FIT DATA SELECTION BASED ON PROJECT FEATURES FOR SOFTWARE EFFORT ESTIMATION MODELS

Koji Toda, Akito Monden, Ken-ichi Matsumoto

Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama
Ikoma Nara 630-0192, Japan
{koji-to, akito-m, matumoto}@is.naist.jp

ABSTRACT
To construct a better multivariate regression model for software effort estimation, this paper proposes a method to automatically select projects as fit data (a dataset for model construction) from a given project data set based on an estimation target's features. As a result of an experimental evaluation using the ISBSG data set, which is one of the most commonly used project data sets for effort estimation studies, the proposed method showed better estimation performance than the conventional method (of constructing a model using all project data). The median of MRE (Magnitude of Relative Error) was improved from 0.552 to 0.383, and also the median of MER (Magnitude of Error Relative) was improved from 0.457 to 0.381. While regression models were often constructed using all available project data, this paper showed the necessity of fit data selection, and showed that the proposed method is one of the effective and systematic means of doing the selection.

KEY WORDS
Effort estimation, Multivariate regression. Fit data selection,

1. Introduction

Multivariate regression modeling is a simple but widely used method for software effort estimation [1][2]. This is because software tools for model construction are available, various variable selection methods such as forward stepwise selection are available, and these give reasonable estimation performance.
However, few studies have been made on how to prepare a proper fit dataset for the model construction. Indeed, a regression model's estimation performance greatly depends on the fit data. Generally speaking, to construct a better model, fit data should contain "homogeneous" projects whose development environments, processes or application domains are the same. For example, if the estimation target is a mainframe system development, fit data should include mainframe projects and should not include Windows application ones or embedded system ones since the development process and the required reliability are usually quite different. However, sometimes we should not do such a selection because selecting projects reduces the size of the fit dataset and this can decrease the estimation performance of the constructed model. Moreover, there is no common definition of "homogeneous projects." So far, selection of fit data has been done in an ad hoc manner.
This paper proposes an automatic fit data selection method based on an estimation target's project features. Our basic idea is to select as many candidates for fit data sets as possible, each having at least one similar feature with the target project, build regression models using each candidate, and select the best model (and its fit data candidate) having the best "goodness of fit" to the data. For example, assuming that the estimation target has features "business category = banking, programming language = C, architecture = client-server system", we select all the banking system projects as one of the fit data candidates from past projects. We also select C language projects and client-server system projects as other candidates. If the size of a candidate data set is too small, then we remove it from the candidates. Additionally, a total set (including all past projects) is also used as a fit data candidate because if sizes of all other fit data candidates were not large enough, a model constructed from all past projects might be the best choice.
To evaluate the effectiveness of the proposed method, we conducted an experiment using the ISBSG (International Software Benchmarking Standard Group) dataset [3] , which is one of the most commonly used project data sets for effort estimation studies [4][[5]. In the experiment, we compared our method with a naive regression model built from all available project data (i.e. without selection).
In the rest of this paper, Section 2 describes problems of fit data selection. Section 3 provides details of the proposed method. Section 4 describes an experiment to evaluate the proposed method. Section 5 gives experimental results and discussion. Finally, Section 6 summarizes and outlines future work.

Table 1. An example of project data set

Management Attributes		Project attributes			Architecture			Requirement			Size				
Project ID	Dept. code	Development type	Business area type	Application area type	Platform	Job	Database	Capability	Security	Portability	SLOC (Planned)	SLOC (Recorded)	Effort (Planned)	Effort (Recorded)	…
06S101	Industrial Dept. 1	New Development	Finance	Customer management	Windows	Interaction	DB2	Medium	High	N/A	10000	14239	12 staff month	16 staff month	…
06G201	Industrial Dept. 2	Re Development	Retail	Ordering	UNIX	Batch	Oracle	High	High	Low	28000	30940	60 staff month	68 staff month	…
06G01	Public Work Dept.	Enhancement	Government	Personnel affairs	Windows	Interaction	My SQL	Medium	High	Medium	8000	7900	12 staff month	8 staff month	…
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…

2. Problem of Fit Data Selection

In general, an effort estimation model is constructed from past project data available in a company. Table 1 shows a typical example of a project data set. Each row represents a project and each column represents a feature of a project. For example, the third row in the table represents a project whose Project ID is 06S101, Dept. code is “Industrial Dept. 1”, etc.

