Analogy-based Effort Estimation: A Systematic Mapping of Literature

ELIANE MARIA DE BORTOLI FÁVERO[1]
ANDREY RICARDO PIMENTEL[2]
ROBERTO PEREIRA[2]
DALCIMAR CASANOVA[1]

Technological Federal University of Paraná - UTFPR
Academic Department of Informatics
Via do Conhecimento, Km 01, S/N
85503-390 - Pato Branco (PR) - Brazil[1]
Federal University of Paraná - UFPR
Department of Informatics Rua Evaristo F. F. da Costa, 418 - Jardim das Américas
80050-540 - Curitiba (PR) - Brazil[2]
[1](elianedb,dalcimar)@utfpr.edu.br
[2](andrey,rpereira)@inf.ufpr.br

Abstract. Many research initiatives have been developed in the field of Software Engineering, including the area of software estimation. Software effort estimation techniques based on analogy are applied from historical data of projects, obtained in the early stages of software development. In this context, this paper presents a systematic mapping of the literature, aim to elicit the state of the art on analogy-based software effort estimation techniques , indicating challenges and research opportunities. The mapping was done for the period from 2007 to 2017 and was conducted separately for each of the selected sources. The articles found were reviewed according to previously established research and selection criteria, according to the study objectives. Note that the model of estimation by analogy has received more attention and is presented as a promising and feasible technique in relation to the others. The techniques of Adaptive Neuro-fuzzy Inferences (ANFIS), Collaborative Filtering (FC), Radial Basis Functions (FBR) and Deep Learning present a gap to be explored. The results point to a demand for simple and practical estimation techniques, with emphasis on the estimation based on analogies and for the exploration of the artifacts generated in the initial phase of development, mainly in the textual format.

Keywords: systematic mapping; analogy; software effort estimation; analogy-based effort estimation.

(Received November 11th, 2018 / Accepted December 03th, 2018)

1 Introduction

Estimating software effort is a challenging and important activity in the software development process. This activity depends on the success of other crucial aspects of a project that directly impact the quality of the software product developed, predominantly the time under which the software was developed and the budget constraints. The success of any particular software project depends greatly on the accuracy of its effort estimates [39]. An accurate estimate assists in contract negotiations, scheduling and synchronization of project activities and efficient allocation of resources. [75], shows an increase in the rate of software design flaws –especially in the year 2012 –which resulted in budget and / or schedule overflows.

There are currently different types of development processes that have differing impacts on planning and estimating software projects. Ways to generate effort estimates to achieve better results have been one of the focuses of the Software Engineering industry for quite some time [31, 11, 41, 7, 10, 4]. These experiments have continuously explored computational techniques individually or in combination, seeking to achieve better levels of precision in effort estimation.

Several classifications for software effort estimation models have been proposed in the last decade and include small differences according to the authors' point of view. For instance, [69] classifies software effort estimation in algorithmic/parametric and non-algorithmic terms. The former includes those that use mathematical or algorithmic models (applied to project attributes) to calculate their estimate, e. g. COCOMO II [15] and Function Point Analysis [24]. The latter relies on machine learning techniques, and uses historical project data to generate learning models to predict future estimates [49], for example, the use of regression models [2, 81] e classification algorithms [46, 59].

Shepperd et al.[67], in contrast, classifies effort estimates in software projects into three categories: 1. Human-centric (e.g. expert judgment); 2. model-based (e.g. COCOMO); and 3. induced prediction techniques [53, 2]. Although they are widely used - especially in agile models, techniques relying on human determination problems because they become very subjective, often generating distorted estimates with excess of optimism, for example. Model-based techniques, by contrast, use replicable methods to produce estimates and can be more objective than those with [45]. Among the many induced predicted techniques, the main types used are linear regression, neural networks, and analogy [46, 59]. In order to use these techniques, it is interesting that the training data available are independent of algorithmic/parametric models.

Chiu et al.[21] highlight a fourth model: the analogy-based effort estimation (ABEE). This method has been widely used and aims to identify similarity between projects already carried out, thus generating the most approximate estimate for a new project. The origin of this method can be attributed to a study performed by [72], which identifies it as a viable approach to predicting estimates. Studies involving ABEE are commonly associated with Artificial Intelligence techniques (also associated with historical project data).

According to [37], the accuracy of the estimate is improved when the analogy is combined with another technique to generate estimates. Artificial Intelligence techniques are useful regardless of the mode used to approximate similar data. Fuzzy systems, genetic algorithms, case-based reasoning, and collaborative filtering are techniques that improve ABEE performance.

Approaches based on analogy have shown promise in the field of software effort estimation, and its use has increased among researchers in this area [39]. Authors, such as [37], classify analogy-based technique as a machine learning technique. This technique has been advocated as a potential method for efficient effort estimation, since it allows modeling the complexity between the effort and the variables included in the context of the software project (e.g. team data, project data), elements which have a relationship that is normally not linear. Wen et al.[78] carried out a systematic review of the literature in which they identified eight types of machine learning techniques. The Case-based reasoning (CBR) and artificial neural networks (ANN) were the most used techniques for estimating effort, representing 37% and 26%, respectively.

Idri et al. [37] in turn performed a systematic literature review on ABEE and found out that these techniques outperform other prediction techniques. Some advantages of this estimation technique as highlighted by the authors include: 1) they present a tendency to produce acceptable estimates, surpassing other estimation techniques (e. g. human-centered models and model-based estimates); 2) they generate models that relate the effort and attributes of the project context, despite the inherent complexity, resulting in reproducible models to produce estimates [46] (e. g. linear regression, neural networks and analogies [68, 50] and 3) unlike prediction techniques such as Artificial Neural Networks –such as black boxes –ABEE techniques are similar to human reasoning, which makes use of analogies in previous experiences in facilitating effort estimation.

However, ABEE techniques are still limited because they can not adequately handle categorical attributes measured on a nominal or ordinal scale, such as the complexity level of a requirement and/or the area of a project [39]. For this reason, the study considers the application of complementary techniques, such as machine learning.

ABEE is fit for both agile and traditional models as long as the estimation approach is based on previous team experiences to estimate software projects. A more latent disadvantage in agile models is that the estimation process is usually conducted in discussion among team members, as there are no formal conventions for the use of historical data. Another disadvantage of both development models is that there could be some inattention to exact detail in the process, these data may not have been properly recorded and could generate improper results.

