad University, Kerman, Iran
[b] Department of Software Engineering, Universiti Teknologi Malaysia (UTM), Skudai 81310, Johor Bahru, Malaysia

ARTICLE INFO

Article history:
Received 5 May 2013
Received in revised form
15 August 2013
Accepted 16 August 2013
Available online 10 September 2013

Keywords:
Localization
Classification
Effort estimation
Estimator
Software project

ABSTRACT

Accurate estimation of software development effort is strongly associated with the success or failure of software projects. The clear lack of convincing accuracy and flexibility in this area has attracted the attention of researchers over the past few years. Despite improvements achieved in effort estimating, there is no strong agreement as to which individual model is the best. Recent studies have found that an accurate estimation of development effort in software projects is unreachable in global space, meaning that proposing a high performance estimation model for use in different types of software projects is likely impossible. In this paper, a localized multi-estimator model, called LMES, is proposed in which software projects are classified based on underlying attributes. Different clusters of projects are then locally investigated so that the most accurate estimators are selected for each cluster. Unlike prior models, LMES does not rely on only one individual estimator in a cluster of projects. Rather, an exhaustive investigation is conducted to find the best combination of estimators to assign to each cluster. The investigation domain includes 10 estimators combined using four combination methods, which results in 4017 different combinations. ISBSG, Maxwell and COCOMO datasets are utilized for evaluation purposes, which include a total of 573 real software projects. The promising results show that the estimate accuracy is improved through localization of estimation process and allocation of appropriate estimators. Besides increased accuracy, the significant contribution of LMES is its adaptability and flexibility to deal with the complexity and uncertainty that exist in the field of software development effort estimation.

1. Introduction

Planning, scheduling, managing and all other important aspects of a software project depend considerably on an accurate estimation of development effort (Jones, 2007). In recent years, accurate development effort estimation has become a challenging issue in the management of software projects (Azzeh, 2012; Li et al., 2009a). Basically, the special characteristics of software projects make the process of estimation more difficult than it may seem. Unlike other types of projects (building construction, material production and so on), developers are confronted with an intangible product whose specifications may not be completely obvious at the early stages of the project. Due to inconsistency and vagueness of software project attributes, an accurate estimation of development effort seems to be unreachable in a dataset comprised of heterogeneous projects. This means that it is impossible to propose an estimation model that is suitable for use in different types of software projects.

Recently, it has been proved that global effort estimation models (those constructed on whole dataset) may be inefficient for use in a subset of dataset (local regions) (Bettenburg et al., 2012; Menzies et al., 2011; Posnett et al., 2011). In order to remedy the problem of the global models, prior studies have utilized project clustering to construct localized estimation models (Aroba et al., 2008; Cuadrado et al., 2007; Khatibi Bardsiri et al., 2012a; Kocaguneli et al., 2012; Lin et al., 2012). The proposed localized models have been constructed using a blind clustering method such as c-means (Aroba et al., 2008), k-means (Lin, 2010), M5(Rodrguez et al., 2006), and GAC (Kocaguneli et al., 2012) with the exception of a model proposed by Khatibi Bardsiri et al. (2013) in which the type of software projects has been considered in the clustering process. Although the localized effort estimation models have shown a considerable improvement in the estimate accuracy, they have only utilized one type of effort estimation model in the local regions. For example, analogy based estimation (ABE) (Kocaguneli et al., 2012), neural network (Benala et al., 2012; Khatibi Bardsiri et al., 2012a), regression (Aroba et al., 2008; Bettenburg et al., 2012), a combination of ABE and particle

* Corresponding author at: Department of Computer Engineering, Bardsir Branch, Islamic Azad University, Kerman, Iran.
Tel.: +60127373053.
E-mail address: kvahid2@live.utm.my (V.K. Bardsiri).

0952-1976/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.engappai.2013.08.005

swarm optimization (PSO) algorithm (Khatibi Bardsiri et al., 2013) as well as a combination of COCOMO and PSO (Lin et al., 2012) have been employed in the local regions, separately.

Despite the efforts made to find the most appropriate model in the field of software development effort estimation (Heiat, 2002; Keung et al., 2012; Menzies et al., 2006), there is no strong agreement as to which individual estimation model is the best. Therefore, relying on only one type of estimation model threatens the performance and generalization of the current localized effort estimation models. Different types of software projects may need different types of estimators. This is what has been lost in prior studies. Indeed, in-depth performance analysis of an individual estimator or a combination of estimators in the local regions can facilitate selection of the best estimators to assign to particular types of software projects. This paper aims to propose a high performance model called localized multi-estimator (LMES) in which the process of effort estimation is localized and the most efficient combination of estimators is assigned to each type of project. In fact, LMES is an improved version of the model developed in (Khatibi Bardsiri et al., 2013).

The rest of this paper is organized as follows: the literature review is presented in Section 2. In Section 3, the proposed estimator is explained. The experimental design is described in Section 4. The numerical results are presented in Section 5. The results are discussed in Section 6. Section 7 includes description of threats to validity. Finally, conclusion and future work are explained in Section 8.