Multivariate regression modeling is one of the most commonly used modeling methods that use a data set like that in Table 1. However, since the model itself is quite simple, it is difficult to represent all types of projects by one model. For example, the productivity of business systems is 8 to 18 times greater than that of real-time embedded systems [6]. This means, real-time embedded systems require 8 to 18 times greater effort than business systems even if their product sizes are the same. Although this difference can be given as a Boolean (0 or 1) variable in a model, 8 to 18 times greater effort cannot be characterized since the regression model represents only additive relationships among variables.

Hence, to construct a better multivariate regression model, selecting “homogeneous” projects that have the same project features as the target project (system type, development environment, etc.) is important in fit data selection. Nevertheless, sometimes we should not perform the selection because it reduces the size of a fit dataset and this can decrease the estimation model performance of the one constructed from the fit data. Moreover, there is no common definition of “homogeneous projects.” So far, selection of fit data has been done in an ad hoc manner by practitioners.

3. Fit Data Selection Method

Here we propose a fit data selection method based on an estimation target’s project features. Our basic idea is to select as many candidates for fit data sets as possible, each having at least one similar feature with the target project, build regression models using each candidate, and select the best model (and its fit data candidate) having the best “goodness of fit” to the data. Below we explain the procedure using a simple example shown in Figure 1. In the figure, the target project has features “Language = COBOL”, ”Architecture = Stand Alone” and ”Dev.(Development) Type = Re-Dev.(Redevelopment)”. The procedure is shown as follows (STEPS 1-3).

In the Figure 1 and procedure, one similar feature is used for fit data selection however proposed method is not limited only one to select. If greater than one combination among features is used for selection, prediction accuracy will be higher than one feature is used. In this paper, proposed method’s feature for selection is one because of space limitation.

[STEP 1] Candidates for fit data sets are selected from a past project data set. In this step, we choose one feature from an estimation target project and select projects from all past fit data including the feature. Selected projects are treated as candidates. We perform this procedure for each feature included in the target project and all candidates are treated as “fit data candidates”. For example, since the

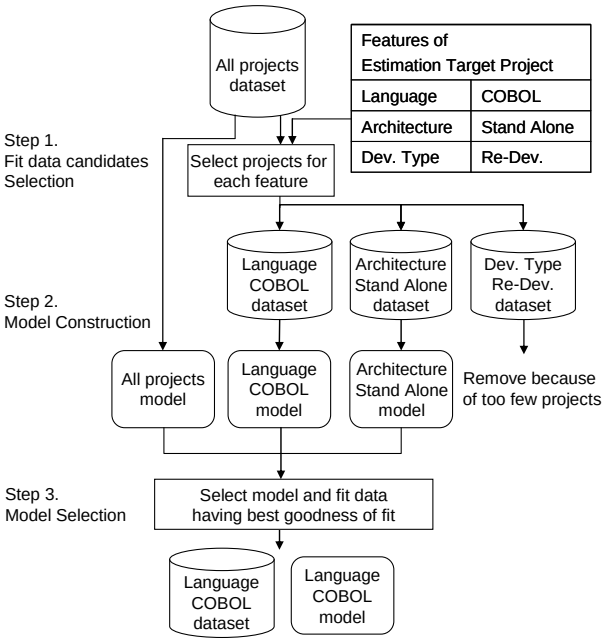

Figure 1. Procedure of fit data selection

target project has three features (COBOL, Stand Alone and Re-Dev.), each fit data candidate contains COBOL language projects, Stand Alone system projects and redevelopment projects.

If the size of a candidate is smaller than a minimum size (e.g. 10 projects), we remove this candidate from the candidate set. In this paper we call this size "minimum fit data size." In addition to the resultant candidates, we also add a dataset containing all past projects (a total set) as a candidate because selection sometimes causes a negative effect on model construction. In this step, we calculate the threshold (for example median, quantile deviation, etc.) from all past projects about each numeric variable because a numeric variable cannot use fit data selection (if a candidate having same value in one numeric variable are collected from past data, the candidate contain a few projects).