Another challenge (especially in agile models) is the scarcity of data and project requirements available in the early stages of the development process. The basic requirement specification used in these models is the user history (or user requirements), and is generally written without formal convention [35]. A user story is a brief specification of user needs [25]. Informality is related to the lack of standardization in the specification of user stories, which are usually presented as unstructured text, which generates the need for adequate text exploration techniques in order to generate good characteristics to be used by techniques of Artificial Intelligence.

This paper presents a systematic mapping of the literature on the application of ABEE techniques in different contexts, looking at the discovery of research opportunities in the estimation of software effort. This systematic mapping is a necessary step to elicit the state of the art in ABEE, considering that the studies in this category of estimation have increased considerably in recent years.

The main goals of this paper are: 1. to provide a mapping of the studies in ABEE regarding: publication sources, research approach, types of contribution, techniques used, etc. and 2. to identify the resources used as inputs to the ABEE estimation process in order to identify other possible research gaps.

The next section presents the study plan developed to conduct this systematic mapping, followed by the presentation of the answers to the research questions and the discussions about the results and possible areas for ABEE research. Finally, the conclusion about this mapping is presented.

2 Methodology

The systematic mapping process followed the guidelines of [58] for drawing up the study. This plan specifies search expressions, search strategies, search engines, inclusion/exclusion criteria of studies, and data systematization, analysis and synthesis.

Systematic mapping studies are a type of systematic literature review that aims to collect and organize research articles related to a specific topic [58, 3, 42]. This type of study requires a careful and detailed search process, with well-defined inclusion and exclusion criteria [18]. According to [58], a systematic mapping usually presents broader research questions than a systematic review, primarily concerned with the structuring of a research area.

2.1 Research questions

Goal: this systematic mapping aims to identify studies related to ABEE techniques, being designed to provide answers to the following research questions:

- Q1.What is the distribution of ABEE publications over time? Justification: to verify how ABEE in research has evolved in the last years.
- Q2.What are the most common publication sources for ABEE publications? Justification: identify sources of publication that are more relevant to the topic.
- Q3.What are the countries of origin of ABEE publications and its main authors? Justification: to monitor the research carried out in the area and enable the exchange of experiences.
- Q4.What types of contributions do the ABEE studies present? Justification: to verify the volume of practical and theoretical work carried out in ABEE.
- Q5.What databases are used in the selected studies? Justification: to know the characteristics of the databases used in the studies carried out
- Q6.What solutions (methods, techniques, models) have been proposed in ABEE? Justification: to map the diversity of ABEE solutions available in the literature, independent of the development process adopted, and classify them to identify trends and / or common aspects within this area.
- Q7.What resources are used as input to the mapped estimation techniques in order to generate ABEE? Justification: to identify resources used as inputs to prediction techniques already studied.

2.2 Search strategy

The search was planned to cover the largest number of studies about software effort estimation, without specifying, at that moment, the estimation and development process model. Thus, the search expressions presented in Table 1, for each search engine, were selected and calibrated. These search engines were used because they appeared as the main sources of search and as the most used digital libraries for systematic studies [17, 26, 37].

Table 1: Search expressions

IEEE Explorer	**Number of articles returned**
((âestimating softwareâ OR âestimation softwareâ OR âeffort estimationâ OR âestimation of softwareâ OR âsoftware effortâ) AND ("Abstract":âestimation softwareâ OR "Abstract":"estimating softwareâ OR "Abstract":"effort estimationâ) AND ("Document Title":âestimating softwareâ OR "Document Title":âeffort estimationâ OR "Document Title":âsoftware effortâ OR "Document Title": âestimation softwareâ OR "Document Title":âestimation of softwareâ))	580
ACM Digital Library	
(+estimating +software +estimation +software +effort +estimation +estimation +of +software +software +effort) AND acmdlTitle:(+estimating +software +effort +estimation +software +effort +estimation +software +estimation +of +software)	91
Science Direct	
- With at least one of the words in the title: "estimating software" OR "estimation software" OR "estimation of software" OR "effort estimation" OR "estimating software" OR "software effort"	109
Springer	
- With at least one of the words in the title: "estimating software" OR "estimation software" OR "estimation of software" OR "effort estimation" OR "estimating software" OR "software effort"	174

The research considered papers published from 2007 to the 2017, including the first quarter of 2018, and was conducted separately for each search source. Other relevant search sources such as Google Scholar and DBLP were not used because their results were included in the other sources consulted.

2.3 Inclusion/Exclusion Criteria

The selection of the studies occurred from the reading of the title, summary and keywords of the resulting publications. Two filters were applied on the 954 studies retrieved by the search expressions (see Table **??**) following inclusion criteria were applied in the 1st filter (CI1F):

- CI_1F–1: The study defines and/or presents theoretical and practical aspects aimed at the recovery/ generation of effort estimates in software projects.
- CI_1F–2: The study investigates, compares or evaluates proposed methods for the recovery/ generation of effort estimates in software projects.
- CI_1F–3: The study analyzes the application of methods aimed at the recovery/generation of effort estimates in software projects.

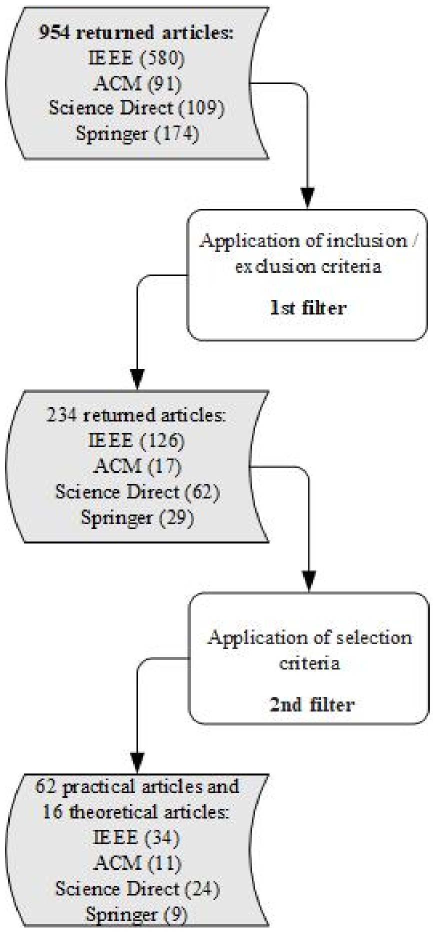

Figure 1: Selection process of publications

The following exclusion criteria for the first filter (CE_1F) were applied excluding papers that:

- CE_1F–1: Does not address an estimate of effort or time in software development projects.