2. Related work

Software development effort estimation has been studied for many years. The process of improving estimation models began with very simple assumptions and has evolved to now include complicated equations and techniques. In order to discuss the existing estimation models, it is necessary to divide all the models into two main groups: algorithmic and non-algorithmic. The algorithmic models are constructed based on fixed and predefined mathematical and statistical equations while the non-algorithmic models rely on learning, inferring and analyzing previous software projects.

Different types of regression, COCOMO (Boehm, 1981) and COCOMO II (Boehm, 2000) are the most important algorithmic models. On the other hand, analogy based estimation (ABE) (Shepperd and Schofield, 1997), classification and regression tree (CART) (Breiman, et al., 1984), expert judgment (Dalkey and Helmer, 1963), and various soft computing techniques such as artificial neural network (ANN) (Ara et al., 2012; El-Sebakhy, 2011; Heiat, 2002), fuzzy logic (Ahmed et al., 2005), neuro-fuzzy (S.-J. Huang and Chiu, 2009), and optimization algorithms (Azzeh, 2011; Ferrucci et al., 2010) are the most common non-algorithmic models.

In general, researchers are interested in combining different estimation models to achieve a high performance model. Due to simplicity and flexibility, ABE has been frequently used in the hybrid effort estimation models proposed over the past few years. In the ABE method, the development effort of a new project is estimated through a comparison between the new project and those completed in the past. Similarity and solution functions are the main components of ABE to measure the similarity level between two projects and to estimate the development effort using similar projects, respectively. The combinations of ABE and the genetic algorithm (Chiu and Huang, 2007; S. J. Huang and Chiu, 2006; Y. F. Li et al., 2009b; Milios et al., 2011), ABE and PSO (Khatibi Bardsiri et al., 2012b; Wu et al., 2010), ABE and ANN (Khatibi Bardsiri et al., 2012a; Y. F. Li et al., 2009a), ABE and grey (Azzeh et al., 2010; C. J. Hsu and Huang, 2011; Song and Shepperd, 2011), ABE and outlier elimination techniques (Seo and Bae, 2012), ABE and principle component analysis (PCA) (Jianfeng et al., 2009), ABE and regression (Mittas and Angelis, 2010) as well as ABE and rough set theory (Li and Ruhe, 2008) are some instances of hybrid models constructed using ABE. Fig. 1 shows the common ABE based effort estimation models.

The complexity and non-normality of software project attributes have been frequently elaborated in prior studies and the generalization of estimation models has been extensively mentioned as a critical issue. Dolado (2001) reported that mathematical-based effort estimation models are unable to be a universal model. It was the first alert for researchers to invent more flexible and reliable estimation models.

Finally, segmented software development effort estimation models were introduced to remedy the problem of inconsistency in software projects (Cuadrado-Gallego et al., 2006; Rodrguez et al., 2006). The authors have proposed a model in which the projects are divided into several clusters using a clustering algorithm (EM and M5). For each cluster, a regression based equation is

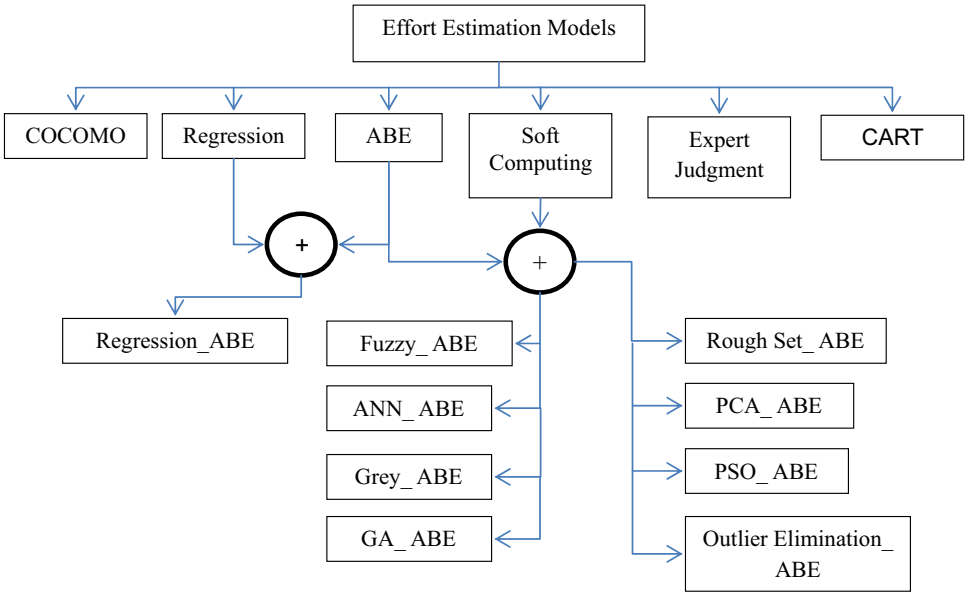

Fig. 1. Common ABE-based effort estimation models.