[STEP 2] We construct regression models each using one of the fit data candidates and calculate the goodness of fit (e.g., residual mean square or adjusted R-square).

[STEP 3] We select the best model and its fit data candidate based on the goodness of fit. In Figure 1, the COBOL language model and its fit data were selected.

These steps are simplified ones where all project features (Language, Architecture, Dev. Type, etc.) are given as categorical variables. However, project data sets usually contain quantitative variables (e.g. Function Points and Project Length); therefore, we translate the quantitative variables into categorical ones by partitioning each quantitative variable by a given threshold in STEP 1. For example, if the median is used as a threshold and we choose one numeric variable of the target project and its value is over or equal to the threshold, the numeric variable is treated as a categorical variable and named "variable name + high"; and if it is under the threshold, the numeric variable is treated as a categorical one and named "variable name + low."

Concretely, if the threshold of Project Length is "12 months" and the estimation target project's Project Length is "10 months", we select projects from all past projects that satisfy their Project Length of less than 12 months and treat them as fit data candidates "Project Length: low". Likewise, if the target project's "Project Length" is "15 months", we select projects satisfying over or equal to 12 months and call them " Project Length: high".

The detail of the procedure is shown in appendix. In this paper, we defined each quantitative variable's threshold as its median value in the experiment (Section 4).

4. Experiment

4.1 Overview

The goal of this experiment is to evaluate the proposed method using actual project data. We used the ISBSG dataset established by the International Software Benchmarking Standard Group (Section 4.2). When building regression models, we used the total development effort (denoted "Summary Work Effort" in the ISBSG dataset) as an objective variable and we used other variables (such as Function Point) that can be measured by the end of the design phase as predictor variables.

In the experiment, we set the minimum fit data size to be 10. Therefore, if one of the fit data candidates contained less than 10 projects, this candidate was removed from the candidate sets. Note that "minimum fit data size = 10" may not be the best choice, but finding the best minimum fit data size is out of the scope here. The main goal of this experiment is to compare the proposed method (i.e. automatic fit data selection) with the conventional method (i.e. using all projects as fit data).

Number of feature to select fit data is one in this experiment, because used dataset has not so many projects that almost fit data candidates are not include over minimum fit data size after fit data selection when number of feature selection is greater than one.

As measures of "the goodness of fit" of a constructed model, Residual Mean Square (RMS), adjusted R-square (adj.R^2), Least Absolute Deviation (LAD), etc. have been used [7]. In this experiment, we evaluated two commonly used criteria: (1) residual mean squared (RMS) and (2) adjusted R-squared (adj.R^2) to identify the best fit data candidate.

We used the stepwise multivariate regression analysis as a modeling technique. The forward stepwise selection method was used for the variable selection.

Table 2. Features involved in the experimental dataset

Features	Scale	Examples
Count Approach	Nominal	COSMICFFP, IFPUG, NESMA, etc.
Function Points	Ratio	20, 45, 80, 250, etc.
Summary Work Effort	Ratio	100, 160, 380, 1200, etc.
Effort Plan	Ratio	15, 32, 50, 100, etc.
Effort Specify	Ratio	20, 36, 68, 140, etc.
Development Type	Nominal	New Development, Enhancement or Redevelopment
Architecture	Nominal	Client Server, Stand Alone or Multi-tier
Primary Programming Language	Nominal	COBOL, Java, C, etc.
Recording Method	Nominal	Productive Time Only Recorded, Stuff Hours, etc.
Resource Level	Ordinal	Level 1 - 4

4.2 Dataset

We used the ISBSG dataset in this experiment. The dataset is collected from 20 countries, from 1989 to 2004 and contains 3026 projects each having (at most) about 100 feature variables. However, since it contains a lot of missing values, not all the projects are usable for building regression models.