- CE_1F–2: Uses only parametric or algorithmic models.
- CE_1F–3: Presents only future works.
- CE_1F–4: Are duplicates of a study already selected.
- CE_1F–5: Do not present any type of investigation, comparison, evaluation or application of methods aimed for ABEE in software projects.

When there were duplicates of articles referring to the same study, the most recent one was considered. For the first filter, the title, abstract and keywords were read, if necessary, the whole text was analyzed. For the remaining 234 articles inclusion criterion of the second filter (CI2F), considered the entire publication (full article and summary article published in conferences or journals), should have contained at least one of the following elements for the recovery/generation of effort estimates in software projects:

- CI_2F–1: are practical studies in ABEE;
- CI_2F–2: General (includes surveys, reviews and systematic mapping in ABEE).

When applying the criteria of the second filter, 78 relevant articles were retrieved, distributed among the selection criteria presented, of which 62 are classified as practical studies, which will be explored in the results of this mapping. The Figure 1 shows the search methodology used in this mapping, as well as the number of studies returned for each search in their respective digital library and the amounts of studies remaining after the application of the first and second filters.

2.4 Extraction of data

To answer the research questions of this systematic mapping was defined a set of data to be extracted from the selected articles was defined. Table 2 presents the data extracted from the selected articles.

Table 2: Data to be extracted from publications

Attributes	**Research questions**
Article title	Q1
Year of publication	Q1
Source of publication	Q2
Name (s) of the author(s)	Q3
Country of publication	Q3
Type of contribution (theoretical, systematic review of literature, systematic mapping, Surveys, practice - tool, model, method, technique, comparison)	Q4
Database used in the search: name, public/ private	Q5
IA techniques applied to the estimation: Fuzzy Systems, Fuzzy Analogy, Genetic Algorithms, Artificial Neural Networks, Statistical Models, Decision Trees, Naive Bayes, Case-based Reasoning (CBR), Regression Models â CART, MLR, SWR, Collaborative Filtering, Bees Algorithm, Similarity Measures , Support Vector Machines, Radial Base Function, Differential Evolution, Classical Analogy, Bayesian Regression	Q6
By Analogy (Yes/No): Consider the existence of historical data with or without the use similarity measures	Q6
In agile process model: Yes / No	Q6
Resources used as input in estimation techniques: numerical, textual, mixed	Q7

2.5 Threats to Validity

This systematic mapping draws on a protocol theoretically advocated by [58], enabling a correct and consistent research process, guaranteeing aspects such as generability and descriptive validity. Moreover, it is important to cite potential threats to the validity of the research: i) the use of search expressions and ii) the theoretical validity when evaluating inclusion and exclusion criteria and data extraction.

Regarding the search expressions it should be noted that it was not possible to use the same search expression on all search engines due to differences in the input format for this expression, which may not guarantee complete coverage of all related, relevant studies that were returned. Regarding the theoretical validity, different interpretations for the inclusion and exclusion criteria may occur, as well as for the data to be extracted from the studies, which depends on the researcher's bias, which is common when this analysis is done individually by the researchers.

3 Mapping Results

This section presents the results related to the proposed systematic mapping, according to the research questions previously defined. Throughout the presentation of the results, the necessary discussions are development, highlighting data and techniques used and pointing out to research gaps.

Regarding the distribution of ABEE publications over time (Q1), Figure 2 shows the distribution of selected publications over the last ten years. It can be observed that there has been a growth in research in the field of software effort estimation, mainly in ABEE, the focus of this mapping, especially perceived from 2014, and rising in the following years, especially in 2016. This can be partially explained by the increasing need to improve software development processes and, because of this, the importance of an estimate of quality.

According to [29], the model of analogy-based estimation has received more attention and is presented as a promising and viable technique compared to other methods. One of the latent research aspects of analogy-based effort estimation is how to predict the effort of software projects when they are described by mixed numeric and categorical data [4]. In addition, [38] cites that models by analogy can be easily understood by users, as they resemble the human approach to problem-solving, unlike "black-box" models such as artificial neural networks.

The Table 3 presents the most common publication sources for ABEE studies (Q2), showing that 22% of the selected publications were published in a small set of journals and another 5% on specific conferences/symposium, with the most outstanding journals being the Journal of Systems and Software (JSS), Information and Software Technology (IST), Institution of Engineering and Technology (IET) and Applied Soft Computing. Another 73% was published in various newspapers and conferences, with only few occurrence in each.

Among the countries with the highest number of publications in ABEE, (Q3) are Marocos (12), Canada (10), India (8), China (6) and Iran (6), Jordan and USA appear with 5, United Arab Emirates e Japan appear with 3, Malaysia, Thailand and United Kin appear with 2 publications.

Figure 2: Number of publications by year after 2nd filter

For responding to Q4, which refers to the types of contributions presented by the studies in ABEE, it is results show that: 79% (62 publications) of them are practical contributions, especially those that are classified as methods/techniques, be they innovations or improvements of existing methods/techniques. Only 21% (16 publications) of the contributions are theoretical (systematic reviews of the literature, systematic mapping, surveys).

Table 3: Publication sources vs. occurrence

Title	Type	Number	%
Journal of Systems and Software (JSS)	Journal	6	10
Institution of Engineering and Technology (IET)	Journal	4	6
Applied Soft Computing	Journal	3	5
Symposium On Applied Computing (SAC)	Symposium	3	5
Other	Journals and Conferences	46	74

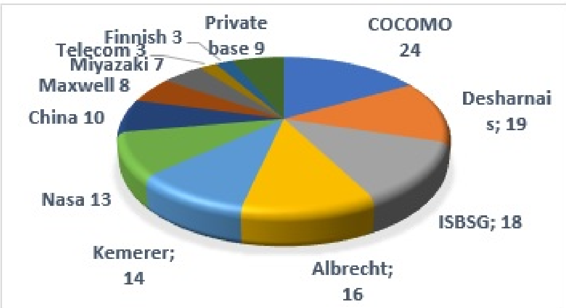

Figure 3: Databases used in surveys

In order to answer Q5, about the databases used in the studies, it was observed among the practical selected studies that approximately 94% (135) of them Figure 3 used historical data from public databases available for research on the development of software projects (e.g. ISBSG, Desharnais, Albrecht). Only 6% (9) of the studies were carried out using private databases obtained from software development organizations. The studies, carried out specifically with agile effort estimation techniques, were based on data provided by industry engineers and other academic studies performed. The study of [**?**] points out that their study was the first time a database was used and was made available for future studies involving requirements in agile models. As specified in Table 2 and Figure 4, the most used techniques in ABEE are Statistical Models (STM) and Fuzzy Systems (FS), followed by Classical Analogy (CA), Regression Models (RM), Case Based Reasoning (CBR) and Artificial Neural Network (ANN).

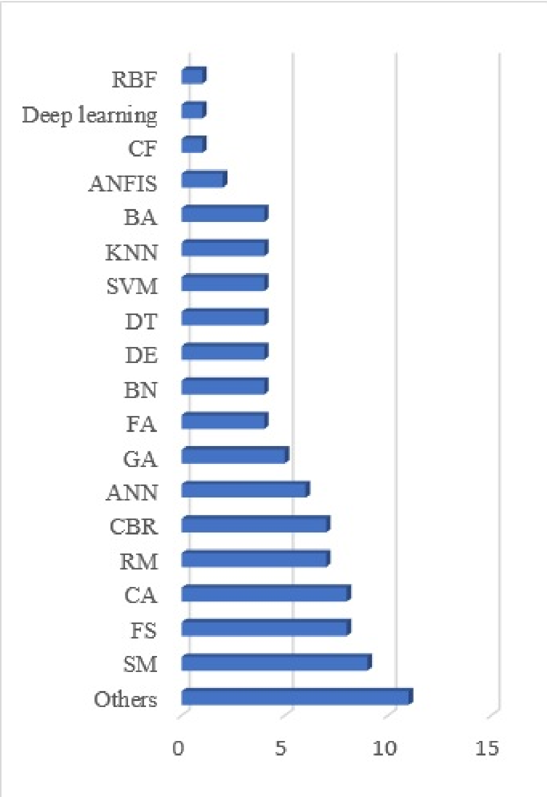

Figure 4: Most used techniques

The Figure 4 relates to the content to Q6, about the solutions presented by the studies analyzed. Upon review of the studies presented, which included the use of Adaptive Neuro-fuzzy Inferences (ANFIS), Collaborative Filtering (FC), Radial Basis Functions (FBR) and Deep Learning, it is possible to identify gaps in research particularly the exploitation of textual artifacts. The study by [22, 23], which applied deep learning to explore the texts of requirements in order to obtain an estimate of the effort of software projects. Other techniques that only have one occurrence and have potential for further research (included in âOtherâ) in which [62] analyzed the Firefly Algorithm and [52] which analyzed the Satin Bower Bird Optimization Algorithm. These three studies present a gap in research, especially in relation to the exploration of textual exploitation.

Considering Q7, only 11% (7 studies) are applied to the agile models of development Table 4, all of which are practical studies, including theoretical and practical studies on estimation by analogy.

The Table 3 presents the practical studies developed in the context of software estimation by analogy and that make use of textual requirements in the initial phase. Each study listed has the techniques used, the development model and its objective.

The Q8 investigates the type of data used as input in the estimation techniques studied Figure 5. Only 13% (8) of the practical studies analyzed use the text exploration of artifacts generated in the initial stages of development (e.g. user stories or other requirements documents) as input to the ABEE. From these, 50% (4) apply to the context of agile models and another 50% to traditional models of development.

The other 87% (54) of the studies presented in this table use other basic design attributes and software size metrics (e.g. points per function, points per story, COSMIC functional size).

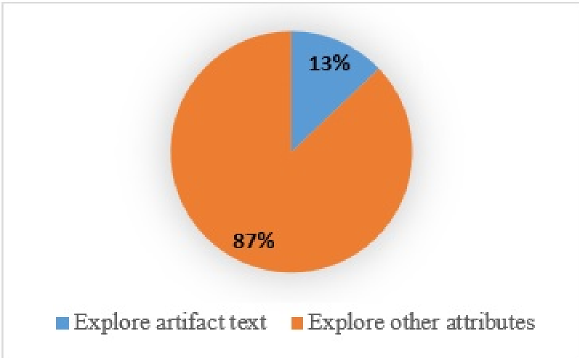

Figure 5: Input data for the estimation process

Table 4: Studies in effort estimation by analogy/development process model

	Number of studies	%	Identification of the publication
Agile models	7	11	[22, 30, 51, 2, 57, 27, 40]
Others	55	89	[45, 69, 20, 21, 10, 11, 9, 37, 82, 38, 39, 12, 73, 61, 29, 4, 49, 48, 9, 28, 43, 46, 47, 7, 19, 55, 81, 56, 6, 66, 32, 1, 63, 52, 5, 8, 14, 80, 79, 62, 59, 70, 77, 54, 38, 71, 13, 65, 34, 60, 36, 33, 64, 74]

Table 5: Studies on software effort estimation using textual requirements

Publication ID	Techniques used	Exploited data	Development model	Overview
[51]	ME, AM (various algorithms)	user history text + history attributes (id, project, estimate)	agile	An automated estimation methodology called "Self-Estimating" was proposed, complementing the Agile Planning Poker model. The Self-Estimating Leverage features extracted from user stories and their actual effort time. The approach is justified by evaluating alternative machine learning algorithms for prediction. It was shown that the selected machine learning methods performed better than Planning Poker's estimates in later phases of a project. This estimation approach is evaluated for accuracy, applicability and value, and results are presented in real-world environment.
[2]	linear regression, SVM, RBF	text of user stories	agile	A method was proposed to predict the effort based on user stories produced from agile models. The proposed method is based on the extraction of predictors from user histories and was applied to two agile software projects in the industrial context. It has been shown that this effort estimate works reasonably well if the user stories are written in a structured way.
[35]	Stanford Parser [44] (tagger POS of Brill [16] and a morphological stemmer), classifiers Naive Bayes, and a logistic classifier	requirement text	traditional	This work aims to develop a tool that automatically realizes a faster approach of the COSMIC size without requiring the formalization of the requirements, demonstrating its applicability in popular agile processes.
[22]	Deep learning: Long Short Term Memory (LSTM) and Recurrent Highway Networks (RHN)	user story text	agile	The model consists of a combination of two powerful deep learning architectures: long-term memory (LSTM) and recurrent road network (RHN). The forecasting system is trainable from start to finish with raw input data for forecast results without any manual resource engineering. The model learns from the point estimates of the team's previous story to predict the size of new stories. It is used in conjunction with (rather than a substitute for) existing estimation techniques practiced by the team.
[81]	Semantic analysis of block and word-level requirements and regression algorithms	requirement text	traditional	An initial software size estimation method has been proposed which can extract semantic features from natural language requirements automatically and construct size estimation models for a project by analogy.