Therefore, we removed projects and variables (features) containing missing values, and as a result we built an initial project dataset of 109 projects and 10 variables having no missing values. Table 2 shows a list of features included in the initial dataset. In the table, "Summary Work Effort" is the objective variable and the other 9 features are predictor variables.

Details of variables are described in [8]. These 10 variables include 4 ratio scale variables, 5 nominal scale ones and 1 ordinal scale one. Nominal scale and ordinal scale variables are each converted into a set of Boolean (1 or 0) variables.

In the experiment, we split the initial dataset into fit and test randomly, each with the same size. We repeated this operation 10 times.

4.3 Evaluation Criteria

To evaluate the estimation error, we used MRE (Magnitude of Relative Error) [9] and MER (Magnitude of Error Relative) [10]. If MRE is too large, that means the effort is over-estimated, and if MER is too large, that means the effort is under-estimated. In other studies, Absolute Error is often used, however we did not use this criterion because using both MRE and MER is enough to evaluate both over- and under-estimation.

To evaluate the estimation performance of models, we used the mean of MRE and MER (namely, MMRE and MMER), the their median (MdMRE and MdMER), the their variance (VMRE and VMER) and their Pred(25) (Pred(25).MRE and Pred(25).MER) [11].

The formulas of MRE, MER and Pred(25).MRE are shown as follows. In the formulas, E denotes actual objective variable, denotes its predicted actual variable and N denotes the number of projects.

Magnitude of Relative Error (MRE):

$$MRE = \left| \frac{E - \hat{E}}{E} \right|$$

Magnitude of Error Relative (MER):

$$MER = \left| \frac{E - \hat{E}}{\hat{E}} \right|$$

Percentage of relative error deviation within 25 (Pred(25)):

$$Pred(25).MRE = \frac{1}{N} \sum_{i=1}^{N} \begin{cases} 1 & if\ MRE_i \le 25 \\ 0 & otherwise \end{cases}$$

5. Result and Discussion

5.1 Result

The results are shown in Table 3 and Table 4, Figure 2 and Figure 3. In these tables and figures, "RMS" indicates the proposed method using the residual mean square (as a goodness of fit) and "adj.R^2" indicates that using the adjusted R-squared. "Conventional" means that a regression model was built using all available projects (i.e. without fit data selection).

As we compared the proposed method using "RMS" with that using "adj.R^2", the former showed better performance in all evaluation criteria (significant difference ($p<0.05$) was seen in MMRE, MMER, MdMRE, MdMER, Pred(25).MRE and Pred(25).MER). Therefore, next we compare the proposed method using "RMS" with the conventional method.

As we compared the proposed method "RMS" with the conventional method (of constructing a model using all project data), "RMS" showed better performance in all criteria. For example, MdMRE was 0.383 in the proposed method, while it was 0.552 in the conventional method. There was significant difference ($p<0.05$) in all criteria.

Table 3. Result of experiment (MRE)

	Conventional	The proposed method with RMS	The proposed method with adj.R^2
MMRE	1.181	0.783	0.977
MdMRE	0.552	0.383	0.507
Pred(25).MRE	24.6	36.3	30.9
VMRE	7.623	2.706	3.817

Table 4. Result of experiment (MRE)

	Conventional	The proposed method with RMS	The proposed method with adj.R^2
MMER	1.529	0.829	1.129
MdMER	0.457	0.381	0.418
Pred(25).MER	28.0	36.9	29.8
VMER	155.103	2.173	9.323

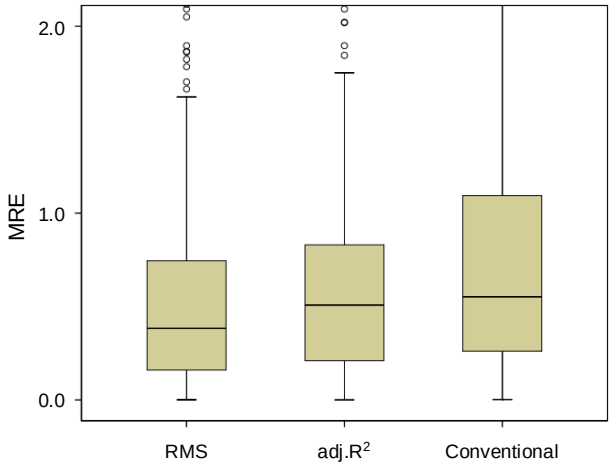

Figure 2. Boxplots of estimation performance (MRE)