[40]	Artificial neural networks	requirement text	agile	The proposed method uses techniques, such as word merges, to create a system that can estimate effort using only basic project management metrics and textual task descriptions. An artificial neural network was used to automate the task of estimating effort. The method was evaluated with real data from industrial software projects. The results outweigh some of the related libraries.
[56]	Decision tree C 4.5, bayesian networks	requirement text	traditional	From a UML use case diagram or a list of use-case names, automate the COSMIC and IFPUG FPA functional size approximation. The proposed method consists of a two-step process to approximate the functional size of applications based on use-case objectives. First, the names of the use cases, expressed in natural language, were assigned to each of the thirteen categories. In the second step, information on use-case objective categories and historical data was used to construct prediction models and use them to approximate size in COSMIC and Function Points.
[6]	multiple regression analysis	requirement text	traditional	To streamline measurement of size and effort estimation, this study explores the correlations between measures in the problem domain, such as the number of distinct nouns and distinct verbs in the requirement artifacts, and the solution domain measures, such as the number of classes and methods in the corresponding object-oriented software. Twelve commercial software projects were analyzed and multiple analysis were implemented to develop an estimation model for solution domain metrics in terms of problematic domain metrics. The results suggest that for the projects examined, it is possible to use problematic domain measures to make plausible predictions for solution domain metrics.

All studies presented in Table 5 make use of textual requirements of the initial phases of the development process, of which, three specifically explore user histories [51, 2, 22]. Moreover, five of these studies explore the text of other initial requirements [35, 81, 40, 56, 6].

Of the studies presented in Table 3, it is interesting to note that only one of them integrates data extracted from initial textual requirements with attributes of user history (e.g. complexity, priority) and none of the studies integrates data extracted from the text of the initial requirements with attributes of the project and development team.

4 Challenges and Opportunities

The systematic mapping has identified other studies (research, mapping or systematic review of literature), but none of them is a specific review of studies presenting techniques in ABEE.

Regardless of the development model adopted, the existing techniques vary greatly at their precision level, with little consensus in relation to different development contexts. For [76], more studies are needed on estimates in the context of the development of agile software that, also takes into account other predictors of effort, which can be checked as the results of the article.

Considering software development in general, initial requirements are generally textual and, therefore appear as a potential resource to be explored to obtain estimates based on a model by analogy. In agile processes (such as Scrum) –as in many traditional processes –the estimation is usually completed by human participation and is usually based on the experience and historical data of past projects; these projects generally focus on the development effort of each functionality without considering the context in which these projects were conducted.

It should be considered that human factors can also affect the development of projects, because they involve subjective aspects (e. g. the turnover of team members, the lack of experience of one or more team members in certain types of projects or technologies, etc). Such aspects are difficult to measure, and increase the likelihood of possible estimation errors. Therefore, both project-related factors and human factors should be considered when making estimations. However, the systematic mapping study revealed an absence of both factors in the estimation process.

When analyzing the databases used in the estimation studies Figure 3, it was observed that most of them store data from the project context, with little emphasis on the human data involved in development. In addition, only one study used a requirements-based database, which is geared to agile models and is in English. It would be invaluable if this study was made available in other languages and accompanied by the consideration of other facts besides the development effort.

From the point of view of the artifacts used in the estimation process, a considerable portion (50%, 4 studies) of studies within the agile context and others portion of 50% in the traditional context aim at the exploration of textual artifacts to estimate development effort. These studies used the text of requirements and their estimation metric, both individual and used in combinationâas attributes in the initial requirements. Both variants (individual or in combination) neglected to consider data extracted from the text of user stories to the context of the project (especially human factors) –fundamental in estimating effort.

Most of the studies presented use Artificial Intelligence techniques based on learning in the estimation of software effort. These techniques present limitations in the use of variables, either in terms of quantity or its format (numerical or textual). For this reason, models for automatic generation of estimates fail to consider some variables or do not use textual variables; some even still are semi-automatic models and employ the experience of the project manager and the team to insert these variablesâa subjective process with room for error. Thus, an opportunity would be to integrate AI techniques appropriately in order to include all the variables involved in the estimation process. Retrieving the most approximate estimate from historical data of projects (ABEE) presents itself as a promising field of research, always aiming at better results.

ABEE methods require human intervention, either in the recording of previous project data, in the analysis and implementation of the requirements, in the validation of the results obtained from the estimation, among other activities.

Based on the results from the mapping, it is possible to identify critical aspects involved in ABBE. Fig 6 draws on the Organizational Onion to show that these aspects are related to different aspects of formality in an organization, where technical aspects are part of formal aspects that exist in the context of informal activities and practices.

As Figure 6 suggests, informality seems to play an important role in software estimation. Activities carried out in a random manner, that is, there is not necessarily a pre-established pattern for its realization, not for its registration. For example, when done manually, estimates depend on discussions among team members about their experience in similar projects to arrive at

Figure 6: Informal, formal and technical levels of the agile estimation process

the most accurate estimate possible, and for this, historical data are not always available, nor are ways to measure the effort destined to these meetings to estimate a project. As previously mentioned, involving human aspects makes estimation somewhat subjective, since it involves different experiences, expectations, motivations, points of view, etc.; for this reason, the importance of the formalization of the activities carried out at the informal level is important, and detailed more specifically in Figure 6. Especially in agile process models, the role of personal decision making is an imperative because it ensures the dynamics and agility intrinsic to the development process. In this process model, the estimation of effort for the realization of a functionality usually occurs through the use of algorithmic models (e.g. estimation of points by history or points by use cases), which are assigned by the members of the development team [57]. When historical data is available, estimate is done in an analogous way, often manually or semi-automatically, which does not rule out the human role in this process, at least to evaluate the resulted estimate.

Therefore, an important contribution would be the formalization of the development process, as shown in the Formal level of Figure 6, which would allow all these aspects to be considered in the generation of the estimates, thus reducing the human effort in the estimation process, resulting in more accurate estimates generated from actual data based on the context of each project (e.g. type of project, complexity, level of developer experience, technology, among others).

On the hand, involving human aspects makes estimation somewhat subjective, since it involves different experiences, expectations, motivations, points of view, among others. On the other hand, it is a rich source of information that carries relevant knowledge from the context where the software development takes place, including people, the environment, organizational structure, stakeholders and so on. Therefore, the formalization of the activities carried out at the informal level is important to help taking advantage from this informal richness. Especially in agile process models, the role of people in decision making becomes critical in order to ensure the dynamics and agility intrinsic to the process.

In agile models, requirements are defined in a rather lean way, with minimal bureaucracy. In the case of generating a feasible estimate from data extracted from system's requirements (e.g. from user stories texts), this specification of requirements requires a standard structure. According to [22], although some initiatives can be found, this standard structure is still non-existent or not widley accepted in agile models, suggesting the need for research to reduce this gap.

Such an structure may make it to formalize requirements in order to ensure that they present the information necessary to generate a consistent estimate, preserving the richness of the informal context without overloading or restricting the activity.

Approaches for the automatic or semi-automatic generation for agile estimation, based on artifacts created in the initial phase of development, are still presented as a challenge. It is important to be aware of not losing the essence and principles of agile models, taking advantage of their flexibility and paving room for formal and automatic approaches.

5 Conclusion

This paper presented the main results from a systematic mapping of Analogy-based software effort estimation studies. The mapping covered studies published from 2007 to 2017 (including the first quarter of 2018), indexed by the main digital libraries (IEEE-Explorer, ACM Digital Library, Springer, Science Direct) available for Computer Science research.

Based on the results of the mapping, it was possible to observe that the model of estimation by analogy has received more attention and is presented as a promising and feasible technique in relation to other methods. One possible reason for that is its similarity with techniques that use classic analogies. The techniques of Adaptive Neuro-fuzzy Inferences (ANFIS), Collaborative Filtering (FC), Radial Basis Functions (FBR) and Deep Learning present a gap to be explored, as they have still been little explored for estimation purposes.

In agile models, the estimate is usually generated based on the experience of the people involved in the development process, even if there is a development history. Typically, the development history stores only data from the individual project itself, with little or no data on human aspects involved. This can be explained because the estimation process in agile environments is still essentially manual, with direct human intervention. In addition,

there are limitations in existing project management tools for the effective exploitation of this historical data. Therefore, there is a demand for research on the software effort estimation process by analogy in the agile context allied to the use of data extracted from real agile contexts.

6 Bibliograph References

References

[1] Abnane, I., Idri, A., and Abran, A. Empirical evaluation of fuzzy analogy for software development effort estimation. In *Proceedings of the Symposium on Applied Computing*, pages 1302–1304. ACM, 2017.

[2] Abrahamsson, P., Fronza, I., Moser, R., Vlasenko, J., and Pedrycz, W. Predicting development effort from user stories. In *Empirical Software Engineering and Measurement (ESEM), 2011 International Symposium on*, pages 400–403. IEEE, 2011.

[3] Acuña, S. T., Castro, J. W., Dieste, O., and Juristo, N. A systematic mapping study on the open source software development process. In *Evaluation & Assessment in Software Engineering (EASE 2012), 16th International Conference on*, pages 42–46. IET, 2012.

[4] Amazal, F. A., Idri, A., and Abran, A. Improving fuzzy analogy based software development effort estimation. In *Software Engineering Conference (APSEC), 2014 21st Asia-Pacific*, volume 1, pages 247–254. IEEE, 2014.

[5] Araújo, R. d. A., Oliveira, A. L., and Meira, S. A class of hybrid multilayer perceptrons for software development effort estimation problems. *Expert Systems with Applications*, 90:1–12, 2017.

[6] Ayyildiz, T. E. and Koçyigit, A. A case study on the utilization of problem and solution domain measures for software size estimation. In *Software Engineering and Advanced Applications (SEAA), 2016 42th Euromicro Conference on*, pages 108–111. IEEE, 2016.

[7] Azzeh, M. Adjusted case-based software effort estimation using bees optimization algorithm. *Knowlege-Based and Intelligent Information and Engineering Systems*, pages 315–324, 2011.

[8] Azzeh, M. and Nassif, A. B. A hybrid model for estimating software project effort from use case points. *Applied Soft Computing*, 49:981–989, 2016.

[9] Azzeh, M., Nassif, A. B., and Minku, L. L. An empirical evaluation of ensemble adjustment methods for analogy-based effort estimation. *Journal of Systems and Software*, 103:36–52, 2015.

[10] Azzeh, M., Neagu, D., and Cowling, P. I. Analogy-based software effort estimation using fuzzy numbers. *Journal of Systems and Software*, 84(2):270–284, 2011.

[11] Bardsiri, V. K., Jawawi, D. N. A., Bardsiri, A. K., and Khatibi, E. Lmes: A localized multi-estimator model to estimate software development effort. *Engineering Applications of Artificial Intelligence*, 26(10):2624–2640, 2013.

[12] Bardsiri, V. K., Jawawi, D. N. A., Hashim, S. Z. M., and Khatibi, E. Increasing the accuracy of software development effort estimation using projects clustering. *IET software*, 6(6):461–473, 2012.

[13] Benala, T. R. and Bandarupalli, R. Least square support vector machine in analogy-based software development effort estimation. In *Recent Advances and Innovations in Engineering (ICRAIE), 2016 International Conference on*, pages 1–6. IEEE, 2016.

[14] Benala, T. R. and Mall, R. Dabe: Differential evolution in analogy-based software development effort estimation. *Swarm and Evolutionary Computation*, 38:158–172, 2018.

[15] Boehm, B., Abts, C., and Chulani, S. Software development cost estimation approachesâa survey. *Annals of software engineering*, 10(1-4):177–205, 2000.

[16] Brill, E. A simple rule-based part-of-speech tagger proceedings of the 3rd anlp san francisco.

[17] Britto, R., Freitas, V., Mendes, E., and Usman, M. Effort estimation in global software development: A systematic literature review. In *Global Software Engineering (ICGSE), 2014 IEEE 9th International Conference on*, pages 135–144. IEEE, 2014.

[18] Budgen, D., Turner, M., Brereton, P., and Kitchenham, B. Using mapping studies in software engineering. In *Proceedings of PPIG*, volume 8, pages 195–204. Lancaster University, 2008.

[19] Cerpa, N., Bardeen, M., Astudillo, C. A., and Verner, J. Evaluating different families of prediction methods for estimating software project outcomes. *Journal of Systems and Software*, 112:48–64, 2016.