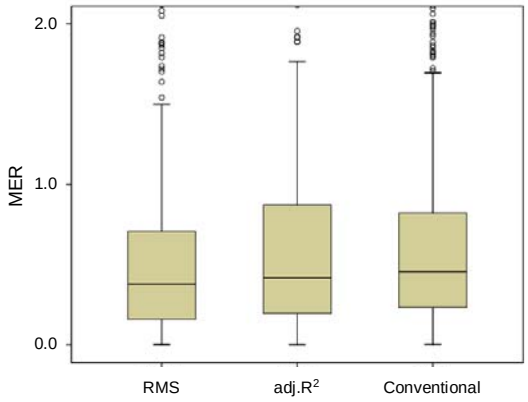

Figure 3. Boxplots of estimation performance (MER)

5.2 Discussion

The experimental result showed RMS was better than adj.R^2 in the proposed method. First, we discuss the reason why such difference occurred. One possible interpretation is that RMS is more robust than adj.R^2 against outliers in the dataset. This is because RMS is calculated from the (average) residual between actual values and their estimates, while adj.R^2 is calculated from the coefficient of correlation between them. If an outlier is contained in the dataset, adj.R^2 is much more affected by the outlier than RMS.

Next we focus on the selected fit data in the proposed method (RMS). Since each fit data set is selected based on a single project feature (e.g. fit data containing projects of “Architecture = Stand Alone”), we counted the number of selections for each project feature (Table 5). In Table 5, for example, “Architecture | Stand Alone | 16” indicates that a model that consists of projects whose architecture was stand alone was selected 16 times for effort estimation in the experiment (of 10 times repetition). Features that have not been selected in experiment are not shown in Table 5.

As shown in Table 5, ratio scale variables (function points, effort plan and effort specify) were frequently selected. In particular, the model of “Function Points = Low” was the most frequently selected (204 times), while the “Function Points = High” model was not selected at all. Likewise, “Low” is more frequently selected than “High” in “Effort Plan” and “Effort Specify”.

This indicates that to estimate the effort of small projects, we should build models from small projects only, while for large projects we should build models using both small and large projects.

Notably, there was a case where all past project data was used to build a model. This indicates that, although fit data selection is required for most projects, there still exists a case where selection is not required for some projects.

Table 5. The number of selections for each project feature

Features		The number of selection in proposed method (RMS)
All Past Data	----	1
Count Approach	COSMIC FFP	1
Count Approach	IFPUG	13
Function Point	Low	204
Effort Plan	High	1
Effort Plan	Low	61
Effort Specify	High	1
Effort Specify	Low	62
Development Type	Enhancement	7
Development Type	New Development	24
Architecture	Client Server	10
Architecture	Stand Alone	16
PPL*	COBOL	4
Recording Method	PTOR**	78
Recording Method	SHR***	13
Resource Level	1	6
Resource Level	2	32
Resource Level	4	6

* PPL : Primary Programming Language
** PTOR : 'Productive' time only (recorded)
*** SHR : Stuff hours (recorded)

6. Conclusion

In this paper, we proposed a fit data selection method for constructing regression models based on features of the target project. As a result of an experimental evaluation using the ISBSG data set, the proposed method showed better estimation performance than the conventional method (of constructing a model using all project data). The median of MRE (Magnitude of Relative Error) was improved from 0.552 to 0.383, and the median of MER (Magnitude of Error Relative) was improved from 0.457 to 0.381.

While regression models were often constructed using all available project data, this paper showed the necessity of fit data selection, and showed that the proposed method is an effective and systematic means of performing the selection.

The proposed fit data selection method can be applied to any types of model-based estimation, e.g. the neural network model. Also it can be applied to models to solve other problems, e.g. linear discriminate models and logistic regression models for the two-class partitioning problem. In the future, we would like to apply our method to these other models.