[20] Chinthanet, B., Phannachitta, P., Kamei, Y., Leelaprute, P., Rungsawang, A., Ubayashi, N., and Matsumoto, K. A review and comparison of methods for determining the best analogies in analogy-based software effort estimation. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 1554–1557. ACM, 2016.

[21] Chiu, N.-H. and Huang, S.-J. The adjusted analogy-based software effort estimation based on similarity distances. *Journal of Systems and Software*, 80(4):628–640, 2007.

[22] Choetkiertikul, M., Dam, H. K., Tran, T., Pham, T., Ghose, A., and Menzies, T. A deep learning model for estimating story points. *arXiv preprint arXiv:1609.00489*, 2016.

[23] Choetkiertikul, M., Dam, H. K., Tran, T., Pham, T. T. M., Ghose, A., and Menzies, T. A deep learning model for estimating story points. *IEEE Transactions on Software Engineering*, 2018.

[24] Choi, P. S. P. S. V., Soonhwang. A rule-based approach for estimating software development cost using function point and goal and scenario based requirements. *Expert Systems with Applications*, 39(1):406–418, 2012.

[25] Cohn, M. *Agile estimating and planning*. Pearson Education, 2005.

[26] Costa, L. A. and Salvador, L. d. N. Ambiente de aprendizagem presencial e virtual integrados com a computação ubíqua: Um mapeamento sistemático da literatura. In *Memorias del XX Congresso Internacional de Informática Educativa, TISE*, volume 11, pages 211–220, 2015.

[27] Dragicevic, S., Celar, S., and Turic, M. Bayesian network model for task effort estimation in agile software development. *Journal of Systems and Software*, 127:109–119, 2017.

[28] El Bajta, M. Analogy-based software development effort estimation in global software development. In *Global Software Engineering Workshops (ICGSEW), 2015 IEEE 10th International Conference on*, pages 51–54. IEEE, 2015.

[29] Ezghari, S., Zahi, A., and Idri, A. A learning adaptation cases technique for fuzzy analogy-based software development effort estimation. In *Complex Systems (WCCS), 2014 Second World Conference on*, pages 492–497. IEEE, 2014.

[30] Garg, S. and Gupta, D. Pca based cost estimation model for agile software development projects. In *Industrial Engineering and Operations Management (IEOM), 2015 International Conference on*, pages 1–7. IEEE, 2015.

[31] Hamdy, A. Genetic fuzzy system for enhancing software estimation models. volume 4, pages 227–232, 2014.

[32] Hosni, M. and Idri, A. Software effort estimation using classical analogy ensembles based on random subspace. In *Proceedings of the Symposium on Applied Computing*, pages 1251–1258. ACM, 2017.

[33] Hosni, M., Idri, A., Nassif, A. B., and Abran, A. Heterogeneous ensembles for software development effort estimation. In *Soft Computing & Machine Intelligence (ISCMI), 2016 3rd International Conference on*, pages 174–178. IEEE, 2016.

[34] Huang, J., Li, Y.-F., Keung, J. W., Yu, Y. T., and Chan, W. An empirical analysis of three-stage data-preprocessing for analogy-based software effort estimation on the isbsg data. In *Software Quality, Reliability and Security (QRS), 2017 IEEE International Conference on*, pages 442–449. IEEE, 2017.

[35] Hussain, I., Kosseim, L., and Ormandjieva, O. Approximation of cosmic functional size to support early effort estimation in agile. *Data & Knowledge Engineering*, 85:2–14, 2013.

[36] Idri, A. and Abnane, I. Fuzzy analogy based effort estimation: An empirical comparative study. In *Computer and Information Technology (CIT), 2017 IEEE International Conference on*, pages 114–121. IEEE, 2017.

[37] Idri, A., azzahra Amazal, F., and Abran, A. Analogy-based software development effort estimation: A systematic mapping and review. *Information and Software Technology*, 58:206–230, 2015.

[38] Idri, A., Hosni, M., and Abran, A. Improved estimation of software development effort using classical and fuzzy analogy ensembles. *Applied Soft Computing*, 49:990–1019, 2016.

[39] Idri, A., Hosni, M., and Abran, A. Systematic literature review of ensemble effort estimation. *Journal of Systems and Software*, 118:151–175, 2016.

[40] Ionescu, V.-S. An approach to software development effort estimation using machine learning. In *Intelligent Computer Communication and Processing (ICCP), 2017 13th IEEE International Conference on*, pages 197–203. IEEE, 2017.

[41] Kazemifard, Z. A. N. M. A. M. F., M. Fuzzy emotional cocomo ii software cost estimation (fecsce) using multi-agent systems. *Applied Software Computing Journal*, 11(12):2260–2270, 2011.

[42] Keele, S. Guidelines for performing systematic literature reviews in software engineering. In *Technical report, Ver. 2.3 EBSE Technical Report. EBSE*. sn, 2007.

[43] Khatibi, E. and Bardsiri, V. K. Model to estimate the software development effort based on in-depth analysis of project attributes. *IET Software*, 9(4):109–118, 2015.

[44] Klein, D. and Manning, C. D. Accurate unlexicalized parsing. In *Proceedings of the 41st annual meeting of the association for computational linguistics*, 2003.

[45] Kocaguneli, E., Gay, G., Menzies, T., Yang, Y., and Keung, J. W. When to use data from other projects for effort estimation. In *Proceedings of the IEEE/ACM international conference on Automated software engineering*, pages 321–324. ACM, 2010.

[46] Kocaguneli, E., Menzies, T., Bener, A., and Keung, J. W. Exploiting the essential assumptions of analogy-based effort estimation. *IEEE Transactions on Software Engineering*, 38(2):425–438, 2012.

[47] Li, J., Ruhe, G., Al-Emran, A., and Richter, M. M. A flexible method for software effort estimation by analogy. *Empirical Software Engineering*, 12(1):65–106, 2007.

[48] Manapian, A. and Prompoon, N. Software time estimation model for requirements change based

on software prototype profiles using an analogy estimation method. In *Computer Science and Engineering Conference (ICSEC), 2014 International*, pages 366–371. IEEE, 2014.

[49] Manikavelan, D. and Ponnusamy, R. Software cost estimation by analogy using feed forward neural network. In *Information Communication and Embedded Systems (ICICES), 2014 International Conference on*, pages 1–5. IEEE, 2014.

[50] Menzies, T., Chen, Z., Hihn, J., and Lum, K. Selecting best practices for effort estimation. *IEEE Transactions on Software Engineering*, 32(11), 2006.

[51] Moharreri, K., Sapre, A. V., Ramanathan, J., and Ramnath, R. Cost-effective supervised learning models for software effort estimation in agile environments. In *Computer Software and Applications Conference (COMPSAC), 2016 IEEE 40th Annual*, volume 2, pages 135–140. IEEE, 2016.

[52] Moosavi, S. H. S. and Bardsiri, V. K. Satin bowerbird optimizer: A new optimization algorithm to optimize anfis for software development effort estimation. *Engineering Applications of Artificial Intelligence*, 60:1–15, 2017.

[53] Najadat, A. I. S. Y., H. Predicting software projects cost estimation based on mining historical data. *ISRN Software Engineering*, pages 1–8, 2012.

[54] Nanda, S., Soewito, B., et al. Modeling software effort estimation using hybrid pso-anfis. In *Intelligent Technology and Its Applications (ISITIA), 2016 International Seminar on*, pages 219–224. IEEE, 2016.

[55] Nassif, A. B., Capretz, L. F., and Ho, D. Analyzing the non-functional requirements in the desharnais dataset for software effort estimation. *arXiv preprint arXiv:1405.1131*, 2014.

[56] Ochodek, M. Functional size approximation based on use-case names. *Information and Software Technology*, 80:73–88, 2016.

[57] Panda, A., Satapathy, S. M., and Rath, S. K. Empirical validation of neural network models for agile software effort estimation based on story points. *Procedia Computer Science*, 57:772–781, 2015.

[58] Petersen, K., Vakkalanka, S., and Kuzniarz, L. Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 64:1–18, 2015.

[59] Phannachitta, P., Keung, J., Monden, A., and Matsumoto, K. A stability assessment of solution adaptation techniques for analogy-based software effort estimation. *Empirical Software Engineering*, pages 1–31, 2016.

[60] Rao, P. S., Reddi, K. K., and Rani, R. U. Optimization of neural network for software effort estimation. In *Algorithms, Methodology, Models and Applications in Emerging Technologies (ICAMMAET), 2017 International Conference on*, pages 1–7. IEEE, 2017.

[61] Ren, X., Dai, Y., and Zhou, L. Application in effort estimation of collaborative filtering. In *Computational Intelligence and Design (ISCID), 2013 Sixth International Symposium on*, volume 1, pages 330–333. IEEE, 2013.

[62] Resmi, V., Vijayalakshmi, S., and Chandrabose, R. S. An effective software project effort estimation system using optimal firefly algorithm. *Cluster Computing*, pages 1–10, 2017.

[63] Rijwani, P. and Jain, S. Enhanced software effort estimation using multi layered feed forward artificial neural network technique. *Procedia Computer Science*, 89:307–312, 2016.

[64] Rumjaun, S. D. N. S. S., Gutteea, K. A., and Nagowah, L. Effortest-an enhanced software effort estimation by analogy method. *ADBU Journal of Engineering Technology*, 5(2), 2016.

[65] Sánchez, R. and Pinto-Roa, D. P. Design of software effort estimation models an approach based on linear genetic programming. In *Computer Conference (CLEI), 2017 XLIII Latin American*, pages 1–10. IEEE, 2017.

[66] Satapathy, S. M. and Rath, S. K. Empirical assessment of machine learning models for effort estimation of web-based applications. In *Proceedings of the 10th Innovations in Software Engineering Conference*, pages 74–84. ACM, 2017.

[67] Shepperd, M. Software project economics: a roadmap. In *2007 Future of Software Engineering*, pages 304–315. IEEE Computer Society, 2007.

[68] Shepperd, M. and Kadoda, G. Comparing software prediction techniques using simulation. *IEEE Transactions on Software Engineering*, 27(11):1014–1022, 2001.

[69] Shivhare, J. and Rath, S. K. Software effort estimation using machine learning techniques. In *Proceedings of the 7th India Software Engineering Conference*, page 19. ACM, 2014.

[70] Silhavy, R., Prokopová, Z., and Silhavy, P. Algorithmic optimization method for effort estimation. *Programming and Computer Software*, 42(3):161–166, 2016.

[71] Singh, S. P. and Kumar, A. Software cost estimation using homeostasis mutation based differential evolution. In *Intelligent Systems and Control (ISCO), 2017 11th International Conference on*, pages 173–181. IEEE, 2017.

[72] Sternberg, R. J. Component processes in analogical reasoning. *Psychological Review*, 84:353–378, 1977.

[73] Thamarai, I. and Murugavalli, S. Using differential evolution in the prediction of software effort. In *Advanced Computing (ICoAC), 2012 Fourth International Conference on*, pages 1–3. IEEE, 2012.

[74] Thamarai, I. and Murugavalli, S. An evolutionary computation approach for project selection in analogy based software effort estimation. *Indian Journal of Science and Technology*, 9(21), 2016.

[75] The Standish Group International. The chaos report, 1995.

[76] Usman, M., Mendes, E., Weidt, F., and Britto, R. Effort estimation in agile software development: a systematic literature review. In *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*, pages 82–91. ACM, 2014.

[77] Velarde, H., Santiesteban, C., Garcia, A., and Casillas, J. Software development effort estimation based-on multiple classifier system and lines of code. *IEEE Latin America Transactions*, 14(8):3907–3913, 2016.

[78] Wen, J., Li, S., Lin, Z., Hu, Y., and Huang, C. Systematic literature review of machine learning based software development effort estimation models. *Information and Software Technology*, 54(1):41–59, 2012.

[79] Wu, D., Li, J., and Bao, C. Case-based reasoning with optimized weight derived by particle swarm optimization for software effort estimation. *Soft Computing*, pages 1–12, 2017.

[80] Zare, F., Zare, H. K., and Fallahnezhad, M. S. Software effort estimation based on the optimal bayesian belief network. *Applied Soft Computing*, 49:968–980, 2016.

[81] Zhang, C., Tong, S., Mo, W., Zhou, Y., Xia, Y., and Shen, B. Esse: an early software size estimation method based on auto-extracted requirements features. In *Proceedings of the 8th Asia-Pacific Symposium on Internetware*, pages 112–115. ACM, 2016.

[82] Zhang, W., Yang, Y., and Wang, Q. Using bayesian regression and em algorithm with missing handling for software effort prediction. *Information and software technology*, 58:58–70, 2015.