Acknowledgements

This work was conducted as a part of the StagE Project, the Development of Next Generation IT Infrastructure, supported by the Ministry of Education, Culture, Sports, Science and Technology.

References

[1] B. Boehm, *Software engineering economics* (Englewood Cliffs, NJ: Prentice Hall, 1981).

[2] L. C. Briand, T. Langley & I. Wieczorek, A replicated assessment and comparison of common software cost modeling techniques, *Proc. International Conf. on Software Engineering (ICSE '00)*, The Limerick, The Ireland, 2000, 377-386.

[3] International Software Benchmarking Standards Group, ISBSG estimating benchmarking and research suite release 9, International Software Benchmarking Standards Group, VIC, Australia, 2004.

[4] R. Jeffery, M. Ruhe, & I. Wieczorek, Using public domain metrics to estimate software development effort, *Proc. 7th International Software Metrics Symposium (METRICS '01)*, England, London, 2001, 16 -27.

[5] E. Mendes, C. Lokan, R. Harrison, & C. Triggs, A replicated comparison of cross-company and within-company effort estimation models using the ISBSG database, *Proc. 11th IEEE International Software Metrics Symposium (METRICS '05)*, Italy, Como, 2005, 36.

[6] L. Putnam & W. Myers, *Measures for excellence* (Englewood Cliffs, NJ: Prentice Hall, 1992).

[7] C. Lokan, What should you optimize when building an estimation model?, *Proc. 11th IEEE International Software Metrics Symposium (METRICS '05)*, Italy, Como, 2005, 34.

[8] International Software Benchmarking Standards Group, *The benchmark release 6* (Australia, VIC: International Software Benchmarking Standards Group, 2000).

[9] S. D. Conte, H. E. Dunsmore & V. Y. Shen, *Software engineering metrics and models* (Menlo Park, CA: The Benjamin/Cummings Publishing Company, 1986).

[10] B. A. Kitchenham, S. G. MacDonell, L. M. Pickard, & M. J. Shepperd, What accuracy statistics really measure, *IEE Proceedings Software*, *148*(3), 2001, 81-85.

[11] M. Jørgensen, Experience with the accuracy of software maintenance task effort prediction models, *IEEE Transactions of Software Engineering*, *21*(8), 1995, 674-681.

Appendix

; Step 1 … Building fit data set candidates
Let $P := p_1,\ldots p_n$ a project set including all past projects.
Let $p_t \notin P$ be the target project that needs effort estimation.
Let $V := v_1,\ldots,v_k$ be a set of project attribute.
Let $v_i(p_j)$ be an attribute value of i-th project attribute of project p_j.
Let $threshold_i$ be the threshold of a numerical variable v_i.
For all x ($x = 1,\ldots n$){
  If v_x is a categorical variable then{
    Build a project set $R_x \subseteq P$ where
all $r \in R_x$ satisfies $v_x(r) = v_x(p_t)$ and
        all $s \in P\text{-}R_x$ satisfies $v_x(r) \neq v_x(p_t)$
  }else{
    If $v_x(r) < threshold_x$ then{
    Build a project set $R_x \subseteq P$ where

all $r \in R_x$ satisfies $v_x(r) < threshold_x$ and
all s$\in P\text{-}R_x$ satisfies $v_x(r) \geq threshold_x$
}else{
Build a project set $R_x \subseteq P$ where
all $r \in R_x$ satisfies $v_x(r) \geq threshold_x$ and
all $s \in P\text{-}R_x$ satisfies $v_x(r) < threshold_x$
}
}
}
; Step 2 … Building estimation models
Let *min_projects* be the minimum number
of project required to build an estimation model.
For all x ($x = 1,\ldots n$){
If $|R_x| >$ *min_projects* then build a model m_x
using R_x as a fit dataset.
}
Build a model m_{all} using P as a fit dataset

; Step 3 … model selection
Let M be a set of all models built in step 2.
Let $A(m_x)$ be the goodness of fit of model m_x.
Select a model $m_x \in M$ where $A(m_x)$ is
the best among all $A(m_i \in M)$.